Buletinul Stiintific al Universitatii "Politehnica" din Timisoara, ROMANIA
Seria AUTOMATICA si CALCULATOARE
PERIODICA POLITECHNICA, Transactions on AUTOMATIC CONTROL and COMPUTER SCIENCE
Vol.49 (63), 2004, ISSN 1224-600X

Developments in DoS Research and Mitigating Technologies

Valer BOCAN

Department of Computer Science and Engineering, Politehnica University of Timişoara, Bd. V. Pârvan, 300223 Timişoara, Romania
E-mail: vbocan@dataman.ro, WWW: http://www.dataman.ro

***Abstract* - This paper is a survey on the problem of denial of service attacks and the proposed ways to defend against them. Of particular concern are the distributed attacks that an adversary can carry out by recruiting innocent targets to aid the attack. Since attacks come in many forms and shapes and are generally easy to carry out given the free availability of tools such as Trinoo, defense is difficult because the line between legitimate and unauthorized traffic cannot be drawn precisely. We describe the current state of facts that allows DoS attacks and the potential mitigation technologies.**

***Keywords:* denial of service, attack, mitigating technology**

I. INTRODUCTION

Denial of service attacks are a major cause of incorrect operation in the Internet and are arguably the most serious threat that the Internet community faces today. The first major attack brought down University of Minnesota's network in August 1999. About six months later, in February 2000 a Canadian teenager attacked some of the Internet's most important sites: Yahoo, CNN, Amazon, Buy and eBay. Since then, attacks seemed to be on the rise [23].

Unfortunately, users are more interested in software that has new features rather than solid software with few or no flaws. Besides, security does come for a price. Modern software expends a huge number of cycles to draw pretty three-dimensional windows with alpha blending that provide little or no functional improvement at all. Although security is one of the major problems in the industry, many are unwilling to spend as many cycles on the security as they spend on drawing their windows [8]. There are also too many users that don't care whether their system is safe or it can be used as a target or as a launching pad for malware of all sorts [12].

The false sense of security is probably worse than the lack of security. There are still too many under skilled system administrators out there that leave their systems' doors wide open by not applying the latest patches and not conforming to standard procedures. Add the fact that the number of directly connected homes, schools, libraries or other public entities has grown exponentially lately and you are beginning to see the whole dimension of the phenomenon [6].

Security threats can be categorized as follows: [15]

- breaches of confidentiality
- failure of authenticity
- unauthorized denial of service

While the first two have been extensively analyzed in the literature, denial of service attacks have not received the proper attention up until recently, namely after the February 2000 events [2].

A. Definition of the DoS Attack

A denial of service attack on a network could take one of the two possible forms. A malicious party (a.k.a. the attacker) could cause the network not to transmit messages it should be sending in order to offer service to a subset or all of its clients. On the other end of the spectrum, the network could be caused to send messages, which it should not be sending. By far the most common form of DoS in today's networks is causing excessive bogus traffic (a.k.a. flooding the network) in the direction of a particular server, which in the end will prevent legitimate users from getting the service they could otherwise be receiving from that server [18].

There are several common attack methods known by the Internet community. They are divided into two main categories: flood attacks and malformed packet attacks:

B. Flood Attacks

Flood attacks are quite common and they intend to saturate networks links in order to crash routers and switches or flood systems with more traffic that they can handle. Unfortunately, tools required to mount such attacks are freely available on the Internet and even malicious users with little or no experience can use them.

- **Smurf Flood Attack** is a common DoS attack known as a *reflector* attack. An attacker sends a small number of ICMP echo packets to a broadcast address that defines several hosts. The replies from all those hosts are sent simultaneously to the victim, exhausting all the available bandwidth and possibly processing power.

