Denial of service in public key protocols

Pasi Eronen
Helsinki University of Technology
Telecommunications Software and Multimedia Laboratory
pasi.eronen@iki.fi

Abstract

Network denial of service attacks have become a widespread problem on the Internet. However, denial of service is often considered to be an implementation issue by protocol designers. In this paper I present a survey of the literature on designing denial of service resistant communication protocols. I consider several different types of resources vulnerable to resource consumption attacks, and outline countermeasures against such attacks. I also describe how these countermeasures are used in the ISAKMP/IKE and Photuris protocols, and give overview of design recommendations for future protocols.

1 Introduction

Denial of service (DoS) means "the prevention of authorized access to resources or delaying of time-critical operations" [22]. During the last couple of years, network denial of service attacks—which deny or degrade access to some network service—have become a problem on the Internet. The most publicized attacks have been against well-known web sites such as Yahoo and Amazon [12, 37].

CERT's note on denial of service [9] identifies three different types of network DoS attacks. First, consumption of scarce, limited or non-renewable resources. Second, destruction or modification of configuration information, and third, physical destruction or modification of network components.

In this survey, I focus on resource consumption attacks, and how they could be prevented—or at least made more difficult—by taking availability into account in the design of communication protocols. In protecting the availability of actual systems, it is, of course, important to consider other attacks as well.

I also limit the study to what I consider to be a typical case on the Internet: protecting the availability of a server from attackers trying to deny service to legitimate clients. Thus, protecting clients from DoS, and protocols involving more than two parties, are beyond the scope of this paper.

The rest of the paper is organized as follows. The next section presents a couple of different resources vulnerable to DoS attacks, with examples in each category. Section 3 identifies a number of different countermeasures. Sections 4–6 describe how these countermeasures

are used in two actual protocols, ISAKMP/IKE and Photuris. Section 7 presents some ideas how denial of service problem should be addressed in protocol design, and finally, Section 8 presents my conclusions and suggestions for further research.

2 Resource consumption attacks

Resource consumption attacks work by consuming some scarce, limited or non-renewable resources. Most of the widespread network DoS attacks have been of this type: for example, the TCP SYN flooding attacks and the distributed DoS attack on Yahoo, Amazon, and other popular web sites. The details of both of these attacks are discussed below.

Resource consumption attacks have become common because they are quite easy to mount, difficult to defend against, and hard to trace to their source. Furthermore, there are many types of resources which could be consumed. The next sections describe the most common resources used for denial of service attacks.

2.1 Memory

One of the earliest widespread DoS attacks was the TCP SYN flooding attack. It works by filling the table reserved for half-open TCP connections in the operating system's TCP/IP stack. When the table becomes full, new connection can't be opened until some entries are removed from the table (due to handshake timeout). The details are described in [7, 19, 44].

This attack can be done using fake IP addresses, so tracing it to its source is difficult. Of course, the table of connections can be filled without spoofing the source IP address. Often the space available for fixed tables, such as the half-open TCP connection table, is much less than the total RAM of the system. This makes the attacks even easier.

2.2 Network bandwidth

The distributed DoS attack carried out against Yahoo, Amazon, and other popular web sites worked by filling their network connection with junk traffic [12, 37]. The traffic was generated from hundreds of different sources—computers who had earlier been compromised by crackers, and had attack software installed on them.

In general, protecting systems from this type of attack is difficult, since it requires mechanisms distributed across the Internet. Some countermeasures to make tracing such attack easier are described in Section 3.2.

However, there are other types of network bandwidth attacks. In some protocols, the client sends only a small amount of data, but gets a large response from the server. This could be described as "traffic amplification" [8, 10]. This effect can be used to attack either the "amplifier" directly, or the amplifier can be used to attack some third party (by faking the source address, so that the amplified traffic is sent to the final target). This kind of attack could be made more difficult by server-side actions; e.g. designing services so that large responses aren't generated to untraceable requests.

2.3 Computational resources

Some authors (e.g. Juels and Brainard in [25]) have suggested that strong authentication, based on a public-key infrastructure, could prevent or mitigate denial of service attacks such as SYN flooding.

