

# **CONTRIBUȚII LA CREȘTEREA DISPONIBILITĂȚII, SCALABILITĂȚII ȘI SECURITĂȚII SISTEMELOR DE COMUNICAȚIE**

Teză destinată obținerii  
titlului științific de doctor inginer  
la  
Universitatea "Politehnica" din Timișoara  
în domeniul ȘTIINȚA CALCULATOARELOR  
de către

**ing. Valer Bocan**

Conducător științific: prof. dr. ing. Vladimir Crețu  
Referenți științifici: prof. dr. ing. Mircea Petrescu  
prof. dr. ing. Victor Patriciu  
prof. dr. ing. Mircea Vlăduțiu

Ziua susținerii tezei: 12.06.2009

Seriile Teze de doctorat ale UPT sunt:

- |                        |   |
|------------------------|---|
| 1. Automatică          | 7. Inginerie Electronică și Telecomunicații |
| 2. Chimie              | 8. Inginerie Industrială                    |
| 3. Energetică          | 9. Inginerie Mecanică                       |
| 4. Ingineria Chimică   | 10. Știința Calculatoarelor                 |
| 5. Inginerie Civilă    | 11. Știința și Ingineria Materialelor       |
| 6. Inginerie Electrică |   |

Universitatea „Politehnica” din Timișoara a inițiat seriile de mai sus în scopul diseminării expertizei, cunoștințelor și rezultatelor cercetărilor întreprinse în cadrul școlii doctorale a universității. Seriile conțin, potrivit H.B.Ex.S Nr. 14 / 14.07.2006, tezele de doctorat susținute în universitate începând cu 1 octombrie 2006.

Copyright © Editura Politehnica – Timișoara, 2009

Această publicație este supusă prevederilor legii dreptului de autor. Multiplicarea acestei publicații, în mod integral sau în parte, traducerea, tipărirea, reutilizarea ilustrațiilor, expunerea, radiodifuzarea, reproducerea pe microfilme sau în orice altă formă este permisă numai cu respectarea prevederilor Legii române a dreptului de autor în vigoare și permisiunea pentru utilizare obținută în scris din partea Universității „Politehnica” din Timișoara. Toate încălcările acestor drepturi vor fi penalizate potrivit Legii române a drepturilor de autor.

România, 300159 Timișoara, Bd. Republicii 9  
tel. 0256 403823, fax. 0256 403221  
e-mail: editura@edipol.upt.ro

## Cuvânt înainte

Scrierea acestui cuvânt înainte este la fel de anevoioasă ca și cercetarea din cadrul activității de doctorat, pentru că emoțiile și trăirile momentului strălucesc în comparație cu puterea de expresie a cuvintelor. Am să încerc totuși să aștern câteva rânduri de recunoștință celor ce au contribuit la formarea mea ca om, cerând iertare pentru stângăcia cu care o fac.

Mulțumesc domnului prof. dr. ing. Vladimir Crețu, conducătorul științific de doctorat, pentru ajutorul, sfaturile și îndrumarea continuă de pe parcursul lungului urcuș al activității mele de doctorat. De asemenea, recunoștința mea se îndreaptă către domnul prof. univ. dr. Victor Valeriu Patriciu de la Academia Tehnică Militară din București pentru ideile și sfaturile pertinente pe care le-am primit cu fiecare ocazie. Colegii mei din cadrul Alcatel-Lucent Romania mi-au fost de un nesperat ajutor în cercetare prin discuții și facilitarea accesului la documente pe care altfel cu greu le-aș fi putut găsi.

Deși nu este specialistă în domeniu, țin să mulțumesc soției mele Viorica pentru devotamentul ei, pentru răbdarea și dragostea cu care m-a înconjurat în momentele grele. Pot spune că fără sprijinul nemijlocit al soției mele, această teză de doctorat nu ar fi existat. Mulțumesc și părinților mei Vasile și Dorina pentru anii grei și lungi de-a lungul cărora m-au ocrotit și mi-au oferit tot ajutorul de care am avut nevoie.

Mai presus decât tuturor, îi mulțumesc Bunului Dumnezeu pentru că mi-a ajutat să ajung până la acest hotar, că mi-a dăruit atâtea satisfacții pe plan profesional și personal și pentru grija nemăsurată pe care mi-o poartă mie și familiei mele.

Autorul,

Mai 2009

*Ficei mele Ioana Carina, a cărei naștere mi-a luminat existența...*

Bocan, Valer

**Contribuții la creșterea disponibilității, scalabilității și securității sistemelor de comunicație**

Teze de doctorat ale UPT, Seria 10, Nr. 17, Editura Politehnica, 2009, 158 pagini, 39 figuri, 8 tabele.

ISSN: 1842-7707

ISBN: 978-973-625-881-7

Cuvinte cheie: comunicație, disponibilitate, scalabilitate, securitate, GSM, DoS, SSO, puzzle, distribuție, conținut

Rezumat: Teza de doctorat prezintă soluții pentru ameliorarea disponibilității, scalabilității și a securității sistemelor de comunicație. Sub acoperirea acestui deziderat integrator, se abordează trei obiective majore:

- Creșterea disponibilității și scalabilității sistemelor de autentificare
- Studiul și analiza problemei vulnerabilității rețelelor GSM la atacuri DoS și elaborarea de propuneri pentru ameliorarea rezistenței
- Propunerea unei arhitecturi de distribuție digitală a conținutului care îmbunătățește într-o manieră semnificativă scalabilitatea și maniera de cooperare între deținătorii drepturilor de autor

## Cuprins

|       |   |    |
|-------|---|----|
| 1.    | INTRODUCERE .....   | 11 |
| 1.1   | FORMULAREA PROBLEMEI .....  | 11 |
| 1.2   | SCOPUL.....   | 13 |
| 1.3   | ORGANIZAREA TEZEI.....  | 14 |
| 2.    | SECURITATE, DISPONIBILITATE ȘI SCALABILITATE ÎN SISTEME DE<br>COMUNICAȚII.....                  | 17 |
| 2.1   | CARACTERISTICI ALE SISTEMELOR DE COMUNICAȚII .....  | 17 |
| 2.1.1 | Securitatea .....   | 17 |
| 2.1.2 | Disponibilitatea .....  | 18 |
| 2.1.3 | Scalabilitatea.....   | 18 |
| 2.2   | SECURITATEA ȘI DISPONIBILITATEA SISTEMELOR DE AUTENTIFICARE.....                                | 19 |
| 2.2.1 | Autentificarea bazată pe parole.....  | 19 |
| 2.2.2 | Autentificarea bazată pe adresă.....  | 20 |
| 2.2.3 | Autentificarea criptografică .....  | 20 |
| 2.2.4 | Protocolul SSL .....  | 21 |
| 2.2.5 | Deficiențe ale protocoalelor de autentificare actuale .....                                     | 23 |
| 2.3   | SECURITATEA ȘI DISPONIBILITATEA SISTEMELOR SINGLE SIGN-ON (SSO) ..                              | 25 |
| 2.3.1 | Tipuri de sisteme SSO.....  | 25 |
| 2.3.2 | Proprietăți ale sistemelor SSO .....  | 29 |
| 2.3.3 | Deficiențe ale sistemelor SSO actuale .....   | 33 |
| 2.4   | SECURITATEA ȘI DISPONIBILITATEA REȚELELOR GSM.....  | 34 |
| 2.4.1 | Mecanisme de securitate în rețelele GSM.....  | 36 |
| 2.4.2 | Deficiențe ale rețelelor GSM .....  | 40 |
| 2.5   | SECURITATEA ȘI SCALABILITATEA SISTEMELOR DRM .....  | 42 |
| 2.5.1 | Deficiențe ale sistemelor DRM .....   | 43 |
| 2.6   | CONCLUZII .....   | 46 |
| 3.    | CONTRIBUȚII LA CREȘTEREA DISPONIBILITĂȚII ȘI SCALABILITĂȚII<br>SISTEMELOR DE AUTENTIFICARE..... | 49 |
| 3.1   | CLIENT PUZZLES .....  | 49 |
| 3.1.1 | Descriere .....   | 50 |
| 3.1.2 | Tipuri existente de puzzle-uri .....  | 51 |
| 3.1.3 | Crearea unui puzzle .....   | 52 |
| 3.1.4 | Rezolvarea puzzle-ului .....  | 52 |
| 3.1.5 | Dificultatea puzzle-ului.....   | 53 |
| 3.1.6 | Neajunsuri ale soluției Client Puzzle.....  | 54 |
| 3.1.7 | Propuneri pentru ameliorarea experienței clientului .....                                       | 55 |
| 3.2   | ASPECTE PRACTICE DE IMPLEMENTARE.....   | 61 |
| 3.2.1 | Algoritmul SSL Handshake .....  | 61 |
| 3.2.2 | Algoritmul SSL Handshake cu distribuție egală de efort.....                                     | 62 |
| 3.2.3 | Algoritmul SSL Handshake cu distribuție adaptivă de efort ...                                   | 64 |

|                      |  |     |
|----------------------|--|-----|
| 3.2.4                | Prototipul software .....  | 66  |
| 3.2.5                | Performanță .....  | 67  |
| 3.3                  | DETECTAREA ȘI AMELIORAREA LOCALĂ A ATACURILOR DoS .....              | 69  |
| 3.3.1                | Caracteristici măsurabile .....                                      | 70  |
| 3.4                  | DETECȚIA EURISTICĂ A ATACURILOR CU SSO-SENSE .....                   | 73  |
| 3.4.1                | Scurtă fundamentare matematică .....                                 | 74  |
| 3.4.2                | Aplicarea detecției euristice în autentificare .....                 | 75  |
| 3.4.3                | Rezultate experimentale .....  | 77  |
| 3.5                  | CONCLUZII .....  | 78  |
| 4.                   | CONTRIBUȚII LA CREȘTEREA DISPONIBILITĂȚII REȚELELOR TIP GSM          | 81  |
| 4.1                  | ATACURI ASUPRA REȚELELOR GSM .....                                   | 81  |
| 4.1.1                | Clasificarea DREAD a vulnerabilităților .....                        | 82  |
| 4.1.2                | Clasificarea vulnerabilităților rețelelor GSM după sistemul          |     |
| DREAD                | 82   |     |
| 4.1.3                | Clasificarea vulnerabilităților după gravitate .....                 | 90  |
| 4.2                  | ATACURI DoS ÎN REȚELE GSM .....                                      | 91  |
| 4.3                  | PROFILUL ATACATORULUI .....  | 95  |
| 4.4                  | ECONOMIA ATACULUI .....  | 96  |
| 4.5                  | CREȘTEREA DISPONIBILITĂȚII REȚELELOR GSM PRIN PREAUTENTIFICARE ..... | 97  |
| 4.5.1                | Impactul preautentificării asupra capacității de semnalizare .....   | 99  |
| 4.5.2                | Avantaje și limitări ale soluției propuse .....                      | 100 |
| 4.6                  | CONCLUZII .....  | 100 |
| 5.                   | CONTRIBUȚII ÎN DOMENIUL SISTEMELOR DE DISTRIBUȚIE DIGITALĂ           |     |
| A CONȚINUTULUI ..... |  | 103 |
| 5.1                  | ARHITECTURĂ SCALABILĂ DE DISTRIBUȚIE DIGITALĂ A CONȚINUTULUI .....   | 104 |
| 5.1.1                | Distribuția conținutului .....                                       | 106 |
| 5.1.2                | Retransmisia conținutului .....                                      | 110 |
| 5.1.3                | Redistribuția conținutului .....                                     | 111 |
| 5.1.4                | Riscuri potențiale .....   | 112 |
| 5.1.5                | Avantajele arhitecturii .....  | 112 |
| 5.2                  | IMPLEMENTARE .....   | 113 |
| 5.2.1                | Structuri de date pentru descrierea conținutului .....               | 113 |
| 5.2.2                | Entități care manipulează conținutul .....                           | 114 |
| 5.2.3                | Operații matematice .....  | 122 |
| 5.3                  | MĂSURĂTORI DE PERFORMANȚĂ .....                                      | 126 |
| 5.3.1                | Faza de inițializare .....   | 126 |
| 5.3.2                | Faza de operare .....  | 128 |
| 5.4                  | AVANTAJE ALE SISTEMULUI DE DISTRIBUȚIE PROPUSE .....                 | 130 |
| 5.5                  | CONCLUZII .....  | 135 |
| 6.                   | CONCLUZII FINALE ȘI PERSPECTIVE .....                                | 137 |

|     |                                    |     |
|-----|------------------------------------|-----|
| 6.1 | CONCLUZII FINALE .....             | 137 |
| 6.2 | PERSPECTIVE .....                  | 139 |
|     | ABREVIERI ȘI ACRONIME .....        | 141 |
|     | BIBLIOGRAFIE .....                 | 147 |
|     | LISTA PUBLICAȚIILOR PERSONALE..... | 155 |





## Lista figurilor

|  |     |
|--|-----|
| FIG. 1 MECANISMUL DE PRINCIPIU AL UNUI SISTEM „PSEUDO-SSO” .....   | 26  |
| FIG. 2 MECANISMUL DE PRINCIPIU AL UNUI SISTEM „TRUE-SSO” .....   | 27  |
| FIG. 3 FLUXUL INFORMAȚIONAL ÎNTR-UN SISTEM SSO GENERIC.....  | 28  |
| FIG. 4 CERCUL DE ÎNCREDERE BAZAT EXCLUSIV PE RELAȚII JURIDICE.....   | 33  |
| FIG. 5 ATACUL DOS ÎMPOTRIVA UNUI SISTEM SSO LIBERTY (INDIFERENT DE PROTOCOL).....                              | 34  |
| FIG. 6 PROBLEME DE SECURITATE ALE REȚELELOR FĂRĂ FIR .....   | 36  |
| FIG. 7 AUTENTIFICAREA ÎN REȚELE GSM .....  | 38  |
| FIG. 8 PROCESUL DE ASIGNARE A UNUI CANAL ÎN GSM .....  | 41  |
| FIG. 9 ATACUL DE TIP DoS ÎNTR-O REȚEA GSM .....  | 42  |
| FIG. 10 SCALABILITATE REDUSĂ ÎNTR-UN SISTEM DE DISTRIBUȚIE AL CONȚINUTULUI, CU DRM. ....                       | 44  |
| FIG. 11 ATAC ASUPRA MODULULUI DE REDARE DINTR-UN SISTEM DRM .....  | 46  |
| FIG. 12 PRINCIPIUL CLIENT PUZZLES .....  | 50  |
| FIG. 13 SCHEMA UNUI ATAC PUTERNIC .....  | 54  |
| FIG. 14 EVOLUȚIA COMPARATIVĂ A DIFICULTĂȚII PUZZLE-URILOR TP ȘI CP.....  | 56  |
| FIG. 15 PROTOCOLUL SSL HANDSHAKE.....  | 62  |
| FIG. 16 PROTOCOLUL SSL HANDSHAKE CU DISTRIBUȚIE EGALĂ DE EFORT .....   | 63  |
| FIG. 17 PROTOCOLUL SSL HANDSHAKE CU DISTRIBUȚIE ADAPTIVĂ DE EFORT.....   | 66  |
| FIG. 18 SCHEMA PROTOTIPULUI THRESHOLD PUZZLES .....  | 66  |
| FIG. 19 COMPARAȚIE A REZULTATELOR SIMULĂRIILOR PE PROTOCOLUL SSL.....  | 69  |
| FIG. 20 ATAC DoS ASUPRA SISTEMELOR SSO (ÎN PARTICULAR IMPLEMENTAREA LIBERTY) .....                             | 70  |
| FIG. 21 DETECTAREA UNUI POSIBIL ATAC ASUPRA SISTEMELOR SSO FOLOSIND ORDINEA DE EXECUȚIE<br>A PROTOCOLULUI..... | 71  |
| FIG. 22 MODUL DE ESTIMARE A NIVELULUI DE RISC SSO-SENSE.....   | 73  |
| FIG. 23 FOLOSIREA DETECȚIEI EURISTICE LA NIVELUL VECTORULUI DE AUTENTIFICARE .....                             | 76  |
| FIG. 24 MESAJUL CHANNEL REQUEST .....  | 92  |
| FIG. 25 MESAJUL CHANNEL REQUIRED .....   | 92  |
| FIG. 26 MESAJUL CHANNEL ACTIVE .....   | 93  |
| FIG. 27 MESAJUL CHANNEL ACTIVE ACKNOWLEDGED .....  | 93  |
| FIG. 28 MESSAGE IMMEDIATE ASSIGNMENT .....   | 94  |
| FIG. 29 PROCESUL DE ALOCARE A CANALULUI ÎN GSM .....   | 94  |
| FIG. 30 ATACURI DENIAL OF SERVICE (DoS) ÎN REȚELE GSM .....  | 95  |
| FIG. 31 PROCES DE ASIGNARE A CANALULUI ÎN GSM REZISTENT LA ATACURI DoS.....                                    | 97  |
| FIG. 32 MESAJUL PREAUTHENTICATION BEACON .....   | 98  |
| FIG. 33 CALCULAREA RĂSPUNSULUI DE PREAUTENTIFICARE .....   | 99  |
| FIG. 34 MESAJUL CHANNEL REQUEST (EXTENDED).....  | 99  |
| FIG. 35 ARHITECTURA SISTEMULUI SCALABIL DE DISTRIBUȚIE A CONȚINUTULUI.....                                     | 104 |
| FIG. 36 RETRANSMISIA CONȚINUTULUI PRIN CLIENȚI PROXY .....   | 110 |
| FIG. 37 COMPARAȚIE BROADCAST – UNICAST PENTRU UN CLIENȚ .....  | 133 |
| FIG. 38 COMPARAȚIE BROADCAST – UNICAST PENTRU 5 CLIENȚI .....  | 134 |
| FIG. 39 COMPARAȚIE BROADCAST – UNICAST PENTRU 10 CLIENȚI .....   | 134 |
| FIG. 40 COMPARAȚIE BROADCAST – UNICAST PENTRU 50 DE CLIENȚI.....   | 134 |
| FIG. 41 COMPARAȚIE BROADCAST – UNICAST PENTRU 100 DE CLIENȚI .....   | 135 |



## 1. Introducere

Societatea modernă a identificat o nouă necesitate vitală a sa: comunicația. Aceasta reprezintă forța care animează comunitățile de indivizi sau chiar indivizii în sine, forță care stă chiar la baza progresului umanitar. Felul în care omul a comunicat de-a lungul timpului s-a schimbat și continuă să se schimbe pe măsură ce societatea evoluează. Dacă la început mijloacele de comunicare erau primitive, anevoioase, cu latență mare și eficacitate generală redusă, astăzi schimbul de informație se produce cvasi-instantaneu pe tot cuprinsul globului. Comunicația este factorul determinant într-o multitudine de ramuri ale activității. Lipsa de coordonare pe un șantier, sincope în alimentarea cu subansamble a unei linii de producție, imposibilitatea alertării asupra unui eveniment produs sau iminent care pune vieți în pericol, reprezintă tot atâtea situații în care lipsa de comunicare duce la perturbarea ordinii firești sau mai rău, duce la pagube ce altfel ar putea fi prevenite și evitate.

Scopul acțiunii de comunicare este de a transporta informația (cunoștințele) între indivizii care compun societatea pe diverse căi și modalități, într-o manieră care să asigure integritatea și eventual confidențialitatea comunicării. Comunicarea implică transmisia de informație între entități situate la diverse niveluri, iar eterogenitatea acestora (oameni – sisteme de calcul – linii de comunicație) implică ierarhizarea infrastructurii. Fiecare nivel al infrastructurii este capabil să înțeleagă doar acea informație care îi parvine într-o formă codificată potrivit cu nevoile sale. Spre exemplu, omul percepe ușor informație pe cale audio-vizuală, deci este nevoie ca nivelurile ierarhic inferioare să îi prezinte informația în această formă. Acest lucru se repetă până la nivelurile aflate la baza infrastructurii de comunicație.

Lipsa conjuncturală a comunicării face ca activitatea unor părți din societate să fie perturbată, de la simple inconveniente legate de imposibilitatea efectuării unui apel telefonic până la consecințe grave ca pierderi de vieți omenești și pagube materiale (luând ca exemplu nefuncționarea unui sistem de avertizare la tsunami). Pentru ca situațiile extreme să fie reduse cât mai mult posibil, sistemul de comunicație trebuie să fie de încredere atât din punctul de vedere al confidențialității cât mai ales din punctul de vedere al disponibilității și stabilității. Un sistem de comunicație care prezintă dese întreruperi în funcționare sau variații largi ale calității serviciului va slăbi încrederea consumatorului, lucru ce va duce în cele din urmă la abandonarea utilizării lui, în detrimentul utilității sau confortului social.

### 1.1 Formularea problemei

Să luăm o formă de comunicare între indivizi, spre exemplu simplul act al vorbirii. Situațiile în care vorbirea este împiedicată de o cauză oarecare sau

perturbată din cauza multitudinii de voci din apropiere sunt situații excepționale cărora indivizii societății le găsesc mijloacele necesare pentru evitare sau corectare. În situațiile în care comunicația este afectată, individul identifică cauzele și ia măsurile de corecție necesare, spre exemplu se deplasează într-o direcție care să-i permită o comunicare bună cu interlocutorul. Se observă că părțile implicate în comunicație participă activ la îmbunătățirea comunicației prin acțiunile pe care le întreprind.

Cum societatea virtuală este o oglindire cvasi-fidelă a societății reale [75], situația este similară în cazul comunicației electronice: imposibilitatea comunicării a două entități poate fi provocată de un atac de interzicere a accesului (Denial of Service – DoS), iar comunicarea cu o rată redusă poate fi dată de lipsa scalabilității sistemului de comunicație. Pentru ameliorarea acestor stări de fapt, sistemele implicate în comunicație trebuie să dispună de metode și protocoale de acțiune pentru detectarea, eliminarea și evitarea cauzelor care perturbă comunicația. Putem distinge trei parametri ai sistemelor de comunicație care definesc și dau o măsură calității comunicației:

- Disponibilitatea infrastructurii de comunicație – posibilitatea ca procesul de comunicație să aibă loc
- Scalabilitatea sistemelor de comunicație – capacitatea de a suporta volume crescute de trafic/încărcare fără degradarea severă a indicatorilor de performanță
- Securitatea sistemelor de comunicație – capacitatea de a funcționa în prezența unor factori perturbatori

**Disponibilitatea** unui sistem de comunicație reprezintă capacitatea acestuia de a fi gata a efectua o transmisie într-un timp rezonabil. Lipsa disponibilității se poate datora fie întreruperii fizice a căii de comunicație (fire telefonice, cablu de rețea, emițător radio) sau se poate datora unui atac de tip Denial of Service (DoS) care împiedică transferul de date pe toată perioada sa. În prezent, rezistența la astfel de atacuri nu este un obiectiv de proiectare pentru majoritatea sistemelor de comunicație.

**Scalabilitatea** unui sistem de comunicații este capacitatea acestuia de a se adapta la diverse scenarii de încărcare. În cazul creșterii valorilor traficului, degradarea calității serviciului trebuie să se facă gradual și proporțional, evitându-se astfel căderi rapide ale serviciului care ar putea fi exploatate ulterior de un atacator. În general protocoalele de distribuție a conținutului deservesc clienții unul câte unul, lucru ce reprezintă o gâtuire a performanței care s-ar putea obține prin abordarea paralelă a distribuției.

**Securitatea** sistemului de comunicații reprezintă capacitatea sistemului de a funcționa în condiții normale sau apropiate de normal sub acțiunea unor factori externi perturbatori. În mod tradițional când vorbim de securitate ne gândim la

protecția și confidențialitatea datelor transmise de sistem, iar literatura de specialitate abundă de protocoale și procedee de securizare a informațiilor aflate în tranzit prin diverse medii. Disponibilitatea și scalabilitatea sunt factori care afectează în mod direct nivelul de securitate al unui sistem de comunicații, de aceea considerăm că o abordare corectă a îmbunătățirii securității presupune și o creștere a celor doi factori.

## 1.2 Scopul

Scopul acestei lucrări de cercetare este de a prezenta soluții pentru ameliorarea disponibilității, scalabilității și a securității sistemelor de comunicație.

Rețelele GSM, ca parte a unui sistem larg de comunicații și datorită răspândirii și valorii ridicate de piață de care se bucură, ar putea constitui puncte vulnerabile în infrastructură. Atacurile asupra rețelelor GSM pot cauza pagube importante, de aceea acțiunile subversive sunt tentante pentru atacatori. Rațiunile atacurilor asupra rețelelor GSM par să nu difere de cele îndreptate împotriva rețelelor de calculatoare cum ar fi pierderi în venituri, impact social, etc. Securitatea în GSM se referă doar la partea radio și este proiectată să împiedice doar atacatorii mai puțin experimentați. Deoarece resursele radio sunt limitate, sistemul GSM are facilități puternice de contabilizare a resurselor dar în cazul unor terminale neconforme cu standardul, resursele se pot aloca abuziv, neputându-se face nimic pentru constrângerea și recuperarea acestora. Lucrarea de față ridică problema vulnerabilității rețelelor GSM la atacuri DoS și face propuneri de ameliorare a rezistenței.

În general, protocoalele de autentificare se bazează pe operații criptografice asimetrice și pentru că acestea sunt operații complexe și scumpe din punctul de vedere al resurselor, avem de-a face cu o vulnerabilitate la atacuri de tip Denial of Service (DoS) în cele mai multe cazuri. Dezechilibrul între resursele alocate de client pentru execuția protocolului și cele alocate de server permite clientului să angajeze în mod repetat resurse importante ale serverului. Acest lucru se traduce prin posibilitatea încărcării deliberate de către atacator a serverului de autentificare cu cereri false pentru care se alocă resurse de calcul, în detrimentul cererilor legitime care sunt deservite într-un timp mai îndelungat sau în cazul extrem, deloc. Lucrarea de față prezintă o posibilă soluție la acest dezechilibru, și anume creșterea artificială a costului de execuție al protocolului pentru client, proporțional cu efortul cerut serverului. Astfel, în cazul autentificărilor sporadice încărcarea pe client este neglijabilă, dar dacă clientul emite cereri repetate către server cu scopul de a-l supraîncărca, încărcarea asupra sa crește proporțional.

Multitudinea de dispozitive electronice portabile care ne înconjoară a atras implicit și nevoia acestora de comunicare. De la simple mesaje scrise sau e-mail, aceste dispozitive sunt acum capabile de a reda conținut digital audio și video pe

care îl obțin fie de la o sursă locală cum ar fi un card de memorie fie de la o sursă online de tip streaming sau download. Consumatorii de entertainment digital mobil sunt în special tinerii, și odată cu lansarea unui album de muzică nou sau al unui film, utilizatorii vor provoca creșteri momentane ale valorilor traficului care în lipsa unei infrastructuri suficient dimensionate pot constata degradări ale serviciului consumat, cum ar fi timpi de așteptare mare, viteze de transfer scăzute, etc. Dimensionarea corespunzătoare a infrastructurii trebuie astfel dublată de eficientizarea transportului de informații. Pe piață există deja un număr de sisteme de distribuție digitală a conținutului, care se concentrează pe argumentele tehnice privitoare la mecanismul de redare a conținutului pe dispozitivele autorizate. Scalabilitatea unor astfel de sisteme este redusă și nu poate face față unor vârfuri de trafic și în plus nici nu se are în vedere colaborarea între părțile care partajează drepturile asupra unui conținut digital (autori). Lucrarea de față prezintă o arhitectură de distribuție digitală a conținutului care îmbunătățește într-o manieră semnificativă scalabilitatea și maniera de cooperare între deținătorii de drepturi de autor.

### 1.3 Organizarea tezei

**Capitolul 2** este o trecere în revistă a problematicei securității, disponibilității și scalabilității sistemelor de comunicații, cu accent pe evidențierea neajunsurilor ce survin datorită deficiențelor de design. Astfel, sistemele de autentificare, sistemele Single Sign-On și rețelele GSM sunt vulnerabile la atacuri DoS, fiindu-le astfel afectată disponibilitatea. Vulnerabilitatea survine în urma alocării de resurse valoroase (putere de calcul, frecvențe de comunicație, etc.) în condițiile lipsei unui control strict. Deficiențele în design nu au ca urmare doar scăderea disponibilității, ci în cazul sistemelor de distribuție a conținutului pot afecta scalabilitatea. Datorită abordării secvențiale a procesului, acest tip de sistemele întâmpină dificultăți în manipularea unor cantități crescânde de date.

**Capitolul 3** prezintă contribuțiile la creșterea disponibilității și scalabilității sistemelor de autentificare. Ideea principală este de a egala efortul depus de client cu cel depus de server pentru o operație de autentificare, lucru ce se realizează printr-un *client puzzle* (în esență inversarea parțială a unei dispersii). Pentru ca această soluție să fie fezabilă în cazul clienților cu puteri de calcul care variază în limite largi (de la un simplu PDA sau SmartPhone până la rețele distribuite de calculatoare) s-au introdus conceptele de *threshold puzzle* respectiv *adaptive threshold puzzle*. Sistemele de autentificare de tip Single Sign-On prezintă particularități de comportament în cazul unui atac, exploatate cu ajutorul motorului de detecție *SSO-SENSE*. Capitolul prezintă fundamentarea matematică a acestor concepte precum și aspecte practice de implementare și măsurători de performanță.

**Capitolul 4** prezintă principalele vulnerabilități ale rețelelor GSM, clasificându-le după gravitate după o metrică folosită în software, ceea ce reprezintă

o premieră. Rezultatul deloc surprinzător este că cea mai gravă amenințare asupra unei rețele GSM este atacul Denial of Service (DoS), vulnerabilitate similară sistemelor de autentificare dar pe partea radio.

**Capitolul 5** propune o arhitectură de distribuție a conținutului menită să amelioreze problemele existente în prezent. Arhitectura prezentată abordează distribuția prin prisma difuzării multiple (broadcast), utilizând un procedeu criptografic care să permită decodificarea conținutului de către entități selectate. Deservirea mai multor clienți simultan și posibilitatea ca distribuția să fie preluată de entități de tip proxy duce la creșterea scalabilității generale a sistemului.

**Capitolul 6** prezintă concluziile tezei și prezintă perspectivele de cercetare în domeniu.

Publicațiile personale sunt listate în anexa specială de la sfârșitul lucrării.





## 2. Securitate, disponibilitate și scalabilitate în sisteme de comunicații

Pentru a fi utile scopului în care au fost proiectate și construite, sistemele de comunicații trebuie să fie la dispoziția utilizatorilor la momente oportune și în condiții bine stabilite, în plus trebuie să aibă un comportament previzibil în orice fel de condiții de încărcare.

În comunicații, securitatea este o condiție care rezultă din stabilirea și menținerea unor măsuri de protecție care mențin starea de inviolabilitate față de acte ostile sau influențe de acest fel, cum ar fi: impersonarea participanților, efectuarea de operații privilegiate, capturarea traficului în vederea analizei ulterioare, afectarea disponibilității sistemului prin acțiuni subversive, etc. Autentificarea, autorizarea și criptarea sunt măsuri pentru controlul participanților și sunt mecanisme interne sistemului de comunicație ca parte integrantă a acestuia.

Factorii care influențează sistemele de comunicație sunt disponibilitatea și scalabilitatea. Disponibilitatea este gradul în care un sistem sau subsistem este operabil și într-o stare stabilă, la momentul începerii unei proceduri, când procedura este declanșată la un moment aleatoriu în timp. Scalabilitatea este factorul care influențează în mod direct capacitatea de adaptare la încărcare. Spunem că un sistem de comunicație este scalabil atunci când poate deservi un trafic crescător într-o manieră în care degradarea performanței se face liniar sau subliniar.

În cadrul acestui capitol se face o introducere în problema securității, disponibilității și scalabilității sistemelor de comunicație cu accent pe probleme actuale.

### 2.1 Caracteristici ale sistemelor de comunicații

#### 2.1.1 Securitatea

Conceptul de securitate în sistemele de comunicații se referă la o condiție care rezultă din stabilirea și menținerea unor măsuri de protecție care mențin starea de inviolabilitate față de acte ostile sau influențe de acest fel. Starea de securitate se obține prin stabilirea și verificarea identității participanților de comunicație, autorizarea participanților și protecția datelor vehiculate.

Autentificarea este actul prin care se verifică identitatea pretinsă de un utilizator. O persoană care dorește să efectueze o operație bancară trebuie să prezinte lucrătorului de la ghișeu o formă de identificare, uzual un act cu fotografie. Lucrătorul compară imaginea cu fizionomia persoanei, iar în caz de potrivire procedează la îndeplinirea cererii. În caz contrar, cererea este refuzată.

Autorizarea se referă la pasul ulterior autentificării, când odată stabilită identitatea, utilizatorul este verificat dacă are voie (are drepturi) să efectueze operația solicitată. Autorizarea se face pe baza unor liste de control acces.

În cele din urmă, operația (în cazul nostru comunicația) poate avea loc după un schimb de chei între participanți din care să rezulte o cheie temporară de sesiune cu care comunicația va fi protejată.

### 2.1.2 Disponibilitatea

Gradul în care un sistem sau subsistem este operabil și într-o stare stabilă, la momentul începerii unei proceduri, când procedura este declanșată la un moment aleatoriu în timp, se numește disponibilitate.

Disponibilitatea unui sistem de comunicații se poate defini ca:

$$A = \frac{T_{UpTime}}{T_{UpTime} + T_{DownTime}}$$

Unde  $T_{UpTime}$  și  $T_{DownTime}$  sunt duratele de timp așteptate pentru funcționarea, respectiv nefuncționarea sistemului.

Nefuncționarea sistemului de comunicații poate următoarele cauze:

- Nefuncționarea dispozitivelor hardware implicate în comunicare
- Nefuncționarea căii de comunicare
- Imposibilitatea realizării autentificării/autorizării/criptării, fapt ce duce la refuzul comunicației

Deși primele două cauze pot fi provocate și deliberat, considerăm că defectele la nivelul hardware pot surveni în orice moment și din cauze altele decât rezultatul unui atac. În ceea ce privește imposibilitatea autentificării, a autorizării sau a criptării într-un sistem de comunicație, aceasta poate surveni în urma unui atac de tip Denial of Service (DoS).

Atacurile de tip DoS sunt principala cauză de indisponibilitate a sistemelor și de aceea este nevoie de măsuri de apărare cât mai eficiente.

### 2.1.3 Scalabilitatea

Scalabilitatea este o noțiune în general dificil de definit. În general spunem că scalabilitatea este factorul care influențează în mod direct capacitatea de adaptare la încărcare. Un sistem de comunicație scalabil deservește traficul crescător într-o manieră în care degradarea performanței se face liniar sau subliniar. Un sistem scalabil poate să-și mărească corespunzător capacitatea când beneficiază de o îmbunătățire a infrastructurii hardware pe care rulează. Similar, un sistem

software este scalabil dacă poate suferi îmbunătățiri de structură care să-i mărească capacitatea pe aceeași platformă hardware.

## 2.2 Securitatea și disponibilitatea sistemelor de autentificare

Stabilirea identității participanților la o comunicație poartă numele de *autentificare* și are ca rezultat autenticitatea, însemnând că partea care verifică (*verificatorul*) poate fi sigur că partea verificată (*verificatul*) este cel care pretinde că este. Uzual, tehnicile de autentificare se împart în trei categorii fundamentale:

- *Autentificare prin cunoștințe* (ceva ce utilizatorul știe: coduri PIN, coduri de tranzacție, parole)
- *Autentificare prin posesie* (ceva ce utilizatorul are: chei, carduri de identificare sau alt fel de dispozitive fizice)
- *Autentificare prin proprietăți* (identificarea biometrică a utilizatorului cum ar fi identificarea feței, imagini ale retinei, șabloane vocale, amprente)

Având în vedere multitudinea de site-uri Internet, autentificarea preponderentă astăzi este cea prin cunoștințe, urmată la mare distanță de cea prin posesie.

### 2.2.1 Autentificarea bazată pe parole

În cele mai multe sisteme distribuite și rețele de calculatoare, protecția resurselor se realizează prin login direct folosind parole, cu transmiterea în clar a acestora. Această autentificare are mai multe inconveniente, din care amintim doar câteva:

- Utilizatorii tind să selecteze parole neuniform distribuite. Această problemă este binecunoscută și nu este neapărat legată de rețele de calculatoare și sisteme distribuite. [35][50]
- Nu este convenabil pentru un utilizator care are mai multe conturi pe host-uri diferite să își amintească parola pentru fiecare, și de asemenea să o introducă la fiecare schimbare a host-ului. În schimb, utilizatorul va alege să fie recunoscut de rețea ca întreg și nu de host-urile individuale.
- Transmisia parolei este expusă la captură pasivă.

În principal datorită acestui ultim punct, autentificarea bazată pe parolă nu este potrivită în rețele de calculatoare și sisteme distribuite. Parolele trimise prin rețea sunt foarte ușor compromise și pot fi folosite ulterior pentru impersonarea utilizatorului.

În unele situații, autentificarea prin parolă este chiar deranjantă. Spre exemplu, la începutul telefoniei mobile în Statele Unite ale Americii, telefoanele mobile foloseau ca parolă internă la efectuarea unui apel chiar numărul telefonului

respectiv pentru ca centrala să poată factura fiecare apel în mod corect. Acest mod simplist de abordare a problemei a dus rapid la atacuri prin impersonare, rezultatul net fiind posibilitatea ca un atacator să efectueze convorbiri în contul altei persoane.

### 2.2.2 Autentificarea bazată pe adresă

O alternativă – nu neapărat mai sigură – la autentificarea prin parolă este autentificarea prin adresă. Aceasta presupune că identitatea unei surse se poate deduce din adresa acesteia conținută în pachete. Ideea de bază este că fiecare host memorează identitatea celorlalte host-uri care au acces la resursele sale. În sistemul de operare UNIX, fiecare host are un fișier numit */etc/hosts.equiv*. Utilizatorii cu același cont pe ambele sisteme pot folosi utilitarele „r” fără a specifica vreo parolă.

Ideea de host-uri credibile nu este o soluție la problema autentificării în rețele de calculatoare. De fapt, acest tip de autentificare chiar pune probleme mai mari din punct de vedere al securității. Dacă un atacator reușește să intre pe contul unui utilizator dintr-un sistem, securitatea este compromisă pe toate sistemele care au încredere în acel sistem. În plus, administratorul de sistem nu poate da drepturi preferențiale unor utilizatori.

În funcție de mediul concret, autentificarea prin adresă este chiar mai puțin sigură decât cea bazată pe parolă. Avantajul îl constituie însă comoditatea în folosire și de aceea unele sisteme au ales să o implementeze.

### 2.2.3 Autentificarea criptografică

Ideea din spatele autentificării criptografice este că A își dovedește identitatea către B prin efectuarea unei operații criptografice asupra unei entități cunoscute de ambii participanți sau oferită de B. Operația criptografică efectuată de A se bazează pe o cheie criptografică. Aceasta poate fi fie o cheie secretă sau o cheie privată dintr-un sistem cu chei asimetrice.

În general, autentificarea criptografică este mai sigură decât autentificarea bazată pe parolă sau pe adresă. Tehnicile bazate pe dovezi *zero-knowledge* pot oferi mecanisme de autentificare puternice [101][70], însă acestea necesită calcule matematice destul de complexe și prezintă mai multe facilități atractive pentru autentificare. Caracteristica definitorie este că părțile care se autentifică dovedesc că știu secretul fără a transfera efectiv informația către verificator.

În ciuda aparentei simplități, proiectarea sistemelor reale este foarte dificilă. O serie de protocoale publicate au prezentat erori de securitate substanțiale sau subtile. În timpul ultimei decade, eforturile de cercetare s-au concentrat în crearea de utilitare pentru dezvoltarea protocoalelor de autentificare și distribuție a cheilor cu o anumită asigurare formală a securității. Realizările cele mai notabile sunt logica

BAN [19] și logica GNY [41]. În loc de a produce protocoale specifice, aceste metodologii se utilizează pentru a verifica un set de afirmații presupus adevărate.

O implementare cunoscută a autentificării criptografice este protocolul SSL, folosit în toate browserele populare cum ar fi Internet Explorer și Firefox. Paragraful următor face o descriere a acestui protocol și a proprietăților acestuia.

#### 2.2.4 Protocolul SSL

Protocolul SSL este un protocol de securitate care oferă comunicare secretă prin Internet. Scopul principal al acestuia este de a oferi confidențialitate și încredere între două aplicații care comunică. Protocolul constă din două straturi: La cel mai de jos nivel, plasat deasupra unui protocol sigur de comunicație (spre exemplu TCP) se află protocolul *SSL Record*. Acesta este folosit pentru încapsularea protocoalelor de nivel mai înalt, cum ar fi protocolul *SSL Handshake* care permite serverului și clientului să negocieze un algoritm de criptare și cheile criptografice corespunzătoare înainte ca aplicațiile să schimbe vreun mesaj. Un nivel superior al protocolului ar putea sta deasupra acestuia în mod transparent.

Protocolul SSL oferă securitatea conexiunii, având trei proprietăți:

- Conexiunea este privată. Criptarea se utilizează după ce are loc schimbul inițial pentru definirea cheii secrete. Criptografia simetrică se utilizează pentru codificarea datelor (DES, RC4, etc.)
- Identitatea părților se autentifică utilizând criptografie asimetrică (RSA, DSS, etc.)
- Conexiunea este de încredere. Transportul mesajului include verificarea acestuia cu un MAC parametrizat. Pentru calcularea MAC-ului se folosesc funcțiile de dispersie SHA, MD5, etc.

Scopurile protocolului SSL, în ordinea priorităților sunt:

1. **Securitatea criptografică:** SSL ar trebui să se utilizeze pentru conexiune sigură între două părți.
2. **Interoperabilitate:** Programatori independenți ar trebui să fie capabili de a dezvolta aplicații SSL care să funcționeze cu succes fără ca aceștia să aibă cunoștință de codul scris de altcineva.
3. **Extensibilitatea:** SSL încearcă să furnizeze un cadru în care să se integreze metode noi de criptare (simetrică sau asimetrică), acest lucru contribuind la evitarea creării unui protocol nou (riscând implicit să se introducă noi slăbiciuni) și evitarea scrierii unei biblioteci de securitate noi.
4. **Eficiența relativă:** Operațiile criptografice sunt scumpe din punctul de vedere al timpului, în special criptografia cu chei publice. Din acest motiv, SSL a înglobat un mecanism de caching al sesiunii pentru a reduce numărul de conexiuni ce trebuie stabilite în totalitate.

SSL este un protocol stratificat. Pe fiecare strat, mesajele pot conține câmpuri pentru lungime, descriere și conținut. SSL primește mesajele de transmis,

fragmentează informația în blocuri prelucrabile, opțional comprimă datele, aplică un MAC, codifică și în final transmite rezultatul. Datele recepționate sunt decriptate, verificate, decomprimate și reasamblate, apoi oferite protocolului superior ierarhic.

O sesiune SSL se poate afla într-una dintre stările predefinite ale protocolului. Este responsabilitatea protocolului SSL Handshake să coordoneze stările clientului și ale serverului, permițând ca sistemele să funcționeze în mod consistent, în ciuda faptului că stările nu sunt exact paralele. În mod logic, starea este reprezentată de două ori, o dată ca starea curentă în operare și (în timpul protocolului de handshake) ca starea în așteptare. În plus, se mențin stări separate ale scrierii și citirii. Când clientul sau serverul recepționează un mesaj de schimb a specificației cifrului, se copiază starea în așteptare în starea curentă de citire. Când negocierea este completă, serverul și clientul interschimbă mesajele de specificare a cifrului și comunică prin noul cifru asupra căruia s-a convenit.

O sesiune SSL poate include mai multe conexiuni sigure, în plus participanții pot avea mai multe sesiuni simultane. Starea sesiunii poate cuprinde următoarele elemente:

- **Identificator de sesiune:** O secvență arbitrară de octeți aleasă de server pentru a identifica o sesiune activă sau reluabilă.
- **Certificat al egalului:** Certificat X.509 al părții cu care se comunică. Acest element poate să fie nul.
- **Metoda de compresie:** Numele algoritmului utilizat pentru a comprima datele înainte de codificare.
- **Specificația cifrului:** Specifică algoritmul de codificare a datelor (cum ar fi nimic, DES, AES, etc.) și un algoritm MAC (cum ar fi MD5 sau SHA). De asemenea definește atributele criptografice cum ar fi dimensiunea tabelii de dispersie.
- **Secretul master:** O valoare secretă de 48 de octeți cunoscută de client și de server.
- **Capacitatea de reluare:** Un indicator care arată dacă o sesiune se poate utiliza pentru a iniția noi conexiuni.

Starea conexiunii poate include următoarele elemente:

- **Valori aleatoare pentru client și server:** Secvențe de octeți alese de client și de server pentru fiecare conexiune.
- **Secret MAC pentru scrierea pe server:** Secretul utilizat în operații MAC pe datele scrise de server.
- **Secret MAC pentru scrierea pe client:** Secretul utilizat în operații MAC pe datele scrise de client.
- **Cheia de scriere a clientului:** Cheia pentru datele codificate de server și decodificate de client.
- **Cheia de scriere a serverului:** Cheia pentru datele codificate de client și decodificate de server.
- **Vectori de inițializare:** Când se utilizează un cifru bloc în modul CBC, se păstrează un vector de inițializare pentru fiecare cheie. Acest câmp este inițializat de protocolul SSL Handshake.

