R209 Essay: Cryptographic Protocols

Chongyang Shi (*cs940*)

November 17, 2017

1 Summaries of research

The classic paper by Burrows et al. [1] produced formalised logic to verify the security of published authentication protocols at the time. The paper started by setting out postulates and important distinctions in the logic, as well as rules for idealising protocols and deriving legal annotations. Taking into account formalised goals of authentication protocols, they proceeded to verify published authentication protocols in a process of idealising the protocol, checking **believes** and **sees** at each stage of the protocol, and verifying resulting states. Various subtle differences between protocols leading to insecurities were discovered in the process. A possible critique to the logic used is the assumption that all beliefs are equally weighted, which may not be the case in practice, due to varying reliabilities of knowledges leading to these beliefs. For example, the risk of client and server clocks going out of synchronisation is greater than a public key prime number being factorised, even though both will invalidate corresponding beliefs.

The book chapter by Anderson [2] produced a survey on attacks on systems whose security components are separated and interact through APIs, with case studies to illustrate recurring vulnerabilities. Past vulnerabilities described include hardware security modules exploited by replaying past encrypted keys or planting chosen MAC keys; decryption of IBM 4758 DES through a meet-in-the-middle birthday attack; and attacks on system call wrappers by exploiting the concurrent nature of modern processors. As many of the vulnerabilities discussed were exploited via transactions that are not widely used or necessary features, the author concluded by calling for minimal inclusion of transactions in secure modules. A possible alternative structure of the chapter is to organise sections by recurring vulnerability, such as the chosen zero plain text attack used on both VSM and IBM PIN generation, and draw case studies to illustrate how each recurring vulnerability has been exploited in different ways.

A recent paper by Beurdouche et al. [3] demonstrated vulnerabilities in state machines of popular TLS implementations. Based on a reference state machine produced from TLS specifications, they performed automated protocol-aware state machine fuzzing on the state machines of TLS implementations. They discovered various flaws in these implementations that can allow server and client impersonation, rolling back forward secrecy, and protocol downgrade attacks. The authors also produced a verified state machine of their own that is secure from these flaws. They concluded with a discussion on challenges faced in achieving full security theorems for OpenSSL. Through responsible disclosure, developers of TLS implementations tested have produced patches for the vulnerabilities identified.

2 Key themes of research

2.1 Reliability of protocol assumptions

For security protocol designers, it is nearly always necessary for them to make a set of assumptions on the parties involved in the protocol, as well as the environment in which the protocol resides. Examples of such assumptions can be seen in Burrows et al. [1, Sec. 1, 2.2], which assumes that both parties of the authentication are trustworthy, and are using secure encryptions with their secret keys. However, sometimes these assumptions can be over-optimistic and lead to security issues in imperfect implementations, such as system call wrappers being only secure on non-concurrent processors [2, 18.3] and TLS assuming strict message ordering and distinguishable optional messages [3, Sec. II].

2.2 Unverified information should not be trusted

Cryptographic protocols often break when unverified external information are relied upon to perform secure functionalities. The idealisation process applied to authentication protocols by Burrows et al. [1, 2.4] strips out clear text messages, as they can be easily forged. If the timing and origin of terminal key-producing transactions can be verified by VSM, the XOR-to-null-key attack described by Anderson [2, 18.2.1] would be a lot harder to perform. Unauthenticated *ServerFinished* messages also allow server impersonation by an attacker against the Java TLS implementation, as described by Beurdouche et al. [3, V-A].

2.3 Unnecessary components in authentication protocols

In some cases, not all parts of a cryptographic protocol are useful to a user, and their inclusion may impact performance or even introduce vulnerabilities. Formal logic analysis by Burrows et al. [1, 4.1, 5.1] discovered unnecessary procedures in both Kerberos and Andrew file system that could reduce performance. Anderson [2] highlighted several vulnerabilities in hardware security modules brought in through implementing normally unnecessary transactions. Anderson also recommended module implementers to carefully minimise transactions. Rarely used insecure ciphersuites included by TLS implementations for backwards compatibility have also allowed some downgrade attacks to take place [3, V-D].

3 Ideas in current context

Burrows et al. [1, 6.1] noted the reliance on the third-party key server to correctly sign certificates and generate fresh secrets in the Needham-Schroeder protocol [4]. This is no longer a common feature in modern public key systems owing to developments in attacks against communications between the client and the key server, such as DNS poisoning of the key server's hostname. The role of a modern public key server such as a PGP key server is limited to distributing public keys and revocation notices of clients [5, 4.3]. Some other protocols are still in wide-use today, such as Kerberos and X.509, with continuing security improvements.

In security verification of authentication protocols, Burrows et al. [1, 2.5] utilised logical formulas analogical to Hoare logic, which has also been part of bases for a wide range of formal languages, such as the communicating sequential processes (CSP) [6]. Designed for describing distributed programs, CSP has been used in conjunction with model checkers for detecting errors in key exchange protocols. Lowe [7] in 1996 used CSP to discover a flaw in the original Needham-Schroeder public key protocol, and subsequently proved the security of his adapted version of the protocol. Similar analysis has been done by Lowe and Roscoe [8] on the TMN protocol in 1997.

