WIKIPEDIA

Forward secrecy

In cryptography, **forward secrecy** (**FS**), also known as **perfect forward secrecy** (**PFS**), is a feature of specific key agreement protocols that gives assurances that session keys will not be compromised even if long-term secrets used in the session key exchange are compromised. For HTTPS, the long-term secret is typically the Private signing key of the server. Forward secrecy protects past sessions against future compromises of keys or passwords. By generating a unique session key for every session a user initiates, the compromise of a single session key will not affect any data other than that exchanged in the specific session protected by that particular key. This by itself is not sufficient for forward secrecy which additionally requires that a long-term secret compromise does not affect the security of past session keys.

Forward secrecy protects data on the transport layer of a network that uses common SSL/TLS protocols, including OpenSSL, when its long-term secret keys are compromised, as with the Heartbleed security bug. If forward secrecy is used, encrypted communications and sessions recorded in the past cannot be retrieved and decrypted should long-term secret keys or passwords be compromised in the future, even if the adversary actively interfered, for example via a man-in-the-middle attack.

The value of forward secrecy is that it protects past communication. This reduces the motivation for attackers to compromise keys. For instance, if an attacker learns a long-term key, but the compromise is detected and the long-term key is revoked and updated, relatively little information is leaked in a forward secure system.

The value of forward secrecy depends on the assumed capabilities of an adversary. Forward secrecy has value if an adversary is assumed to be able to obtain secret keys from a device (read access) but is either detected or unable to modify the way session keys are generated in the device (full compromise). In some cases an adversary who can read long-term keys from a device may also be able to modify the functioning of the session key generator, as in the backdoored Dual Elliptic Curve Deterministic Random Bit Generator. If an adversary can make the random number generator predictable, then past traffic will be protected but all future traffic will be compromised.

The value of forward secrecy is limited not only by the assumption that an adversary will attack a server by only stealing keys and not modifying the random number generator used by the server but it is also limited by the assumption that the adversary will only passively collect traffic on the communications link and not be active using a Man-in-the-Middle (MITM) attack. Forward secrecy typically uses an ephemeral Diffie-Hellman key exchange to prevent reading past traffic. The ephemeral Diffie-Hellman key exchange is often signed by the server using a static signing key. If an adversary can steal (or obtain through a court order) this static (long term) signing key, the adversary can masquerade as the server to the client and as the client to the server and implement a classic Man-in-the-Middle attack.[1]

Contents

History

The term "perfect forward secrecy" was coined by C. G. Günther in 1990[2] and further discussed by Whitfield Diffie, Paul van Oorschot, and Michael James Wiener in 1992[3] where it was used to describe a property of the Station-to-Station protocol.[4]

Forward secrecy has also been used to describe the analogous property of password-authenticated key agreement protocols where the long-term secret is a (shared) password.[5]

In 2000 the IEEE first ratified IEEE 1363, which establishes the related one-party and two-party forward secrecy properties of various standard key agreement schemes.[6]

Definition

An encryption system has the property of forward secrecy if plain-text (decrypted) inspection of the data exchange that occurs during key agreement phase of session initiation does not reveal the key that was used to encrypt the remainder of the session.

Example

The following is a hypothetical example of a simple instant messaging protocol that employs forward secrecy:

1. Alice and Bob each generate a pair of long-term, asymmetric public and private keys, then verify public-key fingerprints in person or over an already-authenticated channel. Verification establishes with confidence that the claimed owner of a public key is the actual owner.
2. Alice and Bob use a key exchange algorithm such as Diffie–Hellman, to securely agree on an ephemeral session key. They use the keys from step 1 only to authenticate one another during this process.
3. Alice sends Bob a message, encrypting it with a symmetric cipher using the session key negotiated in step 2.
4. Bob decrypts Alice's message using the key negotiated in step 2.
5. The process repeats for each new message sent, starting from step 2 (and switching Alice and Bob's roles as sender/receiver as appropriate). Step 1 is never repeated.

Forward secrecy (achieved by generating new session keys for each message) ensures that past communications cannot be decrypted if one of the keys generated in an iteration of step 2 is compromised, since such a key is only used to encrypt a single message. Forward secrecy also ensures that past communications cannot be decrypted if the long-term private keys from step 1 are compromised. However, masquerading as Alice or Bob would be possible going forward if this occurred, possibly compromising all future messages.

Attacks

Forward secrecy is designed to prevent the compromise of a long-term secret key from affecting the confidentiality of past conversations. However, forward secrecy cannot defend against a successful cryptanalysis of the underlying ciphers being used, since a cryptanalysis consists of finding a way to decrypt an encrypted message without the key, and forward secrecy only protects keys, not the ciphers themselves. A patient attacker can capture a conversation whose confidentiality is protected through the use of public-key cryptography and wait until the underlying cipher is broken (e.g. large quantum computers could be created which allow the discrete logarithm problem to be computed quickly). This would allow the recovery of old plaintexts even in a system employing forward secrecy.

