Preventing Denial of Service Attacks on the Protocol Level

Mattias Hallberg matha181@student.liu.se
Anders Lindahl andli747@student.liu.se
Pontus Viking ponvi042@student.liu.se

TDDC03 Information Security
Linköpings universitet

June 2, 2003

Abstract

Denial of Service attacks has become more common with the increasing popularity of the Internet. Many of these attacks exploit weaknesses in common Internet protocols. In this report we look at some of these attacks, where the protocol (rather than a single application) play a critical role. However, our main focus is on some common vulnerable protocols. We list their weaknesses and possible defence mechanisms. Our conclusion is that the major problem is in the IP protocol and the possibility of forging the sender address of data packets. Defence against this should be deployed as close to the attacker as possible.

1 Overview of the Internet

Internet is a connection of many *autonomous systems* (AS), typically networks for large organisations like universities, *ISPs* (Internet Service Providers) and big corporations. Routing inside an AS can be done in several ways, but the suggested *IETF* (Internet Engineering Task Force) standard from 1990 is the *OSPF* (Open Shortest Path First) protocol (defined in RFC 2328 [19]). Routers on the border of an AS use *BGP* (Border Gateway Protocol) [22] to exchange routing information.

Routing protocols are needed for routers to interchange information about what possible routes exist, the load on routes etc.

Inside the local networks the traffic to and from workstations flow, for example HTTP traffic in TCP/IP packets or *DNS* (Domain Name System) queries in UDP/IP packets. The network administrator might be using *SNMP* (Simple Network Management Protocol) to monitor network equipment like routers and switches.

More on this can be read in Tanenbaums book "Computer Networks" [33].

An important part of Internet is the Domain Name System (DNS). An attack on key DNS servers (for example large ISPs) would make the net useless for users relying on that DNS for name lookups.

The DNS system is hierarchical, and at the top are the (in 2003) 13 DNS root servers located in different locations around the world. A *DoS* (Denial of Service) attack based on *flooding* (sending a large amount of packets) with ICMP, TCP SYN, fragmented TCP, and UDP was launched on these in 2002 [37], but failed to disturb traffic from an end user point of view.

2 Attack Types

We have gathered some known attacks as examples, dividing them in two categories. Malformed network traffic attack problems where applications are not prepared to handle erroneous or unexpected traffic. The other category is protocol misuse, when an aspect or feature of a protocol is misused. There are of course overlap and combinations of these categories.

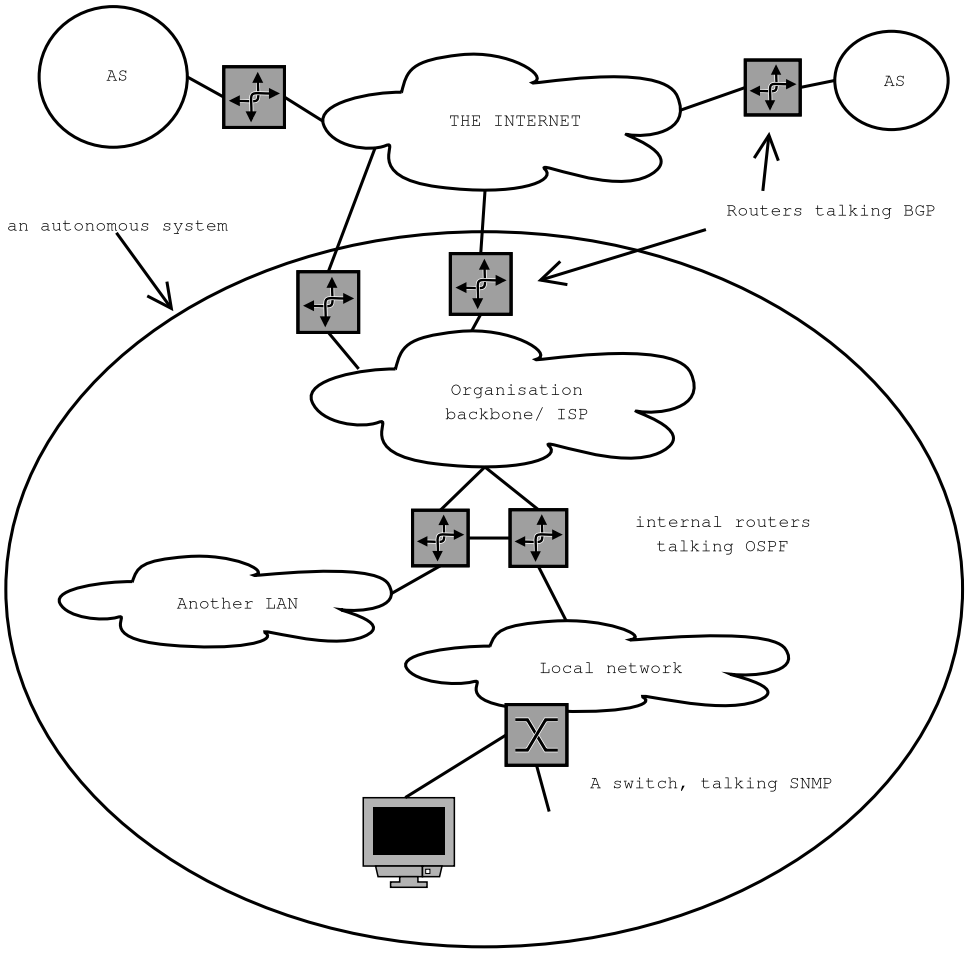

Figure 1: An example of the network hierarchy when autonomous systems (AS) are connected to the Internet

