ICMP attacks against TCP

Fernando Gont
UTN/FRH, OpenBSD Project

FIRST Technical Colloquium
Buenos Aires, Argentina, October 5th-7th, 2005

Fault Isolation in the Internet Architecture

- The Internet Architecture relies on the Internet Control Message Protocol (ICMP) for the fault isolation function (i.e., for finding out network error conditions).

But ICMP is insecure, and hard to secure....

- ICMP messages can originate from any system of the Internet. You can't tell ahead of time which intermediate router will find a network error condition.
- IPsec does not help in this area (unless you assume you have or can set up dynamically a security association with every Internet system)

Generation of ICMP messages

- When a system detects a network error condition, it will usually issue an ICMP error message, to signal the error condition to the sending host.
- In order to allow the ICMP message to be demultiplexed, a piece of the original datagram that triggered the error message will be included. Namely, the entire IP header plus the first 64 bits of the original datagram's payload will be included.
- The IETF specifications do not recommend any type of checks on the received ICMP error messages

Case of TCP, as long as the ICMP payload contains the correct {source IP, source TCP port, destination IP, destination TCP port}, the error message will be passed to the corresponding transport protocol instance, and the corresponding action will be performed

ICMP attacks against TCP

ICMP can be used to perform a variety of attacks against TCP and other similar protocols. They include:

- Blind connection-reset attacks
- Blind throughput-reduction attacks
- Blind performance-degrading attacks

Information needed to attack

- Server-side IP address (usually known)
- Client-side IP address (usually known)
- Server-side TCP port (usually known)
- Client-side TCP port (usually not known, but can be guessed).

Most systems choose their "ephemeral ports" from some subset of the whole port number space. Thus, in practice, fewer packets than 65K are needed (in some implementations, as few as 4K). With a 128 kbps communications link, an attacker would need only a few seconds to perform any ICMP-based attack against TCP.

The weakest link in the chain

- None of the existing counter-measures (TCP MD5 option, IPSec, etc.) that help to protect TCP connections from other attacks (e.g., "Slipping in the window") will help to protect them from ICMP-based attacks.
- Fewer packets are required to perform ICMP-based attacks than those required for other attacks (e.g., "Slipping in the window").

This makes ICMP-based attacks the most trivial attacks that can be performed against TCP and similar protocols

Blind connection-reset attack

- Blindly resetting an arbitrary TCP connection

TCP's reaction to network errors

- The IETF specifications divide errors into "soft errors" and "hard errors".
- TCP's policy of reaction to network errors depends on the type of error being reported.
- If the network problem being reported is a "soft error", TCP will just record the error, and repeatedly retransmit its data until they either get acknowledged, or the connection times out.
- If the network problem being reported is a "hard error", TCP will immediately abort the corresponding connection.

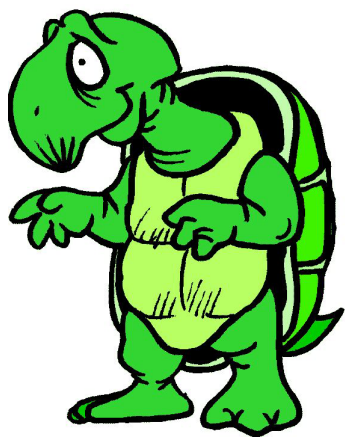
Blind connection-reset attack

An attacker could forge an ICMP error message that indicates a “hard error”, and thus reset an arbitrary TCP connection, even being off-path.

The most affected application protocols are those that rely on long-lived connections.

If the target is BGP, this attack could DoS entire networks

Blind throughput-reduction attack



- Blindly reducing the throughput of a TCP connection

Congestion control in the Internet Architecture

- The Internet Architecture provides little support for congestion control at the network layer. The only mechanism provided is a “choke packet” named “ICMP Source Quench”.

- Systems are supposed to send ICMP Source Quench messages to advice the sending host to slow down the rate at which it is transmitting data (the recommended practice is to put the connection in the “Slow Start” phase of TCP’s congestion control). In theory, they can be used both for congestion control (i.e., by routers) and for flow control (i.e., by end-systems).