Weak perfect forward secrecy

Weak perfect forward secrecy (wPFS) is the weaker property whereby when agents' long-term keys are compromised, the secrecy of previously established session-keys is guaranteed, but only for sessions in which the adversary did not actively interfere. This new notion, and the distinction between this and forward secrecy was introduced by Hugo Krawczyk in 2005.[7][8] This weaker definition implicitly requires that full (perfect) forward secrecy maintains the secrecy of previously established session keys even in sessions where the adversary *did* actively interfere, or attempted to act as a man in the middle.

Protocols

Forward secrecy is present in several major protocol implementations, such as SSH and as an optional feature in IPsec (RFC 2412). Off-the-Record Messaging, a cryptography protocol and library for many instant messaging clients, provides forward secrecy as well as deniable encryption.

In Transport Layer Security (TLS), cipher suites based on Diffie–Hellman key exchange (DHE-RSA, DHE-DSA) and elliptic curve Diffie–Hellman key exchange (ECDHE-RSA, ECDHE-ECDSA) are available. In theory, TLS can choose appropriate ciphers since SSLv3, but in everyday practice many implementations have refused to offer forward secrecy or only provide it with very low encryption grade.[9] TLS 1.3 leaves ephemeral Diffie–Hellman as the only key exchange mechanism to provide forward secrecy.[10]

OpenSSL supports forward secrecy using elliptic curve Diffie–Hellman since version 1.0,[11] with a computational overhead of approximately 15% for the initial handshake.[12]

The Signal Protocol uses the Double Ratchet Algorithm to provide forward secrecy.[13]

On the other hand, among popular protocols currently in use, WPA does not support forward secrecy.

Use

Forward secrecy is seen as an important security feature by several large Internet information providers. Since late 2011, Google provided forward secrecy with TLS by default to users of its Gmail service, Google Docs service, and encrypted search services.[11] Since November 2013, Twitter provided forward secrecy with TLS to its users.[14] Wikis hosted by the Wikimedia Foundation have all provided forward secrecy to users since July 2014[15] and are requiring the use of forward secrecy since August 2018.

Facebook reported as part of an investigation into email encryption that, as of May 2014, 74% of hosts that support STARTTLS also provide forward secrecy.[16] TLS 1.3, published in August 2018, dropped support for ciphers without forward secrecy. As of February 2019, 96.6% of web servers surveyed support some form of forward secrecy, and 52.1% will use forward secrecy with most browsers.[17]

At WWDC 2016, Apple announced that all iOS apps would need to use App Transport Security (ATS), a feature which enforces the use of HTTPS transmission. Specifically, ATS requires the use of an encryption cipher that provides forward secrecy.[18] ATS became mandatory for apps on January 1, 2017.[19]

The Signal messaging application employs forward secrecy in its protocol, notably differentiating it from messaging protocols based on PGP.[20]

German security-focused email provider Mailbox.org uses PFS and HSTS for messages in transit.[21]

See also

- Forward anonymity
- Diffie–Hellman key exchange
- Elliptic curve Diffie–Hellman

References

1. "tls - Does Perfect Forward Secrecy (PFS) make Man-in-the-Middle (MitM) attacks more difficult?" (https://security.stackexchange.com/questions/117101/does-perfect-forward-secrecy-pfs-make-man-in-the-middle-mitm-attacks-more-di). *Information Security Stack Exchange*. Retrieved 2020-10-11.
2. Günther, C. G. (1990). *An identity-based key-exchange protocol*. Advances in Cryptology EUROCRYPT '89 (LNCS 434). pp. 29–37.
3. Menzies, Alfred; van Oorscot, Paul C; Vanstone, SCOTT (1997). *Handbook of Applied Cryptography* (https://archive.org/details/handbookofapplie0000mene). CRC Pres. ISBN 978-0-8493-8523-0.
4. Diffie, Whitfield; van Oorschot, Paul C.; Wiener, Michael J. (June 1992). "Authentication and Authenticated Key Exchanges" (http://www.scs.carleton.ca/%7Epaulv/papers/sts-final.pdf) (PDF). *Designs, Codes and Cryptography*. **2** (2): 107–125. CiteSeerX 10.1.1.59.6682 (https://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.59.6682). doi:10.1007/BF00124891 (https://doi.org/10.1007%2FBF00124891). S2CID 7356608 (https://api.semanticscholar.org/CorpusID:7356608). Retrieved 2013-09-07.
5. Jablon, David P. (October 1996). "Strong Password-Only Authenticated Key Exchange". *ACM Computer Communication Review*. **26** (5): 5–26. CiteSeerX 10.1.1.81.2594 (https://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.81.2594). doi:10.1145/242896.242897 (https://doi.org/10.1145%2F242896.242897). S2CID 2870433 (https://api.semanticscholar.org/CorpusID:2870433).
6. "IEEE 1363-2000 - IEEE Standard Specifications for Public-Key Cryptography" (https://standards.ieee.org/findstds/standard/1363-2000.html). *standards.ieee.org*. Retrieved 2018-06-14.
7. Krawczyk, Hugo (2005). *HMQV: A High-Performance Secure Diffie-Hellman Protocol* (http://eprint.iacr.org/2005/176). Advances in Cryptology – CRYPTO 2005. Lecture Notes in Computer Science. **3621**. pp. 546–566. doi:10.1007/11535218_33 (https://doi.org/10.1007%2F11535218_33). ISBN 978-3-540-28114-6.
8. Cremers, Cas; Feltz, Michèle (2015). "Beyond eCK: perfect forward secrecy under actor compromise and ephemeral-key reveal" (https://www.cs.ox.ac.uk/people/cas.cremers/downloads/papers/CrFe2012-eckpfs.pdf) (PDF). *Designs, Codes and Cryptography*. **74** (1): 183–218. CiteSeerX 10.1.1.692.1406 (https://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.692.1406). doi:10.1007/s10623-013-9852-1 (https://doi.org/10.1007%2Fs10623-013-9852-1). S2CID 53306672 (https://api.semanticscholar.org/CorpusID:53306672). Retrieved 8 December 2015.