- **Numere de secvență:** Fiecare participant menține numere de secvență pentru mesajele transmise și recepționate pentru fiecare conexiune. Când un participant trimite sau recepționează un mesaj de modificare a specificațiilor cifrului, acest număr se pune pe zero. Numere de secvență sunt de tipul *uint64* și nu trebuie să depășească valoarea  $2^{64} - 1$  [37].

Sistemul este la fel de puternic ca cel mai slab algoritm de autentificare și de distribuție a cheilor. Astfel trebuie să se utilizeze doar funcții criptografice sigure, verificate. Chei publice scurte, chei simetrice de 40 de biți sau mai puțin și servere anonime trebuie utilizate cu precauție. Implementările și utilizatorii trebuie să aleagă cu grijă autoritățile de certificare acceptabile, o autoritate mincinoasă putând provoca pagube imense.

#### 2.2.5 Deficiențe ale protocoalelor de autentificare actuale

Deși autentificarea criptografică oferă siguranță participanților la comunicație, aceasta este susceptibilă la atacuri de tip DoS. În cele ce urmează se va face o scurtă prezentare a acestui tip de atac și a cauzelor sale.

##### 2.2.5.1 Atacuri DoS

Un atac DoS poate lua una din cele două forme posibile. Un atacator poate cauza netransmiterea de către rețea a mesajelor pe care ar trebui să le transmită în mod normal clienților săi. De cealaltă parte se află rețelele care trimit mesaje pe care nu ar trebui să le trimită. De departe cel mai cunoscut atac DOS este cauzarea de trafic fals (inundarea rețelei) în direcția un server particular, lucru care în final va duce la împiedicarea clienților legitimi să obțină serviciul pe care îl cer de la acel server.

Comunitatea Internet cunoaște o serie întreagă de atacuri DOS, cum ar fi atacurile prin inundare (flooding) și atacurile prin pachete modificate.

O cauză a atacurilor este că dialogul inițial între părți are loc înaintea unei minime preautentificări. Serverul este incapabil de a distinge traficul legitim de cel fals, fapt ce are toate șansele să rămână așa. Impunerea necesității de a autentifica toate cererile ar însemna un atac DOS în sine, din moment ce serverul ar fi ocupat cu verificarea semnăturilor digitale, indiferent dacă acestea sunt sau nu valide. Această nouă cale de atac ar fi la fel de periculoasă ca și simpla umplere a tabelii TCP, în cazul unui atac de tip TCP SYN.

O altă cauză a atacurilor DoS este lipsa contabilizării resurselor. Spatscheck și Peterson [80] consideră că sunt trei ingrediente cheie pentru protejarea de atacuri DOS:

[1] *contabilizarea* tuturor resurselor consumate de un client

- [2] *deteția* momentului când resursele alocate unui client depășesc un prag prestabilit
- [3] *constrângerea* – capacitatea de a revendica resursele blocate după detectarea unui atac prin alocarea de resurse minime unui atacator, lucru ce înseamnă automat evitarea unui atac DOS ulterior

În perioada când Internetul însuși era proiectat, contabilizarea resurselor era scopul cu prioritatea cea mai mică, lucru ce ne afectează astăzi cel mai mult. În contrast cu rețelele de telefonie omniprezente unde folosirea resurselor este atent controlată, proiectanții rețelei Internet nu au părut să acorde importanță acestui aspect. Astfel, serverele alocă aceeași putere de calcul tuturor cererilor care sosesc la un moment dat ceea ce împiedică o degradare elegantă a performanței în cazul unui atac sau în cazul unei încărcări excesive.

Scenariul anterior este oarecum similar cu mecanismul rudimentar de procesare a pachetelor de intrare datorită arhitecturii bazate pe întreruperi a subsistemului de rețea. Toate sistemele de operare implementează acest tip de arhitectură care s-a dovedit neadecvat în condiții de încărcare mare. Pachetele de la intrare sunt procesate cu prioritatea maximă pentru ca apoi să fie distruse pentru că nu există nicio aplicație care să le deservească. Această situație se numește *receiver livelock*. Mai mult, chiar dacă există o aplicație care să deservească aceste pachete, prioritatea procesului nu este luată în calcul. Astfel, aplicațiile cu prioritate mică primesc aceeași cantitate de trafic ca cele cu prioritate mai mare. În lucrarea lor, Druschel și Banga propun o arhitectură cu procesare întârziată care se bazează pe demultiplexarea timpurie a pachetelor și procesarea lor în funcție de prioritatea destinatarului. Ei susțin că noua arhitectură ar îmbunătăți stabilitatea, justetea și capacitatea sistemelor în condiții de încărcare mare în timp ce performanța nu ar avea de suferit în condiții normale.

Crosby și Wallach [27] au descris o nouă modalitate de atac bazată pe design-ul intrinsec al protocoalelor. Noua clasă de atacuri de bandă îngustă exploatează deficiențele structurilor de date în diferite aplicații. Structurile de date folosite în mod uzual au un timp de rulare în cazul mediu mult mai bun decât în cazul cel mai defavorabil. Spre exemplu, atât tabelele de dispersie cât și arborii binari degenerază în liste înlănțuite atunci când la intrare se prezintă date alese corespunzător. Folosind banda tipică a unui modem, autorii citați au adus un server Bro în pragul colapsului; în șase minute de la începutul atacului, serverul ignora 71% din trafic și consuma întreaga putere de calcul disponibilă.

#### 2.2.5.2 Importanța luptei împotriva atacurilor DoS

Având în vedere tendința globală a piețelor de a se muta on-line, atacurile DoS se dovedesc mult mai periculoase decât s-a prevăzut la început întrucât acestea pot bloca victimele pe durate lungi de timp. De la momentul la care a început atacul și până când acesta este detectat și eliminat, victima este paralizată și nu poate



răspunde la cererile legitime. Pentru site-urile comerciale mari aceasta se traduce prin pierderi financiare importante.

Deși atacurile DOS nu amenință integritatea și securitatea datelor în mod direct, nu există nici un motiv ca un altfel de atac să nu urmărească distrugerea și alterarea lor. Atacurile pot viza date critice, cauzând astfel mai multe daune decât atacul DoS în sine.

Acest fel de atacuri în lanț pot avea loc dacă protocoalele continuă dialogul cu atacatorul chiar și după detectarea unor anomalii în dialogul purtat între părți. Ideea de bază în spatele protocoalelor rezistente (numite protocoale *fail-stop* sau *fail-safe*) este ca schimbul de mesaje să înceteze cu orice client care nu urmează cursul firesc al protocolului.

## 2.3 Securitatea și disponibilitatea sistemelor Single Sign-On (SSO)

În prezent, utilizatorii care doresc să acceseze servicii on-line trebuie să folosească un număr de credențiale (perechi nume utilizator și parolă) pentru a accesa fiecare serviciu la care sunt înregistrați. Această fapt are neajunsuri din punctul de vedere al securității, deoarece utilizatorii se dovedesc incapabili a reține un număr mare de parole de calitate, în plus costurile apelurilor la help desk (pentru resetarea parolelor) sunt importante tocmai din acest motiv.

Un sistem SSO abordează problema într-o manieră diferită. Utilizatorul se autentifică sistemului SSO cu credențialele proprii, apoi orice acces ulterior la servicii este automat autentificat de către sistemul SSO, fără intervenția manuală a utilizatorului. În prezent există mai multe arhitecturi SSO care vor fi prezentate în continuare, fiecare cu avantajele și dezavantajele sale [67].

### 2.3.1 Tipuri de sisteme SSO

Sistemele SSO permit autentificarea automată a utilizatorilor către furnizorii de servicii. Autentificarea în sine implică identificare, deci un sistem SSO trebuie să aibă suport pentru managementul credențialelor utilizatorului. Întrucât acestea pot lua diverse forme, acestea vor fi referite prin sintagma „identități SSO”.

Se pot deosebi două tipuri de sisteme SSO. Primul dintre acestea, numit „pseudo-SSO”, funcționează ca un intermediar între utilizator și aplicațiile pe care acesta le accesează. Autentificarea inițială (numită *autentificare primară*) se face între utilizator și sistemul SSO. La fiecare acces la o aplicație care necesită credențialele utilizatorului, sistemul SSO intervine și efectuează pașii necesari pentru ca autentificarea să fie reușită. Pentru fiecare acces la aplicație se efectuează o autentificare într-o modalitate transparentă pentru utilizator. Aplicația poate să nu fie informată că are loc o autentificare comandată de către sistemul SSO.

Întrucât credențialele SSO sunt specifice fiecărui furnizor de servicii (aplicații software, etc.) relația identitate SSO/furnizor este  $n : 1$ , adică orice identitate dată corespunde unui singur furnizor.

Schema de principiu al unui sistem „pseudo-SSO” este prezentată în Fig. 1.

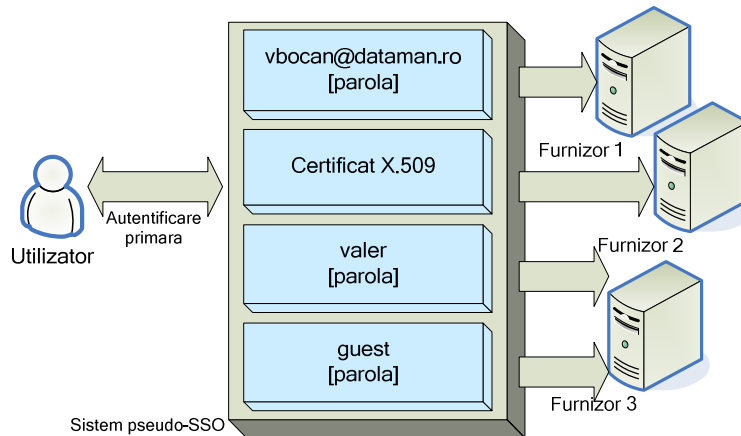


Fig. 1 Mecanismul de principiu al unui sistem „pseudo-SSO”

Un sistem „pseudo-SSO” este de fapt o aplicație care monitorizează comportamentul unor produse software care în mod tradițional ar cere autentificarea utilizatorului în diverse moduri, cum ar fi ferestre de login, prompt, argumente în linia de comandă, etc. Prin mecanisme specifice, aceste cereri de autentificare sunt interceptate și deservite în funcție de credențialele înregistrate pentru produsul în cauză, pentru utilizatorul autentificat curent. Din punct de vedere al mecanismelor de implementare, sistemele „pseudo-SSO” nu sunt altceva decât mecanisme de automatizare ale unor operații care în mod obișnuit ar fi efectuate de către utilizator. La fiecare acces la aplicație se face o autentificare în sine.

Dezavantajul acestui tip de sisteme SSO este manipularea în clar a credențialelor din necesitatea ca acestea să fie livrate tot în clar furnizorilor de servicii. Protejarea credențialelor în drumul lor de la sistemul SSO la aplicație este foarte dificilă, capturarea putându-se face prin diverse mijloace nu foarte avansate. Un alt dezavantaj este necesitatea actualizării scripturilor sau modulelor de interceptare ale ferestrelor de autentificare pentru fiecare produs monitorizat în parte. De regulă există un interval „mort” în care sistemul „pseudo-SSO” nu are cunoștințe despre noile versiuni ale produselor, timp în care clienții trebuie să aștepte. Pentru a-și păstra clienții, companiile care dezvoltă sisteme „pseudo-SSO” trebuie să fie în permanentă alertă, ceea ce duce la un consum crescut de resurse și implicit la un cost ridicat.

Avantajul acestui tip de sisteme SSO este că pentru implementare sunt necesare schimbări minime în infrastructura software, întrucât aplicațiile monitorizate nu trebuie să fie conștiente de noul sistem în funcțiune.

Cel de-al doilea tip de sisteme SSO este fundamental diferit de cel anterior și va fi numit în continuare „true-SSO”. În acest caz, autentificarea primară se realizează către o componentă numită *Authentication Service Provider (ASP)*. Uneori această componentă este numită *Identity Service Provider (ISP)*. Pentru a realiza SSO, ASP trebuie să aibă stabilită o relație cu fiecare furnizor de servicii (SP), relație care de regulă este stabilită printr-un contract. De asemenea trebuie să existe un canal securizat de comunicație între ASP și SP.

Odată ce autentificarea primară a avut loc, la cererea furnizorilor de servicii (SP), ASP-ul realizează autentificarea prin intermediul unui protocol dedicat, folosind așa-numitele *authentication assertions*. Acestea conțin identitatea utilizatorului și starea sa așa cum este cunoscută de către ASP și trebuie transmise în mod securizat, în funcție de specificul implementării. Spre deosebire de sistemele pseudo-SSO, relația identitate SSO / furnizor este  $n : m$ . Aceasta înseamnă că utilizatorul poate alege dintr-o plajă de identități pentru orice SP ales, dar aceeași identitate poate fi utilizată cu mai mulți SP, dacă utilizatorul dispune acest lucru. Astfel este posibilă atribuirea de roluri specifice identităților SSO (care sunt de fapt pseudonime, așa cum au fost definite în [68]).

Schema de principiu a unui sistem true-SSO este prezentată în Fig. 2.

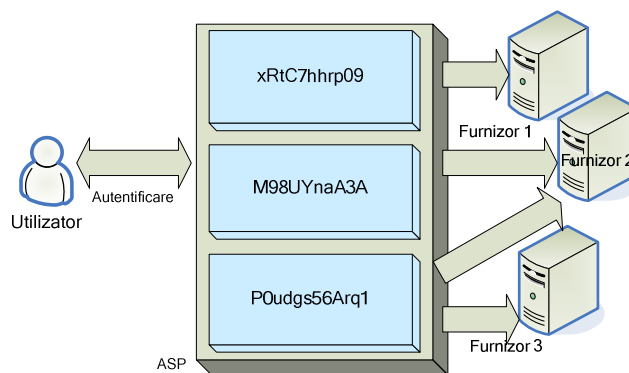


Fig. 2 Mecanismul de principiu al unui sistem „true-SSO”

ASP este un serviciu care deservește cereri provenite de la clienți care au cunoștință de faptul că un sistem SSO este în funcțiune. Dezavantajul acestui sistem este necesitatea modificării produselor software. Acestea trebuie (re)proiectate și implementate urmând specificațiile sistemului SSO în cauză, lucru care nu este întotdeauna ușor pentru aplicații *legacy*.

Avantajul acestei abordări este că autentificarea se face o singură dată, credențialele nu sunt stocate și transmise în clar, putându-se totodată implementa anonimizarea, adică imposibilitatea creării de conexiuni între identitățile separate ale aceluiași utilizator.

Traseul informației între părțile implicate este ilustrat în Fig. 3. În prima fază, utilizatorul efectuează autentificarea primară către ASP (1). Acest pas se efectuează o singură dată într-o sesiune de lucru. La un moment ulterior, utilizatorul dorește să acceseze un serviciu oferit de un furnizor oarecare (2). Furnizorul trebuie să verifice la rândul său identitatea clientului, iar pentru acest lucru va solicita informațiile de la ASP (3.1), care îi trimite un identificator ce reprezintă identitatea utilizatorului (3.2). În acest moment furnizorul de servicii poate trece la îndeplinirea inițială a cererii utilizatorului (4).

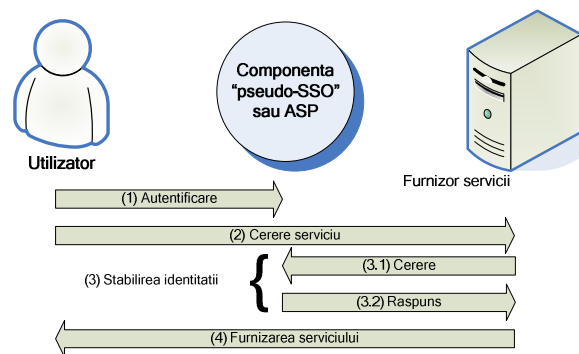


Fig. 3 Fluxul informațional într-un sistem SSO generic

Modulele „pseudo-SSO”/ASP se pot afla pe calculatorul client (cel al utilizatorului) sau se poate afla pe o altă mașină pe care o numim „SSO proxy”. Astfel, deosebim următoarele categorii de sisteme SSO [67]:

- Sisteme pseudo-SSO locale
- Sisteme pseudo-SSO bazate pe proxy
- Sisteme SSO locale
- Sisteme SSO bazate pe proxy

#### 2.3.1.1 Sisteme pseudo-SSO locale

Într-un sistem pseudo-SSO local, componenta pseudo-SSO se află pe sistemul clientului. La începutul sesiunii de lucru, utilizatorul se autentifică componentei care tipic constă dintr-o bază de date care conține identitățile utilizatorului pentru fiecare serviciu înregistrat în parte. O proprietate caracteristică este necesitatea de a păstra credențialele în clar, de unde rezultă necesitatea ca

utilizatorul să aibă încredere în sistemul pe care lucrează. De remarcat că această arhitectură nu se pretează la sisteme publice sau unde accesul nu este restricționat.

#### 2.3.1.2 Sisteme pseudo-SSO bazate pe proxy

Într-un sistem pseudo-SSO local bazat pe proxy, componenta pseudo-SSO se află pe un sistem separat de cel al clientului. Autentificarea primară se efectuează la fel ca în cazul anterior la începutul sesiunii de lucru (eventual și ulterior dacă proxy-ul dorește reautentificarea periodică a clientului). Identitățile pentru fiecare furnizor de servicii în parte sunt stocate pe proxy, într-o bază de date securizată, însă tot este necesară prezentarea în clar a acestora în cazul dialogului cu un furnizor de servicii. Cererile de autentificare pot fi redirectate de către client către proxy printr-un mecanism special destinat acestui scop, fie sunt pur și simplu interceptate fără ca programul în cauză să fie anunțat în vreun fel, astfel încât procesul să fie transparent pentru client. De remarcat că arhitectura se pretează la sistemele publice cu condiția ca proxy-ul să fie securizat corespunzător.

#### 2.3.1.3 Sisteme SSO locale

Un sistem SSO local are ca trăsătură principală furnizorul de identitate (ASP – authentication provider) plasat pe sistemul client. Acesta, cunoscând identitatea clientului dintr-o autentificare primară anterioară, va transmite furnizorilor de servicii identitatea clientului printr-un canal securizat. Între ASP și furnizorii de servicii trebuie să existe o relație de încredere stabilită printr-un contract.

Diferența esențială față de sistemele pseudo-SSO este că ASP-ul realizează autentificarea printr-un protocol special, iar credențialele nu sunt transmise niciodată în clar.

#### 2.3.1.4 Sisteme SSO bazate pe proxy

În acest model, rolul ASP-ului este luat de un server extern clientului, care acționează ca un proxy între client și furnizorii de servicii. Avantajul față de modelul anterior este separarea fizică între client și ASP, în așa fel încât acesta din urmă poate fi mai bine protejat împotriva încercărilor de fraudare. Este important de arătat că un ASP poate impersona oricare identitate pe care o reprezintă, rezultă deci că încrederea clientului și a furnizorilor de servicii este esențială pentru buna funcționare a sistemului.

### 2.3.2 Proprietăți ale sistemelor SSO

Sistemele SSO au o serie de proprietăți trecute în revistă în cele ce urmează:

#### 2.3.2.1 Confidențialitatea

Confidențialitatea are o importanță deosebită în sistemele SSO [100]. Procesul de autentificare nu trebuie să dezvăluie identitatea directă a clientului. De asemenea, este de dorit ca două sau mai multe identități distincte ale aceluiași client să nu poată fi corelate fără consimțământul acestuia. Imposibilitatea corelării identităților implică imposibilitatea determinării identității personale a clientului pe baza credențialelor sale [68].

Sistemele pseudo-SSO nu pot garanta pseudonimitatea (necorelarea identităților), deoarece acestea sunt specifice fiecărui furnizor de servicii și pot conține uneori date personale ale clientului. Astfel, unele sisteme pot cere autentificarea cu certificate, site-urile FTP cer uneori ca numele utilizatorului să fie chiar adresa e-mail, sau codul numeric personal.

#### 2.3.2.2 Acces anonim la rețea

Deși confidențialitatea este o cerință a oricărei implementări SSO, aceasta depinde de modul particular în care este exploatat sistemul. Comunicarea între părțile implicate se realizează pe căi de comunicație „punct la punct”, deci fiecare nod trebuie identificat în mod unic printr-o adresă. Din nefericire, comunicația pe straturile inferioare ale protocoalelor se face fără anonimizare, iar un eventual atacator poate compromite anonimitatea prin analiza pachetelor [4]. Fiecare pachet are în componență printre altele adresa sursă care poate oferi informații despre poziționarea geografică a furnizorului de Internet. Cu această informație, atacatorul poate realiza corelarea identităților fără un efort semnificativ.

Problema determinării sursei traficului este rezolvată prin folosirea unor proxy-uri de anonimizare (ascundere a identității) prin care se realizează comunicația între client și furnizorul de servicii. Proxy-ul are rolul de a înlocui adresa clientului cu propria sa adresă (posibil și alte informații personale), făcând astfel imposibilă identificarea clientului. Nivelul de anonimitate este dat în cele din urmă de numărul de utilizatori care folosesc proxy-ul [68]. Exemple de astfel de servicii sunt Anonymizer [89] și Freedom WebSecure [90].

Modelul de anonimizare cu un singur proxy nu protejează împotriva analizei de trafic și se poate folosi atât timp cât operatorul proxy-ului nu este compromis. Din fericire există scheme – ca cele descrise în [24][40] – care protejează împotriva analizei traficului și distribuie încrederea între proxy-urile implicate. O implementare de acest fel este JAP [91].

Anonimizarea poate fi realizată și la nivelul proxy-ului SSO, într-o manieră transparentă pentru client și pentru furnizorul de servicii. Desigur, întreg traficul clientului trebuie în acest caz rutat prin proxy, însă clientul nu va trebui să aibă încredere într-o altă entitate decât cea în care are deja încredere.

#### 2.3.2.3 Mobilitatea utilizatorului

Sistemele SSO bazate pe proxy, prin însăși structura lor permit ca utilizatorul să fie mobil, deoarece baza de date cu credențialele sale sunt pe o mașină alta decât cea a clientului. Autentificarea se poate realiza de oriunde, în limitele impuse de protocolul de autentificare în sine.

Sistemele SSO locale oferă mobilitate utilizatorului în măsura în care baza de date cu credențialele utilizatorului sunt păstrate extern clientului. Gradul de siguranță este determinat în principal de modul de stocare al acestora, criptat sau în clar. Cele mai cunoscute implementări comerciale sunt Novell Secure Login ([www.novell.com/products/securelogin](http://www.novell.com/products/securelogin)), Passlogic V-Go ([www.passlogic.com/sso](http://www.passlogic.com/sso)), Protcom SecureLogin ([www.protcom.cc](http://www.protcom.cc)), RSA ClearTrust ([www.rsasecurity.com](http://www.rsasecurity.com)).

#### 2.3.2.4 Utilizarea în medii ostile

În unele cazuri este necesar ca furnizorii de servicii să fie accesați din medii ostile, cum ar fi Internet cafe-urile și terminalele publice. Este de dorit ca mașina clientului să nu aibă acces la secretele memorate în sistemul SSO, pentru a evita lansarea de atacuri prin impersonare (asumarea unei identități false). În acest caz este utilă implementarea modelelor cu proxy. Autentificarea primară trebuie în acest caz să se realizeze prin protocoale de autentificare adecvate, rezistente la atacuri replay sau man-in-the-middle.

#### 2.3.2.5 Costuri de instalare și întreținere

La capitolul costuri, instalarea și întreținerea sistemelor pseudo-SSO și true-SSO sunt diferite. În cazul sistemelor pseudo-SSO, costurile de instalare sunt mici deoarece nu trebuie să existe o infrastructură de securitate propriu-zisă, iar furnizorii de servicii nu trebuie să aibă cunoștință de noul modul. Pe de altă parte, dacă furnizorul de servicii schimbă modul de autentificare al clientului, acest lucru trebuie să se reflecte în componenta pseudo-SSO, ceea ce duce la cheltuieli de exploatare. Lucrurile sunt cu atât mai complicate cu cât mediul în care sistemul pseudo-SSO este instalat este mai dinamic.

Situația este exact inversă în cazul sistemelor true-SSO. Cheltuielile legate de instalare sunt mari întrucât este necesară implementarea unei infrastructuri pentru ca dialogul client – furnizor de servicii să se realizeze în mod securizat. Acolo unde se impune este de asemenea necesară semnarea de contracte de servicii și tratate de încredere reciprocă a părților implicate. Costurile de operare sunt în schimb reduse, deoarece schimbările părților nu implică automat modificarea interfeței SSO.

#### 2.3.2.6 Costuri de exploatare

Costurile de exploatare sunt mai mici în cazul sistemelor SSO locale față de cele bazate pe proxy, deoarece se induc costuri legate de operarea proxy-ului. Comunicația între părțile implicate poate fi taxată după un model negociat, mai ales când tot traficul clientului este rutat fizic prin proxy.

#### 2.3.2.7 Relații de încredere

Atât în cazul pseudo-SSO cât și în cazul true-SSO, clientul trebuie să aibă încredere în componenta care stochează credențialele. În cazul sistemelor bazate pe proxy, relația de încredere se poate reglementa prin contracte și politici la diferite niveluri pentru ca integritatea sistemului SSO să fie menținută. În cazul sistemelor SSO locale, utilizatorul trebuie să aibă încredere în componenta software care stochează și prelucrează credențialele.

Dacă la sistemele true-SSO s-a putut evidenția în mod clar posibilitatea stabilirii prin contract a relației de încredere, în cazul pseudo-SSO relația de încredere este difuză, deoarece furnizorul de servicii poate să nu aibă cunoștință despre chiar existența sistemului SSO. Baza de date cu credențiale poate fi stocată criptat sau în clar sau poate fi replicată pe mai multe servere, de unde rezultă că relația de încredere între client, furnizorul de servicii și cel de identitate depinde de implementarea concretă a sistemului SSO. Mai mult, această relație este dinamică, în sensul că se poate schimba odată cu modificarea metodei de autentificare a furnizorului de servicii.

#### 2.3.2.8 Rezolvarea conflictelor

În caz de conflict sau de cercetare legală, este necesar ca operatorul proxy-ului să poată furniza detalii despre tranzacțiile efectuate. Observația este valabilă atât în cazul sistemelor pseudo-SSO cât și true-SSO, însă accentul cade pe sisteme true-SSO unde jurnalul trebuie să fie menținut la zi cu toate tranzacțiile efectuate deoarece sunt implicate semnificativ mai multe entități decât în cazul pseudo-SSO.

#### 2.3.2.9 Medii închise și medii deschise

Mediile închise în general nu au o politică de confidențialitate bine conturată sau dacă există aceasta este considerată de importanță redusă. În acest caz accentul se pune pe costurile de achiziționare, exploatare și întreținere a sistemului SSO. Deoarece aceste costuri sunt mai mici în cazul sistemelor pseudo-SSO, sistemele scalabile la nivel de întreprindere sunt cele mai potrivite pentru recuperarea rapidă a investiției. Arhitectura acestor sisteme este analizată în [26].



În mediile deschise, nevoia de confidențialitate este acută. Nivelul de confidențialitate cerut este oferit doar de către sistemele true-SSO, astfel încât costurile legate de operare și întreținere sunt depășite.

### 2.3.3 Deficiențe ale sistemelor SSO actuale

Implementările SSO curente cum ar fi Liberty [54][55][56], WS-Federation [88], Shibboleth [78] nu se pot impune ca sisteme complete de management al identității într-un mod pur tehnic. Literatura de specialitate menționează necesitatea coroborării părții tehnice a implementării cu partea juridică, sub forma unor înțelegeri și relații de afaceri între părțile implicate (vezi modelul din Fig. 4)

Cu toate acestea însă, înțelegerile pot fi încălcate de participanții la cercul de încredere dacă beneficiile sunt mai mari decât pierderile sau dacă unul sau mai mulți furnizori de servicii sunt compromiși, caz în care participanții onești au de suferit.

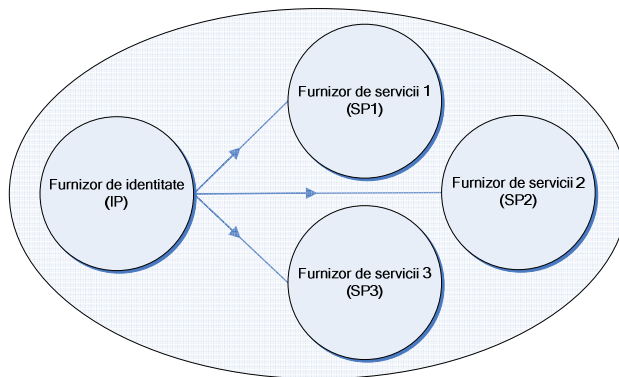


Fig. 4 Cercul de încredere bazat exclusiv pe relații juridice

Pe lângă aspectele legale, sistemele SSO au deficiențe la capitolul disponibilitate, fiind de cele mai multe ori vulnerabile la atacuri DoS. Ideea din spatele unui astfel de atac îndreptat împotriva unui sistem SSO cum este Liberty [54] este foarte de simplă (vezi Fig. 5).

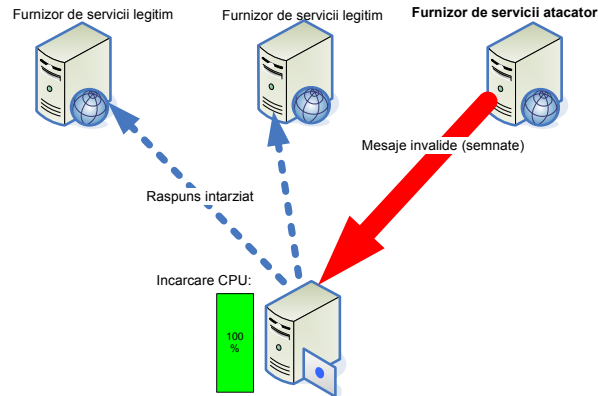


Fig. 5 Atacul DOS împotriva unui sistem SSO Liberty (indiferent de protocol)

Profitând de cerința ca mesajele între entități să fie semnate digital, un furnizor de servicii compromis poate inunda sistemul (în speță furnizorul de identitate) cu mesaje care par legitime, cu semnătură falsă, trimise în cascadă. Nesuspectând nimic, furnizorul de identitate verifică semnăturile digitale ale tuturor mesajelor de intrare, lucru ce duce la epuizarea rapidă a resurselor, ceea ce se traduce fie prin răspunsuri întârziate la cererile furnizorilor legitimi de servicii (degradare de performanță), fie prin absența cu desăvârșire a acestor răspunsuri (indisponibilitate).

Semnarea digitală a fiecărei cereri este o operație extrem de scumpă din punct de vedere al timpului de execuție. Un server care își pune la dispoziție resursele de calcul fără o minimă autentificare prealabilă este în pericol de a fi supraîncărcat cu cereri. În această idee, este foarte curios de aflat de ce dezvoltatorii din Liberty Alliance au trecut cu vederea acest aspect extrem de important. O posibilă explicație ar fi existența cadrului legal în care funcționează sistemul, însă acest lucru nu este suficient pentru protejarea în fapt a participanților onești în sistem.

## 2.4 Securitatea și disponibilitatea rețelelor GSM

În prezent, telefonia fără fir depășește telefonia terestră ca număr de abonați în aproape toate țările Europei și Asiei. Terminalele mobile din generația GPRS/EDGE și 3G facilitează accesul mobil la Internet, iar adoptarea pe scară largă a standardelor IEEE 802.11 și Bluetooth permite integrarea platformelor de telefonie mobilă, laptop și PDA cu suport pentru noi aplicații mobile puternice. Beneficiile imense ale rețelelor universale prezintă un set unic de riscuri care nu pot fi trecute cu vederea, având în vedere dependența din ce în ce mai mare de comunicație.

Tehnologia wireless, fie că e vorba de WiFi (802.11), Bluetooth, GSM sau 3G, este extrem de complexă. Din nefericire, inginerii de telecomunicații nu sunt experți în domeniul securității și prezintă în general tendința de a considera securitatea ca un produs auxiliar ce poate fi oricând adăugat produsului principal, dacă e nevoie. Acest mod de gândire este viciat, întrucât securitatea trebuie să fie în strânsă legătură cu tehnologia radio și telecom. Aceste două domenii trebuie să coexiste și să conlucreze pentru un scop comun, acela de a asigura integritatea, confidențialitatea, autenticitatea și disponibilitatea unui canal de comunicație.

O greșeală care se face în mod curent este că se consideră că procedurile de securitate existente sunt suficient de sofisticate încât să oprească atacurile de orice fel. Acest lucru este greșit, deoarece un atacator va ataca întotdeauna cea mai slabă verigă din sistem, iar acea verigă nu este aproape niciodată sistemul criptografic. Veriga cea mai slabă dintr-un sistem de comunicație fără fir este evident domeniul radio. O judecată de acest fel a dus la implementări cu grave curențe de securitate în protocoalele 802.11b/g WEP și Bluetooth [18]. Aceste sisteme nu au beneficiat de o analiză din punct de vedere al securității în nici una din fazele de proiectare, presupunându-se că mecanismele de securitate comerciale pot fi adăugate ulterior.

Securitatea în rețelele fără fir este o problemă importantă deoarece utilizatorii pot vehicula informații personale, importante sau critice printr-o infrastructură care nu este cu adevărat sigură. Slăbiciunile survin din folosirea multiplelor scheme de securitate incompatibile și greșelile inerente din design-ul protocoalelor. Cel mai mare pericol este ca utilizatorul să perceapă întreaga structură ca fiind sigură, având încrederea că datele sale confidențiale sunt în siguranță. Mediul wireless prezintă multe probleme de securitate, cum ar fi confidențialitatea, autentificarea părților, integritatea, autorizarea, non-negarea și accesibilitatea. Alte probleme pot include ușurința în utilizare, viteza și standardizarea [85].

Se poate afirma că strategia de securitate trebuie planificată și implementată în funcție de tipul datelor transportate și de pierderile estimate în cazul în care are loc o scurgere de informație sau chiar datele sunt modificate în tranzit. De asemenea, trebuie să considerăm faptul că multe probleme de securitate apar datorită implementărilor greșite, interacțiunilor neplanificate între modulele unui sistem, creșteri neplanificate și alte probleme de securitate apărute în urma atacurilor anterioare (vezi Fig. 6).

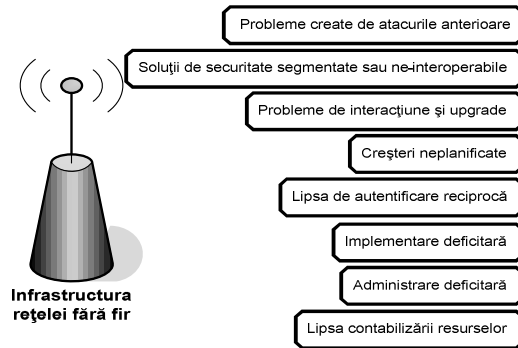


Fig. 6 Probleme de securitate ale rețelelor fără fir

Spre exemplu, un atac de tip Denial of Service, deși în sine acesta nu corupe datele în mod direct, nu sunt motive să nu credem că o acțiune subversivă este în pregătire sau în desfășurare [10].

Pentru a fi eficace cu adevărat, strategia de securitate trebuie aplică de la un capăt la celălalt, adică de la sursă la destinație, indiferent de calea pe care circulă informația. Spre exemplu, WAP oferă securitate prin WTLS (Wireless Transport Security Layer), însă această tehnologie nu este aplicată de la un capăt la celălalt întrucât criptarea are loc doar între terminalul mobil și WAP Gateway [39].

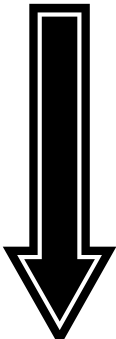
#### 2.4.1 Mecanisme de securitate în rețelele GSM

Securitatea și confidențialitatea în GSM au fost principalele motive pentru care sistemul a fost considerat superior altor sisteme de comunicație mobile. Succesul deosebit a inspirat alte sisteme cum ar fi Code Division Multiple Access (CDMA), Personal Handy Phone System (PHS) și Digital Enhanced Cordless Telecommunications (DECT). O inovație care a contribuit decisiv la popularitatea sistemului GSM a fost introducerea card-ului SIM (*Subscriber Identifier Module*) care a dus la separarea dispozitivului mobil de abonat. Card-ul SIM conține *International Mobile Subscriber Identity* (IMSI) și *Subscriber Identification Key* ( $K_i$ ), ambele fiind folosite pentru autentificarea clientului în rețea. Securitatea GSM se bazează pe trei algoritmi: A3 și A8 pentru autentificare și A5 pentru criptare.

Cu mai mult de 1 miliard de abonați în lume, GSM este o țintă potențială pentru mai multe tipuri de atacuri. Cele mai ușor de pus în practică sunt atacurile care nu necesită tehnologie specială, cum ar fi redirectionarea apelurilor către numere taxate speciale (în funcție de operator), informații false la înregistrarea abonamentelor, fraudă în roaming și furtul de terminale. Sistemele de monitorizare anti-fraudă monitorizează o varietate de indicatori cum ar fi apeluri multiple în același timp, variații largi ale valorilor facturilor, variații largi ale duratelor

convorbirilor (foarte scurte sau foarte lungi), schimbări în tiparele de folosire (indiciu că mobilul a fost furat sau este abuzat) și monitorizarea îndeaproape a clientului în timpul unei perioade probatorii [38].

Tabel 1. Capacitățile unui atacator

|  |   |
|--|---|
| <b>Dificultate</b><br><br> | <b>Capturarea traficului</b><br><br>Reprezintă capacitatea unui intrus de a intercepta traficul și informațiile de semnalizare asociate utilizatorilor GSM. Echipamentul necesar este un telefon mobil modificat.   |
|  | <b>Impersonarea unui utilizator</b><br><br>Reprezintă capacitatea de a trimite date false și/sau informații de semnalizare eronate către rețea cu intenția de a părea că provin de la un utilizator legitim. Echipamentul necesar este un telefon mobil modificat.  |
|  | <b>Impersonarea rețelei</b><br><br>Reprezintă capacitatea de a trimite date false și/sau informații de semnalizare către un utilizator cu intenția de a le face să apară de la o rețea autentică. Echipamentul necesar este un BTS modificat.   |
|  | <b>MITM – Man-In-The-Middle</b><br><br>Reprezintă capacitatea unui atacator de a se plasa între rețea și utilizatorul legitim cu scopul de a intercepta, modifica, șterge, reordona, reda și falsifica data dintre cele două părți. Echipamentul necesar este un BTS modificat în conjuncție cu un telefon mobil modificat. |
|  | <b>Compromiterea Autentificării Rețelei</b><br><br>Intrusul este în posesia unui vector de autentificare compromis (perechi challenge-response, chei de cifrare, chei de integritate, etc.)   |

Sistemul GSM prezintă mai multe probleme de securitate, după cum urmează:

- Comunicația și traficul de semnalizare nu sunt protejate când se comunică cu rețelele fixe, deci rețelele GSM sunt doar atât de sigure cât rețelele fixe la care acestea se conectează.
- Infrastructura GSM nu adresează atacurile active, cum ar fi caching-ul identității, camparea pe un BTS fals, capturarea traficului, etc.
- Interceptarea legală a fost considerată ca o adăugire ulterioară.
- Mecanismele de autentificare și criptografie sunt foarte dificil de îmbunătățit.

- Lipsa transparenței mecanismelor de securitate (utilizatorul nu este conștient cât de sigure sunt de fapt datele sale).

Capabilitățile tehnice ale unui atacator ce pot influența securitatea în rețelele GSM sunt în număr de cinci (vezi Tabel 1). Prima capabilitate este și cea mai simplă de obținut, următoarele implicând investiții mai importante. Se consideră că un atacator care are o anumită capabilitate le are pe toate cele anterioare [38].

Capturarea traficului și impersonarea utilizatorilor au fost două probleme cunoscute la momentul design-ului securității GSM.

#### 2.4.1.1 Autentificarea

Autentificarea clientului se face printr-un algoritm cerere-răspuns simplu, după cum se arată în Fig. 7. Centrul de autentificare GSM (AuC) generează un număr aleator de 128 de biți și îl trimite stației mobile prin intermediul legăturii radio. Acest număr și cheia abonatului ( $K_i$ ) sunt introduse în algoritmul A3 care produce un răspuns semnat (SRES) care la rândul său este trimis înapoi la AuC. Între timp, AuC a calculat deja cantitatea SRES bazată pe aceleași date de intrare și este acum în poziția de a decide dacă stația mobilă este cine spune că este. Sunt câteva probleme cu acest design. Algoritmii A3 (de autentificare) și A8 (de generare a cheilor) sunt specifici fiecărui operator și sunt secrete bine păzite. Acest lucru se traduce prin obscuritate mai degrabă decât securitate. Este binecunoscut faptul că un algoritm secret de autentificare sau de criptare poate fi vulnerabil deoarece nu beneficiază de experiența comunității criptanalitice care ar putea să contribuie la descoperirea erorilor din design.

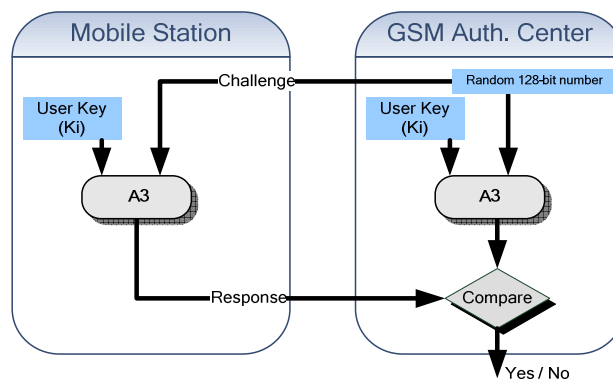


Fig. 7 Autentificarea în rețele GSM

În lumea software, când un program pretinde că implementează un nou algoritm sigur care este de câteva ori mai rapid decât DES sau AES, sunt șanse că

algoritmul nu este nimic mai mult decât o serie de XOR-uri. Cerința ca algoritmul să ruleze pe un smart-card (cum ar fi un SIM) are un profund impact asupra implementării practice. Astfel, 3GPP (3<sup>rd</sup> Generation Partnership Project) sugerează ca implementările implicite ale A3 și A8 să fie o simplă suită de operații XOR, fapt de demonstrează ideea noastră [1]. În mod surprinzător, faptul că SRES are doar 32 de biți are un impact mic asupra securității în cazul unui atac de tip „zi de naștere”<sup>1</sup> deoarece această valoare este utilizată în conjuncție cu cheia aleatoare de la AuC, numărul de capturi necesare pentru un atac cu o probabilitate fezabilă fiind astfel  $1.84 \times 10^{19}$  ( $2^{128/2}$ ) față de 65536 ( $2^{32/2}$ ). Mai multe informații despre atacuri de tip „zi de naștere” se găsesc în [81].

#### 2.4.1.2 Criptarea

Spre deosebire de A3 și A8, standardul GSM specifică algoritmul A5, folosit pentru criptarea vocii, datelor și informațiilor de semnalizare din rețeaua radio. Informația este codificată câte două cadre odată ( $2 \times 114$  biți), câte unul pentru uplink și downlink. În design-ul inițial (numit A5/1), cheia de sesiune K este combinată cu un contor de cadre pentru a inițializa un set de 3 registre care vor produce o ieșire de 228 de biți prin aplicarea XOR la LFSR și textul clar.

O implementare parțială a algoritmului GSM A5 a apărut pe Internet în iunie 1994. La acea vreme s-a susținut faptul că aceea era doar un design timpuriu care are puține asemănări cu algoritmul A5 folosit curent. Se spune că lungimea efectivă a cheii algoritmului A5 este de doar 40 de biți [60].

Neînțelegerile dintre fabricanții de telefoane celulare și guvernul britanic cu privire la licențele de export ale tehnologiilor de criptare în GSM s-au finalizat printr-un compromis în 1993. Națiunilor vest-europene și unor piețe specializate cum ar fi Hong Kong li se permite accesul la tehnologia de criptare GSM A5/1, în timp ce o versiune mai slabă numită A5/2 a fost aprobată pentru export în celelalte țări, incluzând națiunile central și est europene [58]. Aceasta este în principal o problemă politică ce implică drepturile individului și capacitatea agențiilor care asigură ordinea publică de a efectua supravegheri .

Design-ul algoritmului A5 (cu variantele sale A5/1, A5/2 sau mai recenta A5/3) a fost criptanalizat de Barkan, Bihan și Keller [6] care au reușit decodificarea

---

<sup>1</sup> Atacul de tip „zi de naștere” este un atac criptografic, numit astfel deoarece exploatează matematica din spatele paradoxului zilei de naștere din teoria probabilităților. Astfel, dacă o funcție  $f(x)$  are rezultate egale probabilistic în mulțimea  $H$  (cu număr de elemente suficient de mare), se așteaptă găsirea aleatoare a două soluții  $x_1$  și  $x_2$  cu  $f(x_1) = f(x_2)$  după evaluarea a doar  $1,25 \times \sqrt{|H|}$  argumente.

în timp real a fluxului GSM. Acest lucru pune într-o situație sensibilă pe oricine folosește infrastructura GSM pentru comunicații private.

Pentru uz casnic, securitatea în GSM se dovedește a fi net superioară celei din sistemele analogice. Folosirea autentificării, criptării și a numerelor de identificare temporară asigură într-o anumită măsură confidențialitatea și anonimitatea utilizatorilor ca și prevenirea utilizării frauduloase.