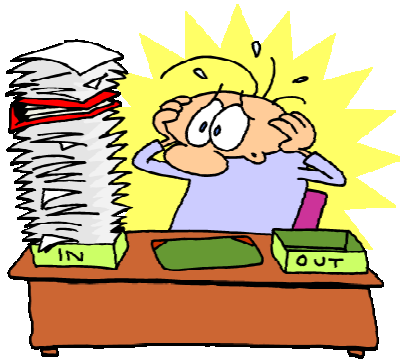
Blind throughput-reduction attack

An attacker could forge ICMP messages to fool the attacked host into thinking the network is congested, thus reducing the throughput of an arbitrary connection, even being off-path.

A continuous stream of ICMP Source Quench messages would reduce the throughput to about **1 packet per RTT** (Round-Trip Time).

If the target application is BGP, this attack might lead to inconsistencies in routing information, with the possibility of DoS entire networks

Blind performance-degrading attack



- Blindly degrading the performance of a TCP connection

How Path-MTU Discovery works

- IP packets are sent with the DF (“don’t fragment”) bit set.
- If a router finds that the packet cannot be forwarded without fragmenting it, it will discard the packet and issue a “fragmentation needed and DF bit set” ICMP error message. The message will include the MTU of the constricting communications link.
- Upon receipt of the ICMP message, the sending TCP will reduce the size of the packets it sends, accordingly.

Blind performance-degrading attack

An attacker could forge an ICMP “fragmentation needed and DF bit set” that reports a low MTU (as low as 68 bytes).

As a result, the attacked host will reduce the size of the packets it sends to the advertised Next-Hop MTU. Only minutes later the Path-MTU would be increased again. Thus,

- Overhead (headers/data ratio) will be increased, leading to throughput reduction.
- To maintain the same throughput, **packet rate** would have to be increased, leading to a degradation of the overall system performance.

If the target is BGP, the attack could DoS entire networks

Solving the problem

"I know the tendency of the human mind is to do anything rather than think, But mental labour is not thought, and those who have with labour acquired the habit of application, often find it much easier to get up a formula than to master a principle.“

- James Clerk Maxwell

“If anything at all, perfection is finally attained not when there is no longer anything to add, but when there is no longer anything to take away”

- Antoine de Saint Exupery

Let's stay tuned in the same frequency

- You cannot "filter all ICMP". You will break, at least, PMTUD.
- You still need PMTUD if you secure your connections by means of IPSec.
- TCP MD5 option will not help to solve this issues: You don't have enough data to recalculate the MD5 signature.
- IP source address spoofing is not needed. Ingress- and egress-filtering won't help you to solve this issues.
- If your argument for not paying attention and not solving this issues is that "it's old stuff", shame on you.
- If someone DoS you with 20-year-old attacks, even more shame on you.

Solving the blind connection-reset vulnerability

- **Are "hard errors" really hard?**

If a so-called "hard error" is reported for a connection that has already been established, then the error cannot be "hard". If it was, you shouldn't have been able to establish the connection!

For connections in any of the synchronized states, all ICMP errors should be considered "soft"

Solving the blind throughput-reduction vulnerability

- RFC 1812 states: *"A router SHOULD NOT originate ICMP Source Quench messages"*
- TCP implements its own congestion control mechanism, which does not use ICMP Source Quench messages.

Hosts should **ignore** ICMP Source Quench messages that are meant for TCP connections

Solving the blind performance-degrading vulnerability

- Require the ICMP message to be "in window" (i.e., refer to data already send but not yet ACKed. You do this for TCP, already!
- Divide PMTUD into to phases: Initial PMTUD, and PMTU Update.
- The first time you discover the PMTU for a connection (i.e. Initial PMTUD phase), perform the traditional PMTUD mechanism (i.e., honor the ICMP messages immediately).
- If the network tries to update the PMTU of your connection, be more cautious. Wait for a RTO (at least), and see if there's progress on the connection. If there isn't, honor the ICMP message. If there is, drop it.
- This way we can achieve for new connections the same convergence time as the traditional PMTUD mechanism (and thus we don't hurt interactive applications), but are still resistant to attack.
- In order to succeed, an attacker should be able to guess a valid TCP SEQ, **and** be able to drop either the data the attacker is sending, or the ACKs the remote TCP is sending. If he can do this, he has already DoS'ed you. (No need to bother with sending ICMP packets). (Ivan Arce's Laziness Principle sketched yesterday).

Lessons learned

"Isn't it ironic… don't you think?"