However, using public key cryptography opens the possibility for another kind of attack. Public key cryptography usually involves expensive computations, such as modular exponentiation. If the attacker can convince the target to perform a large number of such computations, exhausting its CPU resources, legitimate connections won't get any CPU time.

There haven't been any widespread attacks of this type so far, but several protocols have been identified as being vulnerable [20, 21, 29, 30, 32].

2.4 Other resources

Although typically several orders of magnitude larger than RAM, disk space can also be subjected to denial of service. For example, the attacker can cause a large number of error messages to be written to a log, and fill the disk containing the log files. Or, user's mailbox can be filled with junk mail, thus preventing reception of important messages—this attack is usually even easier than filling disk with logs, since users often have mailbox "quotas" much smaller than typical disk capacities.

Disk (or other I/O than network) bandwidth can also be a bottleneck in a system. For example, in a server hosting a large database, queries for attributes which are not indexed probably result in lots of I/O activity. This could happen in a publicly available LDAP server, for instance.

In theory, any finite or non-renewable resource can be the target of a DoS attack, and providing a complete list is thus impossible. Other examples of resources that could be vulnerable are incoming telephone lines on a terminal server, paper at printers or fax machines, money spent buying another resource from a third party, and so forth.

3 Countermeasures

Reacting to a resource consumption attack usually involves at least detecting that an attack is going on, tracing it to (or nearer) its source, and taking administrative or legal action to end the attack.

In this paper I'm focusing only on countermeasures relevant in designing and implementing communications protocols. Thus administrative countermeasures are beyond the scope of this paper.

Countermeasures against DoS attacks can be either preventive or reactive. Preventive countermeasures can help in this by making attacks easier to detect, or trace, and also by making the system more resistant to attacks.

3.1 Detecting attacks

Even detecting that a service is under a denial of service attack can sometimes be difficult. Clients which are denied of service naturally detect it, but the condition isn't always easily noticeable at the server (e.g. TCP SYN attack).

Even when it has been determined that the service is indeed under attack, detecting which part of the incoming traffic belongs to the attack, and which is legitimate traffic, can be difficult. This problem is made more difficult by spoofed IP addresses.

Intrusion detection and reaction systems aim to cut off denial of service attacks by identifying the part of traffic which belongs to the attack, and denying service only to that part while continuing to serve legitimate clients. Most such mechanisms are very ad hoc in nature; a determined attacker can fool them, and they can also produce false positives. For instance, a web proxy for a very large organization naturally produces a large number of traffic, which all looks like it's coming from a single or few IP addresses. To an intrusion detection system this might look like a flooding attack.

3.2 Tracing attacks

Cutting off an attack often requires tracing it to its source. The possibility of tracing probably also discourages attacks, since attackers know they are more likely to get caught. Thus, it can be thought as both preventive and reactive countermeasure.

Forging, or spoofing as it is usually called, of source IP address on the Internet is quite easy. Some techniques (such as cookies, described in the next section) can be used to get some degree of assurance about the source IP address. In analysis of mechanisms for protecting confidentiality and integrity of messages, it is usually assumed that the attacker can modify, replay, and block any packets sent. This naturally allows trivial denial of service, so somewhat weaker assumptions are used when analyzing availability.

There have been several proposals for mitigating the problem of IP spoofing. Ingress filtering [15] means filtering incoming IP addresses which should not occur on the correct link. For example, a central router at an university should filter out outgoing packets whose source address is not within the university's network. This makes the network less likely to be used as a launching pad for attacks, and if deployed widely, should reduce the problem of IP spoofing.

Another proposal, ICMP traceback messages [5], attempts to trace back flooding attacks, even if they have forged source addresses. Other authors have proposed mechanisms for recording the route traveled in the packets themselves [11, 43]. Unfortunately, all of these require modifications to router software, so it remains to be seen if any of them will ever be widely deployed.

3.3 Cookies

Cookie is a piece of data which is generated by the server and given to the client in the beginning of a protocol run. The client has to include this piece of data in the subsequent

messages. The goal of cookies is to prevent attacks which employ IP spoofing. If the client doesn't receive the message containing the cookie, the server will reject further messages because they don't include a valid cookie.