2.1 Malformed Network Traffic

A denial-of-service attack on several Microsoft operating systems was unveiled in may 2000 [35]. Sending a stream of malformed IP fragments to a host made it spend most or all of its CPU resources handling them. This was however not the first time an attack caused by malformed packets where discovered.

In 1997, the Teardrop attack [25] made Linux 2.0, Windows 95 and Windows NT systems stop responding. The problem in Linux was that the re-fragmentation routines could be tricked into calculating a way too big size of the resulting packet and fill up the memory. It seems like there was a similar problem in the Windows IP implementation.

During the following years, several variations of teardrop appeared. Jari Hauito and Tom Weckström published a paper in 1999 where they try out some of them [9].

2.2 Protocol Misuse

Characteristics of protocols and applications can be used for denial-of-service attacks. Mike Kristovich discovered in late 2002 [17] that several game servers could be used as traffic "amplifiers". A similar problem was found with DNS servers in 1999 [26].

The problem in both cases is that a relatively small UDP packet sent to the server results in a much larger response packet back to the sender. By forging the sender address (commonly referred to as *spoofing*) in the initial packet, an attacker could create heavy traffic to the victim.

This is similar to the earlier "smurf" attack [34], where the attacker broadcasts ICMP Echo request (ping) packets with spoofed address. Each of the hosts will (if configured to respond to broadcast pings) then reply with an ICMP Echo reply to the victim.

Another example of protocol misuse is the SYN flooding attack [24] which will be explained in the TCP section.

3 Common Vulnerable Protocols

3.1 IP - Internet Protocol

Aside from the fragmentation attacks above, there are not many attacks on the IP protocol itself. One reason for this could be that the protocol is rather simple and hence it should be easy to do an elegant and secure implementation. The main problem with IP is the possibility of spoofing.

3.1.1 Spoofing and the traceback problem

Spoofing is when an attacker sends a packet with a forged sender address, and is often used to either hide the real source or make the recipient of the packet respond to a victim host.

The problem of finding the real source of a spoofed packet is referred to as the *traceback problem*, and it is an area with active and ongoing research. With proof of the origin of an attack, the victim can contact router administrators (for example the attackers ISP) and have the malicious traffic filtered out.

In an IP traceback, the tracer wants information about the router path the packet has followed from the attacker to the victim. When constructing a traceback algorithm one must make sure the algorithm cannot be controlled by an attacker in a way that could disturb the trace or turn the trace into a denial of service attack itself by making an innocent host appear as the attacker. Susan C. Lee and Clay Shields [18] list three different approaches; Overloading, Trace packet and Query.

Overloading (also referred to as "marking") solves the problem by writing information about the packet route into the IP header. Dawn Xiaodong Song and Adrian Perrig [30] suggest several methods for doing this, using for example encryption techniques from multicast encryption research.

The **Trace Packet** approach makes routers send additional packets with trace information, preferably with a low probability to avoid producing too much traffic overhead. IETF has in a draft proposed using trace packets and HMAC authentication [2], and it seems to be under active development.

Query, asking routers "have you seen this packet" has also been suggested, for example the CITRA system by Sterne et. al.[32]. Because routers must keep a log over packets passing, this will be limited in time and mostly useful for automatic response to attacks.

3.2 TCP - Transmission Control Protocol

TCP, Transmission Control Protocol, is one of the most commonly used protocols on the Internet. It is defined in RFC 793 [12] and RFC 1122 [7]. Both HTTP and FTP are based on TCP, which makes DoS-attacks against TCP a big problem.

3.2.1 How TCP works and why it is vulnerable

TCP is constructed for communication between a single client and a server. To make a connection, the following chain of events takes place. The client sends a SYN packet (SYNchronize) to the server to initiate a connection. The server answers by sending a SYNACK package (ACKnowledge), and the client finally sends an ACK package. This is called the *TCP three-way handshake*. Termination of a connection is performed in a similar way.