- Alanis Morisette

IETF, Part 1: Philosophy & broken principles

A well-known principle applied in the design of TCP/IP protocols is:

"Be liberal in what you accept, and conservative in what you send"

- RFC 1122

For today's environments, this principle should probably be changed to:

"Be conservative in **everything** you do"

IETF, Part 2: Pro-active security?

- The original ICMP specification dates back to 1982
- The Host Requirements RFC dates back to 1989
- The Router Requirements RFC dates back to 1996
- The original Path-MTU specification was issued back in November 1990
- A Path-MTU Discovery mechanism for IPv6 was issued back in August 1996

Yet in October 2005 the IETF specs leave the doors open for these attacks

IETF, Part 3: Slowness

- First version of the draft submitted in August 2004
- The TCPM working Group had **eight** months to adopt the draft as a WG item, or produce alternative work
- Yet in June 2005 we are still discussing if the specifications should be fixed.
- The TCPM WG adopted without **WG consensus** a draft (submitted by Cisco) to “fix” the “Slipping in the window” vulnerability, which requires to guess the TCP SEQ, though. Furthermore, the document proposes a modification to the TCP state machine. (Something supposed to be controversial for the IETF)
- The same people arguing in favor to fix the “Slipping in the Window” vulnerabilities (e.g., TCP-based reset attacks), argue against fixing the ICMP-based reset attacks.

Response time of the industry

- First version of the draft published in August 2004, which addressed the blind connection-reset and the blind-throughput reduction attacks.
- Version –02 published in early December 2004, which addressed the blind performance-degrading attack.

Yet in April 2005 many vendors were not prepared to handle these attacks.

(In October 2005, many vendors still aren't!)

Setting a release date

For CERTs, setting up a disclosure date is likely to be a hard issue:

- More responsive vendors (mainly open source ones) produced patches in terms of weeks
- It took big vendors **months** to patch their systems

Will their disclosure be considered "responsible" if Cisco or Microsoft are still vulnerable?

Cooperation with vendors: terminology

Cooperation: The act of cooperating, or of operating together to one end; joint operation; concurrent effort or labor.

Cooperation with vendors: An oxymoron?

Oxymoron: A rhetorical figure by which contradictory or incongruous terms are conjoined so as to give point to the statement or expression; an expression, in its superficial or literal meaning self-contradictory or absurd, but involving a point.

The so-called "responsible disclosure"

(a widely-adopted, broken mechanism)

- Researchers are faced with a situation in which "completely responsible disclosure" is not possible
- Vendors will try to patent solutions to the vulnerabilities. Possible options are:
  - Make the issue semi-public
  - Hire a lawyer, and issue a patent yourself
  - Announce the vulnerabilities to bugtraq, and make it a vendors' problem

An experience with “cooperation with vendors”

- Researcher described a number of vulnerabilities, proposed fixes to them, and provided audit tools.
- Researcher always available for contact (either e-mail or phone)
- Previews of new versions of the draft available to vendors

However, in response he got:

- Virtually no feedback from vendors (other than Sun Microsystems)
- Patent claims from vendors (and researcher the last party informed about what the patent was about)
- Suggestions that researcher’s activity could have helped terrorism
- Many discussions about getting credit, rather than vendors focusing on patching products
- Vendor’s engineers lobbying at the IETF to **not** adopt the counter-measures as standard recommendations (talk about the height of irony)

The work of independent researchers

- Work is usually done without any type or funding, payment, or support from any organization.
- The community (vendors, and end-users, finally) benefit from the output of the researcher’s work. (Even if he gets some output after weeks, months, of years!)
- They provide “free engineering”: “You have this problem, because of this and this. You can solve it this way. And no, you don’t have to pay me anything”.
- But their work usually depends, at some point, fromm access to equipment or other things. (Believe me, there are some things I cannot do with my P120, for example).
- And an acknowledgement (whether in a vulnerability report, a web site, or wherever) is the only thing that will caught a manager’s or organization’s attention.
- Thus, any discussion about getting or not getting credit for their work, is simply offensive.
- The researcher can, after all, get enough attention by e-mailing bugtraq instead of e-mailing you. You decide.

Questions and Answers

Fernando Gont
fernando@gont.com.ar

More info at:
http://www.gont.com.ar
http://www.li.frh.utn.edu.ar