More precisely, if the server receives a valid cookie from the client, it knows that:

- The attacker is using his real IP address. This address may, of course, belong to a third party computer the cracker has broken into.
- Or, the attacker has access to physical link on the route from the server to the spoofed IP. In reality, the attacker is probably quite "close" to either the server or the spoofed IP, making tracing easier.
- Or, the attacker is able to manipulate the IP routing infrastructure. This is beyond the capabilities of "script kids", and more sophisticated and motivated attackers probably focus on attacks on integrity and confidentiality.

In other words, the idea is to start the protocol with weak authentication (of IP addresses), and possibly later perform stronger authentication. This allows tracing of attacks (with some limitations, as described in the previous section), and probably discourages attacks on computational resources.

An early example of cookies are actually the TCP initial sequence numbers, though their original purpose was to prevent packets from old connections interfering with new connections. After the TCP SYN attacks, some TCP/IP stacks were modified to use the initial sequence numbers as "SYN cookies" to protect against the attack [19, 44].

The cookie approach was much refined during the design of the Photuris protocol. The Photuris specification [27] gives the following requirements for cookies (or "anti-clogging tokens"):

- The cookie must depend on the addresses of the communicating parties.
- Nobody else must be able to forge a cookie that will be accepted by the server.
- The cookie generation and verification must be fast enough so that they don't become subjects to DoS attacks.
- The server must not keep per-client state until the IP address has been verified (i.e. it has received a cookie it generated).

The last requirement is especially important in protecting against memory consumption attacks.

The recommended method for generating the cookies in Photuris is to use a keyed one way hash of both IP addresses, both UDP ports, some locally generated secret value (which must be same for all clients, and must be periodically changed), and some other context-dependent information (see [27] Section 3.3).

The properties of Photuris cookie exchange are described and analyzed by Oppliger in [38]. The idea of of gradually strengthening authentication are analyzed formally by Meadows in [35].

3.4 Storing state in client

The cookie approach can be extended to include some state in the cookie. Any state that would be normally stored in the server is passed to the client. The client passes the state back to the server when sending the next message. The client doesn't have to interpret the state in any way, and can treat it simply as an arbitrary bit string. Encryption and message authentication codes can be used to prevent the client from tampering with the state.

This naturally doesn't work for protocols where the server might be required to take some action before the reception of next message, but is otherwise a quite general approach.

Stateless protocols have others advantages in addition to preventing memory consumption attacks. For example, in NFS recovering from server reboots is easy because no state is kept on the server (except for file locking, which complicates things). Stateless protocols also allows easier load balancing between servers.

The advantages of being stateless, at least in the beginning of a protocol run, were recognized in the security protocol context by Janson et al. [24]. The topic is further explored by Aura and Nikander in [2].

3.5 Re-ordering computations

In typical authenticated versions of the Diffie-Hellmann key agreement protocol, the server has to verify the client's signature in the first message. Since this requires expensive computation, the server can be potentially flooded with requests. In some cases, however, it is possible to modify the protocol so that client has to do some expensive computation first, and the server verifies the signature only after it has verified that the client has done so.

Thus, mounting an attack requires the client to invest the same amount of CPU resources as the server, and this hopefully will make DoS attacks at least somewhat harder. One such modification to the ISAKMP/IKE aggressive mode exchange is described in [32].

3.6 Pricing

Although cookies, stateless connections and re-ordering computations can give some protection against DoS attacks, in some cases more aggressive measures are required to allow service to legitimate users and deny it to attackers.

One such technique is "pricing". This means imposing some deliberate cost to the client, which is small for legitimate users making a small number of requests, but large for an attacker trying to flood the server.

Earliest use of this approach was a proposal by Dwork and Naor to combat junk e-mail [13, 14]. Before accepting an e-mail message, the recipient asks the sender to perform a small computation. The verification of the computation has to be quick, so this doesn't open possibilities for new DoS attacks. Another early use of this technique was against SYN flooding [25].