There have been developments in mitigating Time-Of-Check-To-Time-Of-Use (TOCTTOU) attacks discovered by Watson [9] as mentioned. Mitigations are achieved through sandboxing, which can be done on threads by verifying individual instructions [10], or on user-level through delegation [11]. These mitigations techniques allow system call wrappers on modern concurrent hardware to be more resilient against timing attacks. They however still suffer from other disadvantages mentioned by Anderson [2, 18.3] such as high complexity.

4 Literature review

The formal logic-based methodology developed by Burrows et al. [1] has been the bases of correctness proofs of other modern security protocols, such as secure routing for mobile ad hoc networks [12]. However, this method has also been criticised due to is inability in capturing flaws not reasoned by the underlying logic [13]. Flaws identified by Burrows et al. also became a good guideline for avoiding common flaws when designing new protocols, such as by Diffie et al. [14].

Hardware security modules (HSMs) as discussed by Anderson [2] have also been implemented for vehicular communication security, such as by Kargl et al. [15], who formally verified that their HSM API is secure against root key implanting and key revealing attacks. A notable further security analysis on EMV smartcards by Bond et al. [16] discovered critical flaws in the generation and verification processes of random numbers, significantly reducing perceived security of EMV systems. Yang et al. [17] gave a comprehensive analysis on the state of operating system concurrency attacks in 2012.

Security analysis of TLS implementations by Beurdouche et al. [3] attracted significant academic interest. Another TLS downgrade attack ("Logjam") was discovered by Adrian et al. [18], based on a vulnerability of the TLS protocol itself rather than its implementations, rendering export-grade Diffie-Hellman ciphersuites fully insecure. While not strictly a downgrade attack, Aviram et al. [19] developed another implementation-independent attack ("DROWN") leveraging SSLv2 to decrypt TLS data on a server that supports the legacy SSLv2.

(1212 words according to texcount.)

References

[1] M. Burrows *et al.*, "A logic of authentication," in *Proceedings of the Royal Society of London A: Mathematical, Physical and Engineering Sciences*, vol. 426, no. 1871. The Royal Society, 1989, pp. 233–271.

[2] R. J. Anderson, *Security engineering: a guide to building dependable distributed systems.* John Wiley & Sons, 2010.

[3] B. Beurdouche *et al.*, "A messy state of the union: Taming the composite state machines of tls," in *Security and Privacy (SP), 2015 IEEE Symposium on.* IEEE, 2015, pp. 535–552.

[4] R. M. Needham and M. D. Schroeder, "Using encryption for authentication in large networks of computers," *Communications of the ACM*, vol. 21, no. 12, pp. 993–999, 1978.

[5] A. Whitten and J. D. Tygar, "Why johnny can't encrypt: A usability evaluation of pgp 5.0." in *USENIX Security Symposium*, vol. 348, 1999.

[6] L. Lamport and F. B. Schneider, "The"hoare logic"of csp, and all that," *ACM Transactions on Programming Languages and Systems (TOPLAS)*, vol. 6, no. 2, pp. 281–296, 1984.

[7] G. Lowe, "Breaking and fixing the needham-schroeder public-key protocol using fdr," in *International Workshop on Tools and Algorithms for the Construction and Analysis of Systems.* Springer, 1996, pp. 147–166.

[8] G. Lowe and B. Roscoe, "Using csp to detect errors in the tmn protocol," *IEEE transactions on Software Engineering*, vol. 23, no. 10, pp. 659–669, 1997.

[9] R. N. Watson, "Exploiting concurrency vulnerabilities in system call wrappers." *WOOT*, vol. 7, pp. 1–8, 2007.

[10] M. Payer and T. R. Gross, "Fine-grained user-space security through virtualization," *ACM SIGPLAN Notices*, vol. 46, no. 7, pp. 157–168, 2011.

[11] R. Xu *et al.*, "Aurasium: practical policy enforcement for android applications." in *USENIX Security Symposium*, vol. 2012, 2012.

[12] P. Papadimitratos and Z. J. Haas, "Secure routing for mobile ad hoc networks," in *the SCS Commnication Networks and Distributed Systems Modeling and Simulation Conference (CNDS), San Antonio, TX, January 27-31, 2002*, 2002, pp. 193–204.

[13] M. Bellare and P. Rogaway, "Entity authentication and key distribution." in *Crypto*, vol. 93. Springer, 1993, pp. 232–249.

[14] W. Diffie *et al.*, "Authentication and authenticated key exchanges," *Designs, Codes and cryptography*, vol. 2, no. 2, pp. 107–125, 1992.

[15] F. Kargl *et al.*, "Secure vehicular communication systems: implementation, performance, and research challenges," *IEEE Communications Magazine*, vol. 46, no. 11, 2008.

[16] M. Bond *et al.*, "Chip and skim: cloning emv cards with the pre-play attack," in *Security and Privacy (SP), 2014 IEEE Symposium on.* IEEE, 2014, pp. 49–64.

[17] J. Yang *et al.*, "Concurrency attacks," *HotPar*, vol. 12, p. 15, 2012.

[18] D. Adrian *et al.*, "Imperfect forward secrecy: How diffie-hellman fails in practice," in *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security.* ACM, 2015, pp. 5–17.

[19] N. Aviram *et al.*, "Drown: Breaking tls using sslv2." in *USENIX Security Symposium*, 2016, pp. 689–706.