#### 2.4.2 Deficiențe ale rețelelor GSM

Rețelele GSM, ca parte a unui sistem larg de comunicații și datorită răspândirii și valorii ridicate de piață de care se bucură, ar putea constitui puncte vulnerabile în infrastructură. Atacurile asupra rețelelor GSM pot cauza pagube importante, de aceea acțiunile subversive sunt tentante pentru atacatori. Rațiunile atacurilor asupra rețelelor GSM par să nu difere de cele îndreptate împotriva rețelelor de calculatoare, după cum urmează:

- Atacurile care cauzează întreruperea totală a serviciilor produc pierderi uriașe în venituri, fără a pune la socoteală impactul social uriaș.
- Când nevoia de comunicare este imperioasă, spre exemplu după un atac terorist sau un dezastru natural, atacul DoS asupra rețelei GSM poate avea consecințe dintre cele mai grave, cum ar fi pierderea de vieți omenești și pierderi materiale.

Atacurile care cauzează întreruperi parțiale sau intermitente ale serviciului GSM sunt extrem de greu de depistat. Un client care dorește folosească rețeaua GSM poate avea dificultăți în inițierea apelurilor, încrederea în operatorul rețelei fiind erodată cu posibila consecință de migrare către concurență [14].

Deși tehnologia GSM a fost proiectată având în vedere facilitățile avansate de securitate, acesta fiind și motivul des invocat de superioritate asupra altor sisteme celulare, securitatea se referă doar la partea radio și este proiectată să împiedice doar atacatorii mai puțin experimentați. Deoarece resursele radio sunt limitate, sistemul GSM are facilități puternice de contabilizare a resurselor. Totuși, în cazul în care o stație mobilă (telefon, modem, etc.) nu funcționează conform standardelor, nu se poate face nimic pentru constrângerea și recuperarea acestora, după cum se va vedea în cele ce urmează.

Scenariul tipic pentru partea preliminară a unui apel cu originea de la terminalul mobil (MS – Mobile Station) este următorul (vezi Fig. 8):

- MS-ul cere asignarea unui canal de control de la BSC (Base Station Controller).



- BTS (Base Station Transceiver) decodifică mesajul CHANNEL REQUEST, calculează avansul de timp (adică distanța MS-BTS) și trimite informația completă către BSC printr-un mesaj CHANNEL REQUIRED. De asemenea, se indică tipul de serviciu solicitat.
- După recepționarea și procesarea unui mesaj CHANNEL REQUIRED, BSC-ul informează BTS-ul despre ce tip de canal și care număr de canal va fi rezervat în urma unui mesaj CHANNEL ACTIVE.
- BTS-ul confirmă primirea mesajului printr-un mesaj CHANNEL ACTIVE ACKNOWLEDGE.
- BSC-ul trimite către BTS mesajul IMMEDIATE ASSIGNMENT COMMAND, care la rândul său informează MS-ul de canalul alocat.

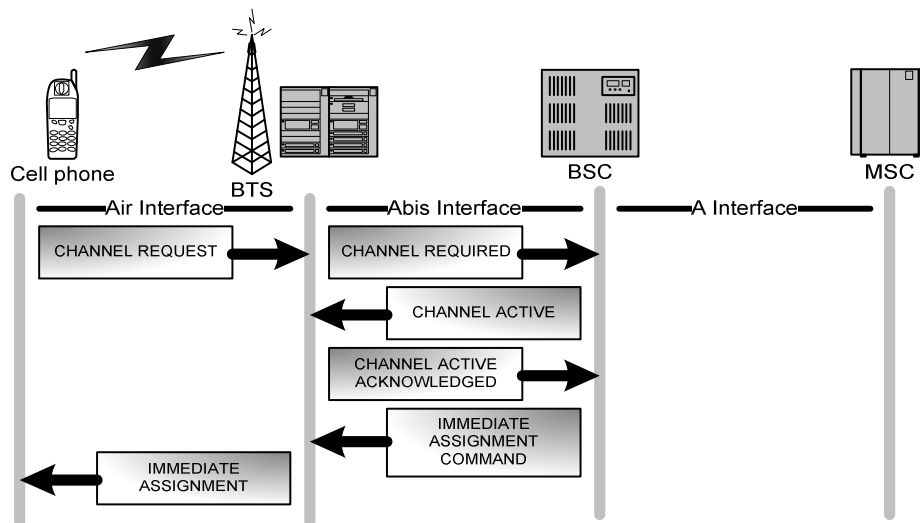


Fig. 8 Procesul de asignare a unui canal în GSM

În acest moment, la cererea unui terminal mobil **neidentificat** (client din perspectiva noastră), BSC-ul a alocat un canal de semnalizare din plaja de canale disponibile. Stația mobilă este acum responsabilă de aderarea la restul protocolului, urmând a cere accesul la serviciu.

Întregul design se bazează pe faptul că stația mobilă va urma în mod corect fiecare pas al protocolului. Ce se întâmplă însă dacă o stație mobilă repetă scenariul anterior și cere mai multe canale de semnalizare fără ca vreodată să continue protocolul până la sfârșit? Deoarece numărul de canale de semnalizare este limitat, rețeaua se congestionează local iar cererile legitime sunt respinse datorită lipsei de canale disponibile. BSC-ul în cele din urmă va dezactiva cererile incomplete, eliberând resursele, dar acest lucru nu înseamnă recuperarea lor deoarece atacul în

sine nu este detectat. Canalele de trafic disponibile nu vor fi oferite clienților legitimi deoarece toate canalele de semnalizare sunt ocupate (vezi Fig. 9).

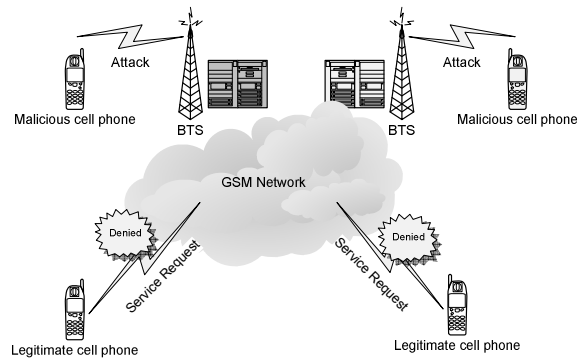


Fig. 9 Atacul de tip DoS într-o rețea GSM

Chiar dacă rețeaua GSM ar efectua o autentificare minimă a stației mobile cerând acestea numărul IMEI sau nivelul de putere recepționat de la cele șase celule învecinate, atacul tot ar fi posibil deoarece aceasta are un control complet asupra informațiilor, putând raporta valori false.

Este interesant de remarcat că întărirea în sine a securității SIM-ului nu este relevantă, întrucât atacul este asupra protocolului de inițiere a apelului în sine și nu asupra SIM-ului [14].

## 2.5 Securitatea și scalabilitatea sistemelor DRM

Sistemele DRM (Digital Rights Management) au ca scop principal aplicarea unor tehnici pentru restricția liberei circulații a conținutului digital (în orice formă, dar mai ales video și audio) în scopul protejării proprietății intelectuale a autorului.

Punctul de vedere al susținătorilor sistemelor DRM este că fără un sistem de protecție al conținutului, care să permită unui grup restrâns de clienți accesul la conținutul digital, rata pirateriei ar crește într-o măsură mare care în final duce la diminuarea profiturilor autorilor. Odată cu declinul vânzărilor, susținătorii DRM spun că factorii creativitate și apoi calitate vor avea de asemenea de suferit.

Susținătorii libertăților civile susțin că folosirea tehnicilor DRM ar trebui limitate sau chiar îndepărtate, întrucât extinderea controlului autorilor de conținut digital până după faza vânzării dăunează expresiei creative și drepturilor consumatorilor. Orice conținut digital ar trebui să fie protejat de legea drepturilor de autor care însă să permită utilizatorului onest folosirea liberă a conținutului în anumite situații, cum ar fi vizionarea sau audiția privată acasă pe dispozitive

diferite, copierea de siguranță, efectuarea de copii pentru membrii familiei, etc. Sistemele DRM actuale nu sunt capabile să facă concesii pentru utilizarea liberă în anumite situații, multe voci afirmând că prin protejarea conținutului digital se aduce atingere drepturilor legale ale consumatorilor.

Printre primele sisteme DRM și poate cel mai contestat este CSS (Content Scrambling System) [96], folosit pentru protejarea filmelor pe suport DVD. Sistemul a fost dezvoltat de DVD Consortium ca un instrument de influențare a producătorilor de hardware de a produce numai sisteme fără anumite caracteristici. Accesul la cheia de decriptare a CSS a fost permis doar producătorilor care au fost de acord să nu includă caracteristici ca digital-out, care ar fi permis ca filmul să poată fi copiat ușor într-o formă fără pierderi. În acest fel DVD Consortium este în cele din urmă capabil de a dicta politica hardware pentru întreaga industrie DVD.

La scurt timp după implementarea sistemului CSS, algoritmul acestuia a fost spart. Au apărut programe cum ar fi DeCSS care permit copierea filmelor și vizionarea copiilor pe dispozitive hardware care în mod normal nu ar fi capabile sau pe sisteme de operare altele decât cele pentru care player-ele software au fost făcute inițial disponibile. În Statele Unite ale Americii și în alte țări ale lumii au apărut legi cum ar fi Digital Millennium Copyright Act (DCMA) care în mod explicit interzic folosirea programelor sau a altor dispozitive de ocolire a protecțiilor DRM.

Deși sistemele DRM sunt cel mai adesea folosite pentru protejarea conținutului video acestea încep să fie folosite și pentru alte tipuri de conținut. Fișierele audio cumpărate din multe magazine online au diverse tipuri de scheme DRM atașate cu scopul de a limita numărul de dispozitive care le pot reda. Mulți producători de publicații electronice folosesc implementări DRM similare pentru a limita numărul de computere pe care acestea se pot vizualiza și chiar numărul de vizualizări.

Problemele de securitate, problemele legate de utilizarea legală și expresia creativă sunt pe primul front al disputei din jurul sistemelor DRM. Această dispută va continua fără îndoială în anii care urmează, susținută de industria media care consideră că DRM este singura modalitate prin care își pot salva modelul de business. Totuși, un număr de inovatori în domeniu au început să exploreze alternative, anticipând înfrângerea sistemelor DRM. Un exemplu este firma Apple condusă de Steve Jobs care a anunțat că piesele muzicale vor fi oferite în magazinul online propriu atât în format protejat cu DRM cât și în format liber.

#### 2.5.1 Deficiențe ale sistemelor DRM

Motivul existenței sistemelor DRM este protejarea conținutului digital și acordarea permisiunilor de redare a acestuia unui subset de clienți autorizați. Scopul final al oricărui atac îndreptat împotriva DRM este obținerea conținutului în clar

pentru a fi ulterior distribuit liber și pentru ca redarea pe diverse dispozitive neautorizate să fie posibilă.

Sistemele DRM pot fi atacate în trei moduri: atacuri asupra infrastructurii, atacuri asupra unității securizate de stocare și atacuri asupra modulului de redare [83].

#### 2.5.1.1 Atacuri asupra infrastructurii

Atacurile de acest tip nu au ca scop principal recuperarea cheilor de conținut sau accesarea conținutului în sine, ci dezactivarea completă a funcției DRM sau perturbarea bunei sale funcționări, în sensul împiedicării furnizării serviciului pentru unul sau mai multe dispozitive (clienți).

Dezactivarea funcției DRM poate avea loc cel mai probabil în cazul unui dispozitiv mobil, aceasta putând surveni ca urmare a unui atac produs fără consimțământul sau știrea proprietarului, prin modificarea neautorizată a softului. Acest atac poate avea ca scop subminarea încrederii utilizatorului în sistemul DRM care protejează un anumit conținut media, ceea ce duce în final la renunțarea completă la sistem și scăderea cotei sale de piață.

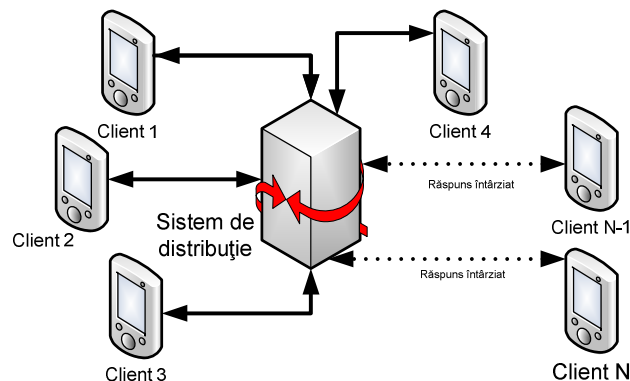


Fig. 10 Scalabilitate redusă într-un sistem de distribuție al conținutului, cu DRM.

Atacul prin împiedicarea furnizării de conținut digital protejat cu DRM poate avea loc la nivel de infrastructură și se traduce în fapt printr-un atac DoS. Acțiunea de protejare cu DRM a unui conținut digital este în sine o operație costisitoare din punct de vedere al resurselor, aceasta presupunând din partea serverului una sau mai multe operații criptografice cu chei publice, operații cu chei simetrice și aplicarea funcțiilor de dispersie asupra conținutului. În cazul în care sistemul de distribuție este de tip centralizat, deservind o mulțime de clienți, acesta devine un

punct de gâtuire datorită slabei proprietăți de scalare. Astfel, clienții care solicită un conținut digital de la server vor primi răspunsul cu întârzieri proporționale cu gradul de încărcare al sistemului (vezi Fig. 10), încărcare care depinde în mod direct de numărul simultan de solicitări.

#### 2.5.1.2 Atacuri asupra unității securizate de stocare

Deoarece orice sistem DRM are în componență un modul criptografic care decodifică conținutul protejat cu ajutorul unor chei memorate în cadrul dispozitivului sau într-un loc ascuns din cadrul sistemului de operare, anumite atacuri pot viza exact această zonă. În funcție de design, atacul poate viza:

- Obținerea cheilor asociate dispozitivului – clonarea dispozitivului mobil vizat se poate face ușor iar conținutul destinat dispozitivului original poate fi redat pe toate clonele. Acest atac are însă un efect temporar, întrucât anumite scheme DRM au un sistem de revocare a cheilor compromise care fac neutilizabile clonele.
- Obținerea cheilor asociate conținutului digital – acest lucru se traduce prin posibilitatea de decodifica conținutul digital și de a îndepărta protecția DRM. Conținutul astfel obținut este liber de orice restricții și poate fi redat pe orice dispozitiv, cu sau fără DRM.

Cea mai cunoscută metodă de stocare securizată este TPM – Trusted Platform Module [92], un dispozitiv criptografic ce se dorește a fi implementat în orice PC nou. Caracteristica cheie ce permite TPM să ofere stocare securizată este capacitatea de a stoca securizat o serie de măsurători. Pentru a realiza acest lucru, TPM are un set de registre numite PCR – Platform Configuration Registers. Un PC echipat cu TPM și un BIOS securizat poate efectua o serie de măsurători ale hardware-ului și software-ului de pe mașina în cauză și să le memoreze în registrele menționate. TPM-ul poate semna criptografic aceste registre și le poate trimite unui terț, spre verificare. Acesta la rândul său, poate verifica măsurătorile și poate afirma că sistemul a pornit în configurația menționată de TPM. Mai mult, TPM nu permite interogarea registrelor de către o mașină a cărei configurație hardware sau software a fost modificată.

#### 2.5.1.3 Atacuri asupra modulului de redare

Acest tip de atac vizează ultima legătură din lanțul DRM și pentru că atacatorul nu are nevoie de cunoștințe de criptografie, acest tip de atac este și cel mai ușor de pus în practică. În principiu atacul constă în interceptarea semnalului digital în clar după ce acesta a fost autorizat și decriptat de către sistemul DRM. Dacă interceptarea se poate face înainte de transformarea semnalului în formă analogică (pentru redare pe TV sau în difuzoare), calitatea semnalului capturat este bună, de multe ori identică cu originalul, ceea ce se traduce într-un atac reușit. Dacă însă interceptarea are loc după transformarea în semnal analogic, captura se

realizează cu pierderi de calitate, atacul având astfel un succes parțial (Fig. 11 Atac asupra modului de redare dintr-un sistem DRM).

Pentru a împiedica interceptarea conținutului digital în tranzit de la mediul de stocare la dispozitivul de redare, compania Microsoft a patentat și implementat în sistemul de operare Windows Vista un mecanism numit Protected Media Path (PMP), care este în esență un set de tehnologii DRM care creează un așa-numit mediu protejat (PE – Protected Environment). Ideea din spatele acestui concept este că aplicațiile de nivel înalt (ex. player-ele video și audio) operează la un nivel înalt de abstractizare, putând da comenzi simple gen „Start”, „Stop”, etc., fără a avea acces la date în sine. Mediul protejat (PE-ul) este disponibil doar aplicațiilor agreate și semnate de Microsoft, alte aplicații neavând acces [97].

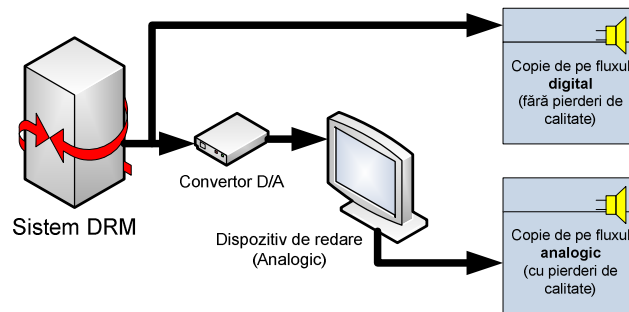


Fig. 11 Atac asupra modului de redare dintr-un sistem DRM

## 2.6 Concluzii

Sistemele de comunicații sunt indispensabile în societatea modernă. Un simplu acces la Internet, un apel de voce sau de date prin telefonie mobilă sau ascultarea muzicii la banalul gadget de buzunar, sunt acțiuni care necesită căi de comunicație. Lipsa sau întreruperea comunicației poate avea urmări dintre cele mai diverse și de cele mai multe ori serioase.

Securizarea căilor de comunicație este un domeniu în care contribuțiile sunt variate. Unul dintre domeniile în care încă există curențe în abordare este creșterea disponibilității și asigurarea scalabilității sistemelor de comunicație, această nișă fiind acoperită de prezenta lucrare de cercetare.

Deși eterogene din punct de vedere tehnologic, s-a avut în vedere un orizont larg de sisteme, cum sunt:

- Sistemele de autentificare
- Sistemele Single Sign-On (SSO)
- Rețelele GSM
- Sistemele de distribuție a conținutului digital

Capitolul de față a făcut o descriere a fiecărui sistem amintit și a scos în evidență deficiențele existente actual, după cum urmează:

- Sistemele de autentificare, sistemele Single Sign-On (SSO) și rețelele GSM sunt vulnerabile la atacuri de tip DoS. Vulnerabilitatea apare din cauza lipsei de control asupra resurselor alocate participanților la comunicație și are de cele mai multe ori ca rezultat degradarea severă a serviciului sau blocarea lui pe perioada desfășurării atacului.
- Sistemele DRM actuale sunt puțin scalabile și indirect sunt susceptibile la atacuri DoS.

Capitolele următoare vor prezenta pe larg contribuțiile aduse în domeniu, ca soluții propuse pentru fiecare dintre deficiențele descrise aici.





### 3. Contribuții la creșterea disponibilității și scalabilității sistemelor de autentificare

Interacțiunea sistemelor de diverse tipuri a dus automat la necesitatea stabilirii cu o oarecare precizie a identității, proces numit uzual autentificare. Procesul de autentificare a fost îndelung analizat în numeroase lucrări, cele mai multe punând în mod firesc accentul pe latura securității.

Cu toate acestea, un nivel de securitate ridicat al protocolului nu reprezintă implicit o condiție suficientă pentru ca autentificarea să se soldeze cu succes. Este nevoie ca autentificarea să aibă loc la modul concret, adică părțile implicate în autentificare să fie disponibile și să execute procesul într-un timp rezonabil.

În general protocoalele de autentificare se bazează pe operații criptografice cu chei publice și pentru că acestea sunt operații complexe și scumpe din punctul de vedere al resurselor, avem de-a face cu o vulnerabilitate la atacuri de tip Denial of Service (DoS). Dezechilibrul între resursele alocate de client pentru execuția protocolului și cele alocate de server permite clientului să angajeze în mod repetat resurse importante ale serverului. Acest lucru se traduce prin posibilitatea încărcării deliberate de către atacator a serverului de autentificare cu cereri false pentru care se alocă resurse de calcul, în detrimentul cererilor legitime care sunt deservite într-un timp mai îndelungat sau în cazul extrem, deloc. Soluția la acest dezechilibru îl reprezintă creșterea artificială a costului de execuție al protocolului pentru client, proporțional cu efortul cerut serverului. Astfel, în cazul autentificărilor sporadice încărcarea pe client este neglijabilă, dar dacă clientul emite cereri repetate către server cu scopul de a-l supraîncărca, încărcarea asupra sa va crește proporțional.

Metoda numită Client Puzzles a fost propusă în literatură pentru echilibrarea efortului pentru cele două părți implicate în autentificare [84]. În acest capitol se propune adoptarea și îmbunătățirea Client Puzzles în cadrul sistemelor de autentificare clasice și a celor de tip Single Sign-On, care să ducă la ameliorarea efectelor atacurilor DoS.

#### 3.1 Client Puzzles

Ideea de bază a acestui mecanism de protecție împotriva atacurilor DoS este că un client care dorește să participe la execuția protocolului de autentificare să-și aloce resursele în mod proporțional cu cele ale serverului, iar în orice punct al execuției protocolului de autentificare, costul rulării pentru client să fie mai mare decât pentru server. Costul pentru client poate fi crescut artificial prin solicitarea rezolvării unui puzzle al cărui grad de dificultate poate fi stabilit cu ușurință de către

server. În același timp, verificarea corectitudinii soluției nu trebuie să fie o povară pentru server deoarece acest lucru ar anula beneficiile acestei tehnici.

Într-o lucrare din anul 1978, Merkle [62] a fost primul care a venit cu ideea de puzzle-uri criptografice dar a aplicat ideea doar pentru schimbul de chei și nu în autentificare. Client puzzle-urile au fost aplicate în atacurile TCP SYN de către Juels și Brainard [48] care menționează că SSL are aceeași slăbiciune și dau o demonstrație riguroasă a caracteristicilor de securitate. Aura, Nikander și Leiwo au aplicat puzzle-urile în protocoalele de autentificare în general [84] iar Dwork și Naor [33] au propus măsuri de reglementare pentru mesajele nesolicitate. Rivest, Shamir și Wagner [72] discută o problema conexasă a criptografiei blocată în timp.

### 3.1.1 Descriere

Înainte să aloce resurse pentru deservirea unei conexiuni serverul trebuie să se asigure că pe întreaga durată a execuției protocolului costul pentru client este mai mare decât pentru server. Costul pentru client se poate mări artificial prin solicitarea rezolvării unui puzzle care trebuie să aibă anumite proprietăți, după cum urmează [84][102]:

- Crearea unui puzzle și verificarea soluției nu necesită resurse importante din partea serverului.
- Costul rezolvării puzzle-ului este ușor de a fi modificat de la 0 la infinit.
- Puzzle-ul poate fi rezolvat pe majoritatea platformelor hardware.
- Precalcularea soluției puzzle-ului este imposibilă.
- În timp ce clientul rezolvă puzzle-ul, serverul nu trebuie să memoreze soluția sau alte informații specifice clientului.
- Același puzzle poate fi distribuit mai multor clienți. Cunoscând soluțiile calculate de unul sau mai mulți clienți nu ajută în calcularea unei noi soluții.
- Un client poate reutiliza un puzzle prin crearea mai multor instanțe ale sale.

Principiul general este arătat Fig. 12.

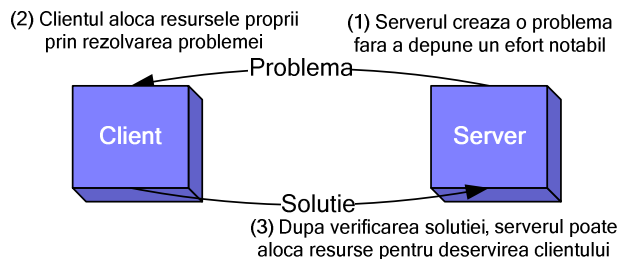


Fig. 12 Principiul client puzzles

### 3.1.2 Tipuri existente de puzzle-uri

#### 3.1.2.1 Inversiunea parțială a unei funcții de dispersie

Prima soluție menționată în literatură este inversiunea parțială a unei funcții de dispersie [48][84]. Ideea de bază este identificarea unei cantități careia aplicându-i-se o funcție de dispersie, să producă un rezultat cu o configurație prestabilită, spre exemplu un număr de biți într-o configurație convenită.

Funcțiile de dispersie sunt primitive criptografice simple care necesită efort de implementare redus pe o varietate de platforme. Deși inversarea parțială a funcțiilor de dispersie este o operație paralelizabilă, propunerile ulterioare de puzzle-uri ale unor autori prezintă la rândul lor neajunsuri. Inversiunea parțială rămâne astfel alegerea din cadrul lucrării de față ca bază de plecare pentru cercetare.

#### 3.1.2.2 Inversiunea logaritmului discret

În [86] se propune folosirea mecanismului Diffie-Hellman pentru a construi un puzzle, având ca bază inversiunea logaritmului discret și se introduce noțiunea de bastion. În accepțiunea lucrării, bastionul este o entitate securizată care protejează un număr de servere.

Ideea de bază este următoarea: serverul și bastionul se pun de acord asupra unui grup ciclic finit  $G$  și unui element generator  $g$ , care se presupune cunoscut de către toți atacatorii. Serverul alege un număr natural  $a$  și trimite  $g^a$  bastionului și similar, bastionul alege un număr natural  $b$  și trimite  $g^b$  serverului. Atât serverul cât și bastionul au acum o cantitate  $(g^a)^b = (g^b)^a = g^{ab}$  care poate fi transmisă parțial clientului, spre exemplu cu un număr  $l$  de biți lipsă din  $b$ . Clientul trebuie să determine valoarea lui  $b$  prin inversarea logaritmului discret.

#### 3.1.2.3 Puzzle-uri blocate în timp

În [72] s-a propus folosirea unui puzzle blocat în timp, care funcționează ca o capsulă de timp care constă dintr-un mesaj ce poate fi descifrat doar după trecerea unui timp.

Se presupune că se dorește criptarea unui mesaj  $M$  pentru o perioadă de  $T$  secunde. Mesajul  $M$  se criptează cu un algoritm cunoscut folosind o cheie  $K$  (spre exemplu RC5),  $C_M = RC5(K, M)$ . Cheia  $K$  se criptează la rândul ei cu  $C_K = K + (a^2)^t \pmod{n}$ , cu  $t = T * S$ , unde  $S$  este numărul de ridicări la putere pe secundă pe care le poate efectua cel care rezolvă puzzle-ul. În lipsa factorizării numărului  $n$ , recuperarea cheii  $K$  se face cel mai eficient efectuând  $t$  operații.

#### 3.1.2.4 Puzzle-uri înlănțuite

Conceptul de puzzle-uri înlănțuite a fost introdus de Ma [59] în 2005 și de Groza [44] un an mai târziu. Construcția puzzle-ului prezentat în [59] începe cu alegerea unui număr aleatoriu inițial  $h_0$ , asupra căruia serverul aplică în mod repetat o funcție de dispersie pentru a obține lanțul  $h_0, h_1, \dots, h_k$  unde  $h_{i+1} = \text{hash}(h_i)$  iar  $k$  este lungimea dorită a lanțului. Acest calcul aduce un avantaj serverului prin stocarea întregului lanț, verificarea soluției reducându-se la o simplă căutare într-o tabelă.

Schema propusă în [44] este similară ca tehnică cu licitațiile puzzle propuse de Wang și Reiter [87], adică cu cât clientul calculează mai multe verigi ale lanțului, cu atât mai multe resurse i se vor aloca de către server.

Deși cele două scheme rezolvă problema calculului paralel a soluției puzzle-ului, acestea prezintă la rândul lor neajunsuri serioase: schema propusă de Ma necesită memorarea soluției fiecărei verigi a lanțului – ceea ce poate duce la epuizarea resurselor de stocare ale serverului – iar cea propusă de Groza este vulnerabilă la atacuri de tip zi de naștere pentru anumite configurații ale lanțului. În plus, această schemă necesită trei operații hash per verigă pentru producerea unui lanț, un cost relativ ridicat în condiții de atac.

#### 3.1.3 Crearea unui puzzle

Periodic (spre exemplu o dată la câteva minute), serverul generează o valoare aleatoare  $N_s$ . Pentru a împiedica atacurile prin ghicire, valoarea trebuie să aibă o entropie de 64 de biți și să nu fie o valoare predictibilă în vreun fel, ca de exemplu amprenta de timp. Această entropie ar trebui să fie suficientă pentru a împiedica un atacator să calculeze perechi « $N_s$ , rezultat». Potrivirile ocazionale rezultate din atacuri de tip „zi de naștere” nu au efecte prea importante în acest caz.

Serverul trebuie să decidă de asemenea și nivelul de dificultate  $k$  al puzzle-ului pe baza unui cumul de măsurători ale condițiilor curente de încărcare. În rezumat, puzzle-ul trimis clienților arată astfel:

«  $N_s, k$  »

- $N_s$  – valoarea aleatoare generată de server (de obicei o cantitate cu mărimea de 64 biți)
- $k$  – nivelul de dificultate a puzzle-ului

#### 3.1.4 Rezolvarea puzzle-ului

Pentru a rezolva puzzle-ul, clientul generează la rândul său o valoare aleatoare  $N_c$ . Scopul acestei valori este dublu. În primul rând, dacă clientul

refolosește valoarea  $N_S$  generată de server, poate construi un nou puzzle prin generarea unei noi valori  $N_C$ . În al doilea rând, fără această valoare, un atacator ar putea calcula puzzle-ul înaintea clientului și ar trimite rezultatul serverului. 24 de biți de entropie ar trebui să fie îndeajuns pentru a împiedica atacatorul de a epuiza întreaga plajă de valori ale  $N_C$  dat fiind faptul că  $N_S$  se schimbă frecvent.

Clientul trebuie să aplice în mod repetat o funcție de dispersie unei cantități iar puzzle-ul se consideră rezolvat când primii  $k$  biți ai cantității  $Y$  sunt egali cu 0.

$$h(C, N_S, N_C, X) = Y$$

- a)  $h$  – funcție criptografică de dispersie, cum ar fi MD5 or SHA
- b)  $C$  – identitatea clientului
- c)  $N_S$  – valoarea aleatoare generată de server
- d)  $N_C$  – valoarea aleatoare generată de client
- e)  $X$  – soluția puzzle-ului

Întrucât serverul schimbă  $N_S$  periodic, atât timp cât acest  $N_S$  se consideră recent, serverul trebuie să mențină o listă de perechi  $N_S - N_C$  în așa fel soluțiile vechi să nu poată fi refolosite.

Deoarece nu se cunoaște nicio metodă ocolitoare pentru a determina  $X$ , singura posibilitate este ca această cantitate să se determine secvențial, adică prin forță brută. Nivelul de dificultate  $k$  (adică numărul de biți 0 de la începutul lui  $Y$ ) dictează cât de mult va lua rezolvarea puzzle-ului. Dacă  $k$  este egal cu 0 nu este necesar nici un efort, iar dacă  $k$  este egal cu 128 (pentru MD5) sau 192 (pentru SHA), clientul trebuie să inverseze o întreagă funcție de dispersie, lucru cvasi-imposibil.

### 3.1.5 Dificultatea puzzle-ului

Parametrul  $k$  reprezintă dificultatea puzzle-ului. Sarcina de a stabili această valoare este destul de dificilă deoarece nu există nicio metrică ce s-ar putea folosi într-o implementare reală. În conformitate cu [28], cea mai potrivită abordare ar fi luarea în calcul a numărului de operații RSA alocate deja. Încărcarea curentă a procesorului și numărul de conexiuni la intrare sunt metrici care depind de un număr de factori care le fac nepotrivite în implementările reale.

Din nefericire, timpul de rezolvare în funcție de dificultatea puzzle-ului urmează o curbă exponențială, aplicabilitatea practică fiind astfel limitată. Pentru a rezolva un puzzle de dificultate  $k$ , clientul trebuie să efectueze în medie aproximativ  $2^{k-1}$  operații. În [84], Aura, Nikkander și Leiwo susțin că valorile rezonabile pentru  $k$  sunt între 0 și 64. Prin experiment am aflat însă că plaja rezonabilă este însă mult mai îngustă.

Pentru a obține o scală mai precisă pentru parametrul de dificultate al puzzle-ului, Juels și Brainard [48] au propus împărțirea problemei în puzzle-uri mai

mici de dificultate egală care ar urma să fie rezolvate separat iar rezultatul total să fie obținut din combinarea rezultatelor parțiale. Aura, Nikkander și Leiwo [84] susțin că aceeași granularitate poate fi obținută din combinarea sub-problemelor de dificultate diferită dar la un cost mai redus pentru server, fără a demonstra practic soluția.

### 3.1.6 Neajunsuri ale soluției Client Puzzle

Client Puzzles este o alegere potrivită securizarea protocoalelor de autentificare împotriva atacurilor DoS atât timp cât atacul nu este distribuit [74]. Deoarece dificultatea puzzle-ului este stabilită după o metrică ce ia în calcul doar angajamentul serverului, ignorând puterea de calcul a clientului (care poate varia în limite largi), eficacitatea metodei poate să fie limitată în anumite cazuri. Puzzle-urile sunt generate la momente precise, momente la care se ia în calcul angajamentul instantaneu al serverului (numărul de operații criptografice care urmează să fie executate într-o perioadă de timp imediat următoare sau orice altă metrică), dar acest lucru poate să nu fie potrivit în toate cazurile.

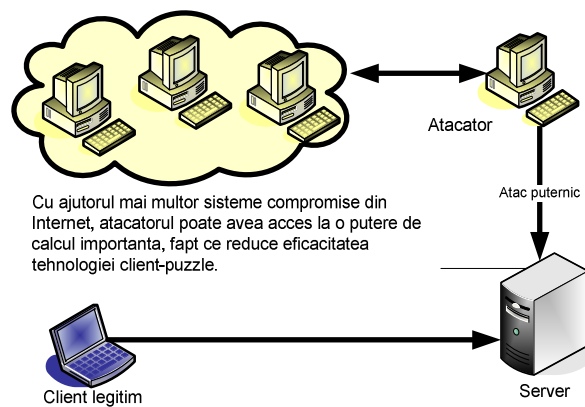


Fig. 13 Schema unui atac puternic

Puzzle-urile sunt astfel vulnerabile la o formă particulară de atac, numită de aici înainte **atac puternic**, datorită naturii paralelizabile a puzzle-ului. Un atac puternic se definește ca un atac de tip *denial-of-service* pus la cale de un atacator cu acces la o putere de calcul importantă. În acest caz, atacatorul este capabil de a rezolva puzzle-urile într-un timp mult mai scurt decât un client legitim, după cum se poate vedea în Fig. 13.

Să presupunem că un server autentifică un număr inițial de clienți legitimi, dificultatea inițială a puzzle-ului fiind zero. Procesul de autentificare își urmează cursul normal, până când are loc un atac puternic. În acest moment, serverul are

tendința de a crește gradual nivelul de dificultate spre valori superioare, pentru a ține pasul cu cantitatea de procesare necesară pentru deservirea cererilor atacatorului. Deși nivelul de dificultate poate fi crescut până la infinit, acest lucru înseamnă un atac DOS în sine îndreptat asupra clienților legitimi care nu ajung niciodată să rezolve un puzzle atât de dificil.

Deși improbabil și greu de pus în practică, un atac puternic este posibil. Dacă un atacator ar avea acces la  $N$  sisteme de calcul conectate (cu  $N$  suficient de mare ca să vorbim de putere de calcul semnificativă), atunci timpul necesar rezolvării puzzle-ului de dificultate  $k$  ar fi proporțional mai mic. Programul SETI [77] și efortul de a sparge algoritmul RSA [32] sunt exemple reale de cum pot conlucra sute de mii de sisteme pentru același scop comun. Puterea cumulată a rețelei *distributed.net* a depășit echivalentul a 160000 de calculatoare PII / 266MHz, lucru ce arată faptul că atacurile puternice sunt posibile.

### 3.1.7 Propuneri pentru ameliorarea experienței clientului

Pentru că în forma actuală experiența utilizatorului în relația cu un sistem de autentificare protejat cu Client Puzzles este suboptimă, propun o serie de modificări ale design-ului actual reunite sub denumirea Threshold Puzzles [11].

#### 3.1.7.1 Limitarea superioară a nivelului de dificultate

Design-ul curent al tehnologiei client puzzle [84] specifică o plajă a dificultății între 0 (nici un efort necesar) și 128 sau 192 (teoretic imposibil, în funcție de tipul de funcție de dispersie folosită). Datorită faptului că dificultatea are o creștere exponențială, implementările acestui mecanism sunt limitate la a folosi valori ale dificultății dintr-o plajă mult mai restrânsă. Nivelurile ridicate ale dificultății ar putea conduce la atacuri DoS îndreptate asupra clienților legitimi, deoarece clienții ar petrece un timp apreciabil pentru găsirea soluției.

Pentru ca percepția dificultății puzzle-ului să fie optimă pentru client trebuie ca timpul de rezolvare al acestuia să fie inferior timpului în care serverul poate răspunde unei cereri de la client având în vedere încărcarea curentă, adică:

$$T_{\text{client}} \leq T_{\text{server}}$$

Definim timpul de rezolvare al puzzle-ului pentru client ca fiind  $T_{\text{client}} = M * t_c$ , unde  $M$  este numărul mediu de operații necesare pentru rezolvarea puzzle-ului iar  $t_c$  este timpul mediu per operație din punctul de vedere al clientului.

$$\text{Știm că } M = \frac{\sum_{i=1}^{2^k} i}{2^k} = \frac{2^k(2^k+1)}{2^{k+1}} \sim 2^{k-1}, \text{ deci } T_{\text{client}} = 2^{k-1} * t_c.$$

Timpul în care serverul poate răspunde unei cereri ia în calcul dimensiunea curentă a cozii de așteptare  $Q$  precum și viteza sa  $t_s$ , astfel:  $T_{server} = Q * t_s$ .

Inegalitatea devine:

$$2^{k-1} * t_c \leq Q * t_s, \text{ de unde obținem valoarea maximă a dificultății } k:$$

$$k \leq \log_2 (Q * t_s / t_c) + 1$$

Deoarece  $k$  trebuie să fie o cantitate pozitivă, pentru a cuprinde și cazul în care încărcarea pe server este redusă (cantitatea de sub logaritmul ar fi subunitară în acest caz), avem:

$$k \leq \log_2 (\max(1, Q * t_s / t_c)) + 1, \text{ unde:}$$

Ec. 1

- $Q$  – dimensiunea curentă a cozii de așteptare
- $t_s$  – timpul necesar efectuării unei operații criptografice, pentru server
- $t_c$  – timpul necesar efectuării unei operații criptografice, pentru client

Pentru  $t_s = 0,003$  și  $t_c = 0,5$  graficul comparativ între dificultatea proporțională a unui client puzzle și dificultatea limitată a unui threshold puzzle arată ca în Fig. 14. Se observă că în cazul threshold puzzle percepția degradării performanței pentru client este mai puțin pronunțată.

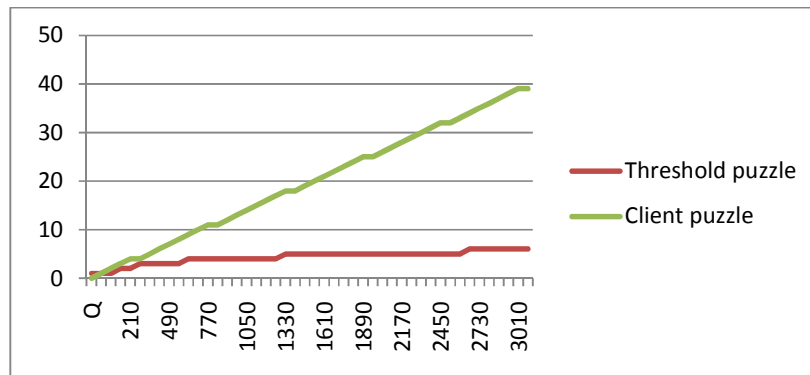


Fig. 14 Evoluția comparativă a dificultății puzzle-urilor TP și CP

### 3.1.7.2 Stabilirea unui timp minim de răspuns

Pentru ca un atac să aibă succes, este necesar ca atacatorul să fie capabil de a trimite cereri numeroase serverului într-un timp scurt, în pofida instalării



tehnologiei client puzzles. Pornind de la observația simplă că orice puzzle, indiferent de nivelul său de dificultate, are un timp de rezolvare, serverul poate să refuze cererile ale căror puzzle a fost rezolvat într-un timp prea scurt.

Ideea de bază constă în adăugarea unei amprente de timp la care serverul a generat valoarea sa aleatoare la lista « $N_S, N_C, X, k$ ». Când serverul primește soluția, acesta poate calcula timpul exact de care a avut nevoie clientul pentru a rezolva puzzle-ul. Acest timp nu ar trebui să fie mai mic de o estimare a serverului bazată pe gradul de dificultate. Dacă este, atunci rezultă că serverul se află sub un atac puternic și ar trebui să înceteze imediat comunicarea cu clientul în cauză.

După cum s-a arătat în paragraful anterior, în medie pentru rezolvarea unui puzzle de dificultate  $k$  este nevoie de  $2^{k-1}$  operații, deci timpul estimativ de rezolvare este:

$$T_e = (2^{k-1}) * T_{\text{operație}}$$

- $T_e$  – timpul estimat pentru rezolvarea puzzle-ului
- $k$  – nivelul de dificultate stabilit de server
- $T_{\text{operație}}$  – timpul mediu de efectuare a unei operațiuni criptografice

### 3.1.7.3 Algoritmul Threshold Puzzles

Algoritmul Threshold Puzzles complet este următorul:

1. La începerea comunicației cu un client, serverul verifică starea sistemului după o metrică proprie. Dacă încărcarea nu depășește un prag considerat critic, execuția algoritmului se oprește, deoarece sistemul de apărare nu este necesar.
2. Serverul generează o valoare aleatoare și unică  $N_S$  (nonce) cu o entropie de 64 de biți și memorează momentul de timp  $T_S$  la care generarea a avut loc.
3. Serverul stabilește un nivel de dificultate  $k$  pentru puzzle și estimează timpul minim  $T_e$  de rezolvare al acestuia. Valoarea dificultății se limitează la un prag superior stabilit, pentru ca o creștere necontrolată a acestuia să nu aibă repercusiuni asupra clienților cu putere de calcul limitată.
4. Serverul creează un puzzle pentru client:

$$\langle N_S, k, T_S \rangle$$

- $N_S$  – valoarea aleatoare generată de server
- $k$  – nivelul de dificultate al puzzle-ului, cu  $k \leq \log_2 (\max(1, Q * t_s / t_c)) + 1$ , unde:
  - a.  $Q$  – dimensiunea curentă a cozii de așteptare
  - b.  $t_s$  – timpul necesar efectuării unei operații criptografice, pentru server
  - c.  $t_c$  – timpul necesar efectuării unei operații criptografice, pentru client

- $T_S$  – timpul la care valoarea aleatoare a fost generată
5. La recepția puzzle-ului, clientul verifică faptul că amprenta de timp  $T_S$  este recentă, generează un număr aleator  $N_C$ , și rezolvă puzzle-ul aplicând în mod repetat o funcție de dispersie unei cantități formate din  $C$ ,  $N_S$ ,  $N_C$  și  $X$ . Puzzle-ul se consideră rezolvat când  $k$  biți consecutivi ai cantității  $Y$  sunt egali cu 0.

$$h(C, N_S, N_C, X) = Y$$

- a)  $h$  – funcție criptografică de dispersie, cum ar fi MD5 or SHA
  - b)  $C$  – identitatea clientului
  - c)  $N_S$  – valoarea aleatoare generată de server
  - d)  $N_C$  – valoarea aleatoare generată de client
  - e)  $X$  – soluția puzzle-ului
6. La primirea rezultatului, serverul:
- a) Calculează timpul necesar clientului pentru a rezolva puzzle-ul:  $T_{solve} = T_S - T_R$ , unde  $T_R$  este timpul de recepție. Dacă acest timp este mai mic decât timpul estimat  $T_E$ , clientul are la dispoziție o putere de calcul mare. În acest caz serverul poate să decidă terminarea comunicației sau întârzierea deliberată a comunicației.
  - b) Verifică dacă clientul a mai furnizat o soluție cu aceiași parametri  $N_S$  și  $N_C$ , caz în care comunicația cu clientul  $C$  încetează.
  - c) Verifică corectitudinea soluției  $X$ .
7. Dacă toate condițiile sunt satisfăcute, serverul poate să aloce resurse pentru execuția protocolului de autentificare a clientului curent.

#### 3.1.7.4 Puzzle-uri adaptate clienților

Variatatea dispozitivelor care accesează diverse servicii este largă, pornind de la sisteme desktop de ultimă generație, laptop-uri în diverse configurații și terminând cu dispozitive mobile cum ar fi PDA-uri și smartphone-uri. Puterea de calcul variază în limite largi la aceste dispozitive și în mod evident, în conjuncție cu tehnologia threshold puzzles, experiența utilizatorului ar fi foarte diferită în condiții de încărcare similare. Acest lucru se întâmplă deoarece un sistem desktop sau laptop este mai puternic decât un PDA, iar în cazul autentificării la același server întârzierea observată de utilizator fiind mult mai mare în cazul PDA-ului.