It is the connection procedure that makes TCP vulnerable. If an attacker sends a SYN with a spoofed source-IP, the attacked server will send its SYNACK to that IP and wait in vain for the final ACK. TCP standards demand a timeout of 75 seconds, during which system resources are allocated to keep record of the connection requests. With

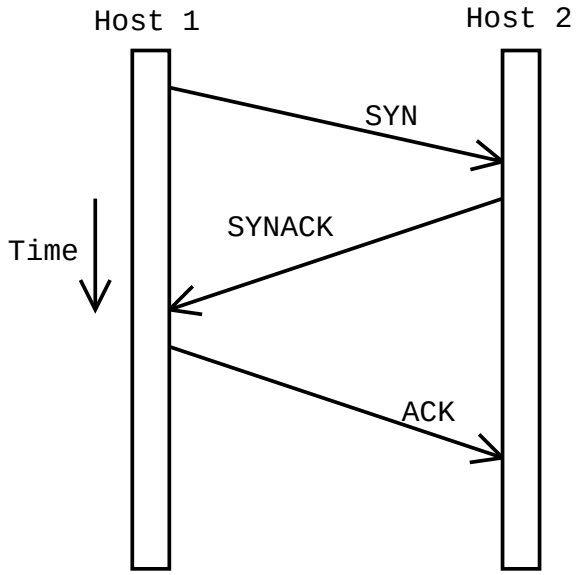

Figure 2: The TCP three-way handshake procedure.

this in mind, it's not hard to imagine an attacker sending a large quantity of spoofed SYNs, constantly filling a servers connection pending queue, before the previous SYNs has timed out. By doing this, the attacker can easily flood the server, and impede legitimate connection attempts, even with small bandwidth capacities of his own.

3.2.2 How we can protect ourselves

The major deficiency with TCP is that it expects all traffic to be legitimate. The ability for an attacker to spoof an IP-address is what is causing most of the problems. There are a few common protection methods against SYN-flooding. In this paper we will present four of those methods. Two of these, Linux cookies and Reset cookies, deals with verifying the legitimacy of the SYN package, while the two other tries to solve the problem in other ways. None of the methods is optimal however, which is showed by Ricciulli, Lincoln and Kakkar in "tcp syn flooding defence" [24].

Linux Cookies

In this model, the incoming SYN packages sequence number, the source address and the destination address, combined with a secret number, are run through a hash function. The secret number is needed to make sure that the attacker cannot know what the cookie will be. If he knew, he could also send a spoofed ACK packet maybe a second later and get the desired resources allocated at the server. The resulting cookie is used as sequence number in the SYNACK, which is sent to the source address as usual. When receiving an ACK, the server simply controls if the sequence number matches the number calculated by the hash function. If it does, the connection is allowed, otherwise the ACK is neglected.

We have found two variations of this approach; Linux cookies and SYN cookies. They differ on the detail whether a secret value is added before the hashing (in Linux Cookies) or if a server-selected secret hash function is used [3] (SYN Cookies). The purpose is the same, to have a unique sequence number identifying a connection.

By doing it this way, no records of connection requests has to be kept locally on the server. According to Ricciulli and colleagues the big problem with this solution is that it doesn't let the server retransmit SYNACKs in case of packet loss [24]; a limitation that breaks TCP semantics. Bernstein, on the other hand, dismisses this as a bogus claim, asserting that SYN-cookies are fully compliant with the TCP protocol [3], and since Linux and SYN cookies are virtually the same, this should also hold for Linux cookies.

Reset Cookies

This is a solution, proposed by Shenk [27], that uses the TCP specifications to create *security associations*. The use of security association is not a part of standard TCP, and its purpose is to make sure the sender of the SYN packet is not a bogus host. When the server receives a SYN package, it controls if the client has an association. If this is the case, the package is treated as usual, but if no association exists, the SYN package is discarded and an illegal SYNACK is sent to the client. In the SYNACK, the sequence number has been replaced by a cookie. When the client receives the faulty SYNACK, it sends a TCP reset, with the servers cookie, back to the server. The server verifies the cookie, thereby verifying that the client is valid, and establishes a security association with the client. When the client tries to reconnect, the

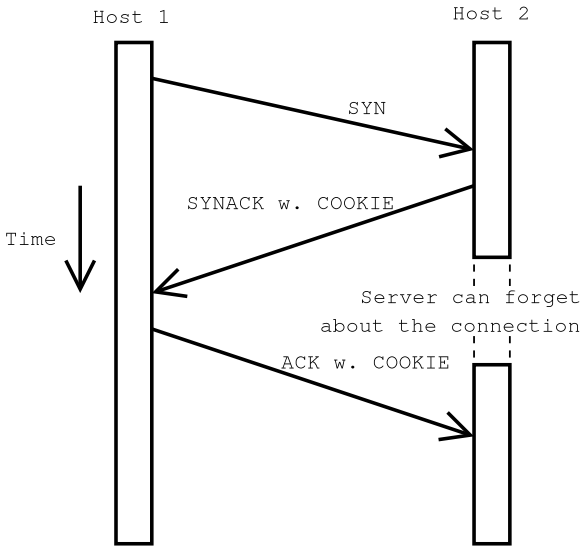

Figure 3: When using Linux cookies, the server does not need to keep information about ongoing connection attempts in memory.

server accepts the SYN and the client is allowed to connect.

As you can see, this solution does not break TCP semantics, but has the flaw that the setup time, for the first connection, is drastically increased.

