WIKIPEDIA

SYN cookies

SYN cookie is a technique used to resist SYN flood attacks. The technique's primary inventor Daniel J. Bernstein defines SYN cookies as "particular choices of initial TCP sequence numbers by TCP servers." In particular, the use of SYN cookies allows a server to avoid dropping connections when the SYN queue fills up. Instead, the server behaves as if the SYN queue had been enlarged. The server sends back the appropriate SYN+ACK response to the client but discards the SYN queue entry. If the server then receives a subsequent ACK response from the client, the server is able to reconstruct the SYN queue entry using information encoded in the TCP sequence number.

Contents

Implementation

In order to initiate a TCP connection, the client sends a TCP SYN packet to the server. In response, the server sends a TCP SYN+ACK packet back to the client. One of the values in this packet is a *sequence number*, which is used by the TCP to reassemble the data stream. According to the TCP specification, that first sequence number sent by an endpoint can be any value as decided by that endpoint. SYN cookies are initial sequence numbers that are carefully constructed according to the following rules:

- let **t** be a slowly incrementing timestamp (typically `time()` logically right-shifted 6 positions, which gives a resolution of 64 seconds)
- let **m** be the maximum segment size (MSS) value that the server would have stored in the SYN queue entry
- let **s** be the result of a cryptographic hash function computed over the server IP address and port number, the client IP address and port number, and the value **t**. The returned value **s** must be a 24-bit value.

The initial TCP sequence number, i.e. the *SYN cookie*, is computed as follows:

- Top 5 bits: **t** mod 32
- Middle 3 bits: an encoded value representing **m**
- Bottom 24 bits: **s**

*(Note: since **m** must be encoded using 3 bits, the server is restricted to sending up to 8 unique values for **m** when SYN cookies are in use.)*

When a client sends back a TCP ACK packet to the server in response to the server's SYN+ACK packet, the client

must (according to the TCP spec) use *n+1* in the packet's *Acknowledgement number*, where *n* is the initial sequence number sent by the server. The server then subtracts 1 from the acknowledgement number to reveal the SYN cookie sent to the client.

The server then performs the following operations.

- Checks the value **t** against the current time to see if the connection has expired.
- Recomputes **s** to determine whether this is, indeed, a valid SYN cookie.
- Decodes the value **m** from the 3-bit encoding in the SYN cookie, which it then can use to reconstruct the SYN queue entry.

From this point forward, the connection proceeds as normal.

Drawbacks

The use of SYN cookies does not break any protocol specifications, and therefore should be compatible with all TCP implementations. There are, however, two caveats that take effect when SYN cookies are in use. Firstly, the server is limited to only 8 unique MSS values, as that is all that can be encoded in 3 bits. Secondly, the server must reject all TCP options (such as large windows or timestamps), because the server discards the SYN queue entry where that information would otherwise be stored.[1]

While these restrictions necessarily lead to a sub-optimal experience, their effect is rarely noticed by clients because they are only applied when under attack. In such a situation, the loss of the TCP options in order to save the connection is usually considered to be a reasonable compromise.

A problem arises when the connection-finalizing ACK packet sent by the client is lost, and the application layer protocol requires the server to speak first (SMTP and SSH are two examples). In this case, the client assumes that the connection was established successfully and waits for the server to send its protocol banner, or resend the SYN+ACK packet; however, the server is not aware of the session and will not resend the SYN+ACK because it discarded the backlog queue entry that would enable it to do so. Eventually, the client will abort the connection due to an application layer timeout, but this may take a relatively long time.[2]

Version 2.6.26 of the Linux kernel added limited support of TCP options, by encoding them into the timestamp.[3]

The newer TCP Cookie Transactions (TCPCT) standard is designed to overcome these shortcomings of SYN cookies and improve it on a couple of aspects. Unlike SYN cookies, TCPCT is a TCP extension and requires support from both endpoints.

Security considerations

Simple firewalls that are configured to allow all *outgoing* connections but to restrict which ports an *incoming* connection can reach (for example, allow incoming connections to a Web server on port 80 but restrict all other ports), work by blocking only incoming SYN requests to unwanted ports. If SYN cookies are in operation, care should be taken to ensure an attacker is not able to bypass such a firewall by forging ACKs instead, trying random sequence numbers until one is accepted. SYN cookies should be switched on and off on a *per-port* basis, so that SYN cookies being enabled on a public port does not cause them to be recognised on a non-public port.[4]

History

The technique was created by Daniel J. Bernstein and Eric Schenk in September 1996. The first implementation (for SunOS) was released by Jeff Weisberg a month later, and Eric Schenk released his Linux implementation in February 1997. FreeBSD implements syncookies since FreeBSD 4.5.[5]

See also

- SYN flood
- TCP Cookie Transactions

References

- D. J. Bernstein's own explanation of SYN cookies (http://cr.yp.to/syncookies.html)

1. [1] (http://cr.yp.to/syncookies/archive), cr.yp.to September 1996
2. András Korn, Defense mechanisms against network attacks and worms (pdf) (http://www.omikk.bme.hu/collections/phd/Villamosmernoki_es_Informatikai_Kar/2012/Korn_Andras/ertekezes.pdf), 2011
3. Patrick McManus, Improving Syncookies (https://lwn.net/Articles/277146/), lwn.net April 2008
4. "Archived copy" (https://archive.is/20130413192449/http://www.iss.net/security_center/reference/vuln/linux-syncookie-bypass-filter.htm). Archived from the original (http://www.iss.net/security_center/reference/vuln/linux-syncookie-bypass-filter.htm) on 2013-04-13. Retrieved 2013-03-17.
5. http://man.freebsd.org/syncookies