Vorbim în acest caz de necesitatea adaptării puzzle-urilor la performanțele fiecărui client în parte, lucru ce ar duce în final la o experiență uniformă a utilizatorului, indiferent de dispozitivul ales. Adaptarea dificultății puzzle-ului este o idee nouă căreia i-am dat un nume sugestiv: Adaptive Threshold Puzzles [12]. Ideea de bază din spatele acestui mecanism este determinarea puterii de calcul aproximative a clientului și adaptarea proporțională a dificultății puzzle-ului în cazul unui atac.

La primul dialog între client și server sau la cererea expresă a clientului poate avea loc determinarea puterii de calcul a clientului. Serverul va trimite un puzzle de explorare care conține o problemă ce trebuie rezolvată de client, timpul de

rezolvare determinând puterea de calcul efectivă. Problema poate fi o inversare parțială a unei funcții de dispersie (procedură asemănătoare celei din cazul client puzzles) sau poate fi o altă metodă total diferită, în funcție de implementarea reală. Este recomandabil ca dificultatea și timpul de rezolvare să aibă o dependență liniară.

Puterea de calcul  $P_C$  a clientului poate fi calculată de către server pe baza timpului în care clientul rezolvă problema. Se poate însă ca un client malițios să întârzie în mod deliberat trimiterea răspunsului, pentru a ascunde adevărata valoare pentru  $P_C$  și pentru ca serverul să-i trimită puzzle-uri cu o dificultate redusă în cazul unui atac, probabil lansat tot de către acesta. De aceea, trebuie găsită o metodă de a încuraja clientul să-și folosească întreaga putere de calcul pe durata rezolvării puzzle-ului de explorare. O soluție ar fi acordarea unui număr limitat de conexiuni pe unitatea de timp proporțional cu puterea de calcul estimată pentru acesta. Spre exemplu, unui client puternic i se alocă un număr maxim de  $N$  conexiuni pe unitatea de timp, iar unui client mai puțin puternic (cum ar fi un PDA sau un telefon mobil) i se vor acorda doar  $N/2$  sau  $N/3$  conexiuni pe aceeași unitate de timp. Dacă un client malițios rezolvă puzzle-ul de explorare într-un timp mare, determinând astfel o putere  $P_C$  mai mică decât are de fapt, numărul de conexiuni pe unitatea de timp va fi proporțional mai mic, iar conexiunile care depășesc acest număr sunt ignorate. În acest fel, atacurile unui client care nu-și raportează corect puterea de calcul, nu sunt posibile.

Complexitatea adaptată  $k$  a puzzle-ului se poate calcula astfel:

$$k_C = \text{round} \left( k \cdot \log_2 \left( \frac{P_C}{P_{ref}} \right) \right)$$

Ec. 2

Unde:

- $P_{ref}$  – Puterea de calcul de referință
- $k$  – Dificultatea de referință (corelată cu  $P_{ref}$ )
- $P_C$  – Puterea de calcul raportată de un client
- $k_C$  – Dificultatea adaptată clientului

Dificultatea de referință se calculează ca și în cazul threshold puzzles:

$k \leq \log_2 (\max(1, Q * t_s / t_c)) + 1$ , unde:

- $Q$  – dimensiunea curentă a cozii de așteptare
- $t_s$  – timpul necesar efectuării unei operații criptografice, pentru server
- $t_c$  – timpul necesar efectuării unei operații criptografice, pentru client

## 3.1.7.5 Algoritmul Adaptive Threshold Puzzles

Având în vedere motivarea din paragraful anterior, algoritmul Adaptive Threshold Puzzles este următorul:

1. La începerea comunicației cu un client, serverul verifică starea sistemului. Dacă încărcarea nu depășește un prag considerat critic, execuția algoritmului se oprește, deoarece sistemul de apărare nu este necesar.
2. Dacă clientul este întâlnit pentru prima oară, serverul trebuie să afle puterea de calcul aproximativă de care dispune clientul, trimițând un puzzle de explorare. Acest puzzle este o simplă inversiune parțială a unei funcții de dispersie. Puzzle-ul trebuie să aibă o complexitate medie, spre exemplu  $k = 6$ . Numărul de cereri permise de client per unitatea de timp vor fi în funcție de puterea raportată de client în acest pas.
3. Serverul generează o valoare aleatoare și unică  $\mathbf{N}_s$  (nonce) cu o entropie de 64 de biți și memorează momentul de timp  $\mathbf{T}_s$  la care generarea a avut loc.
4. Serverul stabilește un nivel de dificultate  $\mathbf{k}_c$  personalizat pentru puzzle și estimează timpul minim  $\mathbf{T}_E$  de rezolvare al acestuia,

$$k_c = \text{round} \left( k \cdot \log_2 \left( \frac{P_c}{P_{ref}} \right) \right)$$

5. Serverul creează un puzzle pentru client:

**«  $\mathbf{N}_s, \mathbf{k}, \mathbf{T}_s$  »**

- $\mathbf{N}_s$  – valoarea aleatoare generată de server
  - $\mathbf{k}$  – nivelul de dificultate a puzzle-ului
  - $\mathbf{T}_s$  – timpul la care valoarea aleatoare a fost generată
6. La recepția puzzle-ului, clientul verifică faptul că amprenta de timp  $\mathbf{T}_s$  este recentă, generează un număr aleator  $\mathbf{N}_c$ , și rezolvă puzzle-ul aplicând în mod repetat o funcție de dispersie unei cantități formate din  $\mathbf{C}$ ,  $\mathbf{N}_s$ ,  $\mathbf{N}_c$  și  $\mathbf{X}$ . Puzzle-ul se consideră rezolvat când  $\mathbf{k}$  biți consecutivi ai cantității  $\mathbf{Y}$  sunt egali cu 0.

**$\mathbf{h}(\mathbf{C}, \mathbf{N}_s, \mathbf{N}_c, \mathbf{X}) = \mathbf{Y}$**

- f)  $\mathbf{h}$  – funcție criptografică de dispersie, cum ar fi MD5 or SHA
  - g)  $\mathbf{C}$  – identitatea clientului
  - h)  $\mathbf{N}_s$  – valoarea aleatoare generată de server
  - i)  $\mathbf{N}_c$  – valoarea aleatoare generată de client
  - j)  $\mathbf{X}$  – soluția puzzle-ului
7. La primirea rezultatului, serverul:
    - a) Calculează timpul necesar clientului pentru a rezolva puzzle-ul:  $T_{solve} = T_s - T_R$ , unde  $T_R$  este timpul de recepție. Dacă acest timp este mai mic decât timpul estimat  $T_E$ , clientul are la dispoziție o putere de calcul mare. În acest caz serverul poate să decidă terminarea comunicației sau întârzierea deliberată a comunicației.

- b) Verifică dacă clientul a mai furnizat o soluție cu aceiași parametri  $N_S$  și  $N_C$ , caz în care comunicația cu clientul C încetează.
  - c) Verifică corectitudinea soluției **X**.
8. Se aplică constrângerea ca numărul de conexiuni de la client per unitatea de timp să nu depășească un prag corespunzător puterii de calcul raportate de acesta la pasul 2.
  9. Dacă toate condițiile sunt satisfăcute, serverul poate să aloce resurse pentru execuția protocolului de autentificare a clientului curent.

### 3.2 Aspecte practice de implementare

Aspectele teoretice expuse în paragrafele anterioare au fost validate în practică. Pentru a reduce timpul de dezvoltare al prototipului software și pentru a elimina problemele inerente ale unor implementări de la zero, am utilizat biblioteca Mentalis Security Library [98] care este o implementare open-source a SSL și TLS în .NET Framework.

#### 3.2.1 Algoritmul SSL Handshake

Profitând de faptul că SSL este un protocol de autentificare popular, dată fiind prezența sa în orice browser, este oportun a verifica considerațiile teoretice asupra acestuia. Pentru înțelegerea schimbărilor propuse în SSL este necesar să prezentăm întâi configurația sa nemodificată.

Protocolului SSL Handshake este descris în Fig. 15. Schimbul de mesaje între client și server conține informații referitoare la capabilitățile criptografice ale clientului precum și alegerea serverului în această privință.

Se observă că mesajul CERTIFICATE conține o semnătură electronică, serverul angajându-și în acest fel resursele fără a cunoaște identitatea clientului și fără să știe dacă acesta este bine intenționat sau nu. Dacă clientul nu intenționează să continue dialogul cu serverul ci dimpotrivă, doar să îl supraîncarce cu cereri inutile care vor fi semnate în cele din urmă, avem de-a face cu un atac DoS tipic.

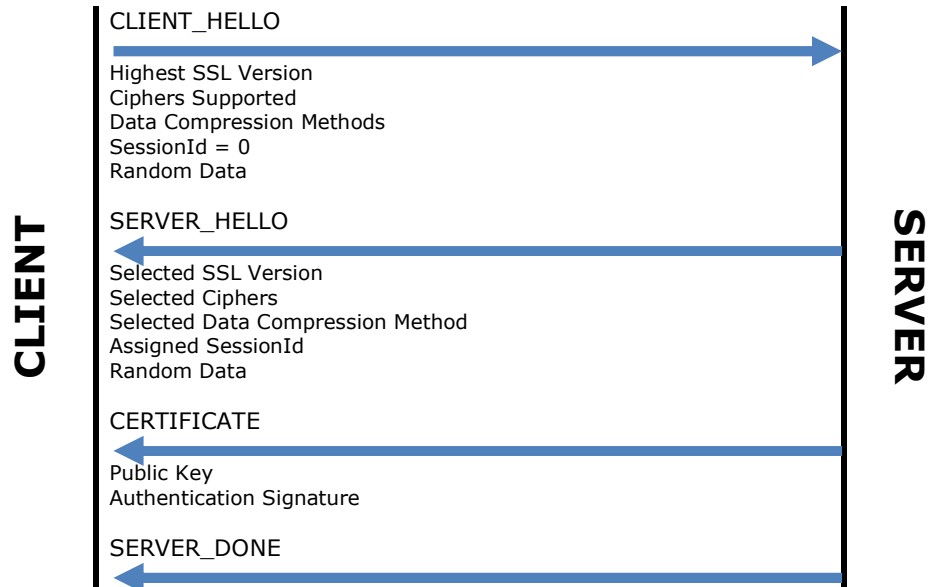


Fig. 15 Protocolul SSL Handshake

Din punct de vedere al disponibilității, angajarea necondiționată de resurse din partea serverului este un neajuns. Tocmai de aceea am făcut propunerea de a utiliza tehnologiile Threshold Puzzles și Adaptive Threshold Puzzles în cadrul sistemelor de autentificare în general și mai cu seamă în cel mai răspândit sistem de autentificare existent, SSL. Modificările aduse protocolului nu afectează credibilitatea sa criptografică ci îi conferă posibilitatea de a egaliza efortul procesului de autentificare pe cele două părți implicate, clientul și serverul.

În cele ce urmează se va prezenta varianta modificată a protocolului, așa cum a fost ea implementată în cadrul prototipului software.

### 3.2.2 Algoritmul SSL Handshake cu distribuție egală de efort

Suportul pentru tehnologia Threshold Puzzles presupune modificarea fluxului de mesaje ale protocolului SSL Handshake și anume introducerea a două mesaje adiționale numite PUZZLE\_CHALLENGE respectiv PUZZLE\_SOLUTION înainte de mesajul SERVER\_DONE care marchează sfârșitul protocolului. Scopul acestei modificări este distribuirea egală a efortului între cele două entități implicate în procesul de autentificare, clientul și serverul. Cantitatea de procesare pe care trebuie să o efectueze serverul trebuie să se mențină sub un prag critic. Dacă pragul este depășit este necesar ca avalanșa de cereri de autentificare să fie controlată, o metodă potrivită fiind ocuparea clientului cu o sarcină cu dificultate proporțională cu întârzierea necesară serverului pentru a ține pasul.

Mesajul PUZZLE\_CHALLENGE conține timpul la care puzzle-ul a fost creat (*TimeStamp*), nivelul de dificultate al puzzle-ului (*Difficulty*) și o valoare aleatoare unică aleasă de server (*ServerNonce*). Acest mesaj este trimis de server ca răspuns la mesajul CLIENT\_HELLO numai în cazul în care încărcarea serverului depășește un prag considerat critic. La rândul său clientul răspunde cu mesajul PUZZLE\_RESPONSE, care conține identitatea sa (*ClientID*), valoarea aleatoare de la server (*ServerNonce*), o valoare unică aleatoare generată de client (*ClientNonce*) și soluția puzzle-ului (*Solution*).

Fluxul mesajelor pentru algoritmul SSL Handshake cu distribuție egală de efort este prezentat în Fig. 16:

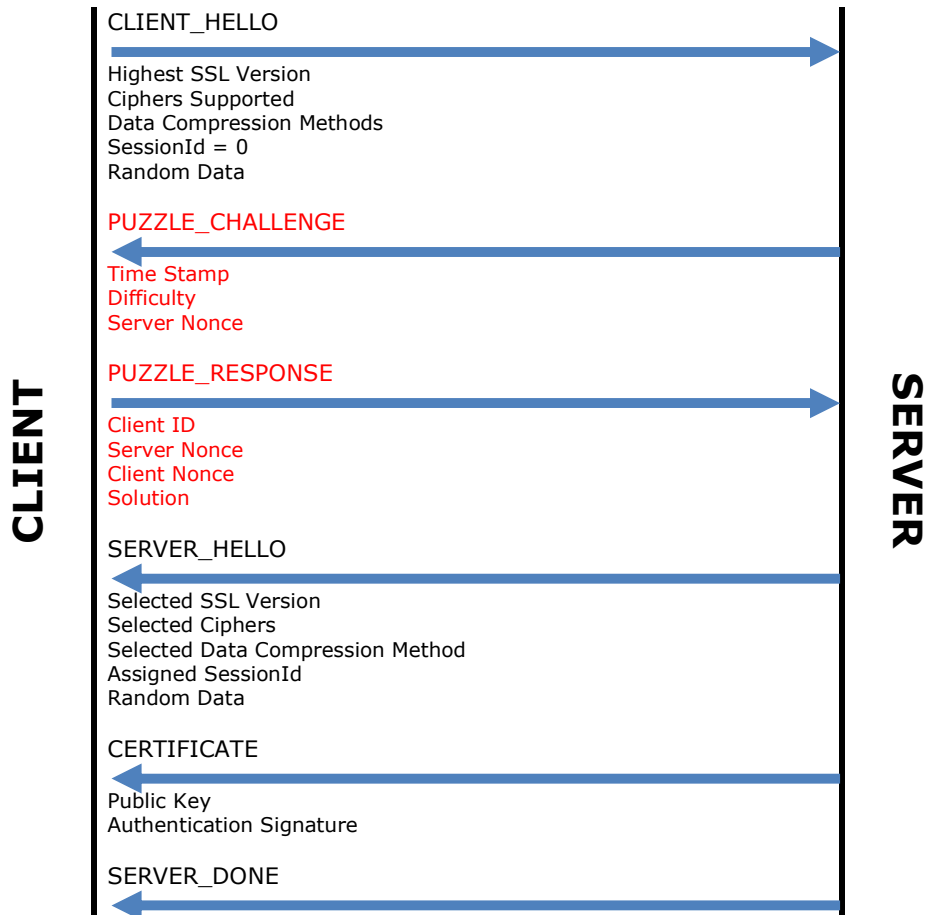


Fig. 16 Protocolul SSL Handshake cu distribuție egală de efort

Adăugarea celor două mesaje suplimentare (PUZZLE\_CHALLENGE și PUZZLE\_SOLUTION) nu schimbă integritatea criptografică a protocolului SSL ci are ca scop ameliorarea disponibilității serverului prin întârzierea clientului cu o durată de timp proporțională cu încărcarea curentă a serverului. Schimbările la nivel criptografic ar presupune o abordare diferită și deosebit de laborioasă pentru a evita introducerea de noi vulnerabilități.

### 3.2.3 Algoritmul SSL Handshake cu distribuție adaptivă de efort

Pentru scenariile în care clienții unui sistem sunt eterogeni din punct de vedere al capacității de calcul (laptop-uri, PDA-uri, etc.) se impune adoptarea tehnologiei Adaptive Threshold Puzzles. La fel ca în cazul distribuției egale de efort din paragraful anterior este necesară modificarea fluxului de mesaje ale protocolului SSL Handshake prin introducerea de mesaje adiționale. Două dintre mesaje sunt cele pe care le-am amintit deja, și anume PUZZLE\_CHALLENGE respectiv PUZZLE\_SOLUTION și au aceleași funcții. În plus, distribuția adaptivă a efortului necesită cunoașterea puterii de calcul a clientului, acest lucru fiind aflat prin altă pereche de mesaje: PUZZLE\_EXPLORE\_CHALLENGE și PUZZLE\_EXPLORE\_SOLUTION. Scopul acestei modificări este distribuirea efortului între cele două entități implicate în procesul de autentificare, clientul și serverul, însă proporțional cu puterea de calcul a clientului. Cantitatea de procesare pe care trebuie să o efectueze serverul trebuie să se mențină sub un prag critic. Dacă pragul este depășit este necesar ca avalanșa de cereri de autentificare să fie controlată, o metodă potrivită fiind ocuparea clientului cu o sarcină cu dificultate proporțională cu întârzierea necesară serverului pentru a ține pasul.

Dacă clientul care solicită autentificarea este necunoscut, serverul are nevoie să cunoască puterea sa de calcul. Această determinare se face prin solicitarea rezolvării unui puzzle de explorare cu dificultate medie, prin mesajul PUZZLE\_EXPLORE\_CHALLENGE. Răspunsul la acest mesaj vine din partea clientului prin mesajul PUZZLE\_EXPLORE\_SOLUTION. Dacă clientul este cunoscut (a mai efectuat cereri de autentificare în trecut) se poate omite pasul de explorare pentru un timp nedeterminat sau pentru un timp stabilit în mod unilateral de către server. Odată cunoscută puterea de calcul a clientului (fie din explorarea curentă sau una petrecută anterior), serverul poate cere clientului rezolvarea unui puzzle, asemenea cu cazul anterior.

Mesajele PUZZLE\_EXPLORE\_CHALLENGE și PUZZLE\_CHALLENGE conțin timpul la care puzzle-ul a fost creat (*TimeStamp*), nivelul de dificultate al puzzle-ului (*Difficulty*) și o valoare aleatoare unică aleasă de server (*ServerNonce*). Clientul răspunde cu mesajele PUZZLE\_EXPLORE\_SOLUTION și PUZZLE\_SOLUTION, care conțin identitatea sa (*ClientID*), valoarea aleatoare de la server (*ServerNonce*), o valoare unică aleatoare generată de client (*ClientNonce*) și soluția puzzle-ului (*Solution*).



Fluxul mesajelor pentru algoritmul SSL Handshake cu distribuție adaptivă de efort este prezentat în Fig. 17:

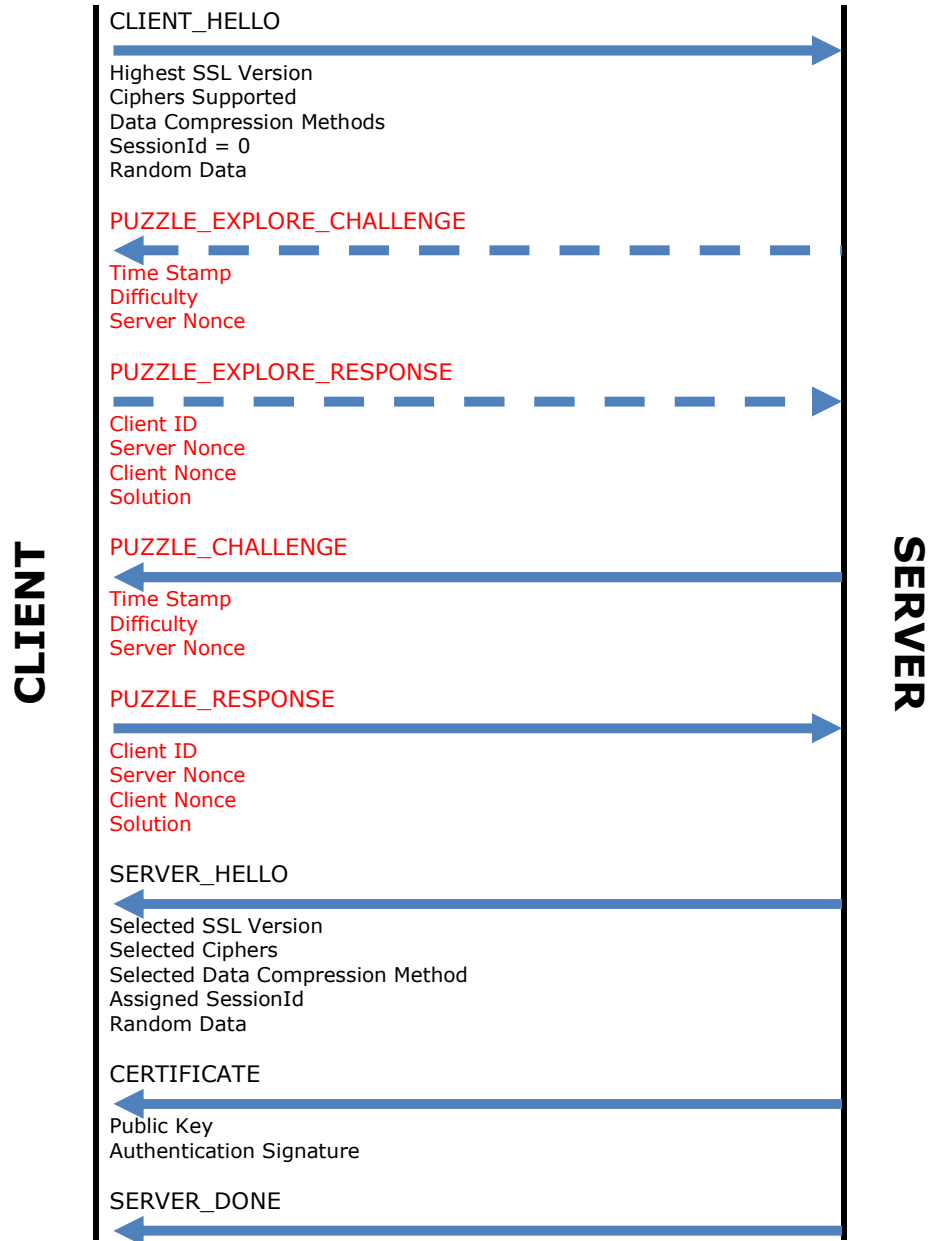


Fig. 17 Protocolul SSL Handshake cu distribuție adaptivă de efort

La fel ca în cazul anterior, adăugarea mesajelor suplimentare nu schimbă integritatea criptografică a protocolului SSL ci are ca scop ameliorarea disponibilității serverului prin întârzierea clientului cu o durată de timp proporțională cu încărcarea curentă a serverului. Schimbările la nivel criptografic ar presupune o abordare diferită și deosebit de laborioasă pentru a evita introducerea de noi vulnerabilități.

Spunem că distribuția efortului este adaptivă deoarece întârzierea clientului se face în funcție de puterea de calcul pe care o are.

### 3.2.4 Prototipul software

Pentru a simula comportarea tehnologiei Threshold Puzzles într-un context real și pentru a evidenția avantajele acesteia față de tehnologia Client Puzzle propusă în literatură, s-a realizat un prototip cu structura indicată în Fig. 18.

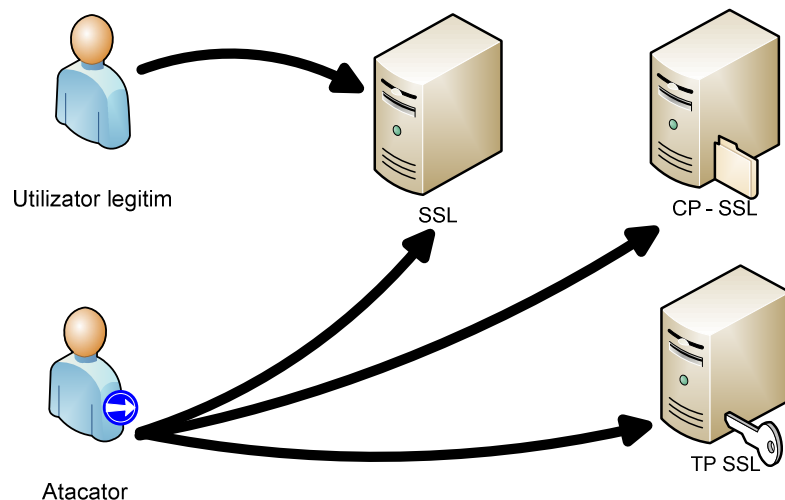


Fig. 18 Schema prototipului Threshold Puzzles

Prototipul conține următoarele module:

- **Utilizator legitim** – un client obișnuit, care dorește efectuarea unei autentificări SSL și urmează execuția protocolului conform cu specificația.

- **Atacator** – un client cu acces la o putere de calcul mare și care efectuează un număr mare de cereri false de autentificare într-un timp scurt.
- **Server SSL** – Un server SSL obișnuit și neprotejat.
- **Server CP SSL** – Un server SSL protejat cu tehnologia *Client Puzzles*.
- **Server TP SSL** – Un server SSL protejat cu tehnologia *Threshold Puzzles*.

Întrucât în condiții de laborator accesul la o putere de calcul semnificativă este dificil, pentru a simula acest lucru s-a convenit ca modulul reprezentând atacatorul să emită o soluție aleatoare – și evident incorectă – urmând ca verificarea pe partea serverului să nu se efectueze. În acest fel atacatorul pare că rezolvă puzzle-ul într-un timp considerabil mai mic decât clienții legitimi. Această modificare nu este în măsură să afecteze în mod semnificativ rezultatele experimentului.

Simularea s-a efectuat pe două sisteme de calcul legate într-o rețea de 100 Mb/s prin care nu a existat trafic perturbator în timpul desfășurării simulării. Pe sistemul reprezentând serverul au fost instalate modulele *Server SSL*, *Server CP SSL* și *Server TP SSL*. Pe sistemul reprezentând clientul au fost instalate modulele *Utilizator legitim* și *Atacator* apoi s-au lansat instanțe multiple ale acestora pentru a simula existența mai multor clienți.

### 3.2.5 Performanță

Pe sistemul de test au fost rulate mai multe scenarii menite să determine comportarea tehnologiilor Client Puzzles și Threshold Puzzles. Aceste scenarii sunt descrise în cele ce urmează:

#### 3.2.5.1 Clienți autentificați de server normal

Pentru a avea un etalon în măsurarea performanței, s-a rulat sistemul în scenariul clasic, adică clienți obișnuiți autentificați de un server SSL normal. În acest fel s-a putut determina timpul mediu necesar pentru execuția completă a protocolului SSL, fără adăugirile specifice Client/Threshold Puzzles.

S-a efectuat un număr de 100 de autentificări a 15 clienți, timpul mediu de răspuns al serverului fiind de 4545 ms.

#### 3.2.5.2 Clienți și atacator autentificați de server normal

În această simulare s-au folosit 14 clienți normali și un atacator. Scopul atacatorului este de a trimite o rafală de mesaje CLIENT\_HELLO către server în scopul de a declanșa un număr egal de răspunsuri semnate de server. Execuția

protocolului nu este finalizată niciodată de către atacator, iar răspunsul serverului este ignorat.

Atacatorul a trimis cereri false de autentificare cu o rată de 30 pe secundă, timp în care încărcarea pe server a fost de 100%. Timpul mediu de autentificare a clienților legitimi a fost de 11320 ms.

#### 3.2.5.3 Clienți și atacator autentificați de server cu Client Puzzles

Această simulare este asemănătoare cu cea anterioară, cu excepția serverului căruia i s-a activat tehnologia Client Puzzles. Protocolului SSL i s-au adăugat practic două mesaje (numite PUZZLE\_CHALLENGE respectiv PUZZLE\_RESPONSE) așa cum s-a descris în paragrafele anterioare, în speranța reducerii numărului de cereri venite pe unitatea de timp de la un client.

Deoarece clientul are acces la o putere de calcul semnificativă, fapt simulat prin generarea unui rezultat aleatoriu (evident, incorect) și lipsa verificării corectitudinii sale pe partea de server, serverul a fost inundat cu cereri de autentificare false, cu aceeași rată de 30 pe secundă ca în cazul anterior. În tot acest timp încărcarea pe server a fost de 100% iar timpul mediu de autentificare a clienților legitimi a fost de 12730 ms. Timpul ușor mai mare decât în cazul precedent indică un overhead mai mare din cauza celor două mesaje suplimentare din protocol.

#### 3.2.5.4 Clienți și atacator autentificați de server cu Threshold Puzzles

Simularea anterioară a fost reluată modificând configurația serverului SSL să folosească tehnologia Threshold Puzzles. În acest caz, timpul mediu de autentificare al clienților legitimi a fost de 4553 ms, aproximativ egal cu cazul în care serverul nu se află sub atac. Acest rezultat se datorează faptului că atacatorul a rezolvat mult prea repede și în mod repetat puzzle-urile solicitate de server, ceea ce a dus automat la blocarea comunicației cu clientul atacator.

#### 3.2.5.5 Concluzie

În timpul unui așa-numit atac puternic, tehnologia Client Puzzles nu a avut efectul scontat deoarece atacatorul este capabil a rezolva puzzle-urile într-un timp foarte scurt și poate încă trimite un număr considerabil de cereri false serverului. Timpul mediu de execuție al protocolului a fost similar cu cel din cazul unui server neprotejat. Un server protejat cu Threshold Puzzles însă a refuzat sistematic cererile însoțite de soluții sosite prea rapid pentru nivelul de dificultate ales, astfel încât timpul de autentificare perceput de clienți a rămas practic același cu cel din condiții normale.

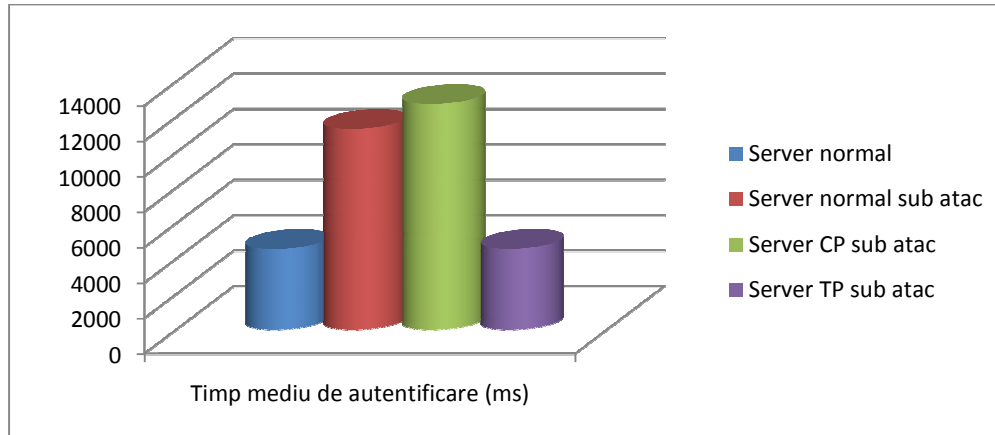


Fig. 19 Comparație a rezultatelor simulărilor pe protocolul SSL

### 3.3 Detectarea și ameliorarea locală a atacurilor DoS

Pentru ca măsurile luate de sistemele de autentificare în vederea ameliorării rezistenței la atacuri DoS să fie eficiente, este necesar ca acestea să cunoască dacă încărcarea sistemului este datorată unui factor temporar și tranzitoriu, unui vârf de sarcină sau într-adevăr, există un atac în desfășurare. Desfășurarea de măsuri de apărare în timpul unor situații tranzitorii introduce de cele mai multe ori o penalitate percepută de clienți ca o degradare a performanței, iar situația este cu atât mai neplăcută cu cât acestea apar mai des.

Datorită naturii complexe a sistemelor de autentificare și datorită multitudinii de configurații în care acestea pot activa, detectarea atacurilor este o problemă relativ dificilă în lipsa unor metrice standard de evaluare a riscului. În cadrul paragrafului de față ne propunem analiza unui sistem de autentificare de tip Single Sign-On, și propunerea unei metrici și unui algoritm de estimare a riscului. Am ales pentru exemplificare suita de protocoale Liberty [55][56].

Sistemele de autentificare de tip Single Sign-On (SSO) în general și implementarea în suita Liberty în particular sunt vulnerabile la atacuri DoS. Cerința ca fiecare mesaj între entitățile sistemului să fie semnat permite unui atacator să inunde rețeaua cu mesaje false pe care entitățile se vor grăbi să le decodifice, doar ca să constate ulterior că mesajul provine de la un client necunoscut, semnătura este invalidă și resursele consumate pentru verificarea semnăturii s-au pierdut inutil. În Fig. 20 se arată mecanismul de principiu al unui atac DoS îndreptat asupra sistemelor SSO. În cazul unui atac, clienții legitimi ai furnizorului de identitate aflat

sub atac vor constata degradarea timpului de răspuns la solicitările lor datorită încărcării artificiale pe furnizorul de identitate [13].

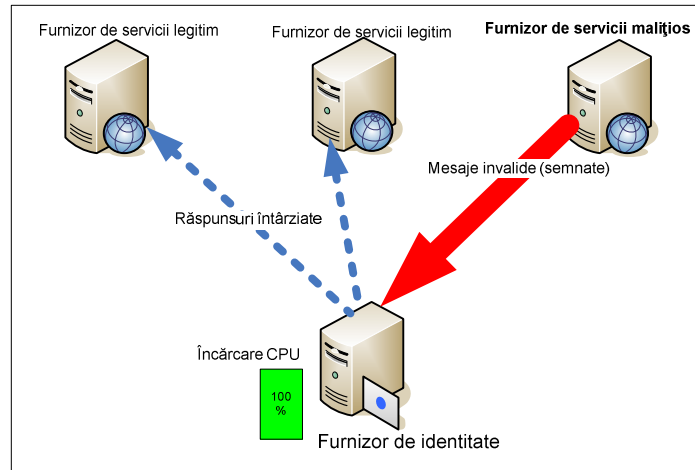


Fig. 20 Atac DoS asupra sistemelor SSO (în particular implementarea Liberty)

Simpla semnare a fiecărui mesaj vehiculat reprezintă în sine un atac DoS. Deși uneori sistemele SSO sunt guvernate și de înțelegeri legale, acest lucru nu este suficient pentru a proteja utilizatorii legitimi de atacuri, fiind nevoie de adoptarea unor măsuri de natură tehnică. Dacă în paragrafele anterioare s-a discutat despre oportunitatea adoptării unor măsuri care să încarce clienții proporțional cu serverul (vezi tehnologiile Threshold Puzzles și Adaptive Threshold puzzles), acum vom vorbi despre metode prin care sistemul de autentificare își dă seama de un atac în desfășurare, fiind capabil să se adapteze situațiilor existente în diverse scenarii folosind inferența bayesiană [23], tehnică folosită și la filtrele spam din sistemele de poștă electronică.

Inferența bayesiană este o alegere potrivită pentru evaluarea stării unui sistem de autentificare sau a unui sistem SSO. Starea sistemului se evaluează și se estimează pe baza unui set de senzori software [79] plasați în anumite puncte din sistem, iar informațiile culese de aceștia sunt agregate la un nivel central care determină gradul de probabilitate ca sistemul supervizat să fie sub atac. În funcție de acest grad de probabilitate se poate lua măsura de a varia plaja de dificultate pentru puzzle.

### 3.3.1 Caracteristici măsurabile

Serverul care asigură serviciul de autentificare trebuie să aibă o măsură exactă a încărcării momentane sau din viitorul apropiat. Întrucât în literatura de specialitate nu am găsit caracteristici măsurabile ale protocoalelor Liberty, am

enumerat cele mai importante caracteristici cu potențial asupra sistemului hardware. Caracteristicile se folosesc ca date de intrare pentru motorul de inferență bayesiană care la rândul său va determina nivelul de siguranță al sistemului SSO.

Caracteristicile măsurabile ale protocoalelor Liberty pot fi diferențiate în mai multe categorii descrise în continuare, în funcție de domeniul vizat.

### 3.3.1.1 Disciplina de execuție a protocolului

O primă caracteristică măsurabilă din această categorie se referă la ordinea de execuție a protocoalelor din suita Liberty. Execuția trebuie să urmeze o ordine aproximativă, un deranj al acestei ordini putând fi semnul unui posibil atac asupra sistemului SSO. Spre exemplu, o cerere *Single Logout* nu ar trebui primită de un furnizor de servicii sau de identitate dacă cererea *Single Sign-On and Federation* a fost primită anterior, cum se vede în Fig. 21 [13].

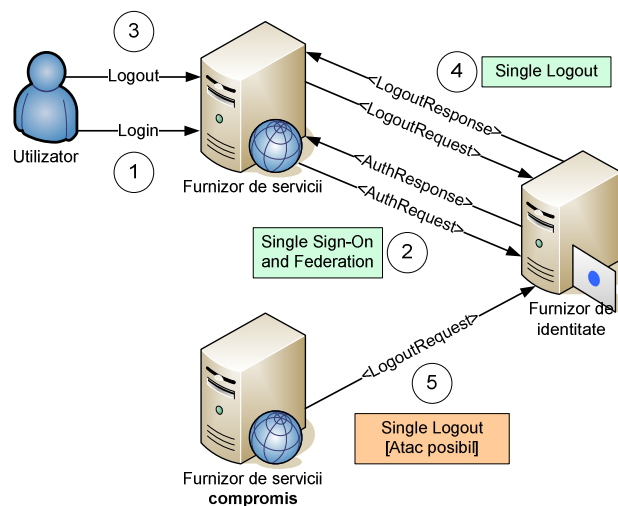


Fig. 21 Detectarea unui posibil atac asupra sistemelor SSO folosind ordinea de execuție a protocolului.

Conținutul cererilor este o altă caracteristică de luat în considerare. Pentru fiecare protocol, cererea are un set de atribute care specifică cum ar trebui procesată aceasta de către entitatea receptoare. O variație largă a acestor atribute din partea unui client anume indică faptul că acesta este neglijent în formularea mesajelor introducând probabil date aleatoare, fapt ce indică un atac împotriva serverului. Spre exemplu, mesajul *AuthRequest* din cadrul protocolului *Single Sign-On and Federation* are un atribut care cere furnizorului de identitate să emită un identificator anonim și temporar în numele clientului, când se schimbă datele între

furnizorii de servicii. Prea multe mesaje *AuthRequest* cu acest atribut setat, recepționate de la același client, ar putea indica un posibil atac DoS asupra furnizorului de identitate.

#### 3.3.1.2 Sănătatea rețelei

Caracteristicile legate de rețea fac parte din această categorie și constau în principal din parametrii traficului pe partea de recepție [5]:

- **Anomalii în operarea rețelei** – aceste tipuri de anomalii includ diferențele semnificative în comportarea rețelei sau funcționare la limita fizică a hardware-ului. Anomaliile se pot distinge printr-o creștere bruscă, aproape verticală a valorilor măsurate pe o perioadă de timp.
- **Anomalii Flash Crowd** – acest tip de anomalie se datorează comportamentului sinergic al unui grup de utilizatori, spre exemplu ca urmare a lansării unui produs software nou. De obicei anomaliile de acest fel se manifestă în puncte bine stabilite din rețea cum ar fi locațiile de download sau mirror-urile.
- **Anomalii de abuz** – sunt datorate fie unor atacuri DoS în desfășurare (prin inundarea rețelei) sau datorită unei activități de scanare a porturilor. Anomalia se deosebește de operarea normală a rețelei și de anomalia Flash Crowd, dar nu întotdeauna este evidentă. Totuși, prin detectarea și măsurarea unor modele de trafic această anomalie poate fi descoperită.

Un senzor software plasat în sistem trebuie să monitorizeze anomaliile din rețea și să furnizeze aceste informații în formă brută sau semi-agregată motorului de detecție a atacurilor. Anumiți parametri (cum ar fi timpul dintre două cereri consecutive, mărimea cererii, numărul de cereri pe secundă de la un client) au fost deja folosiți cu succes în detectarea atacurilor DoS [79][49], însă nu au fost agregați într-un sistem complex de evaluare a stării de sănătate a unui server de autentificare.

#### 3.3.1.3 Sănătatea sistemului gazdă

Nivelul curent de sănătate al sistemului gazdă este un indiciu crucial asupra calității serviciului furnizat. În condiții normale de funcționare și încărcare, sistemul poate oferi servicii cu un factor QoS stabilit prin design. Alterarea condițiilor de natură software sau hardware are ca efect modificarea QoS cu consecințe asupra clienților care constată modificări în calitatea serviciului furnizat de sistem, cum ar fi: întârzieri în furnizarea răspunsului, răspuns cu intermitență, lipsa totală a răspunsului, etc.

Sănătatea sistemului gazdă depinde de factori cum ar fi:



- Cantitatea de memorie disponibilă
- Încărcarea curentă a procesorului, în scenarii uniprocessor
- Încărcarea curentă a procesorului pentru care serviciul are afinitate, în scenarii multiprocessor
- Activitatea I/O a discurilor
- Rezerva de resurse alocate mașinii virtuale, pentru scenariile virtuale
- Starea și nivelul de activitate al driverelor de sistem, servicii, daemons, etc.

### 3.4 Detecția euristică a atacurilor cu SSO-SENSE

Informațiile colectate de senzorii plasați în diverse puncte ale sistemului monitorizat sunt disponibile la un moment dat unui modul software specializat. Acesta poate agrega informațiile în așa fel încât să tragă o concluzie legată de starea momentană a sistemului, în scopul luării măsurilor de apărare împotriva atacurilor sau de limitare ale consecințelor acestora.

Având ca bază teoria inferenței bayesiene, am creat un modul de estimare a nivelului de risc, numit SSO-SENSE. Schema sa de principiu este indicată în Fig. 22.

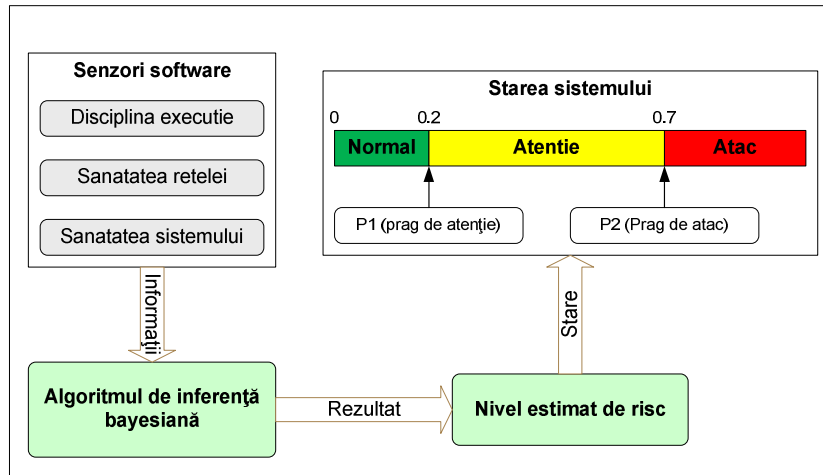


Fig. 22 Modul de estimare a nivelului de risc SSO-SENSE

Senzorii software sunt responsabili cu colectarea informațiilor din sistemul supravegheat. Acei senzori care se ocupă de disciplina de execuție a protocolului trebuie implementați direct în cadrul protocoalelor SSO, în așa fel încât să aibă în mod nemijlocit acces la schimbul de mesaje dintre părți. Senzorii responsabili de sănătatea rețelei și a sistemului pot utiliza indicatorii de performanță existenți în

cadru sistemului de operare, în cazul Windows putându-se utiliza *Windows Management Instrumentation* (WMI).

Informațiile colectate de senzori sunt agregate de algoritmul de inferență bayesiană, din agregare rezultând un nivel estimat de risc. Acest nivel este un număr real între 0 și 1. Valoarea 0 indică funcționarea normală a sistemului cu un grad ridicat de certitudine. O valoare care se apropie de 1 indică prezența unui număr de anomalii în cadrul sistemului, ale căror ponderi pot indica un atac în desfășurare.

Nivelurile pentru pragul de atenție respectiv cel de atac se aleg astfel încât  $0 < P_1 < P_2 < 1$ , cu  $P_1$  și  $P_2$  spațiate în cadrul intervalului astfel încât să asigure o delimitare eficientă a acțiunilor care se petrec în starea de atenție respectiv starea de atac.

#### 3.4.1 Scurtă fundamentare matematică

Pentru a înțelege modalitatea de agregare a informațiilor primite de la senzori, în acest paragraf se face o scurtă fundamentare matematică a teoriei inferenței bayesiene.

Considerăm un set de trei ipoteze reciproc exclusive  $H_1$ ,  $H_2$  și  $H_3$  corespunzătoare stărilor în care se poate afla sistemul – Normal, Atenție, Atac. De asemenea, considerăm un eveniment  $E$  care apare când un senzor software detectează o condiție de prag între caracteristicile măsurabile ale protocolului de autentificare.