BSDI Cookies

Berkeley Software Design Inc. has, according to Ricciulli [24], solved the problem by increasing the server's capacity to deal with connection pending entries. This brute force method is, consequently, no real solution to the problem, but it raises the cost of an attack (more SYN packages are needed to flood the server). This may stop individual attacks, which eliminates large parts of the problem. Coordinated attacks can still be a bother though. Kakkar et al consider this to be the currently best method, for large TCP servers with great kernel memory capacity [24].

Random Drop

Random Drop is a common technique, which also has applications in *congestion control* (techniques for avoiding too many packets in a (part of a) subnet). If a SYN package arrives at a server and its connection pending queue is full, a randomly picked entry in the queue is dropped and the new SYN takes its place. The client of the dropped SYN is notified by a TCP reset. If the dropped SYN was legitimate, all that happened was that the clients first attempt to connect failed and it can easily try again. If it wasn't legitimate; well, that's the reason we have protection in the first place. With a large enough queue, the probability of a successful connection attempt is quite good.

The advantage of this technique is its simplicity. It doesn't require any changes of TCP semantics, it doesn't slow down the connection speed and it works pretty well even against coordinated attacks. The obvious flaw is that it doesn't guarantee a successful connection, but in reality, very few of the other techniques really do that either.

In [24] Kakkar et al proposes a development of Random Drop, where a simple filter mechanism, used on the incoming SYN packages, boosts the efficiency of the method.

3.3 UDP - User Datagram Protocol

UDP is another common Internet protocol, defined in RFC 768 [21]. Due to the fact that it is *connectionless* (does not require a connection procedure similar to TCP) and has a smaller header than TCP it gives less protocol overhead. An UDP header is 64 bits, and consists only of source port, destination port, length and a checksum. Andrew Tanenbaum states in "Computer Networks" [33] that the main reason for using UDP instead of raw IP is the possibility to send to specific ports.

The simplicity of UDP makes it useful for example in question-answer situations (like DNS) and bandwidth-intense applications like multiplayer games or video broadcasting where there is to much overhead with TCP.

In the attacks section we saw attacks based on the fact that UDP is connectionless.

3.3.1 Protecting against UDP attacks

Because UDP is connectionless, there is no simple way for a firewall to track connections and determine when traffic is not associated with a current "connection". One possibility is to filter on UDP ports, which is perfectly adequate for stopping protocols not used (for example SNMP). How-

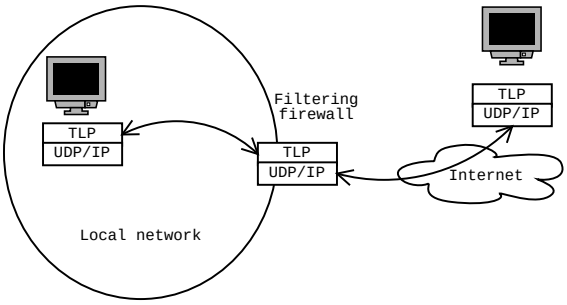

Figure 4: The UDP firewall approach works by encapsulating UDP packets in a TLP packet and having the firewall keep track of initiated UDP connections.

ever, only filtering on ports leave the opened ports unprotected.

Chang and Fung proposed in 2002 a method for implementing a stateful firewall for UDP [5] based on a transport layer proxy. A centralised proxy in the network (for example at the gateway/firewall) keeps track of UPD connections initiated from the internal network and throws away packets not asked for.

With this approach, the host on an internal network is protected from unwanted UDP traffic. However, there is still no way to protect the network resources on the external side of the firewall. An attacker could still flood the network connection all the way to the firewall (like in the game server attack [17]) which could make that connection useless for legitimate traffic.

3.4 ICMP - Internet Control Message Protocol

ICMP is used for traffic control on the Internet. It is connectionless and based on IP, and because of these properties it has been used in spoofed flooding attacks [34]. ICMP packets can be of about forty different types, and there is room in the specification for defining several more types. Some solutions to the traceback problem above suggest a new ICMP packet type [2].

Some of the ICMP messages can be used to interfere with IP routing. The *source quench* type was used earlier to indicate network congestion, but has been abandoned in favour of congestion control in the network layer. The *redirect* type is used to tell routers about ineffective routing paths. This could be used by an attacker to alter the routing table and make eavesdropping possible. The possible problem seems to be generally known, and the suggested default settings are to ignore redirect packets and possibly store them for further inspection.

The seen attacks based on ICMP largely depend on the possibility to spoof the sender IP. Another attack with ICMP appeared in 2002[8], where flooding a Cisco router with random redirect packets made it fill the entire memory with the new suggestions to the routing table. This is however more of a design error in Cisco IOS rather than a problem with ICMP Redirect itself.

3.5 SNMP - Simple Network Management Protocol

SNMP is widely used for management of IP-based networks. As its name suggests, SNMP is simple to use which is the reason for its popularity. But this is also its weakness, which will be explained below. SNMP works through UDP, IP, and often TCP; thereby sharing the weaknesses of those protocols. SNMP is now on its third release [29], and some work has been done to improve security, but many problems have still not been addressed. Another thing to take into consideration is that the first release of SNMP (SNMPv1) is still widely used. For a short overview of SNMP, we recommend SNMP&SNMPv2 by William Stalling [31].