Usually this cost is in terms of processing time, since it is easy to implement, but other

forms (such as paying with actual micropayment systems) are also possible. The computation can be just "junk computation" to prevent denial of service, or it can be "useful" computation for some other purpose [23]. This mechanism is generalized and analyzed by Aura, Leiwo, and Nikander in [3, 29] and formally described by Jakobsson in [23].

Interestingly, the idea of moderately hard computational problems has received other uses as well: e.g. uncheatable benchmarks [6], timed release of secrets [41, 17], partial key escrow [4], and auditable metering of web site use [16].

4 IPSEC architecture

IPSEC protocols provide security services at the IP layer in both IPv4 and IPv6 environments. At the lowest level of the IPSEC architecture are cryptographic algorithms for authentication and encryption, which are used by the Authentication Header (AH) and Encapsulating Security Payload (ESP) protocols.

The parameters required by AH and ESP, such as which algorithms are used and keys, are stored in Security Associations (SAs). Although the SAs can be specified manually, they are often negotiated using some key management protocol. The two most widely used protocols are ISAKMP/IKE and Photuris. Their main purpose is to create security associations with short-term keys from longer-term keys, refresh these keys when necessary, and negotiate other parameters required for the SAs.

The IPSEC architecture is described by Kent and Atkinson in [28].

5 ISAKMP/IKE

ISAKMP/IKE is actually a combination of two protocols, ISAKMP and IKE [33, 18]. ISAKMP provides a framework for authentication and key exchange. IKE, based on the Oakley protocol [39], specifies a key agreement protocol based on this framework.

The IKE negotiation has two phases. In the first phase, the parties negotiate an ISAKMP security association, which provides a secure communication channel between the ISAKMP daemons. In phase 2, this ISAKMP security association is used to negotiate an IPSEC security association for AH or ESP.

ISAKMP and IKE provide a large number of different modes and options, which can be considered to be different subprotocols. Meadows has identified a total of thirteen (!) different subprotocols [34]. This makes understanding, implementation, and analysis of ISAKMP/IKE very challenging.

5.1 Cookies in ISAKMP/IKE

ISAKMP uses cookies as a countermeasure against IP spoofing. The header of each packet contains two fields of 64 bits, the initiator and responder cookies. The initiator cookie is

sent in the first message by the client, and the responder cookie is returned in the reply. All further messages contain both cookies.

The cookies are also used to identify a particular ISAKMP SA during phase 1 of the protocol. Since the cookie is also used to distinguish between different ISAKMP SAs, the protocol requires that the responder cookie is different for each initiator, and each protocol run. The method recommended by the ISAKMP specification is to use a one-way hash of IP addresses, UDP port numbers, locally generated secret value, and date and time.

Unfortunately, this requires a small amount of per-client state be stored on the server after the first message (at least the date and time). Simpson criticizes this harshly in [45].

5.2 Aggressive mode problem

ISAKMP contains two different modes which may be used in the phase 1 negotiation. These are called the "main mode" and "aggressive mode". The aggressive mode requires fewer messages than the main mode, but it doesn't usually provide protection of identities, and also has some other limitations.

The aggressive mode also has a known DoS vulnerability when using public key signatures for authentication [45, 31, 32]. The responder has to sign its response to the first message from the client. This expensive operation is done *before* the cookie has been verified, so it allows the possibility for DoS attack using spoofed IP addresses.

6 Photuris

The Photuris protocol is much simpler than ISAKMP/IKE, since it is not a combination of many layers of different subprotocols. The protocol has three phases: cookie exchange, value exchange, and identification exchange. After these phases, additional messages may be used to refresh the keys or modify other security parameters. [27, 26]

Like the ISAKMP header, the Photuris header also contains fields for initiator and responder cookies, each of 128 bits. In additional to the cookies, a "counter" field is included to distinguish between multiple parallel protocol runs. This removes the need to store any state on the responder after the first pair of messages.

Also, in Photuris the server never performs any expensive computations before receiving the second message (which must contain a valid cookie).

7 Protocol design principles

Based on the discussion in the previous sections, it should be clear that denial of service issues should considered in the design of new protocols. Without going to specific implementation issues, I feel that at least the following recommendations should be used as guidelines in the design.