Bazându-ne pe considerentele anterioare, se poate calcula *constantă de normalizare*  $\Lambda$  astfel:

$$\Lambda = P(E | H_1) \cdot P(H_1) + P(E | H_2) \cdot P(H_2) + P(E | H_3) \cdot P(H_3)$$

Ec. (3)

Inițial se consideră cu un grad mare de siguranță că sistemul nu este sub atac, putem deci asocia câte o probabilitate pentru fiecare dintre stări:

$$P(H_1) = 0.9 \text{ (sistem în stare normală)}$$

$$P(H_2) = 0.09 \text{ (sistem în stare de atenție)}$$

$$P(H_3) = 0.01 \text{ (sistem sub atac).}$$

La apariția unui eveniment  $E$ , evaluăm probabilitățile posterioare ale celor trei ipoteze, astfel:

$$P(H_1 | E) = \frac{P(E | H_1) \cdot P(H_1)}{\Lambda}$$

Ec. (4)

$$P(H_2 | E) = \frac{P(E | H_2) \cdot P(H_2)}{\Lambda}$$

Ec. (5)

$$P(H_3 | E) = \frac{P(E | H_3) \cdot P(H_3)}{\Lambda}$$

Ec. (6)

Cu fiecare eveniment E care se produce în sistem, probabilitățile posterioare ale ipotezelor  $H_1$ ,  $H_2$  și  $H_3$  se modifică. Starea momentană a sistemului după producerea evenimentului E este:

$$S = \text{MAX} [ P(H_1 | E), P(H_2 | E), P(H_3 | E) ]$$

Ec. (7)

Evenimentul E poate fi:

- Încărcarea disciplinei de execuție a protocolului de autentificare sau SSO, inversiunea de mesaje, mesaje lipsă sau cu parametri incompleți.
- Timpul între două cereri succesive.
- Eroare de autentificare.
- Creștere bruscă a încărcării pe CPU pentru o perioadă scurtă de timp.
- Creștere bruscă a traficului pe rețea, exceptând anomalii flash crowd.
- Încercarea de acces la porturi successive (scanare).

#### 3.4.2 Aplicarea detecției euristice în autentificare

Diversele metode de apărare împotriva atacurilor DoS aplicate sistemelor de autentificare își pot îmbunătăți comportamentul și deciziile prin informațiile suplimentare oferite de sistemele de detecție ale atacurilor. În timp ce un sistem complex de detecție a intrușilor aplicat la nivelul global al rețelei poate ajuta într-o manieră generală, detecția atacurilor la nivelul vectorului de autentificare este o problemă specifică. În acest caz atacatorul nu urmărește pătrunderea în sistem și accesul la date confidențiale ci compromiterea în sine a autentificării și împiedicarea accesului clienților legitimi, fapt ce nu interesează sistemele de *intrusion detection* existente actual.

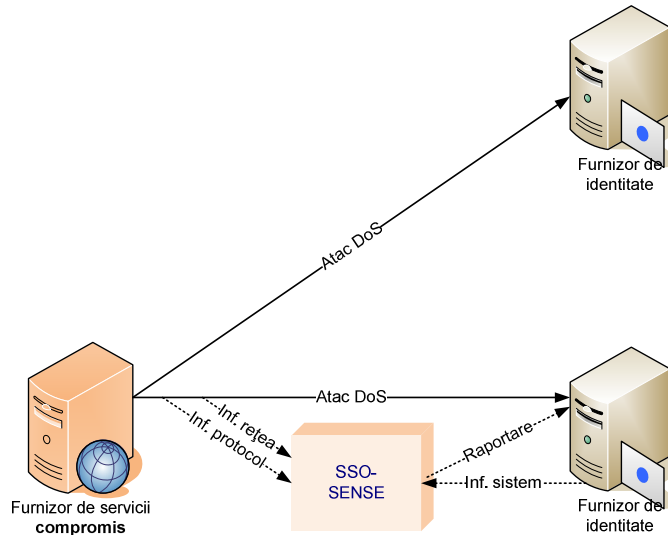


Fig. 23 Folosirea detecției euristice la nivelul vectorului de autentificare

În Fig. 23 un furnizor de servicii compromis din cadrul unui sistem SSO lansează un atac asupra a doi furnizori de identitate. Ambii furnizori sunt protejați împotriva atacurilor de tip DoS, cu diferența că unul beneficiază de un sistem de detecție euristică ce îi permite să ia măsuri mai eficiente și posibil cu impact inferior la nivelul clienților legitimi.

Conform Ec. 1, nivelul de dificultate  $k$  al unui threshold puzzle se calculează astfel:

$$k_{\max} = \log_2 (\max(1, Q * t_s / t_c)) + 1$$

În lipsa informațiilor despre iminența unui atac, dificultatea puzzle-ului poate fi aleasă proporțional cu lungimea cozii cererilor sau poate urma o regulă de creștere dinainte stabilită. Dacă informația cu privire la starea probabilă a sistemului este disponibilă, regula de variație a parametrului  $k$  poate fi:

- **Stare normală** – serverul nu ajunge la încărcarea maximă decât în puține momente de vârf. Este nevoie preponderant de puzzle-uri cu complexitate redusă, care se poate alege în mod neuniform din plaja  $[0; k_{\max}]$ .
- **Stare de atenție** – încărcarea pe server depășește constant un prag de confort, fără ca acest lucru să indice automat un atac în desfășurare. Este nevoie preponderant de puzzle-uri de complexitate medie din plaja  $[0; k_{\max}]$ .

- **Stare de atac** – există indicii că un atac este în desfășurare. Pe durata acestei stări serverul poate lua decizia ca nivelul de dificultate  $k_{max}$  să fie menținut, chiar dacă serverul nu a ajuns la încărcarea maximă.

### 3.4.3 Rezultate experimentale

Prototipul modulului de estimare a nivelului de risc SSO-SENSE a fost implementat în limbajul C#, pe platforma .NET Framework 3.5. Pentru a determina comportamentul modulului de inferență bayesiană, scenariul de test a constat în simularea a două situații care pot fi întâlnite în mod real:

- Autentificări eșuate repetate – sistemul este bombardat cu cereri de autentificare false, despre care atacatorul știe că vor eșua, cu scopul de a încălca serverul.
- Neurmarea cursului firesc al protocolului – aceasta urmărește destabilizarea serverului prin răspunsuri nepotrivite cu specificația, lucru care lasă serverul în așteptare până la un eventual timeout.

La simulare s-a convenit ca starea inițială a sistemului să fie cea normală, încărcarea curentă pe server  $Q = 400000$ , cu  $t_s = 0,003$  și  $t_c = 0,5$ . Dificultatea maximă a puzzle-ului este astfel  $k_{max} = 12$ .

Autentificările eșuate sunt evenimente cu gravitate redusă, de aceea tranziția de la starea normală la starea de atac se face trecând prin mai multe stări de atenție. Nivelul de dificultate determinat de mecanismul puzzle  $k_{max}$  este coroborat cu starea curentă a sistemului. Simularea unui șir de autentificări eșuate duce la următoarea ieșire a prototipului:

```
Status before Authentication Failure: Normal
Status after failure no. 1: Normal (k = 1)
Status after failure no. 2: Warning (k = 4)
Status after failure no. 3: Warning (k = 5)
Status after failure no. 4: Warning (k = 4)
Status after failure no. 5: Warning (k = 4)
Status after failure no. 6: Attack (k = 12)
```

Se observă că după primul eveniment de autentificare eșuată, sistemul rămâne în starea normală, deoarece acest tip de evenimente pot fi provocate în mod curent de utilizatori, fără că acest lucru să însemne neapărat un semn de atac. Următoarele evenimente însă provoacă trecerea sistemului în starea de atenție și în cele din urmă în starea de atac, semnalizând administratorului gravitatea situației.

Evenimentele de tip eroare de protocol sunt mai grave (lucru stabilit inițial prin probabilitățile asociate ipotezelor) și deci implicit convergența sistemului către starea de atac este mai rapidă. Rezultatul rulării în condițiile a 6 evenimente succesive de acest tip este dat mai jos:

```
Status before Protocol Followup Failure: Normal
Status after failure no. 1: Warning (k = 6)
Status after failure no. 2: Warning (k = 4)
Status after failure no. 3: Warning (k = 5)
Status after failure no. 4: Attack (k = 12)
Status after failure no. 5: Attack (k = 12)
Status after failure no. 6: Attack (k = 12)
```

### 3.5 Concluzii

Una dintre soluțiile propuse în literatură pentru ameliorarea rezistenței la atacuri DoS a protocoalelor de autentificare a fost folosirea tehnologiei client puzzles cu scopul de a crește artificial costul de execuție al protocolului pe client, proporțional cu creșterea încărcării pe server în scopul realizării unei bucle de autoreglare. Cu toate acestea însă, datorită naturii paralelizabile mecanismul client puzzles poate fi învins în anumite condiții de performanță ale atacatorului.

Pentru a evita situațiile în care un atacator reușește să rezolve puzzle-urile într-un timp suficient de scurt pentru a fi capabil de a trimite o rafală de cereri către server, în esență crescând nivelul de dificultate al puzzle-ului și blocând astfel clienții legitimi cu calcule inutile, am adus două modificări de design, grupate sub denumirea „Threshold Puzzles”, după cum urmează:

- Limitarea superioară a nivelului de dificultate al puzzle-ului – pentru ca un eventual atac puternic să nu aibă ca rezultat indirect un atac DoS asupra clienților înșiși.
- Stabilirea unui timp minim de răspuns – estimarea timpului minim necesar unui client mediu de a rezolva puzzle-ul.

Pentru scenariile în care clienții unui serviciu sunt eterogeni, având puteri de calcul ce variază în limite largi, tehnologia a fost și mai mult rafinată prin adaptarea încărcării artificiale a clientului în funcție de puterea sa reală de calcul. Acest lucru a pornit de la observația că un PDA sau Smartphone au puteri de calcul mult mai mici decât un laptop sau un desktop. În condiții de încărcare mare a serverului, când valoarea dificultății puzzle-ului este mare, clienții cu putere de calcul redusă vor întâmpina dificultăți în rezolvare, ceea ce este o degradare a experienței utilizatorului. Astfel, am creat conceptul de „Adaptive Threshold Puzzles” care – așa cum îi spune și numele oferă spre rezolvare clienților puzzle-uri cu dificultăți personalizate fiecărui client în parte.

În urma experimentelor realizate cu prototipul implementat am demonstrat că un server protejat cu Threshold Puzzles se comportă în termeni comparabili cu un server în lipsa atacului.

Măsurile de apărare împotriva atacurilor DoS pe care un sistem le are la dispoziție sunt eficace atunci când acesta cunoaște faptul că un atac este în desfășurare. Pentru a determina acest lucru nu este suficientă o metrică simplă, care să implice doar câțiva parametri sistem cum ar fi încărcarea memoriei și a procesorului, deoarece în acest fel am avea de-a face cu numeroase alarme false. Este nevoie de un modul care să analizeze informațiile disponibile la un moment determinat și să ia decizii în consecință, evitând catalogarea evenimentelor tranzitorii drept atacuri. Pentru aceasta am propus folosirea inferenței bayesiene ca bază pentru un modul de detecție a nivelului de risc numit SSO-SENSE.

Contribuțiile aduse de acest capitol pot fi sumarizate după cum urmează:

- Au fost dezvoltate tehnologiile Threshold Puzzles și Adaptive Threshold Puzzles, pentru a acoperi situațiile în care tehnologia Client Puzzles nu este eficientă.
- S-a arătat practic rezistența la atacuri DoS a protocolului SSL, căruia i s-a adaptat tehnologia Threshold Puzzles.
- S-a evidențiat posibilitatea ca atacurile DoS să fie detectate euristic prin inferență bayesiană, ca metodă care vizează vectorul de autentificare al unui sistem.
- Nivelul de dificultate al unui puzzle poate fi ales optim în funcție de starea sistemului de autentificare determinată euristic, adică niveluri de dificultate preponderent reduse pentru un sistem neatacat și niveluri preponderent ridicate sau nivelul maxim pentru un sistem aflat sub atac.





## 4. Contribuții la creșterea disponibilității rețelelor tip GSM

Suntem cvasidependenți de comunicație. Fie că e vorba de mediul personal, cel de afaceri, medical sau de altă natură, comunicația joacă un rol însemnat în viața de zi cu zi a societății. Comunicațiile critice – adică cele care au loc în urgențele medicale, în situații de criză sau calamități naturale – au loc în majoritatea cazurilor pe o infrastructură care nu a fost concepută pentru disponibilitate crescută sau redundanță. Un eventual atac asupra infrastructurii de comunicație – mobilă sau fixă – poate avea un impact nedorit. Deoarece cel mai răspândit standard de comunicație mobilă este GSM, în acest capitol se va aborda problema disponibilității sale, problemele pe care le are design-ul actual precum și modalitățile prin care se pot ameliora efectele atacurilor DoS. Deși tehnologia GSM are o vârstă și tinde să fie înlocuită treptat de 3G, WiMax sau LTE, subiectul este de actualitate întrucât noile tehnologii vor asigura acoperire doar local, în timp ce suprafețele mari de teritoriu sunt acoperite de GSM.

Problemele de securitate ale GSM sunt numeroase, însă o încercare de a le rezolva pe toate în același timp ar fi sortită eșecului. Se impune astfel o clasificare a vulnerabilităților după anumite criterii din care să rezulte o prioritizare în acțiune. În capitolul de față se face o astfel de analiză pentru ca mai apoi să se prezinte soluții la cea mai gravă problemă cu care se confruntă tehnologia GSM.

### 4.1 Atacuri asupra rețelelor GSM

Atacurile asupra unui sistem de comunicații cum este GSM sunt numeroase, complexe și nu întotdeauna evidente. Multitudinea de factori care influențează comunicația, complexitatea modelelor implicate în tehnologie, factorul uman, factorul social, sunt lucruri care afectează într-un fel sau altul siguranța comunicației. Literatura menționează un număr de atacuri posibile asupra sistemelor GSM [38], însă nu indică gravitatea și probabilitatea acestor atacuri. Simpla apreciere subiectivă a gravității nu este o cale precisă, de aceea este necesar un oarecare grad de formalism pentru determinarea celor mai grave și posibile atacuri.

În [16] s-a propus aplicarea unei metodologii de clasificare a atacurilor GSM, metodologie derivată din software și care poartă numele DREAD. Aceasta a fost introdusă de Howard și LeBlanc în [46].

#### 4.1.1 Clasificarea DREAD a vulnerabilităților

Nivelul de risc al unei vulnerabilități, fie la nivel software fie la nivel de infrastructură de comunicații, se poate determina formal având în vedere anumiți factori, după cum urmează:

- **Potențialul de distrugere** – Cât de mare este pierderea dacă vulnerabilitatea este exploatată? Se măsoară distrugerea maximă posibilă și se acordă un scor de la 1 la 10. Scorul 10 este o amenințare care permite atacatorului să ocolească toate mecanismele de protecție și să acționeze în voie în sistemul compromis.
- **Reproductibilitatea** – Cât de ușor se poate reproduce atacul? Se măsoară ușurința cu care o amenințare devine exploatabilă, pe o scară de la 1 la 10. O reproductibilitate înaltă este în interesul oricărui atacator.
- **Exploatabilitatea** – Cât de ușor este ca atacul să fie lansat? Spre exemplu, dacă un programator începător cu un PC acasă poate iniția atacul, scorul acestui factor ar fi clar 10. Dacă însă atacul ar necesita o agenție guvernamentală care ar trebui să investească o sută de milioane de dolari, scorul acestui factor ar trebui să fie 1. De asemenea, se va estima ce nivel de autentificare și autorizare sunt necesare pentru a ataca sistemul. Spre exemplu, dacă un utilizator anonim de la distanță poate ataca sistemul, scorul ar fi 10, spre deosebire de un utilizator local căruia i se cer credențiale puternice, caz în care scorul ar fi mult mai mic.
- **Utilizatori afectați** – Câți utilizatori sunt afectați? Aceasta înseamnă aproximativ procentajul de utilizatori care ar fi impactați de un atac: 91-100% (scor 10), până la 0-10% (scor 1). Trebuie de asemenea gândit și în termeni de număr absolut de utilizatori. Un singur procent din 100 de milioane de utilizatori înseamnă totuși o mulțime de utilizatori.
- **Ușurința în descoperire** – Cât de ușor se poate descoperi vulnerabilitatea? Aceasta este probabil cea mai dificil de determinat metrică și se va presupune întotdeauna ca având valoarea 10, deoarece mai devreme sau mai târziu o vulnerabilitate va fi descoperită și făcută publică.

#### 4.1.2 Clasificarea vulnerabilităților rețelelor GSM după sistemul DREAD

În literatura de specialitate nu există o clasificare a vulnerabilităților GSM după gravitate. În lucrări precum [38] se pune accent doar pe aspectele tehnice ale atacurilor, fără a ține seama de experiența tehnică a atacatorului și resursele necesare pentru a-l pune în practică. Lucrarea de față însă abordează într-o manieră comparativă vulnerabilitățile din GSM și le clasifică în funcție de probabilitatea de exploatare. Această abordare permite prioritizarea măsurilor de ameliorare a securității.

Evaluarea scorului de risc s-a făcut după cum urmează: pentru fiecare vulnerabilitate în parte s-a acordat câte un punctaj de la 1 la 10 fiecărui factor din

sistemul DREAD (potențial de distrugere, reproductibilitate, exploatabilitate, utilizatori afectați, ușurința în descoperire). Media aritmetică a celor 5 punctaje dă scorul de risc.

#### 4.1.2.1 Denial of Service

Interfața radio GSM este vulnerabilă la atacuri DoS deoarece resurse prețioase cum sunt canalele de semnalizare sunt oferite oricui le solicită. Inundarea canalelor de semnalizare cu cereri legitime sau false înseamnă în esență că traficul este paralizat. Inundarea canalului de semnalizare poate fi cauzată de un atacator cu o stație mobilă modificată [14] sau de cereri legitime [34].

| Factor                             | Scor de risc |
|------------------------------------|--------------|
| Potențial de distrugere            | 5            |
| Reproductibilitate                 | 10           |
| Exploatabilitate                   | 7            |
| Utilizatori afectați               | 9            |
| Ușurință în descoperire            | 10           |
| Factor de risc "Denial of Service" | 8.2          |

#### 4.1.2.2 De-registration Spoofing

Un atacator poate falsifica o cerere de de-înregistrare către rețea (IMSI detach). Aceasta înseamnă că utilizatorul este detașat din aria vizitată și astfel toate serviciile terminate la mobil vor fi afectate.

| Factor                                    | Scor de risc |
|---|--------------|
| Potențial de distrugere                   | 3            |
| Reproductibilitate                        | 10           |
| Exploatabilitate                          | 5            |
| Utilizatori afectați                      | 1            |
| Ușurință în descoperire                   | 10           |
| Factor de risc "De-registration Spoofing" | 5.8          |

#### 4.1.2.3 Location Update Spoofing

Acest atac este similar cu cel anterior. Un atacator trimite o cerere falsă de actualizare a locației către o zonă diferită de cea în care utilizatorul se află. Ca și în cazul precedent, serviciile terminate la mobil vor fi afectate.

| Factor                                    | Scor de risc |
|---|--------------|
| Potențial de distrugere                   | 3            |
| Reproductibilitate                        | 10           |
| Exploatabilitate                          | 5            |
| Utilizatori afectați                      | 1            |
| Ușurință în descoperire                   | 10           |
| Factor de risc "Location Update Spoofing" | 5.8          |

#### 4.1.2.4 Camping on a False BTS

Terminalul mobil poate fi „momit” să se campeze pe un BTS fals, făcându-l astfel inaccesibil semnalelor de apel ale rețelei mobile. Alternativ, BTS-ul fals poate acționa ca un intermediar și ar putea permite traficul după bunul plac. Acest atac necesită un BTS modificat.

| Factor                                  | Scor de risc |
|---|--------------|
| Potențial de distrugere                 | 3            |
| Reproductibilitate                      | 10           |
| Exploatabilitate                        | 4            |
| Utilizatori afectați                    | 1            |
| Ușurință în descoperire                 | 10           |
| Factor de risc "Camping on a False BTS" | 5.6          |

#### 4.1.2.5 Passive Identity Caching

În anumite condiții, rețeaua poate cere utilizatorului să își decline identitatea într-o manieră clară, necriptată. Un terminal mobil modificat poate fi folosit pentru a memora informația pentru utilizări ulterioare.

| Factor                                    | Scor de risc |
|---|--------------|
| Potențial de distrugere                   | 2            |
| Reproductibilitate                        | 8            |
| Exploatabilitate                          | 5            |
| Utilizatori afectați                      | 1            |
| Ușurință în descoperire                   | 10           |
| Factor de risc "Passive Identity Caching" | 5.2          |

#### 4.1.2.6 Active Identity Caching

Acest atac este similar cu cel anterior, cu excepția faptului că utilizatorul este „momit” să se campeze pe un BTS fals care la rândul său va cere în permanență ca identitatea să fie vehiculată în clar.

| Factor                                   | Scor de risc |
|--|--------------|
| Potențial de distrugere                  | 2            |
| Reproductibilitate                       | 8            |
| Exploatabilitate                         | 4            |
| Utilizatori afectați                     | 1            |
| Ușurință în descoperire                  | 10           |
| Factor de risc "Active Identity Caching" | 5            |

#### 4.1.2.7 Encryption Suppression

Deoarece stația mobilă nu are cum să autentifice mesajele de pe interfața radio, acesta poate fi „momit” să se campeze pe un BTS fals și să comunice cu atacatorul într-o manieră necriptată. Atacatorul poate elimina comanda de criptare și menține convorbirea necriptată pentru atât timp cât aceasta rămâne nedetectată.

| Factor                  | Scor de risc |
|-------------------------|--------------|
| Potențial de distrugere | 2            |
| Reproductibilitate      | 10           |
| Exploatabilitate        | 3            |

|   |     |
|---|-----|
| Utilizatori afectați                    | 1   |
| Ușurință în descoperire                 | 10  |
| Factor de risc "Encryption Suppression" | 5.2 |

#### 4.1.2.8 Compromised Cipher Key

Acest atac necesită un BTS modificat și posesia de către intrus a unui vector de autentificare compromis, exploatând astfel vulnerabilitatea că utilizatorul nu are control asupra cheii de cifrare. Utilizatorul țintă este „momit” să se campeze pe un BTS/MS fals. Când se efectuează un apel, perechea BTS/MS falsă forțează folosirea cheii de cifrare compromisă.

| Factor                                  | Scor de risc |
|---|--------------|
| Potențial de distrugere                 | 2            |
| Reproductibilitate                      | 8            |
| Exploatabilitate                        | 3            |
| Utilizatori afectați                    | 1            |
| Ușurință în descoperire                 | 10           |
| Factor de risc "Compromised Cipher Key" | 4.8          |

#### 4.1.2.9 Eavesdropping on User Data by Suppressing Encryption

Acest atac necesită un BTS/MS modificat care exploatează vulnerabilitatea că stația mobilă nu poate autentifica mesajele prin rețeaua radio. Utilizatorul este tentat să se campeze pe un BTS fals, iar când acesta încearcă să inițieze un apel, criptarea nu este activată prin falsificarea mesajelor care cer acest lucru. Atacatorul poate însă să creeze o legătură reală și securizată cu rețeaua, permițând trecerea traficului. În acest fel atacatorul poate să captureze date de trafic necodificate.

| Factor                  | Scor de risc |
|-------------------------|--------------|
| Potențial de distrugere | 2            |
| Reproductibilitate      | 10           |
| Exploatabilitate        | 2            |
| Utilizatori afectați    | 1            |
| Ușurință în descoperire | 10           |

|   |   |
|---|---|
| Factor de risc "Eavesdropping on User Data by Suppressing Encryption" | 5 |
|---|---|

$$\text{Risk}_{\text{DREAD}} = (2 + 10 + 2 + 1 + 10) / 5 = 5$$

#### 4.1.2.10 Suppression of Encryption between Target User and True Network

Utilizatorul este tentat să se campeze pe un BTS/MS fals. Când utilizatorul sau rețeaua legitimă inițiază o conexiune, BTS/MS-ul fals modifică parametrii de criptare ai MS-ului pentru a face să pară că există o incompatibilitate între rețea și stația mobilă. Rețeaua poate decide să stabilească o conexiune necifrată. După ce decizia a fost luată, intrusul poate să captureze traficul în voie.

| Factor  | Scor de risc |
|---|--------------|
| Potențial de distrugere   | 2            |
| Reproductibilitate  | 10           |
| Exploatabilitate  | 2            |
| Utilizatori afectați  | 1            |
| Ușurință în descoperire   | 10           |
| Factor de risc "Suppression of Encryption between target User and True Network" | 5            |

#### 4.1.2.11 Eavesdropping on User Data by Forcing the Use of a Compromised Cipher Key

Acest atac necesită un BTS/MS modificat și posesia de către intrus a unui vector de autentificare compromis, exploatând astfel slăbiciunea că utilizatorul nu are un control asupra cheii de cifrare. Utilizatorul țintă este tentat să se campeze pe BTS/MS-ul fals. Când utilizatorul țintă sau intrusul inițiază un serviciu, BTS/MS-ul fals forțează utilizarea unei chei de cifrare compromise pe partea mobilă, în timp ce creează o conexiune cu rețeaua reală folosind un abonament real.

| Factor                  | Scor de risc |
|-------------------------|--------------|
| Potențial de distrugere | 2            |
| Reproductibilitate      | 10           |
| Exploatabilitate        | 2            |
| Utilizatori afectați    | 1            |
| Ușurință în descoperire | 10           |

|  |   |
|--|---|
| Factor de risc "Eavesdropping on User Data by Forcing the Use of a Compromised Cipher Key" | 5 |
|--|---|

#### 4.1.2.12 User impersonation with compromised authentication vector

Acest atac necesită un MS modificat și posesia de către intrus a unui vector de autentificare compromis care va fi utilizat de către rețea pentru a autentifica utilizatorul legitim. Intrusul folosește aceste informații pentru a impersona utilizatorul țintă în rețeaua legitimă.

| Factor   | Scor de risc |
|--|--------------|
| Potențial de distrugere  | 2            |
| Reproductibilitate   | 10           |
| Exploatabilitate   | 2            |
| Utilizatori afectați   | 1            |
| Ușurință în descoperire  | 10           |
| Factor de risc "User impersonation with compromised authentication vector" | 5            |

#### 4.1.2.13 User impersonation through eavesdropped authentication response

Atacul necesită un MS modificat și exploatează slăbiciunea că un vector de autentificare se poate folosi de mai multe ori. Intrusul poate captura răspunsul de autentificare trimis de utilizator și folosește acest răspuns când aceeași întrebare este trimisă ulterior. Intrusul folosește răspunsul capturat pentru a impersona utilizatorul țintă în rețea.

| Factor   | Scor de risc |
|--|--------------|
| Potențial de distrugere  | 2            |
| Reproductibilitate   | 10           |
| Exploatabilitate   | 5            |
| Utilizatori afectați   | 1            |
| Ușurință în descoperire  | 10           |
| Factor de risc "User impersonation through eavesdropped authentication response" | 5.6          |



## 4.1.2.14 Hijacking outgoing calls in networks with encryption disabled

Acest atac necesită un BTS/MS modificat. În timp ce utilizatorul țintă se campează pe un BTS fals, intrusul semnalizează utilizatorul țintă că are un apel de intrare. Acesta la rândul să inițiază procedura de stabilire a apelului, pe care intrusul o permite să se efectueze între utilizator și rețea, modificând elementele de semnalizare în așa fel încât pentru rețea să pară că utilizatorul dorește să inițieze un apel. Rețeaua nu activează criptarea. După autentificare, intrusul taie conexiunea cu utilizatorul și în continuare folosește conexiunea cu rețeaua pentru a efectua apeluri frauduloase pe abonamentul utilizatorului.

| Factor   | Scor de risc |
|--|--------------|
| Potențial de distrugere  | 4            |
| Reproductibilitate   | 10           |
| Exploatabilitate   | 5            |
| Utilizatori afectați   | 1            |
| Ușurință în descoperire  | 10           |
| Factor de risc "Hijacking outgoing calls in networks with encryption disabled" | 6            |

## 4.1.2.15 Hijacking outgoing calls in networks with encryption enabled

Acest atac necesită un BTS/MS modificat. În plus față de atacul anterior, de această dată intrusul trebuie să încerce să suprimă criptarea prin modificarea mesajelor prin care MS-ul informează rețeaua despre capacitățile sale criptografice.

| Factor  | Scor de risc |
|---|--------------|
| Potențial de distrugere   | 4            |
| Reproductibilitate  | 10           |
| Exploatabilitate  | 5            |
| Utilizatori afectați  | 1            |
| Ușurință în descoperire   | 10           |
| Factor de risc "Hijacking outgoing calls in networks with encryption enabled" | 6            |

## 4.1.2.16 Hijacking incoming calls in networks with encryption disabled

Acest atac necesită un BTS/MS modificat. În timp ce utilizatorul țintă se campează pe o stație de bază falsă, un asociat al intrusului efectuează un apel la

numărul utilizatorului țintă. Intrusul acționează ca un intermediar între rețea și utilizatorul țintă până la momentul ulterior autentificării și stabilirii apelului dintre aceștia. Rețeaua nu activează criptarea. După autentificare și stabilirea apelului, intrusul eliberează utilizatorul țintă și folosește conexiunea pentru a răspunde la apelul efectuat de asociatul său. Utilizatorul țintă va trebui să plătească pentru partea de apel din roaming.

| Factor   | Scor de risc |
|--|--------------|
| Potențial de distrugere  | 4            |
| Reproductibilitate   | 10           |
| Exploatabilitate   | 5            |
| Utilizatori afectați   | 1            |
| Ușurință în descoperire  | 10           |
| Factor de risc "Hijacking incoming calls in networks with encryption disabled" | 6            |

#### 4.1.2.17 Hijacking incoming calls in networks with encryption enabled

Acest atac necesită un BTS/MS modificat. În plus față de atacul anterior, intrusul va suprima criptarea.

| Factor  | Scor de risc |
|---|--------------|
| Potențial de distrugere   | 4            |
| Reproductibilitate  | 10           |
| Exploatabilitate  | 5            |
| Utilizatori afectați  | 1            |
| Ușurință în descoperire   | 10           |
| Factor de risc "Hijacking incoming calls in networks with encryption enabled" | 6            |

#### 4.1.3 Clasificarea vulnerabilităților după gravitate

Aplicarea metodologiei DREAD de clasificare a vulnerabilităților, metodă folosită inițial în software, a permis ordonarea acestora în funcție de gravitate (vezi Tabel 2). Se constată astfel că cele mai grave vulnerabilități ale rețelelor GSM se referă la atacurile DoS și posibilitatea ca apelurile să fie „furate” la momentul inițierii lor. Atacurile DoS sunt relativ ușor de pus în practică deoarece tehnologia GSM nu are rezistență prin design împotriva atacurilor de acest tip.

Tabel 2. Clasificarea vulnerabilităților în GSM

| Vulnerabilitate   | Scor de risc |
|---|--------------|
| Denial of Service Attacks   | 8.2          |
| Hijacking outgoing calls in networks with encryption disabled             | 6            |
| Hijacking outgoing calls in networks with encryption enabled              | 6            |
| Hijacking incoming calls in networks with encryption disabled             | 6            |
| Hijacking incoming calls in networks with encryption enabled              | 6            |
| De-registration Spoofing  | 5.8          |
| Location Update Spoofing  | 5.8          |
| Camping on a False BTS  | 5.6          |
| User impersonation through eavesdropped authentication response           | 5.6          |
| Passive Identity Caching  | 5.2          |
| Encryption Suppression  | 5.2          |
| Active Identity Caching   | 5            |
| Eavesdropping on User Data by Suppressing Encryption                      | 5            |
| Suppression of Encryption between Target User and True Network            | 5            |
| Eavesdropping on User Data by Forcing the Use of a Compromised Cipher Key | 5            |
| User impersonation with compromised authentication vector                 | 5            |
| Compromised Cipher Key  | 4.8          |

#### 4.2 Atacuri DoS în rețele GSM

Am văzut că atacurile DoS sunt cea mai gravă amenințare la adresa disponibilității sistemului GSM. Partea cea mai vulnerabilă a sistemului și locul unde atacurile se desfășoară este domeniul radio, mai precis sistemul de management al resurselor radio. Atacul de tip DoS în GSM are loc prin inițierea de către un mobil a unui apel fals care are ca scop alocarea unor resurse radio. Repetarea acestui tip de apel duce la epuizarea resurselor și la imposibilitatea folosirii serviciului de către un utilizator legitim. În acest paragraf se descrie în detaliu mecanismul care face GSM-ul vulnerabil în fața unui atac DoS.

Scenariul tipic pentru partea preliminară a unui apel cu originea la mobil este după cum urmează [45]:

**Pasul 1:** Stația mobilă (MS) cere acordarea unui canal de control de la BSC (Base Station Controller) (Fig. 24).

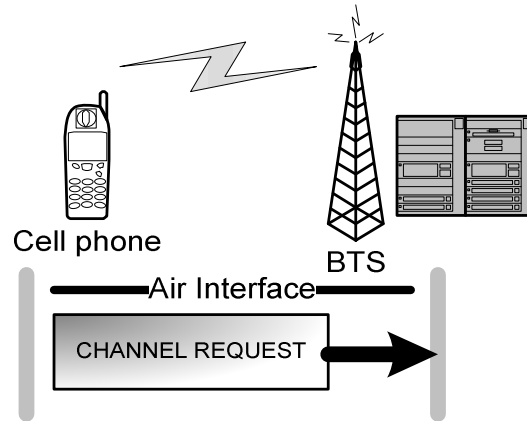


Fig. 24 Mesajul CHANNEL REQUEST

**Pasul 2:** BTS-ul decodifică mesajul CHANNEL REQUEST, calculează avansul temporal (distanța  $MS \square BTS$ ) și trimite informația completă la BSC prin intermediul mesajului CHANNEL REQUIRED. Totodată, se indică și tipul de serviciu solicitat. (Fig. 25)

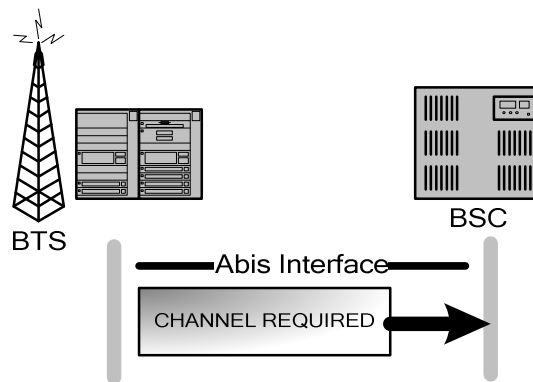


Fig. 25 Mesajul CHANNEL REQUIRED

**Pasul 3:** După recepția și procesarea mesajului CHANNEL REQUIRED, BSC-ul informează BTS-ul de ce tip de canal și ce număr de canal va fi rezervat de un mesaj CHANNEL ACTIVE ulterior (Fig. 26).

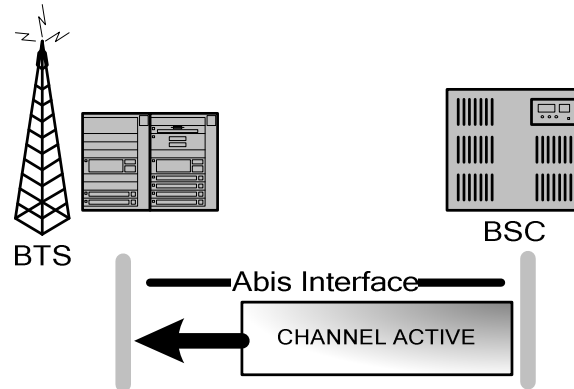


Fig. 26 Mesajul CHANNEL ACTIVE

**Pasul 4:** BTS-ul confirmă primirea prin trimiterea unui mesaj CHANNEL ACTIVE ACKNOWLEDGE (Fig. 27).

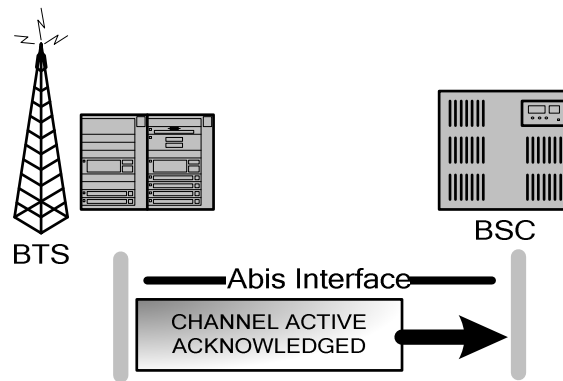


Fig. 27 Mesajul CHANNEL ACTIVE ACKNOWLEDGED

**Pasul 5:** BSC-ul trimite mesajul IMMEDIATE ASSIGNMENT COMMAND către BTS, care la rândul său informează MS-ul despre canalul alocat (Fig. 28).

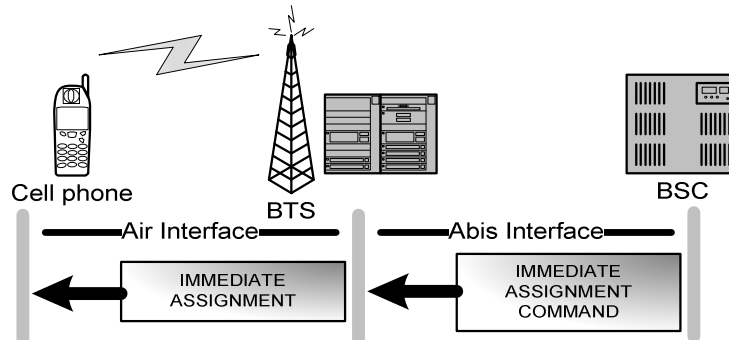


Fig. 28 Message IMMEDIATE ASSIGNMENT

Schimbul complet de mesaje al procesului de alocare de canal este arătat în Fig. 29.

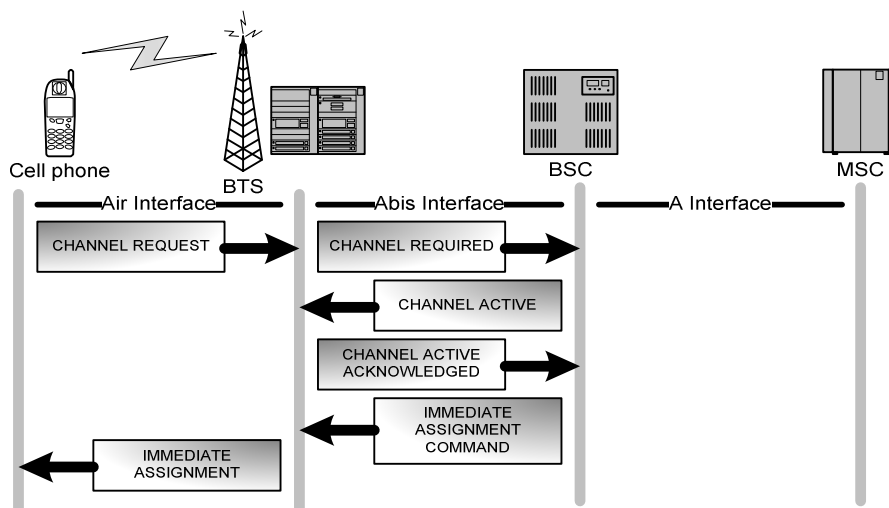


Fig. 29 Procesul de alocare a canalului în GSM

La sfârșitul procesului de alocare a canalului, ca urmare a cererii unei stații mobile **neautentificate** (client din perspectiva noastră), BSC-ul a alocat un canal de semnalizare din grupul de canale disponibile. Stația mobilă este acum responsabilă pentru respectarea restului protocolului, primul pas fiind cererea unui tip de serviciu.

Design-ul se bazează pe faptul că stația mobilă va urma în mod corect fiecare pas din protocol. Ce se întâmplă însă dacă o stație mobilă repetă scenariul anterior și cere mai multe canale de semnalizare fără a continua protocolul până la capăt? Din moment ce numărul de canale de semnalizare este limitat, rețeaua

devine congestionată local și cererile legitime sunt refuzate datorită lipsei de canale disponibile. BSC-ul va termina în cele din urmă cererile incomplete, eliberând resursele, dar acest lucru nu este un management de resurse eficient (resource containment) deoarece atacul în sine nu este detectat. Canalele de trafic disponibile nu vor deservite clienților legitimi deoarece toate canalele de semnalizare vor fi indisponibile (Fig. 30).

Chiar dacă rețeaua ar efectua o minimă autentificare a stației mobile prin cererea numărului IMEI și a puterii celor șase celule învecinate, atacul tot ar fi posibil întrucât stația mobilă are control complet asupra acestor elemente și ar putea raporta valori eronate pentru nivelurile de putere, generând în același timp numere IMEI sau redându-le dintr-o listă precompilată. Pentru a preîntâmpina problemele rețeaua are nevoie de o metodă simplă de verificare a identității clienților.

De remarcat este că îmbunătățirea securității SIM-ului nu este o măsură judicioasă. În timp ce aplicarea unui sistem de securitate bazat pe SIM ar induce costuri reduse de rezolvare a vulnerabilității, soluția ar fi inefectivă întrucât atacul este îndreptat asupra protocolului de stabilire a apelului și nu asupra SIM-ului.

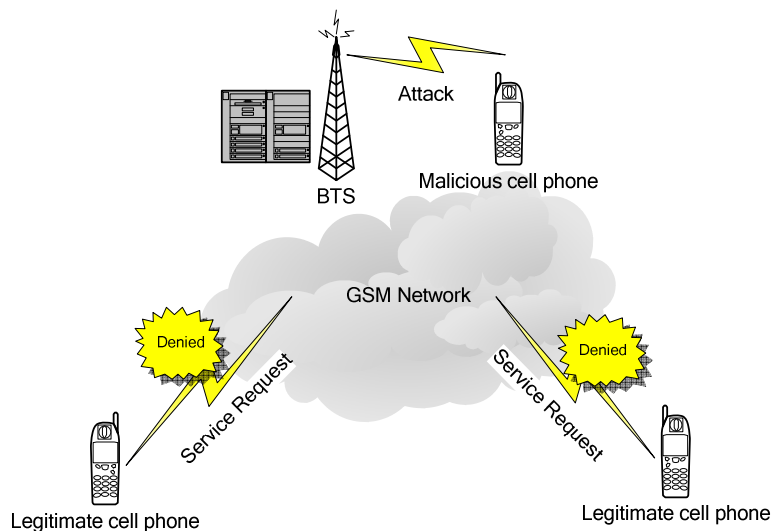


Fig. 30 Atacuri Denial of Service (DoS) în rețele GSM

### 4.3 Profilul atacatorului

Istoria recentă a înregistrat multe încercări de a proteja informația prin ascunderea ei sau cel puțin prin îngreunarea căilor de a o descoperi. Acest procedeu

este cunoscut sub numele de „securitate prin obscuritate” și este cea mai mare sursă de probleme pentru orice sistem care încearcă să-și securizeze datele. Mecanismul de securitate va fi în cele din urmă descoperit, comunicat grupurilor de interese și exploatat.

Cei mai mulți dintre indivizii care atacă sisteme de calculatoare sau de telefonie au motivații personale. Unii dintre ei vor încerca să exploateze slăbiciunile pentru câștiguri personale (acces gratuit la resurse, apeluri la distanță gratuite, apeluri gratuite la numere cu valoare adăugată, etc.) în timp ce alții vor prezenta slăbiciunile grupurilor interesate pentru faimă. Deși atacurile sunt serioase și pot provoca pierderi operatorului de rețea, atacurile Denial of Service sunt de departe cea mai gravă amenințare. Un individ care atacă o organizație vulnerabilă poate paraliza traficul în arii extinse ale rețelei, cauzând pierderi financiare greu de estimat.

Pentru a ataca cu succes o rețea GSM, intrusul trebuie să fie capabil de a modifica codul firmware din telefon într-o manieră potrivită. Această sarcină nu este una simplă întrucât necesită cunoștințe temeinice ale implementării particulare a dispozitivului mobil, codului și chestiunilor tehnice specifice GSM. Acesta este aspectul cel mai provocator și din fericire nu multe persoane au o asemenea experiență.

Intrusul trebuie de asemenea să înțeleagă topologia rețelei pe care o atacă. Dacă aria țintită este locală, o simplă plimbare prin oraș este suficientă pentru a determina locurile aproximative unde BTS-urile sunt localizate și unde stațiile mobile malițioase vor opera. Intrusul nu trebuie să fie în mod necesar prezent, întrucât telefonul mobil ar putea fi preprogramat să lanseze atacul, peisajul orașului oferind o multitudine de locuri unde s-ar putea ascunde astfel de dispozitive mici.

Intrusul trebuie să fie motivat, în funcție de magnitudinea atacului. Este improbabil ca un singur individ poate să fie motivat îndeajuns pentru a încerca un atac semnificativ asupra unei rețele GSM, în special considerând costul mare implicat. Totuși, intrusul poate fi sprijinit de grupuri criminale, caz în care economia atacurilor are o altă magnitudine.

#### 4.4 Economia atacului

Datorită cvasi-universalității și cotei foarte mari de piață a rețelelor GSM, atacurile care ar putea să le afecteze sunt foarte tentante pentru organizațiile criminale. Economia atacurilor asupra rețelelor GSM nu este diferită de cea a atacurilor îndreptate împotriva rețelelor de calculatoare, după cum urmează:

- Atacurile care cauzează oprirea totală a serviciului produc pierderi urișe de venit, fără a menționa latura socială a acestora.



- Când comunicația este vitală, spre exemplu după un atac terorist sau dezastru natural, atacul DoS împotriva rețelelor GSM poate avea consecințe dintre cele mai grave. Lipsa comunicației în astfel de momente poate duce la pierderi de vieți omenești sau proprietate.
- Atacurile care cauzează pauze în furnizarea serviciilor GSM sunt foarte dificil de detectat. Un client care dorește să utilizeze rețeaua poate fi frustrat de imposibilitatea efectuării apelurilor. Dacă situația se repetă, încrederea în operatorul de telefonie poate fi afectată iar clienții pot trece ușor la concurență.