- **TCP SYN Attack** is possible due to the three-way handshaking behavior of the TCP protocol. A client sends a request (SYN) to a server announcing the intention to start a conversation. In return, the server assigns an entry in the table reserved for half-open connections and sends back an acknowledgement message (SYN ACK) signaling the acceptance of the connection request. Now it's the client's turn to send a SYN ACK ACK packet to start the actual communication. A malicious user may never do that and the result is that the entry in the queue for pending connection is blocked until the timeout expires. If the malicious client sends a burst of such requests it may paralyze the activity of a typical 100 MIPS server that can handle around 2000 connections per second [21], the minimum standard TCP connection queue being 2048 slots [5].
- **UDP Flood Attack (Fraggle)** is possible because of the connectionless nature of the UDP protocol. Since no connection procedure is necessary, the attacker may send packets to random ports, causing the victim to allocate CPU cycles in order to determine which application listens to those ports. When it realizes that no applications listen on the ports, the victim will generate a destination unreachable response ICMP packet and will send it to the forged originating address. If enough packets are sent to the victim, the system may go down.
- **ICMP Flood Attack** consists of an attacker sending a large number of echo ICMP packets to the victim. Since the victim cannot keep up with the load, the system may experience performance degradation.
- **E-mail bombing** is another flood attack. Essentially this consists of sending a huge number of e-mails to the target in order to fill the storage space and / or the bandwidth.

With the exception of the UPD Flood Attack, the rest can be avoided by patching the operating system. The UDP attack is difficult to cope with since there may be several applications listening to ports and preventing access by means of firewalls may severely reduce functionality. Such attacks cannot be mitigated unless we find a way to tell legitimate request from the fake ones. In addition to that, mitigating technology must not incur a significant overhead since this may open a new avenue of attack.

C. Malformed Packet Attack

The malformed packet attack is another wide-spread type of DoS attack. The purpose of this attack is to send ill-formed packets to hosts and take advantage of the bad design of the code that processes the packets. Effects range from unacceptable degradation of performance to system crashes.

About half of the security problems on the Internet are caused by buffer overflows. They have been known for about 40 years and there have been good solutions to avoid them for quite about the same time, namely since Algol 60 introduced mandatory array bounds checking. Programmers still refuse to use better tools and this is almost criminal negligence. It is comparable to a car manufacturer making the gas tank out of wax paper [8].

There are several malformed packet attacks and we summarize them below:

- **Ping of Death Attack** consists of sending an ICMP echo packet that is much larger than the maximum IP packet size (64 Kbytes). At destination, some TCP/IP implementations fail to reconstruct the packet, crashing or rebooting the system. Two well known implementations exhibiting this behavior are Windows 95 and some early NT versions.
- **Chargen Attack**. This is a variant of the UDP Flood Attack and uses the port 19 (chargen) of an intermediary system used as an amplifier. The attacker sends a forged UDP packet on port 19 of the intermediary system which in turn replies with a string of characters back to the victim, on its echo service port. The victim then sends back an echo of the string and the loop created rapidly exhausts the bandwidth between the victim and the intermediary system.
- **Teardrop Attack.** Due to poor implementation, some systems fail to correctly cope with packet fragments that have incorrect offsets, making proper reassembly impossible. Instead of gracefully discarding the packets, the implementations in question simply reboot or halt the system.
- **Land Attack.** Astoundingly, some systems crash or reboot when they encounter a forged packet which contains the same address as both the origin and the destination.
- **Win Nuke Attack.** This type of attack is specifically targeted against Windows machines to which attackers send out-of-band data to a specific port, causing the system to crash or reboot.

Another classification of the DoS attacks may be according to the number of parties involved in the attack:

- **Uni-source attacks** – there is only one attacker that targets a single victim.
- **Multi-source attacks** – several hosts (called "zombies") unwillingly participate as attackers, being compromised by the head of the operation. Although more difficult to put into practice, this type of attack is the most dangerous and most difficult to fight against. It is also known as a Distributed Denial of Service (DDoS) attack.

[19] gives a figurative real-world equivalent of the DoS attack: Alice does not like Bob, so she calls multiple pizza delivery parlors and orders one pizza from each, to be

delivered to Bob's house at some given time. When the moment comes, Bob is overwhelmed by the host of pizza deliverers arriving at his house and demanding their money. Simple yet very effective, if Alice had called from a public payphone (essentially disguising her identity), there is nothing Bob or the pizza parlors could do to even hope to find out who played the trick on them in the first place.

II. DENIAL OF SERVICE ATTACKS

A. Cause of DoS Attacks

One obvious cause of TCP SYN attacks is that the preliminary communication takes place before authentication. The server cannot distinguish legitimate traffic from the fake one so there isn't much that can be done here. Imposing the requirement that all requests should first be authenticated would be a DoS attack in its own right because the server would spend a lot of time verifying digital signatures, whether they are real or not. This new avenue of attack proves just as dangerous as merely filling the SYN table for half-open connections.