CERT/CC (CERT Coordination Center) reports a number of vulnerabilities in many SNMP implementations[4]. Unless dealt with, these weak points can cause unauthorised access, DoS-attacks and other unpleasantries. Oulu University Secure Programming Group (OUSPG) tested SNMPv1 and found several weak points, that likely exist in both SNMPv2, and SNMPv3 as well. Namely problems with insecure settings of *community names* (a community is SNMP is similar to a user account with different access levels) and with spoofed UDP source addresses. The details of these weaknesses will not be discussed in this paper, since that would require extensive descriptions of the protocol itself. Instead we will look at what the weaknesses may lead to and what we can do to secure our systems.

3.5.1 What can happen?

CERT has not observed activities or tools that exploit the weaknesses but the risk of system compromise is very high. Most home users will not be directly threatened by this, since SNMP-services usually are disabled by default. SNMPv1 is, however, widely used in network infrastructure. Devices, such as routers and switches, use SNMP, which leads to the threat of large-scale network instability and outage, if someone starts to attack these vulnerabilities. The failure of one main router can cause a whole network to become unstable or unusable.

3.5.2 What we can do?

CERT/CC has a list of suggested actions to reduce the vulnerabilities in an environment using SNMP [4]. These methods are not solutions to the security problems in SNMP. The community name settings will always be a weak point in SNMP, but we can at least make it a little more troublesome for an attacker to cause any damage. The other problem, spoofed UDP source addresses, is not really a problem that can be dealt with on SNMP level.

SNMP patches

Several vendors have released patches, to improve the security of their products.

Disabling SNMP

CERT/CC recommends that all services, that are not explicitly required, should be disabled. This includes SNMP. Unfortunately, some products are still vulnerable to DoS-attacks, even with SNMP disabled [4].

Securing settings

Most SNMP products have the default settings for community strings; "public" for read-only access and "private" for read-write access. These should be changed immediately.

Filtering

Both *ingress* and *egress* filtering should be implemented to make the system more secure. Ingress filtering to prevent attacks against your system, and egress filtering to avoid being the launching pad for attacks against other systems. This is something we recommend any and all Internet users to do, whether they use SNMP or not. If all users firewalled their computers and filtered both incoming and outgoing traffic, many problems would diminish.

3.6 BGP - Border Gateway Protocol

The Border Gateway Protocol [22] is a routing protocol, using TCP, designed to decide what the routing paths should be between larger networks, or Autonomous Systems (AS). Each AS has boundary routers connecting the AS to another AS, and the AS has one or several *BGP speakers*. The speakers are the only ones authorised to communicate to a BGP speaker of another AS. A speaker may be a usual host or a router.

BGP has been designed so that politics can be mixed with the routing. A company, for example, may not be willing to let its out- and inbound traffic pass through a competitors network, or let traffic from other ASes flow through the network to another destination if the network has multiple connections. This policy driven routing is implemented in BGP to allow greater flexibility.

In operation, the BGP protocol establishes a link between two BGP speakers. This is done by sending a message to open and confirm connection parameters. Initially the entire BGP routing table is transfered, and after that updates are sent when the table changes. KeepAlive messages are sent periodically to ensure that the connection still is available.

3.6.1 Attacking BGP

The BGP protocol was not designed with security as the primary aspect. This makes the protocol vulnerable to attacks [28], some of which can have an immediate effect and others of a more information gathering type. The methods available is:

- Using a subverted BGP speaker
- Using an unauthorised BGP speaker
- Masquerading as an authorised BGP speaker

A subverted BGP speaker is an authorised BGP speaker that in some way have been compromised

so it violates the BGP protocol. It can do this by pure misconfiguration, or by causing the speaker to load unauthorised software or configuration information.

An unauthorised BGP speaker is a speaker that is not authorised as a speaker, but has managed to circumvent access control mechanism.

A masquerading BGP speaker is a node that masquerades itself as a authorised speaker. This can be done using IP spoofing or source routing attacks.

An attacker has a couple of approaches to use once he has chosen the method. If not wanting to make a direct denial of service attack against the compromised BGP node, the attacker can use the BGP node to collect information about confidential routing paths not open to the public. With this information the attacker can better plan a later attack and make sure the intended victim cannot escape to easily, or strike at a vulnerable point not visible without the knowledge of the confidential path.

If it is an immediate attack the attacker can try to change the routing paths leading to the victim. This can be done through a subverted BGP speaker, establishing a link between an unauthorised BGP speaker and an authorised BGP speaker, using a masquerading BGP speaker to take the role of an authorised speaker, or subverting the link through which BGP traffic flows.

The attacker can also sniff packets flowing through a compromised or disguised BGP node fairly easy, and this can be used after the attack on the BGP protocol.

The identified damage types [1] when a network as a whole is targeted by the attacker are as follows:

Network congestion More data traffic than can be handled is forwarded through a portion of the network.

Blackhole Large amounts of data traffic is directed to a single router which discards packets as it cannot handle the volume of data.