#### 4.5 Creșterea disponibilității rețelelor GSM prin preautentificare

Dacă cineva ar veni la ușa dumneavoastră și v-ar cere un bun de valoare, i-ați da acel lucru fără a-i stabili mai întâi identitatea? Ați asuma implicit că oricine bate la ușă este un individ onest? Oricât de greu vă vine să credeți, exact acest lucru se întâmplă la inițierea unui apel într-o rețea GSM. În procesul de inițiere a unui apel, în timpul VEA (*Very Early Assignment*) nu se efectuează autentificare sau cifrare [2].

Primul mesaj trimis de telefonul mobil este CHANNEL REQUEST și are lungimea de doar 1 octet. Acesta conține motivul cererii (răspuns la paging, apel de urgență, etc.) și un identificator al tipului de canal pe care stația mobilă îl preferă. Problema cu acest tip de abordare este că BSC-ul își alocă resursele valoroase unei stații mobile neautentificate care ar putea să acționeze în afara standardului.

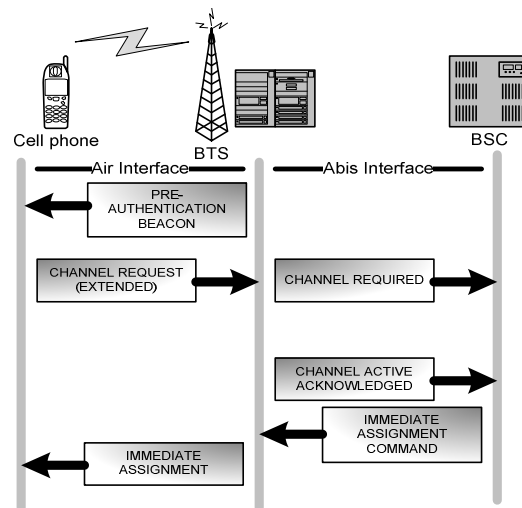


Fig. 31 Proces de asignare a canalului în GSM rezistent la atacuri DoS

Pentru a preveni un atac DoS potențial, trebuie să existe o formă minimă de autentificare la momentul cererii canalului de comunicație, prin urmare propun un nou proces de acordare a canalelor, ilustrat în Fig. 31.

Pasul 1: Când se atinge un prag de congestie al celulei sau al unui grup de celule, la anumite intervale BTS va emite un mesaj PREAUTHENTICATION BEACON care conține o valoare unică de 128 de biți (nonce), cu viață redusă, similar cu cea folosită la autentificare (Fig. 32). Durata de viață limitată este determinată de BSC. La generarea fiecărei valori, BSC-ul va precalcula răspunsul așteptat pentru fiecare cheie de utilizator înregistrată în baza sa de date ( $K_i$ ) pentru ca ulterior verificările răspunsurilor dispozitivelor mobile să se efectueze într-un timp scurt.

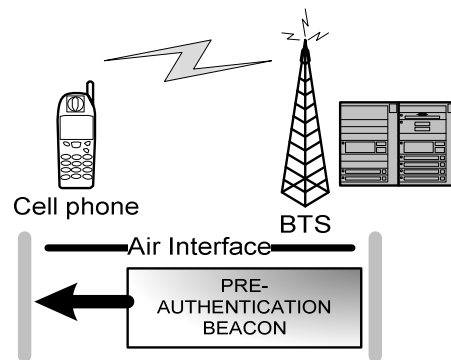


Fig. 32 Mesajul PREAUTHENTICATION BEACON

Valoarea de 128 de biți este suficient de mare pentru a împiedica precalcularea unui spațiu al cheilor statistic semnificativ, în special având în vedere că puterea de calcul a terminalelor mobile este limitată. Stația mobilă memorează ultima valoare primită, fiind folosită pentru cereri ulterioare de alocare de canal, atât timp cât valoarea TTL (Time To Live) permite. Faza de preautentificare funcționează asemănător cu autentificarea în sine. Răspunsul ar trebui însă scurtat pentru a scădea traficul de pe canalul cu acces aleatoriu, care altfel ar deveni congestionat iar acest lucru ar anula avantajul introdus de preautentificare. Propun ca răspunsul să fie redus la 16 biți din cei 32, având în acest fel un spațiu de 65536 de valori, suficiente pentru a evita potrivirile ocazionale în cazul în care atacatorul ar trimite o rafală de cereri false cu răspunsuri aleatoare.

Procesul este ilustrat în Fig. 33.

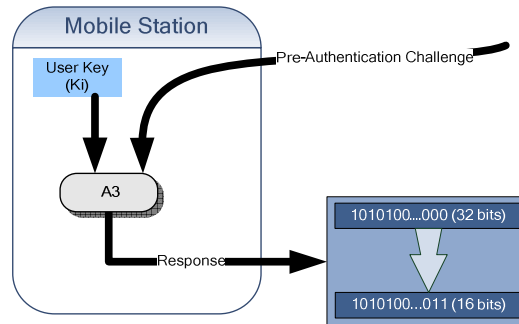


Fig. 33 Calcularea răspunsului de preautentificare

Mesajul CHANNEL REQUEST extins trimis prin canalul de acces aleator (RACH – Random Access Channel) trebuie să conțină atât motivul cererii resursei (ca în versiunea originală) cât și răspunsul de preautentificare de 16 biți.

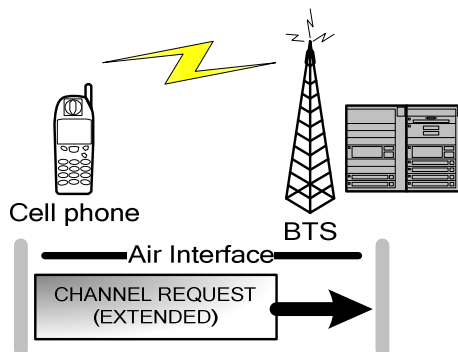


Fig. 34 Mesajul CHANNEL REQUEST (EXTENDED)

#### 4.5.1 Impactul preautentificării asupra capacității de semnalizare

Schema de autentificare propusă necesită transportul unei cantități de 16 biți pe canalul cu acces aleatoriu (RACH – Random Access Channel). Pe RACH, mobilul comunică prin așa numitele rafale de acces (AB – Access Burst), având următoarea structură:

|                |                       |                      |                |                       |
|----------------|-----------------------|----------------------|----------------|-----------------------|
| Tail Bits<br>8 | Syncho Sequence<br>41 | Encrypted Bits<br>36 | Tail Bits<br>3 | Guard Period<br>68,25 |
|----------------|-----------------------|----------------------|----------------|-----------------------|

Informația utilă transportată în AB este de 36 de biți din care primii 8 biți sunt mesajul de acces (3 biți tipul de acces, 5 biți codul aleatoriu de culoare pentru a distinge între mai multe mobile care efectuează simultan o cerere) iar restul de 28 de biți sunt folosiți pentru coduri CRC și de convoluție.

Pentru implementarea preautentificării este astfel nevoie de un AB suplimentar care să transporte valoarea pe 16 biți calculată de mobil urmată de un cod CRC pentru a asigura corectitudinea transmisiei. Legătura dintre cele două AB se poate face prin cei 5 biți aleatorii de culoare la care se adaugă un număr de secvență pentru a putea discrimina componentele mesajului multiplu.

Având în vedere că un cadru are lungimea de 0,234 secunde iar un multcadru de tip RACH are 51 de cadre, avem aproximativ 218 accese RACH posibile pe secundă într-o singură celulă. Deoarece protocolul de stabilire a apelului în noua formă presupune două accese consecutive la canalul RACH, obținem o înjumătățire a capacității de semnalizare la 114 cereri pe secundă într-o celulă.

#### 4.5.2 Avantaje și limitări ale soluției propuse

Efectuarea preautentificării la fiecare cerere de acces la resurse la nivelul BSC-ului asigură continența resurselor. Deoarece canalul RACH pe care se fac accesese are capacitate limitată, schema de preautentificare este slăbită în mod deliberat prin eliminarea a jumătate dintre cei 32 de biți rezultați. Un atacator care ar încerca să lanseze cereri pentru care ghicește valorile de răspuns ar avea o rată de succes de  $2^{-16}$ . Având în vedere faptul că în sistemul GSM se pot efectua aproximativ 784000 de accese de tip burst pe RACH pe oră și presupunând că un terminal mobil folosește toate aceste oportunități pentru a încerca ghicirea valorii de răspuns, vom avea în medie  $784000 * 2^{-16} = \sim 12$  ghiciri corecte pe oră, cantitate neglijabilă care nu permite un atac de tip DoS.

Pentru a putea lansa cu succes un atac, atacatorul trebuie să aibă acces la un număr mare de chei de utilizator ( $K_i$ ) valide și înregistrate în rețea. Entitatea care are o listă cu astfel de chei este OMC-ul, deci atacatorul trebuie să folosească o combinație de mecanisme tehnice și de inginerie socială precum și de informații din interiorul rețelei pentru a obține această listă și a o folosi ulterior pentru precalcularea valorilor de răspuns ale mesajului CHANNEL\_REQUEST (EXTENDED).

#### 4.6 Concluzii

Disponibilitatea și securitatea rețelelor GSM este un subiect actual, în condițiile în care infrastructura de comunicație a societății se bazează în mare măsură pe rețelele mobile, mai ales cele GSM. Capitolul de față propune în premieră clasificarea vulnerabilităților din GSM după o metodă folosită în mod uzual în software. Fiecărei vulnerabilități i se atribuie note corespunzătoare următorilor

factori: potențialul de distrugere, reproductibilitatea, exploatabilitatea, utilizatorii afectați, ușurința în descoperire. Media acestor note dă nivelul de gravitate al vulnerabilității, după care se poate efectua clasificarea. În urma clasificării, pe primul loc se situează atacurile de tip DoS, datorită relativei simplități cu care poate fi lansat și mai ales datorită numărului mare de utilizatori afectați în timpul atacului.

S-a arătat de asemenea că atacurile DoS sunt posibile datorită faptului că rețeaua GSM se bazează pe onestitatea clienților. Resursele din domeniul radio sunt alocate fără nicio autentificare prealabilă, lucru ce duce la blocaje dacă un singur client face cereri în rafală cu scopul de a epuiza numărul de canale disponibile în celulă.

Contribuțiile aduse de acest capitol pot fi sumarizate după cum urmează:

- Vulnerabilitățile sistemului GSM au fost evaluate cu o metodă de clasificare numită DREAD. Rezultatul deloc surprinzător a fost că cea mai periculoasă amenințare la adresa GSM este atacul DoS (denial of service).
- S-a arătat cauza pentru care sistemul GSM este vulnerabil la atacuri DoS, și anume lipsa preautentificării dispozitivelor mobile care pot cere alocarea de resurse în mod repetat.
- S-a arătat de asemenea că un singur atacator este capabil de a dezactiva o întreagă celulă GSM și în anumite cazuri chiar și celulele adiacente.
- Deoarece nu sunt implicate costuri financiare (nu se efectuează nici un apel în sine), costul efectiv al lansării un atac devastator este zero din punctul de vedere al plăților către operatorul vizat.
- În scopul creșterii rezistenței la atacuri DoS, s-a propus introducerea preautentificării terminalelor GSM în cazul inițierii unui apel de către acestea. Preautentificarea impune introducerea în protocolul Channel Assignment a unui mesaj nou de tip baliză numit PREAUTHENTICATION BEACON care transmite terminalelor mobile o valoare challenge care va fi folosită ulterior ca preautentificare în protocolul de inițiere al apelului.
- S-a arătat că impactul preautentificării pentru rețeaua GSM este înjumătățirea capacității de semnalizare pe canalul RACH.



## 5. Contribuții în domeniul sistemelor de distribuție digitală a conținutului

Multitudinea de dispozitive electronice portabile care ne înconjoară a atras implicit și nevoia acestora de comunicare. De la simple mesaje scrise sau e-mail, aceste dispozitive sunt acum capabile de a reda conținut digital audio și video pe care îl obțin fie de la o sursă locală cum ar fi un card de memorie fie de la o sursă online de tip streaming sau download. Mereu în actualitate, problema drepturilor de autor este acum și mai pregnantă. Dispozitivele electronice portabile sunt o piață gata de exploatat de către casele de discuri și cele de film, cu condiția ca veniturile din distribuția de muzică și film să nu fie afectate într-o măsură semnificativă de piraterie.

Consumatorii de entertainment digital mobil sunt în special tinerii. Odată cu lansarea unui album de muzică nou sau al unui film, utilizatorii vor provoca creșteri momentane ale valorilor traficului care în lipsa unei infrastructuri suficient dimensionate pot constata degradări ale serviciului consumat, cum ar fi timpi de așteptare mari, viteze de transfer scăzute, etc. Dimensionarea corespunzătoare a infrastructurii trebuie astfel dublată de eficientizarea transportului de informații.

Drepturile de distribuție pentru un conținut digital sunt deținute arareori de o singură entitate. De cele mai multe ori, conținutul digital este rezultatul colaborării între diverse entități cum ar fi producători video, de sunet, text, interpretare, distribuție, media, etc. Câștigul de pe urma comercializării produsului final se împarte între acestea într-un cadru contractual stabilit anterior. Spre deosebire de distribuția clasică ce presupune vânzarea unui produs tangibil cum ar fi CD sau DVD, în cazul distribuției digitale nu există un control asupra volumului de vânzări exercitat direct de către toate entitățile implicate. Colaborarea în acest caz este de preferat să se producă la nivel interactiv, în sensul că toate părțile implicate să își dea acordul momentan pentru distribuția de conținut pentru un client.

Pe piață există deja un număr de sisteme de distribuție digitală a conținutului. În principal, acestea se concentrează pe argumentele tehnice privitoare la mecanismul de redare a conținutului pe dispozitivele autorizate. Scalabilitatea unor astfel de sisteme este redusă și nu poate face față unor vârfuri de trafic și în plus nici nu se are în vedere colaborarea între părțile care partajează drepturile asupra unui conținut digital.

În cadrul acestui capitol se va prezenta o arhitectură de distribuție digitală a conținutului care îmbunătățește într-o manieră semnificativă scalabilitatea și maniera de cooperare între deținătorii de drepturi de autor. Arhitectura de la care s-a pornit este cea descrisă în [63].

### 5.1 Arhitectură scalabilă de distribuție digitală a conținutului

Marele neajuns al sistemelor de distribuție digitală a conținutului existente astăzi este că în general fac uz de criptografia asimetrică pentru protejarea conținutului (direct, prin criptarea acestuia sau indirect, prin criptarea unei chei simetrice de sesiune), fapt ce limitează drastic scalabilitatea. Arhitectura propusă aici ameliorează acest neajuns prin gruparea clienților și efectuarea operațiilor costisitoare ca timp de un număr mai redus de ori, astfel încât scalabilitatea crește în aceleași condiții de putere de calcul.

Sistemul de distribuție digitală a conținutului propus are structura generală prezentată în Fig. 35.

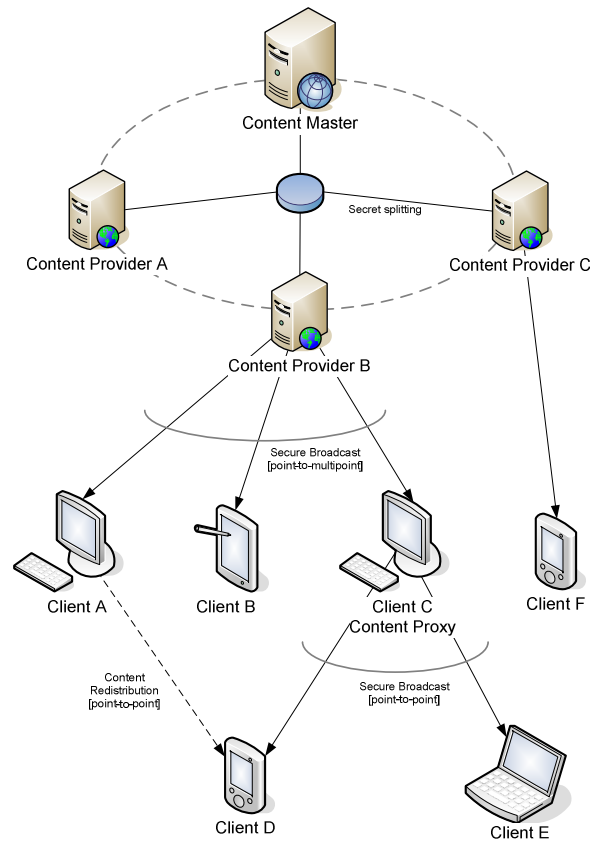


Fig. 35 Arhitectura sistemului scalabil de distribuție a conținutului



Actorii din cadrul sistemului sunt:

- **Content Provider (CP)** este o entitate care deține drepturile totale sau parțiale asupra unui conținut digital. Dacă drepturile deținute sunt parțiale, pentru distribuția conținutului acesta va solicita aprobarea tuturor celorlalți CP implicați și care partajează drepturile de distribuție a respectivului conținut digital.
- **Content Master (CM)** este o entitate de încredere care reprezintă organizația și care controlează activitatea tuturor CP și este implicată indirect în procesul de distribuție a conținutului. Se poate considera că CM este autoritatea care supervizează procesul de aprobare și arbitraj între CP și generează cheile de sesiune pentru activitățile CP.
- **Clienții** sunt dispozitive mobile sau programe software care au dreptul de a reda conținutul multimedia recepționat de la un CP și opțional, ar putea deține drepturile de redistribuție. Pentru ca sistemul de drepturi să fie viabil, clienții trebuie să îndeplinească o serie de condiții specificate în [92]. Mai mult, fiecărui dispozitiv sau program software îi este alocată o pereche de chei (una publică iar alta privată) ce se folosește la schimbul de informații dintre CP și un alt client. Procesul de redistribuție ce are loc între clienții A și D (Fig. 35) este dictat de politicile DRM asociate conținutului (ex. conținutul poate fi redistribuit doar un anumit număr de ori, doar anumitor clienți, etc.). Dacă politicile DRM nu sunt respectate de către client, CP-ul poate să-i revoce drepturile de distribuție.
- **Content Proxy (CPx)** este o funcție a unui client obișnuit care dorește – gratuit sau contra unui avantaj – să-și pună la dispoziție puterea de calcul și banda de comunicație pentru a ameliora încărcarea CM-ului. Orice client poate deveni content proxy dacă dorește acest lucru, semnalizând intenția unui CP. Deoarece conținutul digital este criptat, CPx-ul nu va putea să redea conținutul dacă acesta nu i se adresează în mod direct, dar va putea să trimită mai departe conținutul destinatarilor legitimi. În Fig. 35, clientul C are rolul de content proxy, distribuind datele clienților D și E folosind un broadcast securizat. Prin utilizarea proxy-urilor, CM își poate reduce drastic încărcarea deoarece odată procesate cererile de conținut, distribuția acestora este preluată de un CPx.

Procesul are două părți distincte: **distribuția conținutului** (de la furnizorul de conținut CP la clientul  $C_1$ ) și **redistribuția conținutului** (de la un client  $C_1$  la altul  $C_2$ ). Distribuția poate avea loc fără a fi urmată de redistribuție, însă reciproca nu este valabilă. Se introduc următoarele notații:

$e_A/d_A$  – Perechea de chei public/privată a entității A.

$[D]_{e_A}$  – Datele D criptate cu cheia publică a entității A.

$[D]_{d_A}$  – Datele D semnate cu cheia publică a entității A.

$[D]_K$  – Datele D criptate cu cheia simetrică K.

$h(D)$  – o funcție de dispersie fără coliziuni aplicată datelor D

### 5.1.1 Distribuția conținutului

Furnizorul de conținut (CP) distribuie conținutul digital M și drepturile asociate R, clienților (Ci) cu permisiunea celorlalți furnizori (CP2, ..., CPn) și sub supravegherea furnizorului master (MCP).

- (1) C1, C2, ..., Cn → CP1: Solicitare conținut
- (2) C1, C2, ..., Cn ↔ CP1: Autentificare reciprocă
- (3) CP1 ↔ CP2, ..., CPn, MCP: Acord asupra drepturilor de distribuție
- (4) C1, C2, ..., Cn ↔ CP1: Plată (pas opțional)
- (5) CP1 → C1, C2, ..., Cn :  $[M]_K, [K]_{eS}, X, \eta, \delta, \Lambda$

Furnizorul de conținut așteaptă cereri de distribuție de la clienți și le deservește la intervale prestabilite, spre exemplu o dată la câteva secunde. În acest timp se acumulează un număr de cereri pentru același conținut digital, care sunt grupate în pasul (1). În pasul (2), furnizorul de conținut și clienții se autentifică reciproc. Spre deosebire de arhitectura descrisă în [63], arhitectura de față nu prevede ca plata să se realizeze în acest stadiu întrucât furnizorul de conținut (CP) care a primit cererea nu are autorizarea omologilor săi de a distribui conținutul. În pasul (3), furnizorii de conținut se pun de acord asupra autorizării cererii de distribuție primite de CP și în pasul (4) se efectuează plata.

În pasul (5), furnizorul de conținut generează un lacăt X, criptează conținutul cu o cheie simetrică K ce se folosește o singură dată, criptează K cu cheia de sesiune eS și apoi trimite conținutul criptat împreună cu lacătul X, drepturile  $\eta$ , metadata  $\delta$  și licența pentru conținut  $\Lambda$ . Drepturile  $\eta$  reprezintă o cantitate care descrie cum se va trata conținutul digital pe dispozitivele aprobate iar metadata  $\delta$  sunt informațiile asociate conținutului (numele artistului, albumul, numele melodiei, rata de compresie, etc.). Licența de conținut  $\Lambda$  se definește astfel:

$$\Lambda = [h(M, \eta, \delta, X)]_{d_{CP}}$$

Scopul lui  $\Lambda$  este să certifice acordarea drepturilor  $\eta$  către client pentru conținutul  $M$ . Drepturile  $\eta$  pot fi reprezentate folosind limbaje de autorizare și politici de acces cum ar fi XACML [94] și XrML [95].

Arhitectura propusă prezintă două particularități: în pasul (3) se efectuează punerea de acord a CP asupra drepturilor de distribuție prin tehnica împărțirii secretului [76][75] iar în pasul (5) se face un broadcast securizat pentru clienții cărora le este destinat conținutul, tehnică bazată pe lacătele securizate [25] folosind teorema chineză a restului [93]. Aceste aspecte se vor detalia în paragrafele care urmează.

#### 5.1.1.1 Cooperarea furnizorilor de conținut

Furnizorii de conținut pot partaja drepturile de distribuție, caz în care trebuie ca toți să aprobe și opțional să memoreze cererile clientului. Cooperarea se face prin partajarea unui secret și participarea furnizorilor la reconstrucția sa. Nici un furnizor nu poate obține secretul fără cooperarea celorlalți furnizori participanți.

Considerând un mesaj secret  $M$  de lungime  $m$  și un grup de  $n$  entități care partajează un secret, denumiți  $P_1, P_2, \dots, P_n$ , secretul poate fi împărțit între cei  $n$  după cum urmează:

1. Se generează  $n-1$  numere aleatoare de lungime  $m$ , notate  $R_1, R_2, \dots, R_{n-1}$ .
2. Mesajul  $M$  este criptat astfel:  $S = M \otimes R_1 \otimes R_2 \otimes \dots \otimes R_{n-1}$
3. Secretul  $S$  este distribuit lui  $P_1$ ,  $R_1$  este distribuit lui  $P_2$ ,  $R_2$  lui  $P_3$  și așa mai departe până la  $R_{n-1}$  care este distribuit lui  $P_n$ .

Se observă că singura cale de a obține  $M$  este prin aplicarea funcției XOR bucăților distribuite celor care dețin parte din secret. Aceștia nu trebuie să cunoască cine a primit  $S$  și cine a primit șirurile aleatoare  $R_i$ . Acest lucru face sigură tehnica împărțirii secretului.

În cazul nostru, considerând că schema de distribuție conține un MCP și un număr  $n$  de CP:  $CP_1, CP_2, \dots, CP_n$  care partajează drepturile de distribuție pentru conținutul digital, tehnica funcționează după cum urmează:

1.  $CP_1$  primește o cerere de la client.
2. Dacă  $CP_1$  acceptă cererea (adică dacă clientul este credibil, contul este pe balanță pozitivă, a trecut verificările de bonitate, etc.), acesta va cere MCP-ului – care este autoritatea de încredere – să creeze și distribuie un mesaj secret  $M$  pentru toți furnizorii de conținut (CP).

3. MCP generează un mesaj secret  $M$ , cunoscut numai de el, îl împarte în  $n$  bucăți și le împarte tuturor furnizorilor de conținut (CP) prin utilizarea tehnicii de împărțire a secretului.
4.  $CP_i$  trimite cererea originală a clientului tuturor celorlalți CP, spre informare și o corectă luare de decizie.
5. Dacă un furnizor de conținut este de acord cu conținutul cererii clientului, acesta va pune la dispoziția  $CP_i$  partea sa de secret.
6. Având răspunsurile celorlalți deținători de secrete,  $CP_i$  va reconstrui mesajul  $M$ , care va fi trimis lui MCP pentru validare. Mesajul rezultat  $M$  a fi valid dacă toți furnizorii de conținut și-au dat acordul pentru distribuția conținutului supus discuției.
7. MCP va proceda la verificarea mesajului primit ca răspuns cu mesajul original creat în pasul 3. Dacă validarea se produce cu succes, MCP va autoriza  $CP_i$  să distribuie conținutul digital clientului, altfel va instrui CP să respingă cererea clientului.

#### 5.1.1.2 Broadcast securizat către clienți

Abordarea tradițională a distribuției de conținut presupune ca pentru fiecare cerere de la un client, CP-ul să cripteze conținutul folosind cheia publică a clientului care a făcut cererea [63]. Un număr de  $n$  clienți care solicită conținut digital de la CP va necesita  $n$  criptări, chiar dacă conținutul este identic pentru toți clienții. Acest lucru înseamnă o încărcare inutilă pe serverul care deserveste clienții și se traduce printr-o scalabilitate redusă.

Pentru a reduce încărcarea pe server, se propune ca furnizorul de conținut (CP) care deserveste cererile clienților să folosească o tehnică de distribuție securizată pentru toți clienții care au cerut un anumit conținut. În acest caz, conținutul digital este criptat o singură dată folosind o cheie de sesiune cunoscută numai de CP, fiind trimis clientului împreună cu o cheie de decriptare. Pentru a se asigura că doar clienții legitimi au acces la această cheie de decriptare, aceasta trebuie protejată cu un lacăt. Lacătul poate fi îndepărtat numai de clienții legitimi și este generat prin aplicarea unei tehnici numite teorema chineză a restului (CRT - Chinese Remainder Theorem) [25]:

Tehnica funcționează astfel: Se dă un grup de  $n$  clienți notat  $C$ , format din  $C_1, C_2, \dots, C_n$ , fiecare având o cheie privată  $e_i$ , respectiv una publică  $d_i$ . Se dă de asemenea un set de  $n$  numere pozitive întregi  $N_1, N_2, \dots, N_n$  cu proprietatea că sunt numere prime între ele și sunt cunoscute public în sistem. Din grupul  $C$ , un subset de  $k$  clienți ( $k \geq 2$ ) solicită același conținut digital de la furnizorul de conținut (CP). Dacă în continuare considerăm un set de  $k$  întregi pozitivi  $R_1, R_2, \dots, R_n$ , teorema chineză a restului afirmă că sistemul de congruențe:

$$\begin{cases} X \equiv R_1 \pmod{N_1} \\ X \equiv R_2 \pmod{N_2} \\ \dots \\ X \equiv R_k \pmod{N_k} \end{cases}$$

Ec. (8)

are o soluție comună  $X$ , dată de ecuația:

$$X = \left( \sum_{i=1}^k \frac{L}{N_i} \cdot R_i \cdot f_i \right) \pmod{L}$$

Ec. (9)

unde:

$$1 \equiv f_i \cdot \frac{L}{N_i} \pmod{N_i}$$

Ec. (10)

Cu  $L$  definit ca 
$$\prod_{i=1}^k N_i$$

În schema propusă, furnizorul de conținut va genera o pereche de chei de sesiune  $e_s$  și  $d_s$ , și va cripta conținutul digital o singură dată, folosind  $e_s$ . Furnizorul va genera de asemenea numerele  $R_1, R_2, \dots, R_n$  prin codificarea cheii de decriptare  $d_s$  cu cheia  $e_i$  a fiecărui client, după cum urmează:  $R_i = Enc_{e_i}(d_s)$

Lacătul  $X$  se obține rezolvând sistemul de congruențe și este trimis clienților împreună cu conținutul digital criptat. Fiecare client  $i$  din grupul care a solicitat conținutul digital poate obține  $R_i$  din lacătul  $X$ , iar prin decriptarea cu cheia privată  $d_i$  poate obține  $d_s$ . Odată ce clientul are cantitatea  $d_s$ , acesta poate decripta conținutul digital și îl poate utiliza în conformitate cu politica DRM asociată.

Din tehnica descrisă anterior se observă că lacătul nu poate fi deschis de către clienți nelegitimi. Chiar dacă un client rău intenționat obține restul  $R_i$ , acest nu poate extrage  $d_s$  deoarece nu posedă cheia privată  $d_i$  a clientului legitim  $i$ .

### 5.1.2 Retransmisia conținutului

Prin faptul că un client nu poate decodifica decât conținutul care îi este direct adresat, sistemul de distribuție permite o degrevare suplimentară a serverului la momentul transmisiei de fapt a datelor. Conținutul digital care trebuie livrat la un moment dat în mod normal s-ar plasa într-o coadă care ar deservi clienții care se conectează secvențial sau simultan în limita lățimii benzii de comunicație. În cazul arhitecturii propuse însă este posibil ca mai întâi să se trimită conținutul digital de la un moment dat către clienți mai apropiați geografic de destinatari prin intermediul așa numiților clienți proxy. Acești clienți participă indirect la procesul de distribuție, strict în sensul transportului de informație.

Procesul de retransmisie a conținutului este arătat în Fig. 36. Presupunem că furnizorul de conținut a autorizat toți clienții și a primit acceptul altor furnizori de a trimite conținutul.

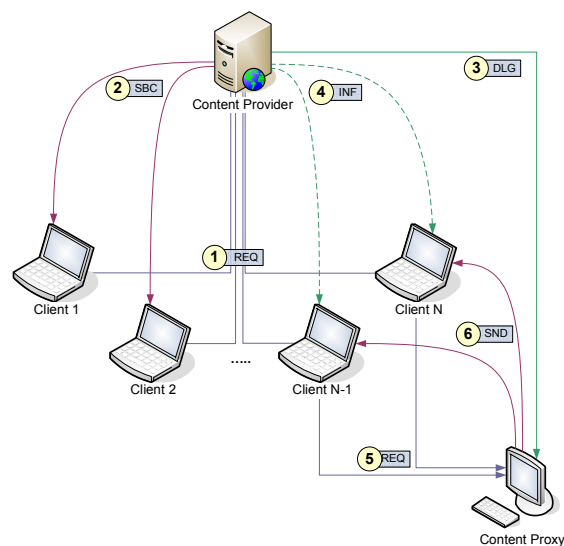


Fig. 36 Retransmisia conținutului prin clienți proxy

- În urma cererii clienților (marcate cu REQ), CP creează conținutul digital într-un mesaj de ieșire (după tehnica descrisă anterior) și îl trimite celor N clienți care l-au solicitat, folosind tehnica difuzării securizate cu lacăt. Când clientul primește mesajul, acesta va trimite o înștiințare (ACK) către CP.
- CP verifică dacă toți cei N clienți au primit mesajul. Dacă sunt clienți care au răspuns cu NAK (în cazul nostru clienții N-1 și N) sau dacă serverul este încărcat, CP va trimite mesajul original la unul sau mai mulți clienți proxy înregistrați și va delega sarcina de a distribui mesajul (DLG).

3. CP informează clienții care nu au recepționat mesajul de existența unui client proxy care este în posesia unei copii a conținutului digital. Din acest moment, CP este degrevat de sarcina de a distribui conținutul.
4. Clienții notificați de CP în pasul 4 vor solicita conținutul digital direct de la proxy, care deține o copie codificată a conținutului și nu va fi capabil să o decodifice decât dacă se află pe lista destinatarilor finali.
5. Proxy-ul va înainta mesajul codificat fiecărui client pentru care a primit notificare.

### 5.1.3 Redistribuția conținutului

Drepturile  $\eta$  acordate inițial de către furnizorul de conținut pot permite clientului  $C_1$  să redistribuie conținutul către un alt client  $C_2$ , urmând protocolul următor [63]:

- (1)  $C_2 \rightarrow C_1$ : Solicitarea conținutului
- (2)  $C_2 \leftrightarrow C_1$ : Autentificare reciprocă
- (3)  $C_1 \rightarrow C_2$ :  $[M]K', [K']eC_2, \eta, \eta', \delta, \Lambda, \Lambda'$
- (4)  $C_2 \leftrightarrow C_1$ : verificarea cantității  $\delta$ , plata (opțională)
- (5)  $C_2 \rightarrow C_1$ :  $\varphi$

Clientul  $C_2$  pornește tranzacția în pasul (1) prin solicitarea lui  $C_1$  a unui conținut digital anume. În pasul (2), cele două părți se autentifică reciproc folosindu-și perechile de chei publice/private. Dacă autentificarea are succes,  $C_1$  decriptează conținutul digital solicitat, generează o cheie temporară simetrică  $K'$  și criptează conținutul cu această cheie. Cheia  $K'$  este apoi criptată cu cheia publică a clientului  $C_2$ .  $C_1$  trimite apoi către  $C_2$  noul conținut criptat, cheia de sesiune  $K'$  criptată cu cheia publică a lui  $C_2$ , drepturile inițiale  $\eta$  și noile drepturi  $\eta'$ . De asemenea,  $C_1$  trimite licența inițială  $\Lambda$  și nouă licență  $\Lambda'$ , definită astfel:

$$\Lambda' = [h(e_1, e_2, M, \eta', \delta, X)]_{d_{C_1}}$$

În pasul (4),  $C_2$  verifică semnătura lui  $C_1$  pe noua licență  $\Lambda'$  și validează  $\eta$  și  $M$  folosind licența  $\Lambda$  originală.  $C_2$  se asigură de asemenea că licența  $\eta'$  poate fi derivată din  $\eta$  și verifică de asemenea  $\delta$  față de tipul de conținut distribuit. Dacă toate verificările trec cu succes,  $C_2$  aprobă tranzacția din pasul (5), trimițând lui  $C_1$  o înștiințare  $\varphi$ , definită ca:

$$\varphi = [h(e_{C_1}, e_{CP}, [M]K', \delta, \eta')]_{d_{C_2}}$$

Înștiințarea  $\varphi$  reprezintă recunoașterea lui  $C_2$  că a primit conținutul  $M$  cu drepturile  $\eta'$ .

#### 5.1.4 Riscuri potențiale

Arhitectura propusă introduce un număr de amenințări, unele fiind comune cu arhitecturi DRM existente, altele fiind noi. Una dintre amenințările cele mai răspândite este modificarea neautorizată a echipamentelor sau a modulelor protejate din interiorul acestora. Rezistența la modificări este dificil de obținut [3][47], se poate deci spune că securitatea este eficace împotriva atacatorilor cu excepția celor mai motivați.

Tehnicile criptografice sunt arareori ținta unor atacuri întrucât atacatorii aleg puncte mai vulnerabile ale unui sistem, cum ar fi sistemul de redistribuție și colaborare între părți. Se pot distinge următoarele tipuri de atacuri:

- **Înlocuirea conținutului** în timpul procesului de redistribuție. Acest lucru se poate întâmpla când un client recepționează o valoare mai mică decât cea comandată, sau clientul proxy înlocuiește conținutul înainte de a fi distribuit mai departe.
- **Copiile de siguranță ale cheilor pe medii nesecurizate** poate duce la compromiterea cheilor deoarece acestea părăsesc mediul securizat oferit de modulul criptografic.
- **Dispozitivele compromise** sunt capabile de a îndepărta mecanismul de securitate care protejează conținutul digital, putând astfel să distribuie conținutul în mod ilegal. Detectarea și izolarea dispozitivelor compromise este esențială pentru sănătatea unui sistem DRM.
- **Compromiterea unui singur provider** duce la căderea totală a sistemului de distribuție. Dacă un singur furnizor de conținut are un comportament care nu este în acord cu sistemul, tehnic acesta nu va putea fi exclus ușor.
- Spre deosebire de alte sisteme care efectuează operații costisitoare (criptografie asimetrică), arhitectura pe care am prezentat-o este mai puțin susceptibilă la **atacuri denial-of-service**, întrucât sarcina de distribuție propriu-zisă este delegată clienților proxy. Totuși, atacurile nu pot fi prevenite în totalitate, dar există căi de ameliorare a efectelor [11][12].

#### 5.1.5 Avantajele arhitecturii

Principalul avantaj al arhitecturii propuse este că scalează foarte bine în diverse scenarii. Când un anumit conținut digital este foarte cerut de către clienți (spre exemplu un album nou de succes, un trailer al unui film sau chiar un film în sine), se poate întâmpla ca foarte mulți dintre aceștia să ceară acest conținut simultan și într-un timp foarte scurt. În mod normal, furnizorul ar trebui să depună un efort susținut pentru satisfacerea cererilor, efort ce presupune putere de calcul și lățime de bandă pentru comunicație. În acest caz însă, furnizorul de conținut poate deservi cererile la un anumit interval (spre exemplu de ordinul secundelor) pentru a



se acumula un număr de cereri, apoi cu o singură operație de criptare serverul va putea deservi toți clienții respectivi, fapt ce duce la o posibilitate importantă de scalare a soluției. În plus, operațiile de transfer de date pot fi preluate de clienți cu rol de proxy, aceștia neputând reda conținutul dacă nu le este adresat.

## 5.2 Implementare

Prototipul sistemului de distribuție a conținutului a fost implementat în limbajul C# pe platforma .NET Framework 3.5 sub numele de SCADDIC. Pentru a reduce din complexitatea prototipului s-au omis funcționalitățile referitoare la confortul și securitatea părților, dezvoltarea concentrându-se pe partea de concept și demonstrare a fezabilității sistemului teoretic. Prototipul păstrează proporțiile diverselor sarcini relevante pentru evaluarea performanței.

Cele trei secțiuni principale ale proiectului sunt:

- Structuri de date pentru descrierea conținutului digital
- Clase pentru modelare diverselor entități care manipulează conținutul
- Bibliotecă de operații matematice și criptografice

Pentru a simplifica infrastructura de comunicație între entitățile care compun sistemul, s-a ales ca acestea să funcționeze pe aceeași mașină. Comunicarea s-a putut realiza astfel prin simple apeluri de funcții și a fost posibil ca biblioteca de operații matematice și criptografice să fie partajată între entități.

### 5.2.1 Structuri de date pentru descrierea conținutului

Structurile de date pentru descrierea conținutului au rolul de a clasifica conținutul digital și a datele auxiliare (metadata). Fiecare conținut este identificat în mod unic de un cod, având date auxiliare asociate care descriu drepturile de redare și descriptori precum titlu, autor, anul lansării, rata fluxului (bit rate), codec-ul folosit, etc. De asemenea, pentru a evita modificarea conținutului în tranzit, este prezentă și o sumă de control.

Clasa **ContentItem** conține informația digitală a conținutului, o sumă de control, precum și datele auxiliare și drepturile de distribuție.

Clasa **ContentMetadata** conține o listă de câmpuri cu valori asociate care au înțeles într-un anumit context dat, spre exemplu lista cu drepturi de redistribuție, informații despre conținutul digital (titlu, autor, bit-rate, etc.).