Another cause of DoS attacks is the less obvious lack of resource accounting. Spatscheck and Peterson [21] consider that there are three key ingredients for defending against DoS attacks:

- *accounting* for all consumed resources per client;
- *detection* when the resources consumed by any given client exceed some limit;
- *containment* – the ability to reclaim the tied resources after detecting an attack by dedicating minimum additional server resources to the task and thus avoiding to fall for a follow-up denial of service attack;

Back in the days when the Internet itself was designed, resource accountability was the lowest priority goal and this is precisely the greatest danger the Internet faces today. As opposed to the ubiquitous phone network where resource usage was carefully controlled, the Internet designers seemed to care less about this aspect. Thus, servers allocate the same amount of CPU cycles to different incoming requests regardless of importance and this prevents a graceful degradation of performance when the system is under heavy load or under attack.

The above scenario is somewhat similar to the rudimentary mechanism of processing incoming requests due to the interrupt-driven architecture of the network subsystem [4]. Virtually all operating systems implement this type of architecture which proved inadequate in high-load environments. Incoming packets are processed with the highest priority and then packets are immediately discarded just because there is no application to service them. This situation is called a *receiver livelock.* More over, even though there is an application to service the incoming packets, the process priority is not taken into account and more times than not there are low priority applications that receive the same amount of incoming traffic as the high priority ones. In their paper, Druschel and Banga propose a lazy receiver processing architecture which rests on early packet demultiplexing, early packet discard and packet processing at the receiver's priority. They claim that the new architecture would improve stability, fairness and throughput of systems under high loads while not suffering from performance degradation under normal conditions.

Crosby and Wallach [3] described a new avenue of attack that is based on the intrinsic design of the protocols. The new class of low-bandwidth denial of service attacks exploits the algorithmic deficiencies in the data structures of common applications. Frequently used data structures have "average-case" expected running time that is far more efficient than the worst case. For example, both binary trees and hash tables can degenerate to linked lists with carefully chosen input. Using bandwidth less than a typical dialup modem, the cited authors brought a dedicated Bro server to its knees; after six minutes of carefully chosen packets, the Bro server was dropping as much as 71% of its traffic and consuming all of its CPU.

B. Significance of Dealing with DoS Attacks

Taking into account the global tendency of the markets to move online, DoS attacks prove more dangerous than originally predicted since they can disable the victims for prolonged periods of time. From the time when the attack is mounted till the time it is detected and recovered from, the victim is virtually paralyzed and cannot respond to legitimate requests. For large commercial sites this translates to losses of billions of dollars of magnitude.

Although DoS attacks do not directly threaten the data in any way, there is no reason not to believe that attacks of some other kind may occur during or after the DoS attack. These follow-up attacks may destroy mission- or life-critical data, causing much more damage than the DoS attack itself which isn't of course, desirable.

This kind of chain attack can actually happen if the protocols involved are not fail-stop or at least fail-safe [11]. The basic idea is that the protocol should automatically halt communication with any host not following the standard protocol execution path.

III. MITIGATING TECHNOLOGIES

The security breaches presented so far have a number of proposed remedies. We can differentiate three approaches here – completely eliminating the attack, mitigating effects of the attack and discouraging the attacker itself. These approaches cannot and may not be used just by themselves; instead they should be used complementarily whenever possible.

A. Eliminating the Possibility of Attack

This is the most desirable way to defend against a DoS attack since the actual attack does not take place and the effect on the victim does not exist. Unfortunately things are not at all simple and threats cannot be eliminated completely.

Selective access to resources

Closed environments (e.g. corporate intranets, military facilities) may benefit from selective access to resources, i.e. only allowing authenticated clients to communicate with servers. This is clearly not a fit for a decentralized environment such as the Internet. [18] cites a known problem with closed environments, i.e. outside intrusions are both not expected and commonly not anticipated. So, the level of preparedness for a security breach, should it ever occur, is very low and the damage grows proportionally high.

Out-of-band signaling

The idea behind out-of-band signaling is that data and control information travels on different physical channels, confusion and interference being thus avoided. This scenario resembles communication between a mobile station and a cellular network (e.g. GSM) where the voice traffic and signals are transmitted in different time slots. [18] cites the case of the land telephony in the 1960s when in-band signaling was in use. One could whistle into the phone receiver and under certain favorable circumstances (the right wavelength and amplitude) the signal, which really was just data, could be interpreted as a control signal (e.g. a free call, etc.).