Looping Data traffic is forwarded along a looping route which swallows more and more packets as they are never delivered outside the loop and more packets are directed into the loop. This results in network congestion.

Partition Some portion of the network believes it is not connected to the rest when it in reality is connected.

Churn Rapid changes in the network forwarding which results in variations in data delivery patterns and affects congestion control techniques.

Instability Instability in the protocol makes a global forwarding state unachievable.

Overload Routing messages make a significant portion of the traffic being forwarded, denying legitimate traffic.

And when the attacker targets a host or part of a network they are:

Starvation Data intended for the network or host is forwarded to a network part that is unable to deliver the traffic.

Eavesdrop Traffic is forwarded through some node which now gets an opportunity to see the traffic and that node was originally not meant to see the traffic at all.

Cut The victim host or network is believed to be unreachable by a part of the network.

Delay Data is forwarded along a much slower path than it in fact could and should.

Looping Traffic is forwarded along a looping part and is never delivered to the host or network.

3.6.2 Proposed solutions to protocol insecurity

As more and more people are beginning to see that the BGP protocol is vulnerable to denial of service attacks more solutions are emerging. Some are major revamps of the BGP protocol while others just introduce some minor additions to the protocol to make it safer.

Session protection with MD5 signature

Protecting the session with an MD5 signature [10] is a simple way to reduce the risk of an attack on the BGP protocol. The MD5 algorithm is applied to the TCP pseudo-header, the TCP header, the TCP segment data if there is any and at last a

key or password known to both ends of the connection. The resulting 16-byte MD5 digest is then inserted into the header. This protects against spoofing since the attacker has to know both the TCP sequence number and the key/password used. This leads to a reduced risk of masquerading BGP speakers. However, it has been shown that the MD5 signature is vulnerable to collision search attacks [6].

S-BGP - Secure BGP

Secure BGP[16] is a make-over of the original BGP protocol and offers good protection, but it comes at a cost in both computing power and hardware requirements.

S-BGP is comprised of three components and these makes the BGP speaker able to validate authenticity and data integrity of the BGP UPDATEs it receives. The BGP speaker also uses the three components to verify the identity and authorisation of the senders. The components are two *Public Key Infrastructures* (PKIs), a new path attribute containing "attestations", and the use of IPsec (defined in RFC 2401 [15], 2402 [13] and 2406 [14] among others).

The two PKIs used are based on X.509 (v3) certificates, defined in RFC 2459 [11]. With these the BGP speaker is able to verify the identities and authorisation of other BGP speakers and of owners of ASes.

When verifying a route, the receiving BGP speaker needs one address attestation, together with one address allocation certificate, from each organisation owning an address block in the route. One route attestation from every S-BGP speaker along the path along with one certificate for each S-BGP speaker along the path to check the signatures on the attestations.

These mechanisms together with the use of IPsec enables good protection against attacks against the protocol. A unauthorised BGP speaker is not able to announce a path without knowing the key used to sign the UPDATE message. Masquerading as a authorised BGP speaker is prevented by the use of IPsec and the PKIs. However, this is achieved by a cost. To compute the validation of signatures the router needs a powerful enough processor to handle this and the rest of its tasks, but more are importantly the increase in RAM requirements which can make an introduction of S-BGP troublesome.

soBGP - Secure Origin BGP

The Secure Origin BGP protocol [20] is a new proposal that is still being worked on. The proposal extends BGP with a new message type, the SECURITY message. This message are to carry the security information within the BGP protocol. The ability to exchange SECURITY between two BGP speakers are negotiated at session startup. The SECURITY message is used to transport three types of certificates, namely:

The Entity Certificate (EC) The ECs are used to verify the existence of an entity in the routing system through a trust model.

The Policy Certificate (PC) The PC provides information about ASes which originate prefixes.

The Authorisation Certificate (AC) The AC provides authorisation information.

A router may rely on a third host do the actual verification of certificates and routes. The router can then focus on the routing task and is relieved from the computational overhead necessary for validating certificates. This makes the implementation of soBGP in the world a bit easier and cheaper for those not willing to replace old routers which are unable to handle the cryptographic tasks. It also makes administration of the Entity Certification database easier since only one needing the database is the host making the validation.

SoBGP is a proposal which has flexibility and security at the same level as S-BGP, but at a lower cost, since validation of certificates can be done by a central entity and thus relieving routers not fast enough.

3.6.3 Using BGP to prevent denial of service

There exists techniques where BGP is used to black-hole an attacked network by setting the next hop to a RFC 1918 [23] network (which is a range of IP networks reserved for private networks unconnected to the Internet). Most routers will route packets to a RFC 1918 network to the null interface.

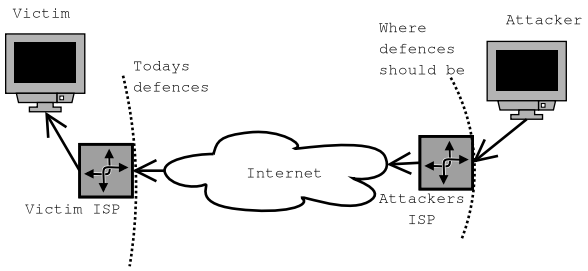

Figure 6: Todays research focus on stopping the attack at the victims ISP or at the victim host itself. If we could stop it at the attackers ISP, the attack would have less effects on the rest of the Internet.