## 5.2.2 Entități care manipulează conținutul

Conform design-ului sistemului de distribuție descris, entitățile care manipulează conținutul sunt:

**Client** – un computer sau un dispozitiv mobil care solicită, consumă sau redistribuie conținutul digital, în concordanță cu licențele și drepturile asociate. Acesta este implementat în clasa **ContentClient**.

```
namespace SCADDICore
{
    /// <summary>
    /// Holds an association of the content item and its lock (we only need this locally, within the
    client)
    /// </summary>
    public class LocalContentItem
    {
        #region Public variables
        /// <summary>
        /// The content item
        /// </summary>
        public ContentItem contentitem;
        /// <summary>
        /// Lock (X) associated to the content
        /// </summary>
        public BigInteger LockX;
        #endregion

        #region Public Constructor
        public LocalContentItem(ContentItem contentitem, BigInteger LockX)
        {
            this.contentitem = contentitem;
            this.LockX = LockX;
        }
        #endregion
    }

    /// <summary>
    /// The Content Client is the requester and the beneficiary of a digital content.
    /// </summary>
    public class ContentClient
    {
        #region Local Variables
        /// <summary>
        /// Content client name (to be distinguished among peers)
        /// </summary>
        public string Name
        {
            get;
            set;
        }

        /// <summary>
        /// A public number given by the Content Master at setup time
        /// </summary>
        public BigInteger N
        {
            get;
            set;
        }

        /// <summary>
        /// Asymmetric key cryptography provider
        /// </summary>
        protected RSACryptoServiceProvider RSA;

        /// <summary>
        /// ASCII Encoder/Decoder
        /// </summary>
        protected ASCIIEncoding ae;

        /// <summary>
        /// Content items stored locally on the client. This resembles the internal memory of a device in
        real-world.
        /// </summary>
    }
}
```

```

protected SortedList<int, LocalContentItem> LocalContentItems;

#endregion

/// <summary>
/// Initializes a new client
/// </summary>
/// <param name="Name">Name of the content client (to be distinguished among peers)</param>
public ContentClient(string Name)
{
    this.Name = Name;
    // Generate a large prime number that acts as N in the Chinese Remainder Theorem
    N = BigInteger.genPseudoPrime(1024, 1, new Random());
    ae = new ASCIIEncoding();
    RSA = new RSACryptoServiceProvider();
    LocalContentItems = new SortedList<int, LocalContentItem>();
}

#region Public Methods
/// <summary>
/// The client receives digital content via this method.
/// </summary>
/// <param name="LockX">The cryptographic lock X, from which the client will extract the session
key used to encrypt the content.</param>
/// <param name="content">The received content.</param>
/// <param name="License">License (lambda) from the content.</param>
/// <param name="ContentMasterPublicKey">The public key of the content master.</param>
public void ReceiveContentFromMaster(ContentItem content, BigInteger LockX, ContentMetadata
License, RSAParameters ContentMasterPublicKey)
{
    Debug.WriteLine("Client " + Name + " received content from master.");
    // Compute R from the lock X and the publicly known value N (established at setup time)
    BigInteger R = LockX % N;
    string OriginalSessionKey = string.Empty;
    try
    {
        // Decode R with private key and get the session encryption key
        OriginalSessionKey = ae.GetString(RSA.Decrypt(R.getBytes(), false));
    }
    catch (Exception)
    {
        // Invalid R, the content is not for this client
    }

    // We now have both the encrypted file and the decryption key
    if (OriginalSessionKey == string.Empty)
    {
        Debug.WriteLine("The key is not for client " + Name + ", therefore it cannot decrypt the
content.");
    }
    else
    {
        // Verify signature with the public key of the content master
        string OriginalHash = License["LicenseHash"];
        byte[] Signature = Convert.FromBase64String(License["Signature"]);
        RSACryptoServiceProvider verifyRSA = new RSACryptoServiceProvider();
        verifyRSA.ImportParameters(ContentMasterPublicKey);
        bool Verified = verifyRSA.VerifyData(ae.GetBytes(OriginalHash), "MD5", Signature);

        if (!Verified) throw new Exception("Invalid content received");

        Debug.WriteLine("Decrypting content with key: " + OriginalSessionKey);
        // Decrypt content and analyze metadata, rights, license
        ContentCrypto.DecryptContent(content, OriginalSessionKey);

        // Generate content license hash (so we can check it whether it is genuine)
        // The hash will be compared against the one received from the master
        // Also, the signature will be verified
        string LicenseHash = Licensing.GenerateContentLicenseHash(content, LockX);
        if (LicenseHash != License["LicenseHash"])
        {
            throw new Exception("License of the content is not valid.");
        }

        // Store received content (delete content first if we have it already in list)
        if (LocalContentItems.ContainsKey(content.ContentItemID))
            LocalContentItems.Remove(content.ContentItemID);
        LocalContentItems.Add(content.ContentItemID, new LocalContentItem(content, LockX));
        // Consume content

        Debug.WriteLine("Content can be consumed.");
        Debug.WriteLine("Received metadata:");
    }
}

```

```

        foreach (string field in content.Metadata.FieldNames)
        {
            Debug.WriteLine(string.Format("Field: {0}, Value: {1}", field,
content.Metadata[field]));
        }

        Debug.WriteLine("Received rights:");
        foreach (string field in content.Rights.FieldNames)
        {
            Debug.WriteLine(string.Format("Field: {0}, Value: {1}", field, content.Rights[field]));
        }
    }
}

/// <summary>
/// Redistributes content to a peer client, should the rights allow that.
/// </summary>
/// <param name="TargetClient">Client to which the content will be sent.</param>
/// <param name="LocalContentID">ID of the content subject to redistribution (in decrypted
state).</param>
public void SendContentToPeer(ContentClient TargetClient, int LocalContentID)
{
    Debug.WriteLine("Client " + Name + " redistributes content to client " + TargetClient.Name +
".");

    // TODO: Create the new set of rights from the original set
    // As of now, the original rights are just as the original ones, but a real implementation
    // would need to alter the rights in some way, ex. to decrease copy count.
    // If the new rights prevent the further distribution of the content, bail out.
    ContentItem OriginalContent = LocalContentItems[LocalContentID].contentitem;

    ContentMetadata NewRights = OriginalContent.Rights;

    // Create a copy of the current content so it does not interfere with our local copy
    ContentItem Content = new ContentItem(OriginalContent);
    // Compute the new content license
    string LicenseHash = Licensing.GenerateRedistributedContentLicense(Content,
LocalContentItems[LocalContentID].LockX, this.PublicKey, TargetClient.PublicKey, NewRights);
    // Digitally sign hash
    byte[] Signature = RSA.SignData(ae.GetBytes(LicenseHash), "MD5");

    ContentMetadata NewLicense = new ContentMetadata();
    NewLicense["LicenseHash"] = LicenseHash;
    NewLicense["Signature"] = Convert.ToBase64String(Signature);

    // Generate a new random key and encrypt the content with this one
    string SessionKey = RandomPassword.Generate(8);
    ContentCrypto.EncryptContent(Content, SessionKey);

    // Encrypt session key with the public key of the target client (receiver)
    RSACryptoServiceProvider targetrsa = new RSACryptoServiceProvider();
    targetrsa.ImportParameters(TargetClient.PublicKey);
    byte[] EncryptedSessionKey = targetrsa.Encrypt(ae.GetBytes(SessionKey), false);

    // Send the content to the client
    //TargetClient.ReceiveContentFromPeer(Content, EncryptedSessionKey, NewRights, NewLicense,
this);
}

/// <summary>
/// Receives content from a peer client
/// </summary>
/// <param name="Content">Content received.</param>
/// <param name="EncryptedKey">Encrypted key used to protect the content. The key can be retrieved
using the private RSA key.</param>
/// <param name="NewRights">Redistribution rights for the current content.</param>
/// <param name="NewLicense">Redistribution license for the current content.</param>
/// <param name="From">Peer from which the content was received.</param>
public void ReceiveContentFromPeer(ContentItem Content, byte[] EncryptedKey, ContentMetadata
NewRights, ContentMetadata NewLicense, ContentClient From)
{
    // Verify license validity
    string LicenseHash = NewLicense["LicenseHash"];
    byte[] Signature = Convert.FromBase64String(NewLicense["Signature"]);
    RSACryptoServiceProvider verifRSA = new RSACryptoServiceProvider();
    verifRSA.ImportParameters(From.PublicKey);
    bool Verified = verifRSA.VerifyData(ae.GetBytes(LicenseHash), "MD5", Signature);

    if (!Verified) throw new Exception("Invalid content received");
    // TODO: Also compare hash of the license, not just the signature
}

```

```

        // Content received from a peer
        byte[] DecryptedKey = RSA.Decrypt(EncryptedKey, false);
        string ContentKey = ae.GetString(DecryptedKey);
        // Decrypt content
        ContentCrypto.DecryptContent(Content, ContentKey);

        Debug.WriteLine("Redistributed content can be consumed by client " + Name + ".");
    }

    /// <summary>
    /// Gets the public key of this content client
    /// </summary>
    /// <returns>The public key</returns>
    public RSAParameters PublicKey
    {
        get
        {
            return RSA.ExportParameters(false);
        }
    }
}
#endregion
}
}
}

```

**Content Provider** – Entitatea care deține copyright-ul asupra conținutului ce se distribuie. Acesta poate funcționa în conjuncție cu alți provideri cu care partajează drepturile. Implementarea este în clasa **ContentProvider**.

```

namespace SCADDICore
{
    /// <summary>
    /// The ContentProvider is responsible for granting/denying individual access to content
    /// by voting through secret splitting
    /// </summary>
    public class ContentProvider
    {
        #region Local Variables
        /// <summary>
        /// Content provider name (to be distinguished among peers)
        /// </summary>
        protected string Name;

        /// <summary>
        /// The list holds the secrets used to decrypt the content with a given ID. Each secret
        /// is used in conjunction with the secrets of other content providers in the secret splitting
        technique.
        /// </summary>
        protected SortedList<int, string> ContentKeyPieces;

        #endregion

        #region Public Constructors
        /// <summary>
        /// Initializes a new content provider
        /// </summary>
        /// <param name="Name">Name of the content provider (to be distinguished among peers)</param>
        public ContentProvider(string Name)
        {
            ContentKeyPieces = new SortedList<int, string>();
            this.Name = Name;
        }

        #endregion

        #region Public Methods
        /// <summary>
        /// Receive a fragment of the key used to encrypt the content identified by ContentID. The rest of
        the fragments
        /// are stored by peer Content Providers. All CPs will participate in the Secret Splitting schema
        when requested.
        /// </summary>
        /// <param name="ContentID">Content identification</param>
        /// <param name="ContentKey">Key</param>
        public void SetContentKey(int ContentID, string ContentKey)
        {

```

```

        ContentKeyPieces.Add(ContentID, ContentKey);
        Debug.WriteLine("CP [" + Name + "] received key " + ContentKey + " for content ID " + ContentID
+ ".");
    }

    /// <summary>
    /// This is the heart of the authorization mechanism. The content provider checks whether ALL
specified
    /// clients are to be granted access to the content with ContentID.
    /// </summary>
    /// <param name="ContentID">Content ID to which access is requested.</param>
    /// <param name="ContentClients">List of clients that request authorization for the
content.</param>
    /// <returns>In case all clients are allowed acces, the provider will proceed with returning its
own
    /// secret key to participate in the secret sharing schema. If at least one client is not allowed,
the provider will return an empty string.</returns>
    public string AuthorizeContentDistribution(int ContentID, List<string> RequestingClients)
    {
        // TODO: As of now, the voting mechanism is not implemented. The content provider blindly
        // provides its key to the Content Master, without asking for reason or other details.
        string ContentKey = string.Empty;
        try
        {
            ContentKey = ContentKeyPieces[ContentID];
        }
        catch (Exception) { }

        Debug.WriteLine("CP [" + Name + "] issued key " + ContentKey + " for content ID " + ContentID +
".");

        return ContentKey;
    }
    #endregion
}
}
}

```

**Content Master** – Este entitatea de încredere în sistemul de distribuție. Cererile care vin de la clienți sunt centralizate și servite aici, cu aprobarea furnizorilor de conținut, care pot restricționa accesul unui client în particular conform propriei voințe. Implementarea este în clasa **ContentMaster**.

```

namespace SCADDICore
{
    /// <summary>
    /// Summary for Content Master
    /// </summary>
    public class ContentMaster
    {
        /// <summary>
        /// Default secret length.
        /// </summary>
        protected const int PASSWORD_LENGTH = 8;

        #region Local Variables
        /// <summary>
        /// Content providers registered with the Content Master.
        /// </summary>
        protected List<ContentProvider> RegisteredContentProviders;
        /// <summary>
        /// Content items registered with the Content Master.
        /// </summary>
        protected SortedList<int, ContentItem> RegisteredContentItems;
        /// <summary>
        /// Hashes for the clear content of the registered items (so we can check the correct decryption
when give a certain password).
        /// </summary>
        protected SortedList<int, string> RegisteredContentHashes;
        /// <summary>
        /// Clients registered with the Content Master.
        /// </summary>
        protected SortedList<string, ContentClient> RegisteredContentClients;
        /// <summary>
        /// Content identifier. Increases with each new registered content.
        /// </summary>
    }
}

```

```

protected static int ContentID = 1;

/// <summary>
/// The RSA object of the content master. This is primarily used to sign content license (lambda)
/// </summary>
protected RSACryptoServiceProvider ContentMasterRSA;

#endregion

#region Public Constructors
public ContentMaster()
{
    RegisteredContentProviders = new List<ContentProvider>();
    RegisteredContentItems = new SortedList<int, ContentItem>();
    RegisteredContentClients = new SortedList<string, ContentClient>();
    RegisteredContentHashes = new SortedList<int, string>();
    ContentMasterRSA = new RSACryptoServiceProvider();
}
#endregion

#region Public Methods

/// <summary>
/// Registers a new CP with the Content Master
/// </summary>
/// <param name="cp">Content Provider to register</param>
public void RegisterContentProvider(ContentProvider newcp)
{
    RegisteredContentProviders.Add(newcp);
}

/// <summary>
/// Registers a new client with the Content Master
/// </summary>
/// <param name="cc">Client to register</param>
public void RegisterClient(ContentClient cc)
{
    RegisteredContentClients.Add(cc.Name, cc);
}

/// <summary>
/// Retrieves a specified content client
/// </summary>
/// <param name="Name">Content client name</param>
/// <returns>Content client information</returns>
public ContentClient GetContentClient(string Name)
{
    return RegisteredContentClients[Name];
}

/// <summary>
/// Registers a media file for use in the SCADDIC system:
/// 1. Generates a random password (the secret that will later be used in the secret splitting
technique)
/// 2. Encrypts the file with the random password
/// 3. Splits the password into equal pieces that will be fed to the content providers
/// 4. Destroys the password so it can no longer be retrieved by an attacker
/// </summary>
/// <param name="ContentFileName">File containing the digital content to be registered</param>
/// <param name="metadata">Metadata of the content being registered (e.g. bit rate, author name,
etc.)</param>
/// <param name="rights">Rights associated to the content being registered (e.g. NO COPIES,
etc.)</param>
/// <returns>The ID of the content being registered.</returns>
public int RegisterContentItem(string ContentFileName, ContentMetadata Metadata, ContentMetadata
Rights)
{
    // Generate a random password
    string Password = RandomPassword.Generate(PASSWORD_LENGTH);
    // Debug.WriteLine("Random password at content registration: " + Password);

    // Let N be the number of currently registered CPs. Generate N-1 secrets of length
PASSWORD_LENGTH.
    int N = RegisteredContentProviders.Count - 1;
    // If no content providers are registered, throw an exception
    if (N == -1) throw new Exception("No content providers are registered. Please register at least
on CP before you can register a content item.");

    string[] R = new string[N];
    for (int i = 0; i < R.Length; i++)
    {
        R[i] = RandomPassword.Generate(PASSWORD_LENGTH);
    }
}

```

```

    }
    // Calculate S defined as Password XOR R1 XOR ... XOR R n-1
    string S = Password;
    for (int i = 0; i < N; i++)
    {
        S = S.XOR(R[i]);
    }

    // Now we have S, R1, R2, ..., Rn-1. We have to distribute these values to CPs.

request. // Get a new ID for the current content. IDs will be sequentially incremented with each new
    int ContentID = GetNewContentID();
    // Distribute S to the first content provider
    RegisteredContentProviders[0].SetContentKey(ContentID, S);
    // Distribute subsequent parts of the secret (R1, R2, ..., Rn-1) to the rest of the CPs.
    for (int i = 1; i <= N; i++)
    {
        RegisteredContentProviders[i].SetContentKey(ContentID, R[i-1]);
    }

Rights); // Create a new content item
    ContentItem newci = new ContentItem(ContentID, File.ReadAllBytes(ContentFileName), Metadata,
    // Add hash to list (while the content is still unencrypted)
    RegisteredContentHashes.Add(ContentID, newci.ContentHash);
    // Encrypt the new content
    ContentCrypto.EncryptContent(newci, Password);
    // Add content to list (so we know it later)
    RegisteredContentItems.Add(ContentID, newci);
    // Return the ID of the content being registered
    return ContentID;
}

/// <summary>
/// Deletes the specified content item from the master store
/// </summary>
/// <param name="ContentID">ID of the content to destroy</param>
public void UnregisterContentItem(int ContentID)
{
    RegisteredContentItems[ContentID] = null;
}

/// <summary>
/// Implements the secret splitting technique.
/// </summary>
/// <param name="ContentID">Content ID being voted by the content providers.</param>
/// <param name="ContentClients">Clients requesting the content.</param>
/// <returns>TRUE if the voting allows the distribution of the content item. Should this value be
false, the content of the DecryptedFile out variable should be discarded.</returns>
protected ContentItem CheckContentProviderAuthorization(int ContentID, List<string>
RequestingClients)
{
    if (RegisteredContentProviders.Count == 0) throw new Exception("Please register at least one
content provider.");

    string S = RegisteredContentProviders[0].AuthorizeContentDistribution(ContentID,
RequestingClients);
    for (int i = 1; i < RegisteredContentProviders.Count; i++)
    {
        S = S.XOR(RegisteredContentProviders[i].AuthorizeContentDistribution(ContentID,
RequestingClients));
    }
    // S should now hold the original secret
    Debug.WriteLine("Decrypted password (after voting): " + S);

    // Let's see what we know about the current content item
    ContentItem original = RegisteredContentItems[ContentID];
    // In order not to make changes into the registered list of content items, we will work on a
copy
    ContentItem decrypteditem = new ContentItem(original);
    // Decrypt content (using S as a password)
    ContentCrypto.DecryptContent(decrypteditem, S);

    // Verify checksum
    if (decrypteditem.ContentHash == RegisteredContentHashes[ContentID])
    {
        // We're ok, the voting allows us to distribute the content
        return decrypteditem;
    }
    else

```



```

    {
        return null;
    }
}

/// <summary>
/// Performs a secure broadcast to the specified clients
/// </summary>
/// <param name="ContentID">ID of the content that will be broadcast</param>
/// <param name="RequestingClients">List of clients (names) that will receive the content</param>
public void BroadcastContent(int ContentID, List<string> RequestingClients)
{
    // Before we can do anything, we must first check whether all content providers agree to grant
the content
    // to the all clients we have.
    #region Check Authorization
    DateTime start = DateTime.Now;
    ContentItem BroadcastMessage = CheckContentProviderAuthorization(ContentID, RequestingClients);
    if (BroadcastMessage == null) throw new Exception("Content with ID " + ContentID + " hasn't
been authorized for all clients.");
    #endregion

    // Generate a random session password that will encrypt the content. The password will be
protected by the lock
    // X which will be opened by authorized clients only.
    string SessionKey = RandomPassword.Generate(8);

    // Calculate lock X
    #region Calculate lock X
    BigInteger X = ComputeLockX(SessionKey, RequestingClients);
    #endregion

    // Compute content license (lambda)
    string LicenseHash = Licensing.GenerateContentLicenseHash(BroadcastMessage, X);

    // Digitally sign license hash
    ASCIIEncoding ae = new ASCIIEncoding();
    byte[] Signature = ContentMasterRSA.SignData(ae.GetBytes(LicenseHash), "MD5");
    ContentMetadata License = new ContentMetadata();
    License["LicenseHash"] = LicenseHash;
    License["Signature"] = Convert.ToBase64String(Signature);

    // Encrypt content with session password
    ContentCrypto.EncryptContent(BroadcastMessage, SessionKey);
    DateTime end = DateTime.Now;
    Debug.WriteLine("Content generated for broadcast in: " + (end - start).TotalMilliseconds + "
ms. =====");

    // Send (broadcast) everything to each client (i.e. the encrypted content and the X)
    foreach (ContentClient cc in RegisteredContentClients.Values)
    {
        // TODO: Uncomment this line when not benchmarking
        cc.ReceiveContentFromMaster(new ContentItem(BroadcastMessage), X, License,
ContentMasterRSA.ExportParameters(false));
    }
}
#endregion

#region Helper Methods
/// <summary>
/// Computes the lock X for the current list of clients requesting
/// </summary>
/// <param name="SessionKey">Key to be protected by the lock X. It will be recovered by any of the
clients in the supplied list.</param>
/// <param name="RequestingClients">Clients requesting the current content. Only these clients will
be able to recover the session key, based on the lock X.</param>
/// <returns>The lock X (see the Chinese Remainder Theorem)</returns>
public BigInteger ComputeLockX(string SessionKey, List<string> RequestingClients)
{
    RSACryptoServiceProvider rsa = new RSACryptoServiceProvider();
    ASCIIEncoding ae = new ASCIIEncoding();
    SortedList<string, BigInteger> Rs = new SortedList<string, BigInteger>();

    // Compute the R coefficient for each client, like so: Ri = SessionKey encrypted with the public
key of the client i
    foreach (string rc in RequestingClients)
    {
        try
        {
            ContentClient client = RegisteredContentClients[rc];
            // Get the public key for the current client
            RSAParameters ClientPublicKey = client.PublicKey;

```

```

        // Initialize our local RSA provider so we can work with it
        rsa.ImportParameters(ClientPublicKey);
        // Encrypt the session key
        byte[] R = rsa.Encrypt(ae.GetBytes(SessionKey), false);
        // Store R in a list of our own
        Rs[client.Name] = new BigInteger(R);
    }
    catch (KeyNotFoundException)
    {
        throw new ArgumentException("Client " + rc + " is not registered with the Content
Master.");
    }
}
// We have the R values, the N values (generated at client registration), so we can finally
compute the lock X using the Chinese Remainder Theorem
ChineseRemainderTheorem crt = new ChineseRemainderTheorem();
foreach (string client in RequestingClients)
{
    crt.AddLinearCongruence(Rs[client], RegisteredContentClients[client].N);
}
return crt.Solve();
}

/// <summary>
/// Returns the current ContentID then increments it.
/// </summary>
protected int GetNewContentID()
{
    return ContentID++;
}

/// <summary>
/// Gets the public key of the content master
/// </summary>
/// <returns>The public key</returns>
public RSAParameters PublicKey
{
    get
    {
        return ContentMasterRSA.ExportParameters(false);
    }
}
#endregion
}
}

```

### 5.2.3 Operații matematice

Operațiile matematice care implică folosirea numerelor foarte mari precum și clasa răspunzătoare pentru rezolvarea congruențelor liniare sunt grupate într-un modul specializat. Pentru lucrul cu numere foarte mari s-a folosit proiectul BigInteger [99], care reprezintă numerele sub formă de string. Rezolvarea congruențelor liniare se face cu clasa LinearCongruence, redată mai jos.

```

namespace SCADDICMath
{
    /// <summary>
    /// The LinearCongruence class solves a linear congruence, as follows:
    ///
    /// If A and B are any integers and N is a positive integer, the linear congruence is:
    ///  $Ax \equiv B \pmod{N}$ 
    ///
    /// Conditions:
    /// B is divided by GCD(A,N) where GCD stands for Greatest Common Divider.
    ///
    /// Solutions:
    /// The set of all solutions is  $\{x_0 + k * N/d\}$  where k is an integer number and  $d = \text{GCD}(A,N)$ .
    /// Keep only the positive solution less than N.
    ///
    ///  $x_0$  is calculated as follows:

```

```

/// Using the Extended Euclidean Algorithm, find two integers R and S such that  $R * A + S * N = d$ .
///  $x_0$  is then  $R * B / d$ 
///
/// The total number of valid solutions is d.
/// </summary>
/// <seealso
cref="http://en.wikipedia.org/wiki/Linear_congruence_theorem#Solving_a_linear_congruence"/>
public class LinearCongruence
{
    #region Congruence Parameters
    /// <summary>
    /// A
    /// </summary>
    public BigInteger A
    {
        get;
        set;
    }

    /// <summary>
    /// B
    /// </summary>
    public BigInteger B
    {
        get;
        set;
    }

    /// <summary>
    /// N
    /// </summary>
    public BigInteger N
    {
        get;
        set;
    }
    #endregion

    #region Public Constructors
    public LinearCongruence(BigInteger B, BigInteger N)
    {
        this.A = 1;
        this.B = B;
        this.N = N;
    }

    public LinearCongruence(BigInteger A, BigInteger B, BigInteger N)
    {
        this.A = A;
        this.B = B;
        this.N = N;
    }
    #endregion

    /// <summary>
    /// Solves a linear congruence in the form  $Ax \equiv B \pmod{N}$ 
    /// </summary>
    /// <returns>An array containing the solutions (if any).</returns>
    public BigInteger[] Solve()
    {
        BigInteger d = A.gcd(N);
        // Check whether we have solutions
        if (B % d != 0) throw new ArgumentException("There are no solutions for linear congruence " + A +
"x \equiv " + B + " (mod " + N + ").");
        // Apply the extended Euclidean algorithm to find out R and S
        BigInteger R, S;
        ExtendedGCD(A, N, out R, out S);
        // Calculate  $x_0$ 
        BigInteger x0 = R * (BigInteger)(B / d);
        // Generate the set of all solutions (there are exactly "d" solutions)
        BigInteger[] Solutions = new BigInteger[d.IntValue()];
        int iSol = 0, k = 0; // This shows the next available index in the solution array

        BigInteger incr = (BigInteger)(N / d);
        while (iSol < d)
        {
            // Calculate the possible solution
            BigInteger Sol = x0 + (k++) * incr;
            // The solution must be positive and less than N (since we're talking modulo N)
            if ((Sol > 0) && (Sol < N))
            {
                Solutions[iSol++] = Sol;
            }
        }
    }
}

```

```

    }
    // Return the solutions of the linear congruence
    return Solutions;
}

/// <summary>
/// Implements the extended Euclidean algorithm.
/// Finds r and s such that Ar + Bs = GCD(A,B) where GCD stands for the Greatest Common Divider.
/// </summary>
/// <param name="A">A</param>
/// <param name="B">B</param>
/// <param name="r">r</param>
/// <param name="s">s</param>
protected void ExtendedGCD(BigInteger A, BigInteger B, out BigInteger r, out BigInteger s)
{
    BigInteger x = 0, y = 1, lastx = 1, lasty = 0;

    while (B != 0)
    {
        BigInteger temp = B;
        BigInteger quotient = (BigInteger)(A / B);
        B = A % B;
        A = temp;

        temp = x;
        x = lastx - quotient * x;
        lastx = temp;

        temp = y;
        y = lasty - quotient * y;
        lasty = temp;
    }
    r = lastx;
    s = lasty;
}

/// <summary>
/// Implements the Euclidean algorithm (finds the greatest common divisor of two numbers).
/// </summary>
/// <param name="A">A</param>
/// <param name="B">B</param>
/// <returns>Greatest common divisor of A and B.</returns>
protected BigInteger GCD(BigInteger A, BigInteger B)
{
    BigInteger Temp;
    while (B != 0)
    {
        Temp = B;
        B = A % B;
        A = Temp;
    }
    return A;
}

#region Overriden Methods
public override string ToString()
{
    return A + "x ≡ " + B + " (mod " + N + ")";
}
#endregion
}
}

```

Rezolvarea sistemelor de congruențe liniare (teorema chineză a restului), se face cu clasa ChineseRemainderTheorem, redată mai jos.

```

namespace SCADDICMath
{
    /// <summary>
    /// The ChineseRemainderTheorem class solves a system of linear congruences such as:
    /// X ≡ A1 (mod m1)
    /// X ≡ A2 (mod m2)
    /// ...
    /// X ≡ Ar (mod mr)
    ///
    /// The solution X can be calculated as (a1 * b1 * M / m1 + ... + a2 * b2 * M / mr) mod M.

```

```

/// where: M is defined as m1 * m2 * ... * mr
/// Each b term can be deduced by solving this linear congruence: b * M / m ≡ 1 (mod m)
/// </summary>
/// <seealso cref="http://mathworld.wolfram.com/ChineseRemainderTheorem.html"/>
public class ChineseRemainderTheorem
{
    #region Local variables
    protected List<LinearCongruence> LinCong;
    #endregion

    #region Public constructor
    public ChineseRemainderTheorem()
    {
        LinCong = new List<LinearCongruence>();
    }
    #endregion

    /// <summary>
    /// Adds a new linear congruence to the equation system, in the form  $x \equiv A \pmod{m}$ 
    /// </summary>
    /// <param name="A">A</param>
    /// <param name="m">m</param>
    public void AddLinearCongruence(BigInteger A, BigInteger m)
    {
        LinCong.Add(new LinearCongruence(A, m));
    }

    public BigInteger Solve()
    {
        // All linear congruences must have been previously loaded
        if(LinCong.Count == 0) throw new ArgumentException("Please specify at least one linear
congruence.");
        // Calculate M = m1 * m2 * ... * mr where r is the number of linear congruences we have in the
system
        BigInteger M = GetM();
        // Calculate Term = a1 * b1 * M / m1 + ... + ar * br * M / mr where r is the number of linear
congruences we have in the system
        BigInteger Term = 0;
        for (int r = 0; r < LinCong.Count; r++)
        {
            BigInteger a = LinCong[r].B;
            BigInteger m = LinCong[r].N;

            // We need to calculate the product for the current r: a * b * M / m
            // b is deduced by solving the linear congruence b * M / m ≡ 1 (mod m)
            LinearCongruence tempC = new LinearCongruence((BigInteger)(M / m), 1, m);
            BigInteger[] solutions = tempC.Solve();
            if (solutions.Length != 1) throw new ArgumentException("Equation is not solvable. Please
check whether all N's are pairwise coprime numbers.");
            // We should have just one solution
            BigInteger b = solutions[0];

            BigInteger Product = a * b * M / m;
            Term += Product;
        }
        // Now we can determine the solution of the equation system: X = Term mod M
        BigInteger X = Term % M;
        return X;
    }

    /// <summary>
    /// M is m1 * m2 * ... * mr where r is the number of equations
    /// </summary>
    protected BigInteger GetM()
    {
        BigInteger result = 1;
        foreach (LinearCongruence lc in LinCong)
        {
            result *= lc.N;
        }
        return result;
    }
}
}

```

### 5.3 Măsurători de performanță

Pentru a valida modelul teoretic și pentru a demonstra fezabilitatea soluției propuse, prototipul a fost supus unor măsurători de performanță. Performanța a fost măsurată pentru faza de inițializare (care are loc o singură dată, la pornirea sistemului) și pentru faza de operare. Pe de o parte, faza de inițializare are o importanță secundară datorită caracterului său tranzitoriu, însă măsurarea acestor timpi ajută la formarea unei opinii legate de timpul de cold-start al unui dispozitiv portabil care ar implementa sistemul SCADDIC. Pe de altă parte, măsurarea performanței în faza de operare oferă indicii precise asupra comportamentului sistemului în timpul funcționării de fapt.

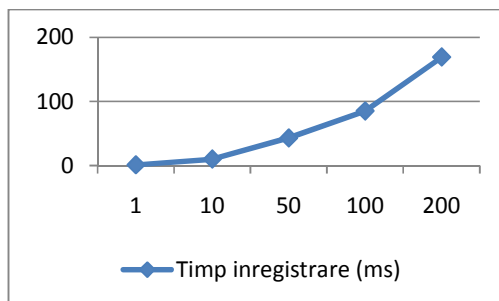
Codul C# al prototipului a fost compilat cu Visual Studio 2008 pe .NET Framework 3.5 și a fost rulat pe un calculator portabil de tip Lenovo T60 (1,80 Ghz, 2 GB RAM). Timpul măsurat în diverse stadii reprezintă duratele necesare diverselor operații criptografice și nu include timpul petrecut de conținut în tranzit prin diverse medii de transmisie.

#### 5.3.1 Faza de inițializare

Faza de inițializare a sistemului SCADDIC începe cu instanțierea Content Master-ului, operație urmată la scurt timp de **înregistrarea furnizorilor de conținut** (Content Provider). Prin operația de înregistrare, furnizorii își anunță prezența, participând ulterior la schema de împărțire a secretului și la votul acordării de permisiuni clienților care solicită primirea unui conținut digital.

Se observă că timpul necesar pentru înregistrarea furnizorilor de conținut crește cvasi-liniar cu numărul furnizorilor de conținut, timpul mediu de înregistrare fiind de 0,9 ms / furnizor.

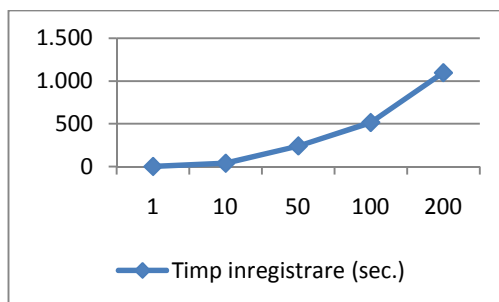
| Numărul furnizorilor de conținut | Timp de înregistrare (ms) |
|----------------------------------|---------------------------|
| 1                                | 1                         |
| 10                               | 10                        |
| 50                               | 43                        |
| 100                              | 85                        |
| 200                              | 169                       |



Tabel 3. Evoluția timpului de înregistrare al furnizorilor de conținut

**Inițializarea clienților** presupune o serie de operații criptografice consumatoare de timp. Aceste operații se efectuează în mod normal la momentul fabricației (în cazul dispozitivelor embedded), deoarece perechea de chei publică și privată nu se modifică pe durata vieții dispozitivului. În cazul prototipului SCADDIC acest lucru are loc la fiecare inițializare a sistemului.

| Număr de clienți | Timp de înregistrare (s) |
|------------------|--------------------------|
| 1                | 1.2                      |
| 10               | 38.6                     |
| 50               | 240.3                    |
| 100              | 515.1                    |
| 200              | 1097.1                   |

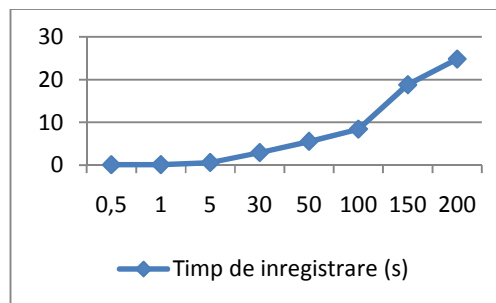


Tabel 4. Evoluția timpului de înregistrare pentru clienți

Timpul necesar pentru înregistrarea clienților este semnificativ mai lung decât cel necesar pentru înregistrarea furnizorilor de conținut datorită operațiilor cu criptografie asimetrică implicate. Timpul mediu de înregistrare pentru un client este de 4,1 sec.

**Înregistrarea conținutului** este operația prin care un furnizor își anunță intenția de a face disponibil conținutul digital aflat în posesia sa. Acest pas are loc după înregistrarea furnizorilor de conținut, deoarece drepturile de distribuție se pot partaja între furnizori. Pentru a evita scurgerile de informații și pentru a proteja secretul conținutului aflat în conservare, adică înainte de a fi oferit clienților, Content Master-ul va cripta fiecare conținut cu o cheie aleatoare care ulterior va fi împărțită între furnizorii înregistrați.

| Dimensiune conținut (MB) | Timp înregistrare (sec) |
|--------------------------|-------------------------|
| 0.5                      | 67                      |
| 1                        | 98                      |
| 5                        | 580                     |
| 30                       | 2918                    |
| 50                       | 5525                    |
| 100                      | 8393                    |
| 150                      | 18775                   |
| 200                      | 24794                   |



Tabel 5. Timpul de înregistrare a conținutului

Timpul necesar pentru înregistrarea conținutului crește cvasi-liniar cu dimensiunea. Timpul mediu de înregistrare este de 110 ms per MB.

### 5.3.2 Faza de operare

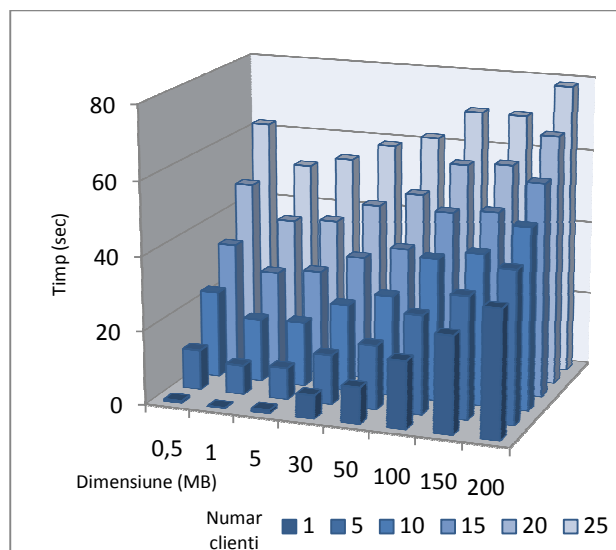
Faza de operare presupune distribuția conținutului în sine, adică trimiterea conținutului digital la clienți. Distribuția este o operație complexă care cuprinde următoarele task-uri:

1. Cererea autorizației furnizorului de conținut
2. Generarea cheii de sesiune
3. Calcularea lacătului X
4. Calcularea valorii de dispersie a conținutului.
5. Criptarea conținutului cu cheia de sesiune.



Evidențierea performanței pentru fiecare dintre aceste task-uri ar dilua măsurarea performanței, de aceea distribuția de conținut a fost simulată ca un tot unitar într-o varietate de condiții, începând de la cazul simplu de 1 client cu conținut de distribuit de 0,5 MB până la 25 de clienți cu conținut de 200MB. Tabelul următor prezintă rezultatele măsurătorilor pe calculatorul de test, în diverse scenarii.

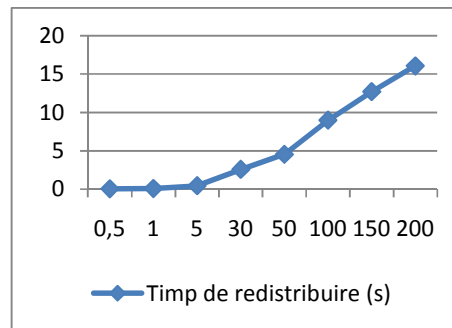
| Număr clienți / Dimensiune conținut (MB) | 1     | 5     | 10    | 15    | 20    | 25    |
|--|-------|-------|-------|-------|-------|-------|
| 0.5                                      | 0.68  | 10.86 | 23.48 | 33.52 | 47.58 | 62.11 |
| 1  | 0.21  | 7.96  | 17.12 | 26.85 | 38.41 | 51.23 |
| 5  | 0.97  | 8.63  | 17.54 | 28.34 | 39.38 | 54.00 |
| 30                                       | 6.37  | 13.47 | 23.6  | 33.37 | 44.80 | 58.77 |
| 50                                       | 9.90  | 17.39 | 27.27 | 36.84 | 48.84 | 61.86 |
| 100                                      | 1.31  | 26.81 | 38.55 | 47.91 | 58.06 | 69.89 |
| 150                                      | 2.28  | 33.14 | 41.00 | 49.07 | 58.95 | 69.79 |
| 200                                      | 34.56 | 41.24 | 49.19 | 57.85 | 67.71 | 78.62 |



Tabel 6. Timpul de distribuție în funcție de numărul clienților și dimensiunea conținutului digital

Timpul necesar pentru pregătirea conținutului pentru difuzare (broadcast) crește liniar cu numărul de clienți și subliniar cu dimensiunea datelor de distribuit. Odată cu creșterea numărului de clienți, sistemul de distribuție este avantajat de dimensiuni mari ale conținutului.

| Dimensiune conținut (MB) | Timp de redistribuție (s) |
|--------------------------|---------------------------|
| 0.5                      | 0.05                      |
| 1                        | 0.09                      |
| 5                        | 0.45                      |
| 30                       | 2.57                      |
| 50                       | 4.53                      |
| 100                      | 8.98                      |
| 150                      | 12.69                     |
| 200                      | 16.04                     |



Tabel 7. Timpul de redistribuție a conținutului

Redistribuirea conținutului este o operație care implică două părți, emițătorul și receptorul. Emițătorul este entitatea în posesia căreia se află conținutul și în plus are drept de redistribuire în mod legal. Receptorul reprezintă entitatea care primește conținutul în baza unei înțelegeri prealabile cu emițătorul. În cazul redistribuirii conținutului nu este nevoie să se calculeze lacătul X, însă pentru protejarea conținutului acesta se va cripta cu o cheie de sesiune care la rândul ei este criptată cu cheia publică a receptorului.

Timpul necesar pentru redistribuirea conținutului crește liniar cu dimensiunea acestuia. Timpul mediu de redistribuție este de 88 ms per MB.

#### 5.4 Avantaje ale sistemului de distribuție propus

Pentru a evalua performanța sistemului de distribuție propus și avantajele sale din punctul de vedere al scalabilității și pentru a putea raporta rezultatele la implementarea din [63], este necesară definirea unei metrice după care să se evalueze performanța. Pentru scopul paragrafului de față, convenim să numim metoda implementată în [63] „unicast” datorită faptului că transmisia datelor se

face în mod punct la punct, iar implementarea propusă în această lucrare se va numi „broadcast” datorită faptului că un număr multiplu de clienți poate fi deservit la un moment dat.

Metrica pe care o propunem aici ia în calcul o serie de parametri comuni celor două implementări (unicast și broadcast), după cum urmează:

1. Timpul de generare a cheii de sesiune
2. Timpul necesar criptării conținutului cu cheia de sesiune
3. Timpul necesar criptării cheii de sesiune (fapt care înseamnă crearea lacătului X care va fi ulterior folosit pentru recuperarea cheii de sesiune)
4. Timpul necesar distribuției conținutului criptat și a cheii de sesiune, de asemenea criptată

Astfel, timpul total de distribuție a unui conținut digital este exprimat de următoarea formulă:

$$T_{dist} = \sum_{i=1}^N Tsk_i + \sum_{i=1}^N Tce_i + \sum_{i=1}^M Tx_i + \sum_{i=1}^N Tsend_i$$

(Ecuția 11)

Unde:

- $T_{dist}$  – Timpul total necesar pentru a transmite toate instanțele conținutului
- $T_{sk}$  – Timpul necesar pentru a genera cheia de sesiune
- $T_{ce}$  – Timpul necesar pentru a cripta conținutul cu cheia de sesiune
- $T_x$  – Timpul necesar pentru a genera lacătul C sau pentru a cripta cheia de sesiune cu cheia publică a receptorului (în cazul redistribuției)
- $T_{send}$  – Timpul necesar pentru a transmite o instanță a conținutului către client
- N – Numărul de instanțe ale conținutului care trebuie transmise
- M – Numărul de clienți la care conținutul trebuie transmis

Pentru scenariul broadcast, numărul instanțelor de conținut care trebuie transmise este  $N = 1$ , deci (Ecuția 11) devine:

$$T_{broadcast} = Tsk + Tce + \sum_{i=1}^M Tx_i + Tsend$$

(Ecuția 12)

Unde:

- $T_{\text{broadcast}}$  – Timpul total necesar pentru a difuza conținutul

Pentru scenariul unicast, numărul de instanțe ale conținutului care trebuie transmise este egal cu numărul clienților, astfel încât  $N = M$ . (Ecuția **11**) devine astfel:

$$T_{\text{unicast}} = \sum_{i=1}^N (Tsk_i + Tce_i + Tx_i + Tsend_i)$$

(Ecuția 13)

Unde:

- $T_{\text{unicast}}$  – Timpul total necesar pentru a trimite conținutul fiecărui client în parte (unicast)

Cele două ecuații scot la iveală faptul că scenariul broadcast poate fi avantajat din punct de vedere al timpului, în condiții de încărcare identică. Pentru a avea o imagine asupra diferențelor între cele două tipuri de sisteme de distribuție, simularea s-a făcut variind dimensiunea conținutului și numărul de clienți care solicită același conținut la un moment dat. Ca urmare a simulărilor premergătoare cu module din cadrul sistemului SCADDIC, s-au obținut următoarele:

- Cheia de sesiune se generează într-un timp de 2 ms.
- Conținutul este criptat la o rată de 73 ms per MB, folosind un algoritm asimetric.
- Lacătul X se calculează în 117 ms pentru un client și are o creștere liniară cu numărul de clienți.
- Pentru unicast, lacătul X nu este calculat și este înlocuit de o criptare cu chei asimetrice, care se consideră că durează tot 117 ms.
- Viteza legăturii dintre emitent și receptor este de 200KB/sec, o valoare considerată tipică pentru un dispozitiv embedded de astăzi.

Timpii redați mai jos au fost obținuți după medierea rulărilor repetate ale simulării.

|                       | 0.5MB | 1MB | 5MB | 30MB | 50MB | 100MB | 150MB | 200MB |
|-----------------------|-------|-----|-----|------|------|-------|-------|-------|
| Broadcast (1 client)  | 3     | 5   | 26  | 156  | 260  | 519   | 779   | 1039  |
| Unicast (1 client)    | 3     | 5   | 26  | 156  | 260  | 519   | 779   | 1039  |
| Broadcast (5 clienți) | 6     | 8   | 29  | 159  | 263  | 522   | 782   | 1042  |
| Unicast (5 clienți)   | 14    | 27  | 131 | 780  | 1299 | 2597  | 3895  | 5194  |

|                         |      |      |      |       |       |       |       |        |
|-------------------------|------|------|------|-------|-------|-------|-------|--------|
| Broadcast (10 clienți)  | 14   | 17   | 38   | 168   | 271   | 531   | 791   | 1050   |
| Unicast (10 clienți)    | 27   | 53   | 261  | 1559  | 2598  | 5194  | 7791  | 10387  |
| Broadcast (50 clienți)  | 295  | 298  | 318  | 448   | 552   | 812   | 1071  | 1331   |
| Unicast (50 clienți)    | 137  | 267  | 1305 | 7796  | 12989 | 25972 | 38954 | 51937  |
| Broadcast (100 clienți) | 1173 | 1175 | 1196 | 1326  | 1430  | 1689  | 1949  | 2209   |
| Unicast (100 clienți)   | 273  | 533  | 2610 | 15593 | 25979 | 51944 | 77909 | 103874 |

Tabel 8. Performanța în secunde pentru diverse tipuri de conținut și număr de clienți

Pentru un singur client, comportamentul în modul broadcast este același cu cel din modul unicast, deci nu se observă un câștig de viteză (Fig. 37). Modul broadcast în sine presupune existența a cel puțin doi clienți care solicită simultan exact același conținut, deci în cazul unei singure cereri modul broadcast se reduce ca funcționare la modul unicast.

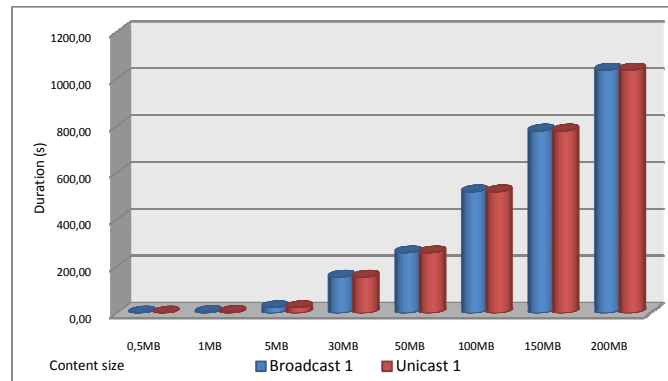


Fig. 37 Comparație Broadcast – Unicast pentru un client

Creșterea numărului de clienți care solicită simultan același conținut digital duce la creșterea performanței sistemului broadcast în comparație cu cel unicast, care în acest caz efectuează o singură dată operațiile criptografice costisitoare. Din figurile Fig. 38 - Fig. 41 se observă creșterea tot mai accentuată a performanței sistemului broadcast odată cu creșterea numărului de clienți deserviți.

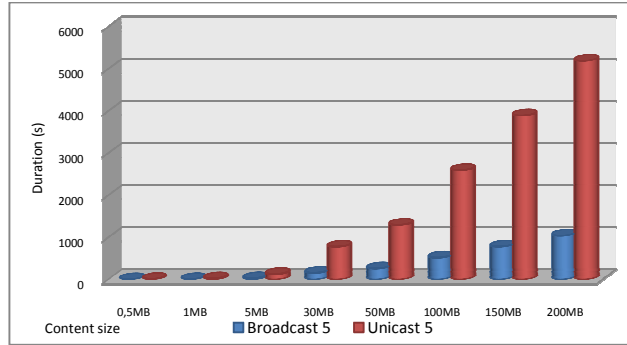


Fig. 38 Comparație Broadcast – Unicast pentru 5 clienți

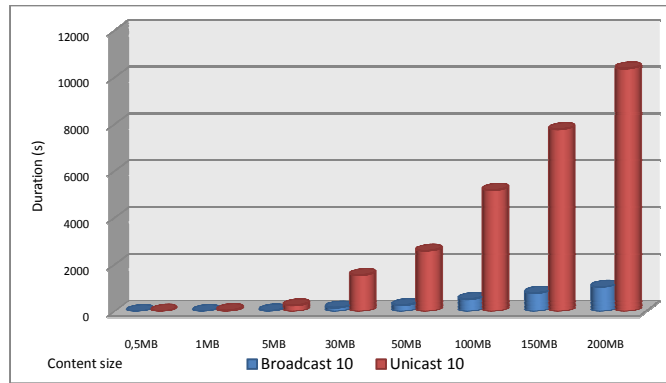


Fig. 39 Comparație Broadcast – Unicast pentru 10 clienți

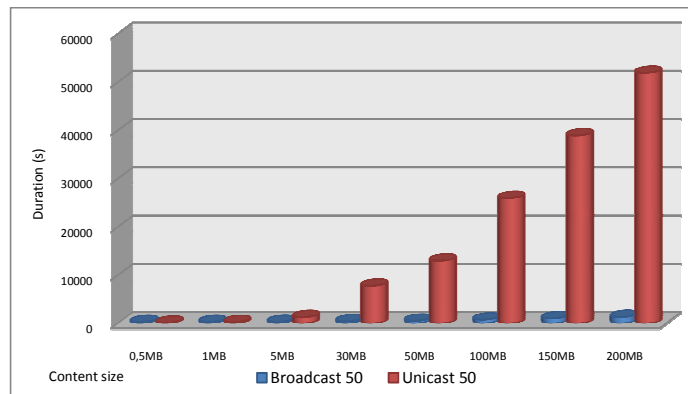


Fig. 40 Comparație Broadcast – Unicast pentru 50 de clienți

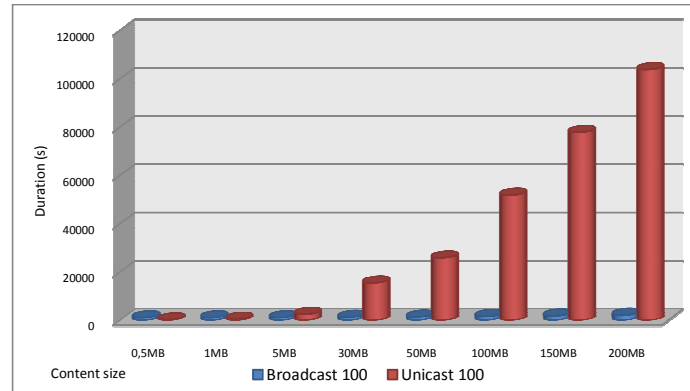


Fig. 41 Comparație Broadcast – Unicast pentru 100 de clienți

## 5.5 Concluzii

Sistemele de distribuție digitală a conținutului existente astăzi sunt în general orientate pe deservirea unui singur client la un moment dat. Arhitecturile de acest tip prezintă neajunsul de a nu fi scalabile, deoarece operațiile de criptare și transmitere a conținutului digital se repetă pentru fiecare client în parte.