Although not based on experiments or at least theoretical work, Schneier claims that out-of-band signaling would alleviate problems related to denial of service attacks. It is yet to be seen whether this claim is true [20].

B. Mitigating the Attack Effect on the Victim

We have seen that there is not much to do in order to prevent attacks. It is therefore desirable to have a response mechanism that allows the system to provide a service within acceptable limits even under severe attack. There are several proposed mitigating technologies and we are presenting them below:

Securing all computers in a network

This would mean the end of zombies as we know them. The attackers would be forced to launch uni-source attacks, reducing the attack magnitude and easing the administrative task of apprehending the criminal. This approach is of little practical interest since securing a system is a relative notion and upgrading and keeping the same level of security in a huge system like the Internet is impossible.

SYN Attacks Countermeasures

In case of SYN attacks there are several proposed approaches [10]:

- *Timeout:* The buffer allocated for a half opened TCP connection is cleared after a certain amount of fixed time. Although this is an easy to implement mechanism, the server should also take into account the legitimate slow connections. If the attacker opens connections very fast compared to a slow user, then this mechanism would not completely solve the problem.
- *Random Dropping:* This method of "dropping" certain connections is similar to the packet marking method used by Random Early Drop (RED) Active Queue Management (AQM) Algorithm. The basic idea is to randomly select connections to drop after the server's resources have passed a certain threshold. The main problem with this approach is that legitimate users are also affected and the attack is never completely ended. If the attacker opens connections with a very high rate, then there is a very minor difference whether or not this algorithm is employed.
- *SYN Cookie:* This method is cited as the strongest method against TCP SYN attacks. The server sends a value “V” that is the hash value of certain parameters of that specific connection and a secret value only known to the server. The server does not allocate any buffer space without receiving the same value “V” in the ACK message that the client is supposed to send. The main assumption here is that the attacker is spoofing IP addresses, so it will never receive the SYN ACK message containing the value “V”. This assumption may not always hold, especially for relatively small local area networks or for Ethernet.

Although categorized as strong, the SYN cookie is not a stateless protocol. This means that the server must store state information for each attempted connection which isn’t desirable.

Ingress filtering

An attacker may forge the source address from which it is launching a DoS attack. The attacker forging its source address will cause the victim to send a SYN ACK packet to an erroneous address, preventing the victim from ever receiving the ACK packet it needs to proceed. In RFC 2267 [9], Ferguson and Senie described network ingress filtering that can prevent attackers from using forged source addresses to launch a DoS attack [17].

Egress filtering

Egress filtering ensures that only IP packets with valid source IP addresses leave the network. This approach is useful when deployed close to the end user, however implementing it for Internet service providers is almost impossible since they frequently need to forward legitimate traffic that is not part of their own address space [17].

Preventing ICMP echo requests

Preventing ICMP echo requests from entering the local network helps alleviate the amplifier problem, where echo replies may be sent to a broadcast address, causing useless traffic and performance degradation.

Disabling unneeded network services

Disabling any unneeded network service is basically narrowing the DoS target such that the attacker has fewer choices. Also, vulnerabilities are discovered regularly and an attacker may use the service in question before the victim becomes aware of the threat and before applying the patch.

Client puzzles

Use of client puzzles prior to committing resources is one of the most cited approaches. The idea behind client puzzles is to slow down the attacker so much that the DoS attack is no longer effective. Before committing resources, sever sends a puzzle to client to solve. The difficulty of the puzzle can be easily changed from zero to infinity to accommodate the current server load. A good puzzle should have the following properties [1, 13]:

1. Creating a puzzle and verifying the solution is inexpensive for the server.
2. The cost of solving the puzzle is easy to adjust from zero to impossible.
3. The puzzle can be solved on most types of client hardware (although it may take longer with slow hardware).
4. It is not possible to precompute solutions to the puzzles.
5. While the client is solving the puzzle, the server does not need to store the solution or other client-specific data.
6. The same puzzle may be given to several clients. Knowing the solution of one or more clients does not help a new client in solving the puzzle.
7. A client can reuse a puzzle by creating several instances of it.