This however black-holes the whole attacked network. D. Turk [36] proposes an enhanced version of this approach. After configuring border routers, a network operator can in the case of an attack advertise a route for the attacked network through BGP to be black-holed, and this on only the border routers where the attack traffic is coming in from. This means that valid traffic entering the Generic Threats to Routing Protocols network from transit points other than those the attack traffic is coming from will be forwarded as usual.

Combining this technique with a sinkhole or sinkhole tunnel, the traffic can be logged for analysis. With a sinkhole all traffic is dumped into a logging device, both legitimate and illegitimate traffic, resulting in the attacked network being denied of service for the legitimate traffic. With the sinkhole tunnel, the traffic is just directed through a tunnel where the traffic is logged and sent on its way again.

4 Conclusions

We believe the research and development focus should be on securing routers and IP. Several of the attacks above are based on IP spoofing, and if the problems with IP could be resolved, then filtering out attacks would be a lot easier. There would also be less need for solutions to the traceback problem.

If routers could filter out traffic with invalid source addresses, and this behaviour of routers could be enforced or encouraged, then spoofing, by both a single attacker and "drones", would be more difficult. If this is done as early in the network route as possible, it is easier to do and more can be gained from it.

All defence techniques stated above fail to address an important aspect of the attacks; even if my host manages to handle a flooding attack, my Internet connection will still be useless (at least in one direction) because it is "filled" with rogue packets.

References

[1] D. Beard. Generic threats to routing protocols, 2003. Work in progress. `http://www.ietf.org/internet-drafts/draft-ietf-rpsec-routing-threats-00%.txt`.

[2] S Bellovin. Icmp traceback messages, 2 2003. `http://www.research.att.com/~smb/`, Work in progress.

[3] D.J. Bernstein. Syn cookies. `http://cr.yp.to/syncookies.html`.

[4] CERT/CC. Cert® advisory ca-2002-03 multiple vulnerabilities in many implementations of the simple network management protocol (snmp), 2002. `http://www.cert.org/advisories/CA-2002-03.html`.

[5] K.P. Chang, R.K.C.; Fung. Transport layer proxy for stateful udp packet filtering. In *Computers and Communications, 2002. Proceedings. ISCC 2002. Seventh International Symposium on*, pages 595 –600, 2002.

[6] H. Dobbertin. The status of md5 after a recent attack. *RSA Labs' CryptoBytes*, 2, 1996. `http://www.rsa.com/rsalabs/pubs/cryptobytes.html`.

[7] Internet Engineering Task Force. Requirements for internet hosts – communication layers, 1989. `http://www.faqs.org/rfcs/rfc1122.html`.

[8] FX <fx@phenoelit.de>. Cisco ios icmp redirect dos, 2002. `http://cert.uni-stuttgart.de/archive/bugtraq/2002/05/msg00206.html`.