Arhitectura propusă în cadrul acestui capitol se construiește pe sistemul de distribuție prezentat în [63] și are avantajul unei performanțe ameliorate și a posibilității de arbitrarie criptografică a drepturilor partajate de furnizori. Performanța arhitecturii este superioară datorită modelului de distribuție de tip broadcast. Acesta presupune codificarea conținutului prin procedee criptografice, permițând astfel ca transmiterea sa să fie simultană și securizată către mai mulți clienți. În plus, acest model de distribuție permite activarea rolului de proxy, adică clienți aflați mai aproape de centrul de distribuție pot acționa ca releu de distribuție, degrevând serverul de sarcini.

Avantajele sistemului de distribuție digitală a conținutului propus pot fi summarize după cum urmează:

- **Scalabilitate** – În momentele de vârf, când serverul este asaltat de cereri performanța sistemului de broadcast propus este foarte bună deoarece distribuția de conținut se face simultan la un număr de clienți cu cereri similare, acumulate la intervale de timp prestabilite. Sistemul propus poate fi de până la câteva ordine de mărime mai scalabil decât varianta din [63], funcție de factori cum sunt numărul de clienți, dimensiunea conținutului de distribuit și viteza conexiunii dintre client și server. Acest fapt a fost dovedit experimental.

- **Distribuție asincronă** – Deoarece conținutul digital este criptat și este disponibil doar unor clienți selectați de către server pe diverse criterii, fluxul digital poate fi răspândit (broadcast) prin rețea fără a exista pericolul ca un client neautorizat să-l poată reda. Sarcina de distribuție care revine în mod tradițional serverului poate fi preluată cu succes de către orice clienți aflați mai aproape de receptor. În acest fel, serverul este liber să efectueze alte operații rezultând un factor de scalabilitate bun.
- **Colaborare între deținătorii de copyright** – Uneori, entități diferite pot deține drepturile asupra diverselor părți ale conținutului digital (coloană sonoră, imagine, traducere, prezentare grafică, etc.), accesul la conținut făcându-se când toate entitățile își dau acordul. Alteori, anumite entități pot cere a fi informate asupra modului în care conținutul digital este accesat. În toate aceste cazuri arhitectura propusă este capabilă să partajeze cu succes drepturile asupra conținutului prin tehnica „secret splitting” și să ofere acces fizic la acesta doar când participanții sunt cu toții de acord.

Tot în cadrul acestui capitol s-a propus o metrică pentru evaluarea performanței sistemului de distribuție. Aceasta ține cont de parametri criptografici din cadrul sistemului și care se adaptează funcție de modelul analizat, mai precis unicast sau broadcast. Această metrică reprezintă numitorul comun ce era necesar pentru a putea compara două sisteme competitive, similare ca funcționalitate dar diferite ca design.



## 6. Concluzii finale și perspective

### 6.1 Concluzii finale

În societatea informatizată a zilelor noastre, securitatea și buna funcționare a sistemelor de comunicație este o problemă de primă importanță. Afacerile noastre, confortul nostru și uneori chiar și viața depind în mare măsură de comunicație, de aceea o atenție deosebită trebuie acordată disponibilității și scalabilității sistemelor care asigură comunicația.

Atacurile reprezintă acțiuni subversive menite să compromită buna funcționare a sistemelor, fiind lansate de indivizi sau grupări care urmăresc obținerea de avantaje materiale sau de alt fel, în detrimentul masei de utilizatori. Deși la nivel legal există metode de constrângere și de pedepsire a acestor practici, acest lucru nu este suficient în majoritatea cazurilor, deoarece atacul poate avea loc de la distanță, în zone cu jurisdicție diferită, inițiatorii putându-și pierde astfel urma. Din punct de vedere tehnic sistemele trebuie să fie pregătite pentru a reduce cât mai mult suprafața de atac și de a limita efectele unui atac care a avut deja loc. Securizarea căilor de comunicație este un domeniu în care contribuțiile din literatură sunt variate. Unul dintre domeniile în care încă există curențe în abordare este creșterea disponibilității și asigurarea scalabilității sistemelor de comunicație, această nișă fiind acoperită de prezenta lucrare de cercetare.

Lucrarea de față a avut în vedere un orizont larg de sisteme:

- Sistemele de autentificare
- Sistemele de tip Single Sign-On (SSO)
- Rețelele de comunicație mobilă de tip GSM
- Sistemele de distribuție a conținutului digital

În cadrul lucrării s-a făcut o descriere a fiecărui sistem amintit și a scos în evidență deficiențele existente în prezent, astfel:

- Sistemele de autentificare, sistemele Single Sign-On (SSO) și rețelele GSM sunt vulnerabile la atacuri de tip DoS. Vulnerabilitatea apare din cauza lipsei de control asupra resurselor alocate participanților la comunicație și are de cele mai multe ori ca rezultat degradarea severă a serviciului sau blocarea lui pe perioada desfășurării atacului.
- Sistemele de distribuție a conținutului actuale sunt puțin scalabile și indirect sunt susceptibile la atacuri DoS.

Contribuțiile aduse de această lucrare cu privire la detectarea timpurie și împiedicarea atacurilor de tip DoS pot fi sumarizate după cum urmează:

- Au fost dezvoltate tehnologiile Threshold Puzzles și Adaptive Threshold Puzzles, pentru a acoperi situațiile în care tehnologia Client Puzzles nu este eficientă [11] [12] [13].
- S-au introdus două versiuni ale schimbului de mesaje Handshake din cadrul protocolului SSL, astfel încât acesta să suporte tehnologiile Threshold Puzzles și Adaptive Threshold Puzzles.
- S-a evidențiat posibilitatea ca atacurile DoS să fie detectate cu o anumită probabilitate înainte ca acestea să aibă loc prin aplicarea inferenței bayesiene [13].
- S-a introdus SSO-Sense, un modul de estimare euristică a nivelului de risc pentru sistemele de autentificare [13].

În ceea ce privește siguranța și disponibilitatea rețelelor GSM, lucrarea aduce următoarele contribuții:

- Vulnerabilitățile sistemului GSM au fost evaluate cu o metodă de clasificare numită DREAD. Rezultatul deloc surprinzător a fost că cea mai periculoasă amenințare la adresa GSM este atacul DoS (denial of service) [16].
- S-a arătat exact cauza pentru care sistemul GSM este vulnerabil la atacuri DoS, și anume lipsa preautentificării dispozitivelor mobile care pot cere alocarea de resurse în mod repetat [103].
- S-a arătat de asemenea că un singur atacator este capabil de a dezactiva o întreagă celulă GSM și în anumite cazuri chiar și celulele adiacente [103].
- Deoarece nu sunt implicate costuri financiare (nu se efectuează nici un apel în sine), costul efectiv al lansării un atac devastator este zero din punctul de vedere al plăților către operatorul vizat [103] [14].
- În scopul creșterii rezistenței la atacuri DoS, s-a propus introducerea preautentificării terminalelor GSM în cazul inițierii unui apel de către acestea. Preautentificarea impune introducerea în protocolul Channel Assignment a unui mesaj nou de tip baliză numit PREAUTHENTICATION BEACON care transmite terminalelor mobile spre rezolvare o mică problemă, după modelul Threshold Puzzles [14] [16].

Lucrarea a introdus de asemenea și o arhitectură de distribuție a conținutului digital cu următoarele avantaje [15]:

- **Scalabilitate** – În momentele de vârf, când serverul este asaltat de cereri performanța sistemului de broadcast propus este foarte bună

deoarece distribuția de conținut se face simultan la un număr de clienți cu cereri similare, acumulate la intervale de timp prestabilite.

- **Distribuție asincronă** – Deoarece conținutul digital este criptat și este disponibil doar unor clienți selectați de către server pe diverse criterii, fluxul digital poate fi răspândit (broadcast) prin rețea fără a exista pericolul ca un client neautorizat să-l poată reda. Sarcina de distribuție care revine în mod tradițional serverului poate fi preluată cu succes de către orice clienți aflați mai aproape de receptor. În acest fel, serverul este liber să efectueze alte operații rezultând un factor de scalabilitate bun.
- **Colaborare între deținătorii de copyright** – Uneori, entități diferite pot deține drepturile asupra diverselor părți ale conținutului digital (coloană sonoră, imagine, traducere, prezentare grafică, etc.), accesul la conținut făcându-se când toate entitățile își dau acordul. Alteori, anumite entități pot cere a fi informate asupra modului în care conținutul digital este accesat. În toate aceste cazuri arhitectura propusă este capabilă să partajeze cu succes drepturile asupra conținutului prin tehnica „secret splitting” și să ofere acces fizic la acesta doar când participanții sunt cu toții de acord.

De asemenea, s-a propus o metrică pentru evaluarea performanței sistemului de distribuție. Aceasta ține cont de parametri criptografici din cadrul sistemului și care se adaptează funcție de modelul analizat, mai precis unicast sau broadcast. Această metrică reprezintă numitorul comun necesar pentru a putea compara sistemele similare ca funcționalitate dar diferite ca design [17].

## 6.2 Perspective

Dacă în anii 1990 și 2000 spectrul radio a fost dominat de generațiile 2/2.5 (GPRS-EDGE) și 3/3.5 (3G-HSDPA) de transmisii de date și comunicații mobile, în perioada următoare spectrul radio va face loc noului venit LTE. Această nouă tehnologie permite rate de download de până la 100 Mbit/sec, uplink de 50 Mbit/sec și timp dus-întors pe interfața radio de sub 10 ms. De asemenea, tehnologia LTE suportă interoperabilitatea cu infrastructurile GSM, CDMA și WCDMA existente.

Cercetarea din perioada următoare vizează următoarele aspecte ale securității LTE:

- Analiza autentificării și distribuției cheilor
- Protecția canalelor de semnalizare în Core Network (NAS) și Radio Network (RRC) și analiza vulnerabilității la atacuri DoS
- Scalabilitatea și capacitatea de ameliorare în caz de atac DoS
- Protecția planului utilizatorului



## Abrevieri și acronime

|   |                 |   |
|---|-----------------|---|
| ◆ | <b>ACL</b>      | Access Control List   |
| ◆ | <b>AES</b>      | Application Environment Specification   |
| ◆ | <b>ANSI</b>     | American National Standards Institute   |
| ◆ | <b>APA</b>      | Authentication and Privilege Attribute  |
| ◆ | <b>API</b>      | Application Programming Interface   |
| ◆ | <b>AS</b>       | Authentication Server   |
| ◆ | <b>ASN.1</b>    | Abstract Syntax Notation 1  |
| ◆ | <b>ATM</b>      | Asynchronous Transfer Mode  |
|   |                 |   |
| ◆ | <b>BAN</b>      | Burrows, Abadi, Needham   |
| ◆ | <b>Bellcore</b> | Bell Communications Research  |
| ◆ | <b>BER</b>      | Basic Encoding Rules  |
| ◆ | <b>BFI</b>      | Swiss Federal Office of Information Technology and<br>Systems                   |
| ◆ | <b>BTS</b>      | Base Transceiver Station (folosit în terminologia<br>GSM)                       |
|   |                 |   |
| ◆ | <b>CA</b>       | Certification Authority   |
| ◆ | <b>CAA</b>      | Certification Authority Agent   |
| ◆ | <b>CAE</b>      | Common Applications Environment   |
| ◆ | <b>CAT</b>      | Common Authentication Technology  |
| ◆ | <b>CBC</b>      | Cipher Block Chaining   |
| ◆ | <b>CCITT</b>    | Consultative Committee on International Telegraphy<br>and Telephony (now ITU-T) |
| ◆ | <b>CD</b>       | Compact Disc  |
| ◆ | <b>CDC</b>      | Certificate Distribution Center   |
| ◆ | <b>CDMF</b>     | Commercial Data Masking Facility  |
| ◆ | <b>CDS</b>      | Cell Discovery Service  |
| ◆ | <b>CEC</b>      | Commission of the European Communities  |
| ◆ | <b>CFB</b>      | Cipher Feedback   |
| ◆ | <b>CSF</b>      | Cryptographic Support Facility  |
| ◆ | <b>CV</b>       | Control Value   |
|   |                 |   |
| ◆ | <b>DAC</b>      | Discretionary Access Control  |
| ◆ | <b>DASS</b>     | Distributed Authentication Security Service                                     |
| ◆ | <b>DCE</b>      | Distributed Computing Environment   |
| ◆ | <b>DEC</b>      | Digital Equipment Corporation   |
| ◆ | <b>DES</b>      | Data Encryption Standard  |
| ◆ | <b>DIT</b>      | Directory Information Tree  |

|   |                |  |
|---|----------------|--|
| ◆ | <b>DNS</b>     | Domain Name Service                                    |
| ◆ | <b>DoC</b>     | U.S. Department of Commerce                            |
| ◆ | <b>DoD</b>     | U.S. Department of Defense                             |
| ◆ | <b>DoD</b>     | U.S. Department of State                               |
| ◆ | <b>DOS</b>     | Atac de tip Denial of Service                          |
| ◆ | <b>DSA</b>     | Digital Signature Algorithm                            |
| ◆ | <b>DSS</b>     | Digital Signature Standard                             |
| ◆ | <b>DSS</b>     | Domain Security Standard                               |
| ◆ | <b>DSSA</b>    | Distributed System Security Architecture               |
| ◆ | <b>DTI</b>     | Directory Information Tree                             |
|   |                |  |
| ◆ | <b>ECB</b>     | Electronic Code Book                                   |
| ◆ | <b>ECMA</b>    | European Computer Manufacturers Association            |
| ◆ | <b>EDI</b>     | Electronic Data Interchange                            |
| ◆ | <b>EES</b>     | Exponential Electronic Signature                       |
| ◆ | <b>EISS</b>    | European Institute for System Security                 |
| ◆ | <b>EKE</b>     | Encrypted Key Exchange                                 |
| ◆ | <b>EPAC</b>    | Extended Privilege Attribute Certificate               |
| ◆ | <b>EU</b>      | European Union   |
|   |                |  |
| ◆ | <b>FAQ</b>     | Frequently Asked Questions                             |
| ◆ | <b>FEAL</b>    | Fast Encryption Algorithm                              |
| ◆ | <b>FIPS</b>    | Federal Information Processing Standard                |
| ◆ | <b>FTP</b>     | File Transfer Protocol                                 |
|   |                |  |
| ◆ | <b>GDA</b>     | Global Domain Agent                                    |
| ◆ | <b>GDS</b>     | Global Domain Service                                  |
| ◆ | <b>GNV</b>     | Gong, Needham, Yahalom                                 |
| ◆ | <b>GSM</b>     | Global System for Mobile Communications                |
| ◆ | <b>GPRS</b>    | General Packet Radio Service                           |
| ◆ | <b>GSS-API</b> | Generic Security Service API                           |
|   |                |  |
| ◆ | <b>HP</b>      | Hewlett-Packard  |
|   |                |  |
| ◆ | <b>IAM</b>     | Institute for Computer Science and Applied Mathematics |
| ◆ | <b>IBM</b>     | International Business Machines Corporation            |
| ◆ | <b>ICL</b>     | International Computers Limited                        |
| ◆ | <b>ICSI</b>    | International Computers Science Institute              |
| ◆ | <b>IDEA</b>    | International Data Encryption Algorithm                |
| ◆ | <b>IDS</b>     | Inter Domain Service                                   |
| ◆ | <b>IEC</b>     | International Electrotechnical Committee               |
| ◆ | <b>IEEE</b>    | Institute of Electrical and Electronic Engineers       |
| ◆ | <b>IETF</b>    | Internet Engineering Task Force                        |
| ◆ | <b>IP</b>      | Internet Protocol                                      |

---

|   |              |  |
|---|--------------|--|
| ♦ | <b>IPSEC</b> | IP Security Protocol                           |
| ♦ | <b>IPST</b>  | IP Secure Tunnel Protocol                      |
| ♦ | <b>IS</b>    | International Standard                         |
| ♦ | <b>ISO</b>   | International Organization for Standardization |
| ♦ | <b>ISODE</b> | ISO Development Environment                    |
| ♦ | <b>IT</b>    | Information Technology                         |
| ♦ | <b>ITU-T</b> | International Telecommunication Union          |
|   |              |  |
| ♦ | <b>JTC1</b>  | Joint Technical Committee 1                    |
|   |              |  |
| ♦ | <b>KDC</b>   | Key Distribution Center                        |
| ♦ | <b>KDS</b>   | Key Distribution Server                        |
| ♦ | <b>KEK</b>   | Key Encryption Key                             |
| ♦ | <b>KTC</b>   | Key Translation Center                         |
|   |              |  |
| ♦ | <b>LAN</b>   | Local Area Network                             |
| ♦ | <b>LEAF</b>  | Login Enrollment Agent Facility                |
| ♦ | <b>LLC</b>   | Logical Link Control                           |
| ♦ | <b>LRA</b>   | Local Registration Authority                   |
| ♦ | <b>LTE</b>   | Long Term Evolution (WiMAX sucesor)            |
|   |              |  |
| ♦ | <b>MAC</b>   | Message Authentication Code                    |
| ♦ | <b>MAN</b>   | Metropolitan Area Network                      |
| ♦ | <b>MD</b>    | Message Digest                                 |
| ♦ | <b>MDC</b>   | Modification Detection Code                    |
| ♦ | <b>MHS</b>   | Message Handling System                        |
| ♦ | <b>MIB</b>   | Management Information Base                    |
| ♦ | <b>MIC</b>   | Message Integrity Code                         |
| ♦ | <b>MIT</b>   | Massachusetts Institute of Technology          |
| ♦ | <b>MKMP</b>  | Modular Key Management Protocol                |
|   |              |  |
| ♦ | <b>NBS</b>   | National Bureau of Standards                   |
| ♦ | <b>NCSC</b>  | National Computer Security Center              |
| ♦ | <b>NCSL</b>  | National Computer Systems Laboratory           |
| ♦ | <b>NetSP</b> | Network Security Program                       |
| ♦ | <b>NII</b>   | National Information Infrastructure            |
| ♦ | <b>NIST</b>  | National Institute of Standards and Technology |
| ♦ | <b>NLSP</b>  | Network Layer Security Protocol                |
| ♦ | <b>NSA</b>   | National Security Agency                       |
|   |              |  |
| ♦ | <b>OFB</b>   | Output Feedback                                |
| ♦ | <b>OMC</b>   | Operations Management Center                   |
| ♦ | <b>OSF</b>   | Open Software Foundation                       |
| ♦ | <b>OSI</b>   | Open Systems Interconnection                   |

---

|   |                    |   |
|---|--------------------|---|
| ◆ | <b>OSI-RM</b>      | OSI Reference Model                                 |
| ◆ |                    |   |
| ◆ | <b>PAC</b>         | Privilege Attribute Certificate                     |
| ◆ | <b>PAS</b>         | Privilege Attribute Server                          |
| ◆ | <b>PC</b>          | Personal Computer                                   |
| ◆ | <b>PEM</b>         | Privacy Enhanced Mail                               |
| ◆ | <b>PGP</b>         | Pretty Good Privacy                                 |
| ◆ | <b>PIN</b>         | Personal Identification Number                      |
| ◆ | <b>PKCS</b>        | Public Key Cryptography Standard                    |
| ◆ | <b>PKM</b>         | Public Key Management                               |
| ◆ | <b>PKP</b>         | Public Key Partners                                 |
| ◆ | <b>PPID</b>        | Primary Principal Identifier                        |
| ◆ | <b>PT</b>          | Privilege Ticket                                    |
| ◆ | <b>PTGT</b>        | Privilege Ticket Granting Ticket                    |
| ◆ | <b>PV</b>          | Protection Value                                    |
| ◆ | <b>PVF</b>         | PAC Validation Facility                             |
| ◆ |                    |   |
| ◆ | <b>RACF</b>        | Resource Access Control Facility                    |
| ◆ | <b>RFC</b>         | Request for Comments                                |
| ◆ | <b>RFT</b>         | Request for Technology                              |
| ◆ | <b>ROM</b>         | Read Only Memory                                    |
| ◆ | <b>RPC</b>         | Remote Procedure Call                               |
| ◆ | <b>RSA</b>         | Rivest, Shamir, Adelman                             |
| ◆ |                    |   |
| ◆ | <b>SACM</b>        | Secure Association Context Manager                  |
| ◆ | <b>SELANE</b>      | Secure Local Area Network Environment               |
| ◆ | <b>SESAME</b>      | Secure European System for Applications in a Multi- |
| ◆ | vendor Environment |   |
| ◆ | <b>SHA</b>         | Secure Hash Algorithm                               |
| ◆ | <b>SHS</b>         | Secure Hash Standard                                |
| ◆ | <b>SKIA</b>        | Secure Key Issuing Authority                        |
| ◆ | <b>SLC</b>         | Secured Logon Coordinator                           |
| ◆ | <b>SMIB</b>        | Security Management Information Base                |
| ◆ | <b>SMS</b>         | Service Management System                           |
| ◆ | <b>SNG</b>         | Secured Network Gateway                             |
| ◆ | <b>SNI</b>         | Siemens Nixdorf Informationsysteme                  |
| ◆ | <b>SNMP</b>        | Simple Network Management Protocol                  |
| ◆ | <b>SNP</b>         | Secure Network Programming                          |
| ◆ | <b>SPKM</b>        | Simple Public-key GSS-API Mechanisms                |
| ◆ | <b>SSE</b>         | Software and Systems Engineering (SSE) Ltd.         |
| ◆ | <b>SSO</b>         | Single Sign-On                                      |
| ◆ |                    |   |
| ◆ | <b>TA</b>          | Trusted Authority                                   |
| ◆ | <b>TAN</b>         | Transaction Authentication Number                   |
| ◆ | <b>TCB</b>         | Trusted Computing Base                              |
| ◆ | <b>TCP</b>         | Transport Control Protocol                          |



---

|     |                |  |
|-----|----------------|--|
| ♦   | <b>TCP SYN</b> | Atac prin epuizarea resurselor asupra protocolului |
| TCP |                |  |
| ♦   | <b>TESS</b>    | The Exponential Security System                    |
| ♦   | <b>TGS</b>     | Ticket Granting Server                             |
| ♦   | <b>TGT</b>     | Ticket Granting Ticket                             |
| ♦   | <b>TLSP</b>    | Transport Layer Security Protocol                  |
| ♦   | <b>TVP</b>     | Time-Variant Parameter                             |
| ♦   | <b>UID</b>     | User Identification                                |
| ♦   | <b>UUID</b>    | Universal Unique Identifier                        |
| ♦   | <b>URL</b>     | Uniform Resource Locator                           |
| ♦   | <b>US</b>      | User Sponsor                                       |
| ♦   | <b>U.S.</b>    | United States                                      |
| ♦   | <b>WIMAX</b>   | Worldwide Interoperability for Microwave Access    |
| ♦   | <b>WG</b>      | Working Group                                      |
| ♦   | <b>WWW</b>     | World Wide Web                                     |



## Bibliografie

- [1] 3rd Generation Partnership Project – *Specification of the GSM-MILENAGE Algorithms: An example algorithm set for Authentication and Key Generation functions A3 and A8*, <http://www.gsmworld.com/using/algorithms/docs/55205-600.pdf>
- [2] Alcatel University – *Introduction to the Alcatel GSM Network*, 2003
- [3] R. Anderson, M. Kuhn, “Tamper Resistance – A Cautionary Note”, Proceedings of the 2<sup>nd</sup> Usenix Workshop on Electronic Commerce, pages 1-11, November 1996.
- [4] Adam Beck, Ulf Moller, Anton Stiglic – *Traffic analysis attacks and trade-offs in anonymity providing systems*, Information Hiding, 4th International Workshop, IHW 2001, volume 2137 of Lecture Notes in Computer Science, pages 245-257, Springer-Verlag, Berlin 2001
- [5] P. Barford, D. Plonka - *Characteristics of network traffic flow anomalies*, in Proc 1st ACM SIGCOMM Internet Meas. Workshop, New York, 2001, pp. 69–74.
- [6] E. Barkan, E. Biham, N. Keller – *Instant Ciphertext-Only Cryptanalysis of GSM Encrypted Communication*, Advances in Cryptology - CRYPTO 2003, Springer Berlin / Heidelberg, Pag. 600, ISBN 978-3-540-40674-7
- [7] S. M. Bellovin, M. Merritt – *Encrypted Key Exchange: Password-Based Protocols Secure Against Dictionary Attacks*, Proceedings of the IEEE Symposium on Security and Privacy, IEEE Computer Society Press, Los Alamitos, CA, 1992
- [8] S. M. Bellovin, M. Merritt – *Augumented Encrypted Key Exchange*, Proceedings of the 1<sup>st</sup> ACM Conference on Communications and Computing Security, November 1993
- [9] Steven M. Bellovin, Michael Merritt – *Limitations of the Kerberos Authentication System*, AT&T Bell Labs
- [10] Valer BOCAN - *Developments in DOS research and mitigating technologies*, Periodica Politehnica, Transactions on Automatic Control and Computer Science, Vol. 49 (63), 2004
- [11] Valer BOCAN - *Threshold Puzzles. The Evolution of DoS-Resistant Authentication*, Periodica Politehnica, Transaction on Automatic Control and Computer Science Vol. 49 (63), 2004, ISSN 1224-600X

- [12] Valer BOCAN, Mihai Fagadar-Cosma - Adaptive Threshold Puzzles, EUROCON - The International Conference on "Computer as a tool", Belgrade, Serbia and Montenegro, 2005
- [13] Valer BOCAN, Mihai Făgădar-Cosma – Towards DoS-resistant Single Sign-On Systems, EUROCON, Belgrade, Serbia, 2005
- [14] Valer BOCAN, Vladimir CREȚU – *Mitigating Denial of Service Threats in GSM Networks*, The International Conference on Availability, Reliability and Security (ARES), Wien, 2006
- [15] Valer BOCAN, Mihai Făgădar-Cosma – *Scalable and Secure Architecture for Digital Content Distribution*, International Conference on Software, Telecommunications and Computer Networks, Croatia, 2006
- [16] Valer BOCAN, Vladimir CREȚU – *Threats and Countermeasures in GSM Networks*, Journal of Networks, Academy Publishers, ISSN 1796-2056
- [17] Valer BOCAN, Vladimir CREȚU - *SCADDIC: The Implementation and Performance of a Scalable and Secure Architecture for Digital Content Distribution*, 7th Conference on Communications, Military Technical Academy, Bucharest, 2008
- [18] Alan Burnett – *Securing the Wireless Internet*, Roke Manor Research Ltd, UK, 2003
- [19] M. Burrows, M. Abadi, R. Needham – "A Logic of Authentication", ACM Operating Systems Review, Vol. 23, 1989
- [20] Computer Emergency Response Team - *CERT advisory CA-2000.01 Denial of service developments*, 2000 (<http://www.cert.org/advisories/CA-2000-01.html>)
- [21] G. A. Champine, D. E. Geer, W. N. Ruh – "Project Athena as a Distributed Computer System", IEEE Computer, Vol. 23, September 1990
- [22] G. A. Champine – "MIT Project Athena – A Model for Distributed Computing", Digital Press, 1991
- [23] E. Charniak - *Bayesian Networks without Tears*, AI Magazine, Winter 1991, pp. 50–63.
- [24] David L. Chaum – *Untraceable electronic mail, return addresses and digital pseudonyms*, Communications of the ACM, 24(2):84-90, 1981
- [25] G.-H. Chiou and W.-T. Chen – "Secure Broadcasting Using the Secure Lock", IEEE Trans. Software Eng., vol. 15, no. 8, pp. 929-934, August 1989.

- 
- [26] Jan de Clercq – *Single Sign-On Architectures*, Infrastructure Security, International Conference, InfraSec 2002, Bristol, UK, 2002
- [27] Scott A. Crosby, Dan S. Wallach – *Denial of Service via Algorithmic Complexity Attacks*, Proceedings of the 12<sup>th</sup> USENIX Security Symposium, 2003
- [28] Drew Dean, Adam Stubblefield – *Using Client Puzzles to Protect TLS*, <http://www.csl.sri.com/users/ddean/papers/usenix01b.pdf>, Proceedings of the 10<sup>th</sup> USENIX Security Symposium, 2001
- [29] Digital Equipment Corporation – *Performance tuning tips for Digital Unix*, iunie 1996, [http://www.abdn.ac.uk/local/apache/manual\\_1.3.4/misc/perf-dec.html](http://www.abdn.ac.uk/local/apache/manual_1.3.4/misc/perf-dec.html)
- [30] D. E. Denning, G. Sacco – “*Timestamps in Key Distribution Protocols*”, Communications of the ACM, Vol. 24, 1981
- [31] Prashant Dewan, Partha Dasgupta, Vijay Karamcheti – *Defending Against Denial of Service Attacks Using Name Secure Resolution*, The 23<sup>rd</sup> International Conference on Distributed Computing, 2003
- [32] The Distributed.net Organization, <http://www.distributed.net>
- [33] Cynthia Dwork, Moni Naor – *Pricing via Processing or Combating Junk Mail*, Proceedings of CRYPTO '92, Springer Verlag, 1992
- [34] William Enck, Patrick Traynor, Patrick McDaniel, Thomas la Porta, “*Exploiting open functionality in SMS capable cellular networks*”, 12<sup>th</sup> ACM Conference on Computer and Communication Security, 2005
- [35] D. C. Feldmeier, P. R. Karn – *UNIX Password Security – Ten Years Later*, Advances in Cryptology – CRYPTO '89, Springer-Verlag, Berlin, 1990
- [36] P. Ferguson, D. Senie – *Network Ingress Filtering: Defeating Denial of Service Attacks which employ IP Source Address Spoofing*, RFC 2267, 1998
- [37] Alan O. Freier, Philip Karlton, Paul C. Kocher – *The SSL Protocol, Version 3.0 (Internet Draft)*, Transport Layer Security Working Group, 1996
- [38] Emmanuel Gadaix – *GSM and 3G Security*, Black Hat Conference Singapore, 2001
- [39] Ghosh and Swaminatha – *M-commerce Security*, Communications of the ACM, February 2001
- [40] David M. Goldschlag, Michael G. Reed, Paul F. Syverson – *Onion routing for anonymous and private Internet connections*, Communications of the ACM, 42(2):84-88, January 1999

- [41] L. Gong, R. Needham, R. Yahalom – “*GNV Logic Fill In*”, Proceedings of the IEEE Symposium on Security and Privacy, 1990
- [42] L. Gong – “*A Security Risk of Depending on Synchronized Clocks*”, ACM Operating Systems Review, Vol. 26, 1992
- [43] L. Gong, T. M. A. Lomas, R. M. Needham, J. H. Saltzer – *Protecting Poorly Chosen Secrets from Guessing Attacks*, IEEE Journal on Selected Areas in Communications, Vol. 11, June 1993
- [44] B. Groza, D. Petrica – *On Chained Cryptographic Puzzles*, 3rd Romanian-Hungarian Joint Symposium on Applied Computational Intelligence (SACI), Timisoara, Romania, May 25-26 2006
- [45] Gunnar Heine – *GSM Networks: Protocols, Terminology and Implementation*, Alcatel SEL Germany, 1998
- [46] Michael Howard, David LeBlanc, *Writing Secure Code*, 2<sup>nd</sup> Edition, Microsoft Press, 2003
- [47] Andrew Huang, “Keeping Secrets in Hardware: the Microsoft Xbox™ Case Study, May 2002, <http://web.mit.edu/bunnie/www/proj/anatak/AIM-2002-008.pdf>
- [48] Ari Juels, John Brainard – *Client puzzles: A cryptographic defense against connection depletion attacks*, Proceedings of the NDSS 1999
- [49] Y. Kim, W. C. Lau, M. C. Chuah, H. J. Chao - *PacketScore: Statistics-based Overload Control against Distributed Denial-of-Service Attacks*, in *IEEE INFOCOM 2004*, Hong Kong, 2004.
- [50] D. V. Klein – “*Foiling the Cracker*”: *A Survey of, and Improvements to, Password Security*, Proceedings of the USENIX UNIX Security II Symposium, USENIX Association, Berkeley 1990
- [51] J. Kohl – *The Kerberos Network Authentication Service (V5) (RFC 1510)*, Digital Equipment Corporation, 1993
- [52] J. T. Kohl, B. C. Neuman, T. Y. Ts'o – *The Evolution of the Kerberos Authentication System*, Distributed Open Systems, IEEE Computer Society Press, Los Alamitos, CA, 1994
- [53] K. Y. Lam, T. Beth – *Timely Authentication in Distributed Systems*”, ESORICS '92 – European Symposium on Research in Computer Security, Springer-Verlag, Berlin, November 1992
- [54] Liberty Alliance Project – *Liberty ID-FF Architecture Overview*, 2003, <http://www.projectliberty.org>

- [55] Liberty Alliance Project – *Liberty ID-FF Protocols and Schema Specification 1.2*, 2003, <http://www.projectliberty.org>
- [56] Liberty Alliance Project – *Liberty Specs Tutorial*, <http://www.projectliberty.org>
- [57] T. M. A. Lomas, L. Gong, J. H. Saltzer, R. M. Needham – *Reducing Risks from Poorly Chosen Keys*, ACM Operating Systems Review, Vol. 23, 1989
- [58] Steve Lord – *Bugwatch: GSM security flaws exposed*, VNU Business Publications Limited, 2003, <http://www.vnunet.com/vnunet/news/2121449/bugwatch-gsm-security-flaws-exposed>
- [59] M. Ma – *Mitigating denial of service attacks with password puzzles*, International Conference on Information Technology: Coding and Computing, 2005, volume 2, pag. 621-626
- [60] David Margrave – *GSM Security and Encryption*, George Mason University
- [61] Catherine Meadows – *A formal framework and evaluation method for network denial of service*, Proceeding of the 1999 IEEE Computer Security Foundations Workshop, Mordano, Italy, 1999
- [62] R. C. Merkle – *Secure Communications Over Insecure Channels*, Communications of the ACM, 1978
- [63] S. K. Nair, B. C. Popescu, C. Gamage, B. Crispo and A. S. Tanenbaum - *"Enabling DRM – preserving Digital Content Redistribution"*, 7<sup>th</sup> International IEEE Conference on E-Commerce Technology, 2005
- [64] R. M. Needham, M. D. Schroeder – *"Using Encryption for Authentication in Large Networks of Computers"*, Communications of the ACM, Vol. 21, December 1978
- [65] R. M. Needham, M. D. Schroeder – *"Authentication Revisited"*, ACM Operating Systems Review, Vol. 21, 1987
- [66] Rolf Oppliger – *Authentication Systems for Secure Networks*, Artech House, Inc., 1996
- [67] Andreas Pashalidis, Chris J. Mitchell – *A Taxonomy of Single Sign-On Systems*, 2003, Royal Holloway, University of London
- [68] Andreas Pfitzmann, Marit Köhntopp – *Anonymity, unobservability and pseudonymity – a proposal for terminology*, Designing Privacy Enhancing Technologies, International Workshop on Design Issues in Anonymity and

Unobservability, number 2009 in Lecture Notes in Computer Science, pages 141 – 160, Springer-Verlag, Berlin 2001

[69] Ping-Herng Denny Lin – *Survey of the Denial of Service Countermeasures*, California State University, Fullerton, 2000

[70] J. J. Quisquater, L. Guillou – “*How to Explain Zero-Knowledge Protocols to Your Children*”, Advances in Cryptology – CRYPTO '89, Springer – Verlag, Berlin 1990

[71] Valentin Razmov – *Denial of Service Attacks and How to Defend Against Them*, University of Washington, 2000

[72] Ronald R. Rivest, Adi Shamir, David A. Wagner – *Time-lock Puzzles and Timed-release Cryptography*, 1996, <http://lcs.mit.edu/~rivest/RivestShamirWagner-timelock.pdf>

[73] Stefan Savage, David Wetherall, Anna Karlin, Tom Anderson – *Practical network support for IP traceback*, technical report UW-CSE-00/02/0, SIGCOMM '00, 2000

[74] Bruce Schneier – *Distributed denial of service attacks*, Crypto-gram newsletter, 2000

[75] Schneier, Bruce - *Secrets & Lies*, Wiley Publishing, Inc., 2004

[76] Bruce Schneier – *Applied Cryptography*, John Wiley & Sons, Inc., 1996

[77] SETI @home Program, <http://setiathome.ssl.berkeley.edu>

[78] Shibboleth Architecture, Protocols and Profiles, Working Draft 02, 22 September 2004, <http://shibboleth.internet2.edu>

[79] C. Siaterlis and B. Maglaris - *Towards Multisensor DataFusion for DoS Detection*, in SAC'04, Nicosia, Cyprus, 2004

[80] Oliver Spatscheck, Larry Peterson – *Defending against denial of service in Scout*, Proceedings of 3rd USENIX/ACM Symposium on OSDI, p. 59-72, 1999

[81] William Stallings - *Cryptography and Network Security, Principles and Practices*, Third Edition, Prentice Hall, 2003

[82] J. G. Steiner, B. C. Neuman, J. I. Schiller – *Kerberos: An Authentication Service for Open Network Systems*, Proceedings of the USENIX UNIX Security Symposium, 1988



- [83] Filip Šuba – *Usability and Security of DRM architectures*, TKK T-110.5190 Seminar on Internetworking, Helsinki University of Technology, 2007
- [84] T. Aura, P. Nikander, J. Leiwo – *DOS-resistant authentication with client-puzzles*, Proceeding of the Cambridge Security Protocols Workshop 2000, LNCS, Cambridge, UK, 2000
- [85] Upkar Varshney – *Network access and security issues in ubiquitous computing*, Workshop on Ubiquitous Computing Environment, Cleveland, 2003
- [86] B. Waters, A. Juels, J. A. Halderman, E. W. Felten – *New Client Puzzle Outsourcing Techniques for DoS Resistance*, 11th ACM Conference on Computer and Communications Security, 2004
- [87] X. Wang, M. K. Reiter – *Defending Against Denial-of-Service Attacks with Puzzle Auctions (Extended Abstract)*, 2003 IEEE Symposium on Security and Privacy (SP'03), pages 78-92
- [88] IBM Corporation, Microsoft Corporation, BEA Systems, RSA Security, Verisig - *Web Services Federation Language (WS-Federation)*, July 2003, <http://www.ibm.com/developerworks/library/ws-fed/>
- [89] Anonimizer – [www.anonymizer.com](http://www.anonymizer.com)
- [90] WebSecure - <http://www.freedom.net/products/websecure>
- [91] JAP Anonymity and Privacy – <http://anon.inf.tu-dresden.de>
- [92] Trusted Computing Group, "Trusted Computing Platform Alliance Main Specification", October 2003, Version 1.2, <http://www.trustedcomputinggroup.org>
- [93] Eric W. Weisstein, "Chinese Remainder Theorem." From *MathWorld* - A Wolfram Web Resource, <http://mathworld.wolfram.com/ChineseRemainderTheorem.html>
- [94] eXtensible Access Control Markup Language (XACML), <http://www.oasis-open.org/committees/xacml>
- [95] XrML: eXtensible Rights Markup Language, <http://www.xrml.org>
- [96] Content Scrambling System, <http://www.dvdcca.org/css/>
- [97] Overview of the Protected Media Path, <http://msdn2.microsoft.com/en-gb/library/aa376846.aspx>
- [98] Mentalis Secure Socket Library, <http://www.mentalis.org/soft/projects/ssocket/>

- [99] BigInteger Project, <http://www.codeproject.com/csharp/biginteger.asp>
- [100] World Wide Web Consortium – *The Platform for Privacy Preferences 1.0 (P3P 1.0) Specification, April 2002*
- [101] A. Fiat, A. Shamir – “How to Prove Yourself: Practical Solutions to Identification and Signature Problems”, *Advances in Cryptology – CRYPTO '86*, Springer – Verlag, Berlin 1987
- [102] Shon Harris – DOS Defense, *Information Security Magazine*, 2001
- [103] Valer BOCAN, Vladimir CREȚU - *Security and Denial of Service Threats in GSM Networks*, CONTI 2004 - Periodica Politehnica, Transaction on Automatic Control and Computer Science Vol. 49 (63), 2004, ISSN 1224-600X
- [104] Valer BOCAN - *Sisteme Single Sign-On sub atacuri Denial-of-Service. Studiu de caz: Proiectul Liberty Alliance*, Referat de doctorat nr. 3, Politehnica University of Timisoara
- [105] Valer BOCAN - *Studiu asupra nivelului de siguranta oferit de protocoalele de autentificare*, Raport de doctorat nr. 2, Politehnica University of Timisoara, 2004
- [106] Valer BOCAN - *Stadiul actual al dezvoltarii sistemelor de securitate pentru retele de calculatoare de inalta siguranta*, Raport de doctorat nr. 1, Politehnica University of Timisoara

## Lista publicațiilor personale

### 2008

Valer BOCAN, Vladimir CREȚU - *SCADDIC: The Implementation and Performance of a Scalable and Secure Architecture for Digital Content Distribution*, 7th Conference on Communications, Military Technical Academy, Bucharest, 2008

### 2006

Valer BOCAN, Vladimir CREȚU - *Threats and Countermeasures in GSM Networks*, Journal of Networks, Academy Publishers, ISSN 1796-2056

Valer BOCAN, Mihai FĂGĂDAR-COSMA - *Scalable and Secure Architecture for Digital Content Distribution*, SoftCOM 2006 - International Conference on Software, Telecommunications and Computer Networks, Split, Croatia

Valer BOCAN, Vladimir CREȚU - *Mitigating Denial of Service Threats in GSM Networks*, ARES 2006 - The First International Conference on Availability, Reliability and Security, Wien, Austria

### 2005

Valer BOCAN, Mihai FĂGĂDAR-COSMA - *Adaptive Threshold Puzzles*, EUROCON 2005 - The International Conference on "Computer as a tool", Belgrade, Serbia and Montenegro

Valer BOCAN, Mihai FĂGĂDAR-COSMA - *Towards DoS-resistant Single Sign-On Systems*, EUROCON 2005 - The International Conference on "Computer as a tool", Belgrade, Serbia and Montenegro

Valer BOCAN - *Sisteme Single Sign-On sub atacuri Denial-of-Service. Studiu de caz: Proiectul Liberty Alliance*, Referat de doctorat nr. 3, Politehnica University of Timisoara

### 2004

Valer BOCAN - *Studiu asupra nivelului de siguranță oferit de protocoalele de autentificare*, Raport de doctorat nr. 2, Politehnica University of Timisoara, 2004

Valer BOCAN - *Developments in DoS Research and Mitigating Technologies*, CONTI 2004 - Periodica Politehnica, Transaction on Automatic Control and Computer Science Vol. 49 (63), 2004, ISSN 1224-600X

Valer BOCAN, Vladimir CREJU - *Security and Denial of Service Threats in GSM Networks*, CONTI 2004 - Periodica Politehnica, Transaction on Automatic Control and Computer Science Vol. 49 (63), 2004, ISSN 1224-600X

Valer BOCAN - *Threshold Puzzles. The Evolution of DoS-Resistant Authentication*, CONTI 2004 - Periodica Politehnica, Transaction on Automatic Control and Computer Science Vol. 49 (63), 2004, ISSN 1224-600X

**2002**

Valer BOCAN - *Stadiul actual al dezvoltării sistemelor de securitate pentru rețele de calculatoare de înaltă siguranță*, Raport de doctorat nr. 1, Politehnica University of Timisoara







Al 35-lea Congres Național al Societății Române de  
Diabet, Nutriție și Bolii Metabolice  
Sinaia 20 - 23 Mai 2009

Timisoara




Coduri de bare - Paymaster SRL - 0722 714798

**Infrastructură de management conferințe**

- plata electronică a taxelor de conferință
- ecusoane personalizate cu coduri de bare
- monitorizare audiență în sălile de conferință
- tipărire diplome cu punctaj calculat
- identificare automată participanți la standul expozantului
- rapoarte statistice de prezență

**Aplicații web și desktop**

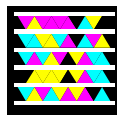
- Design web, site-uri interactive
- Aplicații web cu baze de date
- Aplicații desktop personalizate
- Consultanță

**S.C. Paymaster S.R.L. Timișoara**  
 Web: [www.paymaster.ro](http://www.paymaster.ro)  
 E-mail: [office@paymaster.ro](mailto:office@paymaster.ro)  
 Tel.: 0722 714798



**Citește acest document online**

Versiunea electronică (PDF) a acestui document se găsește la adresa  
<http://www.dataman.ro/publications>.



**Citește acest document pe telefonul mobil sau PDA**

Descarcă pe PDA sau telefonul tău mobil cu cameră aplicația **Microsoft Tag Reader** de la  
<http://gettag.mobi>, apoi scanează acest cod de bare pentru a obține lucrarea în format PDF  
 optimizat pentru dispozitive mobile.