Recommendation 1: Protocol specifications should explicitly state what kind of DoS vulnerabilities exist, and not ignore denial of service as a mere implementation issue.

Recommendation 2: Protocols should verify the client's IP address before creating state or performing expensive computations. This is especially important for protocols which run over UDP, since they don't even have the weak protection offered by TCP sequence numbers.

Recommendation 3: Protocols should avoid creating state on the server at all if the state could be easily stored on the client instead [2].

Aura, Leiwo, and Nikander have written a couple of good papers about the design of DoS resistant protocols [29, 3]. Their earlier paper on stateless connections is also very relevant [2].

Simpson's harsh criticism of ISAKMP/IKE [45] presents some mistakes done in the design on ISAKMP/IKE which should prove helpful to the designers of future protocols. The design principles behind Photuris are described in the Photuris RFC [27] and [46].

The use of formal methods in protocol analysis has usually ignored denial of service. Some results are presented by Meadows in [35], and open issues in the field are discussed in [36].

This paper has addressed only the denial of service aspects in the design of protocols. Aura's dissertation [1] contains a good overview of other security-related aspects in protocol design. Other desirable properties of good Internet protocols, such as efficiency, extensibility, and simplicity are discussed in e.g. [40] and [42].

8 Conclusions

Often denial of service is considered to be an implementation issue by protocol designers. However, the resistance to DoS attacks can be improved by choices taken in the protocol design. In this paper I have presented an overview of resource consumption vulnerabilities, and countermeasures which could be used to defend systems.

Protocol design principles have been analyzed by several researchers, and Section 7 presents a short summary of their recommendations, with references to full publications. Although much of the work has focused on public key cryptographic protocols, many of the principles presented can be applied to other protocols as well.

There are many possibilities for further work. Modification of well known protocols to include better DoS protection, and experiments in the real world, could produce valuable information on the effectiveness of DoS protection mechanisms. Combining the techniques for pricing, tracing, and intrusion detection could also provide practical solutions for DoS protection.

Acknowledgements

I'd like to thank my tutor Bengt Sahlin for his encouraging comments, and Minna Kangasluoma from Nixu Ltd. for a valuable discussion about the topic.

References

[1] Tuomas Aura. *Authorization and availability: Aspects of open network security*. Doctoral dissertation, Helsinki University of Technology, November 2000. Laboratory for Theoretical Computer Science research report HUT-TCS-A64.

[2] Tuomas Aura and Pekka Nikander. Stateless connections. In *Proceedings of International Conference on Information and Communications Security (ICICS '97)*, Lecture Notes in Computer Science volume 1334, pages 87–97, Beijing, China, November 1997. Springer.

[3] Tuomas Aura, Pekka Nikander, and Jussipekka Leiwo. DOS-resistant authentication with client puzzles. In Bruce Christianson, Bruno Crispo, and Mike Roe, editors, *Proceedings of the 8th International Workshop on Security Protocols*, to appear in the Lecture Notes in Computer Science series, Cambridge, UK, April 2000. Springer.

[4] Mahir Bellare and Shafi Goldwasser. Verifiable partial key escrow. In *Proceedings of the 4th ACM Conference on Computer and Communications Security (CCS '97)*, pages 78–91, Zürich, Switzerland, April 1997.

[5] Steven M. Bellovin. ICMP traceback messages. Work in progress, Internet draft bellovin-itrace-00, March 2000.

[6] Jin-Yi Cai, Ajay Nerurkar, and Min-You Wu. Making benchmarks uncheatable. In *Proceedings of the IEEE International Computer Performance and Dependability Symposium (IPDS '98)*, pages 216–226, Durham, North Carolina, September 1998.

[7] CERT Coordination Center. TCP SYN flooding and IP spoofing attacks. CERT Advisory CA-1996-21, available from http://www.cert.org/advisories/CA-1996-21.html, September 1996.

[8] CERT Coordination Center. Smurf IP denial-of-service attacks. CERT Advisory CA-1998-01, available from http://www.cert.org/advisories/CA-1998-01.html, January 1998.