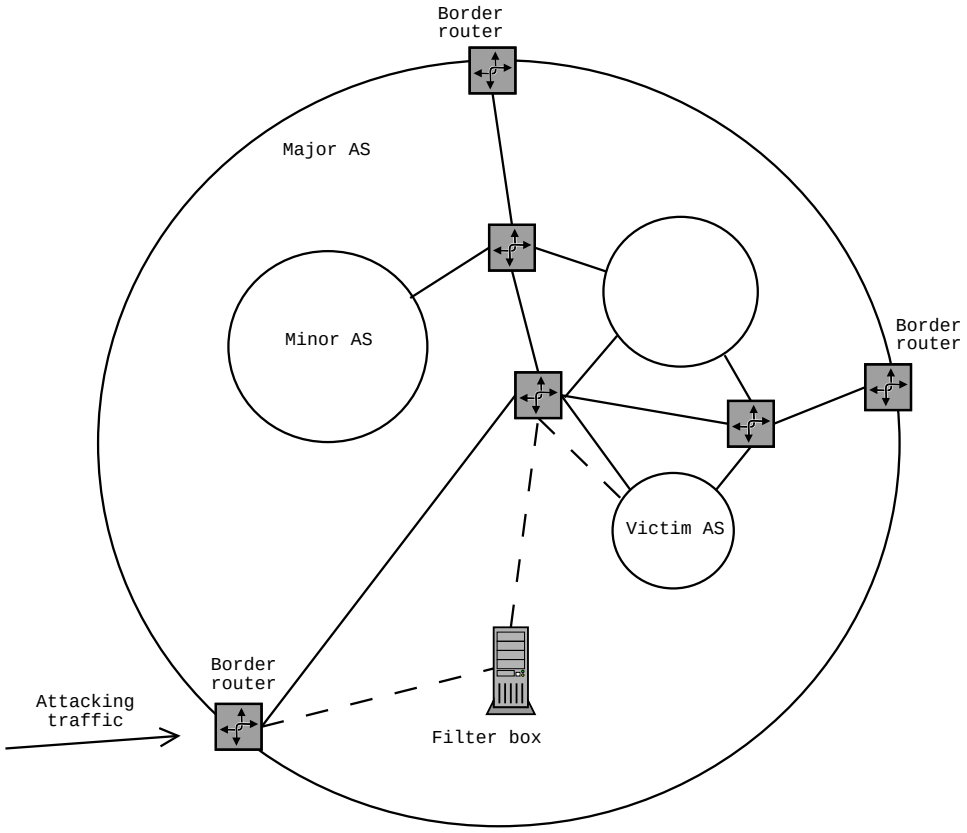

Figure 5: When an attack is detected, border routers are reconfigured to route traffic to the victim AS through a filter box while internal traffic is routed as usual.

[9] Jari Hautio and Tom Weckström. Denial of Service Attacks, 1999. http://www.hut.fi/u/tweckstr/hakkeri/DoS_paper.html.

[10] A Heffernan. Protection of bgp sessions via the tcp md5 signature option, 1998. http://www.faqs.org/rfcs/rfc2385.html.

[11] R. Housley. Internet x.509 public key infrastructure certificate and crl profile, 1999. http://www.faqs.org/rfcs/rfc2459.html.

[12] University of Southern California Information Sciences Institute. Transmission control protocol, 1981. http://www.faqs.org/rfcs/rfc793.html.

[13] S. Kent. Ip authentication header, 1998. http://www.faqs.org/rfcs/rfc2402.html.

[14] S. Kent. Ip encapsulating security payload (esp), 1998. http://www.faqs.org/rfcs/rfc2406.html.

[15] S. Kent. Security architecture for the internet protocol, 1998. http://www.faqs.org/rfcs/rfc2401.html.

[16] S Kent. Secure border gateway protocol (secure-bgp). *IEEE Journal on Selected Areas in Communications*, 18(4):582–592, 2000.

[17] Mike Kristovich. Multi-vendor Game Server DDoS Vulnerability, 2002. http://www.pivx.com/kristovich/adv/mk001/.

[18] C. Lee, S.C. Shields. Challenges to automated attack traceback. *IT Professional*, 4(3):12–18, 2002.

[19] J. Moy. Ospf version 2, 1998. http://www.faqs.org/rfcs/rfc2328.html.

[20] James Ng. Extensions to bgp to support secure origin bgp (sobgp), 2002. EXPIRED INTERNET DRAFT.

[21] J. Postel. User datagram protocol, 1980. http://www.faqs.org/rfcs/rfc768.html.

[22] Y Rekhert. Rfc 1771: A border gateway protocol 4 (bgp-4), 1995. http://www.faqs.org/rfcs/rfc1771.html.

[23] Y. Rekhter. Address allocation for private internets, 1996. http://www.faqs.org/rfcs/rfc1918.html.

[24] Livio Ricciulli, Patrick Lincoln, and Pankaj Kakkar. Tcp syn flooding defense, 1999. citeseer.nj.nec.com/73118.html.

[25] route|daemon9 <route@infonexus.com>. Linux and Windows IP fragmentation (Teadrop) bug, 1997. http://www.insecure.org/sploits/linux.fragmentation.teardrop.html.

[26] scacco. Possible Denial Of Service using DNS, 1999. http://www.geocrawler.com/archives/3/91/1999/7/0/2492001.

[27] E. Shenk. Another new thought on dealing with syn flooding, 1996. http://www.wcug.wwu.edu/lists/netdev/199609/msg00171.html.

[28] B. Smith and J. Garcia-Luna-Aceves. Securing the border gateway routing protocol, 1996.

[29] Inc. SNMP Research International. Snmpv3 specifications and documentation, 2003. http://www.snmp.com/snmpv3/.

[30] A. Song, D; Perrig. Advanced and authenticated marking schemes for ip traceback. *INFOCOM 2001. Twentieth Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings.*, 2:878–886, 2001.

[31] W. Stallings. Snmp and snmpv2 for net. *IEEE Communications Magazine*, 36(3):37–43, 1998.

[32] Dan Sterne and colleagues. Autonomic response to distributed denial of service attacks. *4th Int's Symp. Recent Advances in Intrusion Detection RAID*, pages 134–149, 2001.

[33] Andrew S. Tanenbaum. *Computer networks.* Prentice Hall PTR, 2003.

[34] TFreak. 'smurf' multi-broadcast icmp attack, 1997. http://lists.insecure.org/lists/bugtraq/1997/Oct/0070.html.

[35] <tsabin@razor.bindview.com>. BV-010: Jolt2 - Remote Denial of Service attack against Windows 2000, NT4, and Win9x , 2000. http://www.securityfocus.com/advisories/2240.

[36] D Turk. Configuring bgp to block denial-of-service attacks, 2003. Work in progress. http://www.ietf.org/internet-drafts/draft-turk-bgp-dos-04.txt.

[37] P Vixie. Events of 21-oct-2002, 11 2002. http://f.root-servers.org/october21.txt.