The proposed puzzle is the brute-force reversal of a one-way hash function such as MD5 or SHA. This is a practical choice because the hash functions are computable with a wide variety of hardware and the brute-force testing of different inputs is likely to remain the most efficient way for computing the inverse of these functions. (The difficulty of solving number-theoretic puzzles like factoring may depend heavily on the sophistication of the algorithms used by client.)

Despite the good reputation of client puzzles, Valer Bocan [25] has discovered a vulnerability that allows a malicious client with access to massive computation power to attack a server protected with this technology. Also, Bocan proposes two changes of the client puzzle architecture and introduces the term threshold puzzle.

Progressively stronger authentication

Progressively stronger authentication is an approach based on "promising" system resources according to the level of confidence with respect to the client making the request [14]. Avoiding early strong authentication from the beginning is pertinent since – as outlined in a previous paragraph – this would represent an attack avenue by itself. Starting with weak authentication first (e.g. a cookie) and upon receiving positive feedback (i.e. client responding and following requested steps) the server chooses stronger authentication and finally performs a digital signature verification before actual resource commitment.

System performance monitoring

Observing the system performance and establishing patterns of normal usage is a matter of relative importance. System performance (combined measurement of the CPU usage, disk and network activity and disk space variation) is a side channel whose continuous monitoring may lead to detection of an unauthorized access.

Secure name resolution

Secure name resolution is a brand new mitigating technology proposed by Dewan, Dasgupta and Karamcheti [7]. This technique makes DoS attacks difficult by providing the location of a service only to the pre-registered users and hiding it from all others. In addition, a static service is converted to a relocating service and it relocates whenever attacked. The Domain Name Service (DNS) protocol is modified to provide the encrypted location information to all clients and a Key Server is introduced to provide the decryption key only to the legitimate clients thereby ostracizing unscrupulous clients. An attacker cannot attack without the location information of the service.

Cryptographic salt

Park, Kim, Boyd and Dawson have proposed the use of **cryptographic salt** in specific authentication protocols such as SSL/TLS, SKEME and PACS, where the client authenticates the server by sending a random nonce encrypted under the public encryption key of the server [16]. Although similar to the client puzzle approach, the cited authors claim that the proposed protocol modification is minimal since the changes are at the level of the protocol itself. However, this approach should be used with great care before an in-dept analysis of the cryptographic community is available, since it is well known the fact that any change in a protocol design is susceptible of introducing new vulnerabilities. Nevertheless, this is an interesting idea.

C. Discouraging the Attacker

Should an attack happen, it is highly desirable to have a way to trace the attack back to its initiator in order to apprehend him. This is mostly an administrative effort and should be used to augment previously discussed techniques of coping with a DoS attack.

IP traceback by ISP coordination

IP traceback by ISP coordination means close cooperation between all service providers whose equipment was traversed by the attack wave. This is a very difficult thing to do since policy varies from ISP to ISP and a single break in the chain would lead to the impossibility of catching the criminal. Razmov [18] suggests the formation of some sort of administrative control and mandating ISPs to cooperate, much like governments cooperate in cracking down on international crime. To be really effective, ISPs must have some incentive in participating in this effort, such as creating a list of careless and insecure ISPs that would not be recommended to the public.

IP traceback by probabilistically marking packets

IP traceback by probabilistically marking packets seems to be the most promising approach for discouraging users. It is robust, incrementally deployable and backwards compatible. It is also resistant against IP spoofing and distributed denial of service attacks [23].

The idea behind probabilistically packet marking is to attach partial path information to packets traveling through routers (to which attackers have no access). In the event of a DoS attack, the route of the packets can be reconstructed at the victim provided that there are a sufficient number of packets which is generally not a problem. Implementing this technique would mean a remarkable achievement since no infrastructure changes are necessary, however close cooperation with authorities is still mandatory.

IV. CONCLUSIONS

Complete elimination of DoS attacks is not possible given the current Internet infrastructure. The difficulty lies in differentiating between legitimate and bogus traffic and this state of facts is not going to change soon. We have shown three ways of handling DoS attacks, namely preventing them from happening in the first place, mitigating the effect of an attack in progress and discouraging the attacker. Used in conjunction, these ways provide some degree of protection to the victim of the DoS attack, however immunity is not guaranteed.