[9] CERT Coordination Center. Denial of service attacks. Technical note, available from http://www.cert.org/tech_tips/denial_of_service.html, February 1999.

[10] CERT Coordination Center. Denial-of-service tools. CERT Advisory CA-1999-17, available from http://www.cert.org/advisories/CA-1999-17.html, December 1999.

[11] Thomas W. Doeppner, Philip N. Klein, and Andrew Koyfman. Using router stamping to identify the source of IP packets. In *Proceedings of the 7th ACM Conference on Computer and Communications Security (CCS 2000)*, pages 184–189, Athens, Greece, November 2000.

[12] Jonathan Dube. Web under attack. Article on abcnews.com, 8 February 2000.

[13] Cynthia Dwork and Moni Naor. Pricing via processing or combatting junk mail. In *Advances in Cryptology — CRYPTO '92: 12th Annual International Cryptology Conference, Proceedings*, Lecture Notes in Computer Science volume 740, pages 139–147, Santa Barbara, California, August 1992. Springer.

[14] Cynthia Dwork and Moni Naor. Pricing via processing or combatting junk mail. Technical Report CS95-20, Weizmann Institute of Science, Faculty of Mathematics and Computer Science, Israel, 1995.

[15] Paul Ferguson and Daniel Senie. Network ingress filtering: Defeating denial of service attacks which employ IP source address spoofing. Best Current Practice RFC 2827, IETF, May 2000.

[16] Matthew K. Franklin and Dahlia Malkhi. Auditable metering with lightweight security. In *Financial Cryptography: 1st International Conference Proceedings (FC '97)*, Lecture Notes in Computer Science volume 1318, pages 151–160, Anguilla, British West Indies, February 1997. Springer.

[17] David M. Goldschlag and Stuart Stubblebine. Publicly verifiable lotteries: Applications of delaying functions. In *Financial Cryptography: 2nd International Conference Proceedings (FC '98)*, Lecture Notes in Computer Science volume 1465, Anguilla, British West Indies, February 1998. Springer.

[18] Dan Harkins and Dave Carrel. The internet key exchange (IKE). Standards Track RFC 2409, IETF, November 1998.

[19] B. Harris and R. Hunt. TCP/IP security threats and attack methods. *Computer Communications*, 22(10):885–897, June 1999. Elsevier.

[20] Shouichi Hirose and Kanta Matsuura. Enhancing the resistance of a provably secure key agreement protocol to a denial-of-service attack. In *Proceedings of the 2nd International Conference on Information and Communication Security (ICICS '99)*, Lecture Notes in Computer Science volume 1726, pages 169–182, Sydney, Australia, November 1999. Springer.

[21] Shouichi Hirose and Kanta Matsuura. Enhancing the resistance of a secure key agreement protocol to a denial-of-service attack. In *Proceedings of the 1999 Symposium on Cryptography and Information Security (SCIS '99)*, pages 899–904, Kobe, Japan, January 1999.

[22] ISO 7498-2:1989 Information processing systems – Open Systems Interconnection – Basic Reference Model – Part 2: Security Architecture, 1989.

[23] Markus Jakobsson and Ari Juels. Proofs of work and bread pudding protocols. In *Proceedings of the IFIP TC6 and TC11 Joint Working Conference on Communications and Multimedia Security (CMS '99)*, Leuven, Belgium, September 1999. Kluwer.

[24] Philippe Janson, Gene Tsudik, and Moti Yung. Scalability and flexibility in authentication services: the KryptoKnight approach. In *Proceedings of IEEE INFOCOM '97*, pages 725–736, Tokyo, Japan, April 1997.

[25] Ari Juels and John Brainard. Client puzzles: A cryptographic countermeasure against connection depletion attacks. In *Proceedings of the 1999 Network and Distributed System Security Symposium (NDSS '99)*, pages 151–165, San Diego, California, February 1999.

[26] Phil Karn and William A. Simpson. Photuris: Extended schemes and attributes. Experimental RFC 2523, IETF, March 1999.

[27] Phil Karn and William A. Simpson. Photuris: Session-key management protocol. Experimental RFC 2522, IETF, March 1999.