9. Discussion on the TLS mailing list in October 2007 (https://web.archive.org/web/20170702024607/http://www.ietf.org/mail-archive/web/tls/current/msg02134.html)
10. "A Detailed Look at RFC 8446 (a.k.a. TLS 1.3)" (https://blog.cloudflare.com/rfc-8446-aka-tls-1-3/). *The Cloudflare Blog*. 2018-08-10. Retrieved 2019-02-26.
11. "Protecting data for the long term with forward secrecy" (http://googleonlinesecurity.blogspot.com.au/2011/11/protecting-data-for-long-term-with.html). Retrieved 2012-11-05.
12. Vincent Bernat. "SSL/TLS & Perfect Forward Secrecy" (https://vincent.bernat.im/en/blog/2011-ssl-perfect-forward-secrecy.html). Retrieved 2012-11-05.
13. Unger, Nik; Dechand, Sergej; Bonneau, Joseph; Fahl, Sascha; Perl, Henning; Goldberg, Ian; Smith, Matthew (17–21 May 2015). *SoK: Secure Messaging* (http://www.ieee-security.org/TC/SP2015/papers-archived/6949a232.pdf) (PDF). *2015 IEEE Symposium on Security and Privacy*. San Jose, CA: Institute of Electrical and Electronics Engineers. p. 241. doi:10.1109/SP.2015.22 (https://doi.org/10.1109%2FSP.2015.22). ISBN 978-1-4673-6949-7. S2CID 2471650 (https://api.semanticscholar.org/CorpusID:2471650). Retrieved 4 December 2015.
14. Hoffman-Andrews, Jacob. "Forward Secrecy at Twitter" (https://blog.twitter.com/2013/forward-secrecy-at-twitter-0). *Twitter*. Twitter. Retrieved 25 November 2013.
15. "Tech/News/2014/27 - Meta" (https://meta.wikimedia.org/wiki/Tech/News/2014/27). Wikimedia Foundation. 2014-06-30. Retrieved 30 June 2014.
16. "The Current State of SMTP STARTTLS Deployment" (https://www.facebook.com/notes/protect-the-graph/the-current-state-of-smtp-starttls-deployment/1453015901605223). Retrieved 7 June 2014.
17. Qualys SSL Labs. "SSL Pulse" (https://web.archive.org/web/20190215213454/https://www.ssllabs.com/ssl-pulse/). Archived from the original (https://www.ssllabs.com/ssl-pulse/) (3 February 2019) on 15 February 2019. Retrieved 25 February 2019.
18. https://developer.apple.com/library/ios/releasenotes/General/WhatsNewIniOS/Articles/iOS9.htm SW14
19. "App Transport Security REQUIRED January 2017 | Apple Developer Forums" (https://forums.developer.apple.com/thread/48979). *forums.developer.apple.com*. Retrieved 2016-10-20.
20. Evans, Jon (22 January 2017). "WhatsApp, Signal, and dangerously ignorant journalism" (https://techcrunch.com/2017/01/22/whatsapp-signal-and-dangerously-ignorant-journalism/). *TechCrunch*. Retrieved 18 April 2018.
21. Sven Taylor (7 November 2019). "Mailbox.org Review" (https://restoreprivacy.com/mailbox-org/). Restore Privacy. Retrieved 8 November 2019.

External links

- RFC 2412 IETF, H. Orman. The OAKLEY Key Determination Protocol
- Forward-secure-survey (https://www.cs.bu.edu/~itkis/pap/forward-secure-survey.pdf) An overview
- Perfect Forward Secrecy can block the NSA from secure web pages, but no one uses it (https://web.archive.org/web/20130629200207/http://blogs.computerworld.com/encryption/22366/can-nsa-see-through-encrypted-web-pages-maybe-so) Computerworld June 21, 2013
- SSL: Intercepted today, decrypted tomorrow (http://news.netcraft.com/archives/2013/06/25/ssl-intercepted-today-decrypted-tomorrow.html) Netcraft June 25, 2013
- Deploying Forward Secrecy (https://community.qualys.com/blogs/securitylabs/2013/06/25/ssl-labs-deploying-forward-secrecy) SSL Labs June 25, 2013
- SSL Labs test for web browsers (https://www.ssllabs.com/ssltest/viewMyClient.html)
- SSL Labs test for web servers (https://www.ssllabs.com/ssltest/index.html)