The outcome of this paper is that defense against DoS attacks is a long term battle and it must be deployed globally over various networks. To be effective, technical approaches must also be combined with administrative efforts.

REFERENCES

[1] Tuomas Aura, Pekka Nikander, and Jussipekka Leiwo – ”Dos-resistant authentication with client puzzles.” In Proceedings of the Cambridge Security Protocols Workshop 2000, LNCS, Cambridge, UK, April 2000. Springer-Verlag.

[2] Computer Emergency Response Team, “CERT advisory CA-2000.01 Denial of service developments”, Jan 2000. (http://www.cert.org/advisories/CA-2000-01.html)

[3] Scott A. Crosby, Dan S. Wallach, “Denial of Service via Algorithmic Complexity Attacks”

[4] Peter Druschel and Gaurav Banga, “Lazy receiver processing (LRP): a network subsystem architecture for server systems”, In Proceedings of 2nd USENIX Symposium on OSDI, Seattle, Oct 1996.

[5] Digital Equipment Corporation, “Performance tuning tips for Digital Unix”, June 1996.

[6] (http://www.service.digital.com/manual/misc/perf-dec.html)

[7] Marcel Dekker. Security of the Internet. The Froehlich/Kent Encyclopedia of Telecommunications, vol. 15 pp 231-255, New York, 1997.

[8] Prashant Dewan, Partha Dasgupta, Vijay Karamcheti, “Defending Against Denial of Service Attacks Using Secure Name Resolution”

[9] Niels Ferguson, Bruce Schneier, “Practical Cryptography”, Wiley Publishing, Inc., 2003

[10] P. Ferguson, D. Senie, “Network Ingress Filtering: Defeating Denial of Service Attacks which employ IP Source Address Spoofing”, RFC 2267, January 1998

[11] Orhan Can Ozdural, Dustin Gaskey, „ECE 578 Project – Oregon State University, Electrical and Computer Engineering Dept.“, Spring 2003

[12] Li Gong and Paul Syverson, “Fail-stop protocols: An approach to designing secure protocols”,1998.

[13] Shon Harris. DoS Defense. Information Security magazine, September 2001.

[14] Ari Juels and John Brainard, “Client puzzles: A cryptographic defense against connection depletion attacks.” In S. Kent, editor, Proceedings of NDSS ’99, pages 151–165, 1999.

[15] Catherine Meadows, “A formal framework and evaluation method for network denial of service”, In Proceedings of the 1999 IEEE Computer Security Foundations Workshop, Mordano, Italy, June 1999.

[16] Roger M. Needham, “Denial of service: an example”, Communications of the ACM, vol.37, No.11, pp.42-46, Nov 1994.

[17] DongGook Park, JungJoon Kim, Colin Boyd, Ed Dawson, “Cryptographic Salt: A Countermeasure Against Denial-of-Service Attacks”

[18] Ping-Herng Denny Lin, “Survey of Denial of Service Countermeasures”, California State University Fullerton, November 2000

[19] Valentin Razmov, “Denial of Service Attacks and How to Defend Against Them”, University of Washington, 2000

[20] Bruce Schneier, “Distributed denial of service attacks”, Crypto-gram newsletter, Feb 2000. (http://www.counterpane.com/crypto-gram-0002.html#DistributedDenial-of-ServiceAttacks)

[21] Oliver Spatscheck and Larry Peterson, “Defending against denial of service in Scout”, In proceedings of 3rd USENIX/ACM Symposium on OSDI, pp.59-72, Feb 1999.

[22] The Standard Performance Evaluation Corporation, “SpecWeb96 benchmark results”, 1998. (http://www.specbench.org/osg/web96/results)

[23] Stefan Savage, David Wetherall, Anna Karlin and Tom Anderson, “Practical network support for IP traceback”, Technical report UW-CSE-00/02/01, SIGCOMM ’00, February 2000.

[24] Jaikumar Vijayan, Denial-of-Service attacks on the rise?. CNN.com, Apr. 9, 2002. (http://www.cnn.com/2002/TECH/internet/04/09/dos.threat.idg/)

[25] Valer Bocan, Threshold puzzles: The evolution of DOS-resistant authentication, PERIODICA POLITEHNICA, Transactions on AUTOMATIC CONTROL and COMPUTER SCIENCE, Vol. 49 (63), 2004, in press