[28] Stephen Kent and Randall Atkinson. Security architecture for the internet protocol. Standards Track RFC 2401, IETF, November 1998.

[29] Jussipekka Leiwo, Pekka Nikander, and Tuomas Aura. Towards network denial of service resistant protocols. In *Proceedings of the 15th International Information Security Conference (IFIP/SEC 2000)*, Beijing, China, August 2000. Kluwer.

[30] Kanta Matsuura and Hideki Imai. Protection of authenticated key-agreement protocol against a denial-of-service attack. *Cientifica*, 2(11):15–19, 1999.

[31] Kanta Matsuura and Hideki Imai. Resolution of ISAKMP/Oakley key-agreement protocol resistant against denial-of-service attack. In *Proceedings of Internet Workshop (IWS '99)*, pages 17–24, Piscataway, New Jersey, February 1999.

[32] Kanta Matsuura and Hideki Imai. Modified aggressive mode of Internet key exchange resistant against denial-of-service attacks. *IEICE Transactions on Information and Systems*, E83-D(5):972–979, May 2000.

[33] Douglas Maughan, Mark Schneider, Mark Schertler, and Jeff Turner. Internet security association and key management protocol (ISAKMP). Standards Track RFC 2408, IETF, November 1998.

[34] Catherine Meadows. Analysis of the Internet Key Exchange protocol using the NRL protocol analyzer. In *Proceedings of the 1999 IEEE Symposium on Security and Privacy*, pages 216–231, Oakland, California, May 1999.

[35] Catherine Meadows. A formal framework and evaluation method for network denial of service. In *Proceedings of the 12th IEEE Computer Security Foundations Workshop*, pages 4–13, Mordano, Italy, June 1999.

[36] Catherine Meadows. Open issues in formal methods for cryptographic protocol analysis. In *Proceedings of DARPA Information Survivability Conference and Exposition (DISCEX 2000)*, pages 237–250, Hilton Head, South Carolina, January 2000. IEEE Computer Society Press.

[37] Peter G. Neumann. Inside risks: denial-of-service attacks. *Communications of the ACM*, 43(4):136, April 2000.

[38] Ralf Oppliger. Protecting key exchange and management protocols against resource clogging attacks. In *Proceedings of the IFIP TC6 and TC11 Joint Working Conference on Communications and Multimedia Security (CMS '99)*, pages 163–175, Leuven, Belgium, September 1999. Kluwer.

[39] Hilarie Orman. The OAKLEY key determination protocol. Informational RFC 2412, IETF, November 1998.

[40] Radia Perlman. Folklore of protocol design. Work in progress, Internet draft iab-perlman-folklore-00 (expired), January 1998.

[41] Ronald L. Rivest, Adi Shamir, and David A. Wagner. Time-lock puzzles and timed-release crypto. Technical Report MIT/LCS/TR-684, Massachusetts Institute of Technology, February 1996.

[42] Marshall T. Rose. On the design of application protocols. Work in progress, Internet draft mrose-beep-design-01, November 2000.

[43] Stefan Savage, David Wetherall, Anna Karlin, and Tom Anderson. Practical network support for IP traceback. In *Proceedings of ACM SIGCOMM 2000*, pages 295–306, Stockholm, Sweden, August 2000.

[44] Christoph L. Schuba, Ivan V. Krsul, Markus G. Kuhn, Eugene H. Spafford, Aurobindo Sundaram, and Diego Zamboni. Analysis of a denial of service attack on TCP. In *Proceedings of the 1997 IEEE Symposium on Security and Privacy*, pages 208–223, Oakland, California, May 1997.

[45] William A. Simpson. IKE/ISAKMP considered harmful. *;login:*, 24(6):48–58, December 1999. USENIX Association.

[46] William A. Simpson. Photuris: Design criteria. In Howard Heys and Carlisle Adams, editors, *Selected Areas in Cryptography: 6th Annual International Workshop Proceedings (SAC '99)*, Lecture Notes in Computer Science volume 1758, pages 226–242, Kingston, Ontario, Canada, August 1999. Springer.