Comparison of Modern Network Attacks on TLS Protocol

Oleksandr Ivanov, Victor Ruzhentsev
Department of Information Technology Security,
Kharkiv National University of Radio Electronics
Kharkiv, Ukraine
oa1.ivanov@gmail.com, viktor.ruzhentsev@nure.ua

Roman Oliynykov
Department of Information Systems and Technologies Security,
V.N. Karazin Kharkiv National University
Kharkiv, Ukraine
roliynykov@gmail.com

***Abstract*—The Transport Layer Security (TLS) is cryptographic protocol that provides confidentiality and integrity of data in untrusted networks connections. The protocol is composed of two layers: the TLS Record Protocol for encapsulation of various higher-level protocols and the TLS Handshake Protocol for connection security. Nowadays TLS has become the secure standard of choice for Internet and mobile applications. There are many attacks on the TLS protocol that exploit its vulnerabilities: Cipher Block Chaining (CBC) mode encryption, data compression, using obsolete cypher suites and hash functions. Therefore, it is necessary to identify and examine existing threats of TLS. The paper explores the TLS protocol versions and main differences between them. In addition we consider the brand-new TLS version — TLS 1.3. Well-known network attacks (BEAST, CRIME, BREACH and DROWN), several additional attacks (SLOTH, ROBOT, Lucky 13) and their features are also considered. Finally, we compare reviewed attacks using own criteria that will help to understand the security of the TLS protocol at that moment.**

Keywords—TLS; BEAST; CBC-mode encryption; CRIME; BREACH; HTTP compression; DROWN; ROBOT; Lucky 13

I. INTRODUCTION

Today information security is essential for protection information and communication systems [1]. It includes a broad set of security methods that apply to the following objects:

- Data transmission [2, 3].
- Internet security [4, 5].
- Cryptography [6, 7].
- Data types representations [8, 9].
- Database accesabilitiy protection [10, 11].
- Security of financial structures and related operations [12].

Now it takes a huge role to develop the cryptographic algorithms and ciphers that meet of requirements of post-quantum systems [13, 14]. Ukraine has a big deal in designing stream and block ciphers for post-quantum primitives [6, 7]. Block ciphers have a special place: they are well researched in Ukraine and represented by the various realization of ciphers, hash functions [15, 16] etc.

The TLS protocol is widely used for securing communication over the Internet nowadays [17]. The protocol uses cipher suites, key exchange algorithms and certificates to provide privacy, data integrity and authentication of communicating parties [18, 19, 20]. Originally, TLS was started as the Secure Sockets Layer (SSL). Then it was adopted by the Internet Engineering Task Force (IETF) and specified as TLS 1.0 [17]. Many modern network protocols (e.g., HTTPS, SMTP, FTP, LDAP) use TLS for securing an application-level traffic [20].

Since 2011, this protocol is actively being explored. BEAST and CRIME are the first attacks that proposed by Thai Duong and Juliano Rizzo [21, 22]. These attacks appropriately exploit the vulnerabilities of CBC-mode (Cipher Block Chaining) encryption and HTTP-level compression [23]. Afterwards, information security researchers have found new vulnerabilities of the TLS protocol and proposed a bunch of attacks (e.g. BREACH, DROWN, SLOTH) [24, 25, 26]. As a result, the protocol evolves, and was proposed the new version by internet researchers: TLS 1.3 that is in working draft [27]. Therefore, the issue of finding vulnerabilities and improvements of the TLS protocol is still relevant [23, 27].

This paper explores the TLS protocol and its functioning. Work of the different TLS versions is also considered. The next section provides a brief overview of main attacks (BEAST, CRIME, BREACH, DROWN) on the protocol and some additional attacks too (e.g., ROBOT, Lucky 13) [28, 29]. In final section we compare the TLS attacks with various criteria that will show, in our view, the relevance of the TLS securing.

II. THE TLS PROTOCOL

A. Common Description of The TLS Protocol

The primary goal of the TLS protocol is to provide privacy and data integrity between two communicating applications. The protocol consist of two levels: the TLS Record Protocol and the TLS Handshake Protocol [19, 20].

The Record Protocol is used for encapsulation of various higher-level protocols. The Handshake Protocol allows the server and client to authenticate each other and to negotiate an encryption algorithm and cryptographic keys before the application protocol transmits or receives its first byte of data [19, 20].

Thereby, secure connection between a client and a server supported by this protocols have one and more following properties [20]:

978-1-5386-6611-1/18/$31.00 ©2018 IEEE

- The connection is private (TLS Record). Symmetric cryptography is used for data encryption (e.g., AES). The keys for this symmetric encryption are generated uniquely for each connection and are based on a secret negotiated by another protocol.
- The connection is reliable (TLS Record). Message transport includes a message integrity check using a keyed Message Authentication Code (MAC). Secure hash functions (e.g., SHA-2) are used for MAC computations.
- The peer's identity can be authenticated (TLS Handshake) using asymmetric cryptography (e.g., DSA etc.). This authentication can be made optional, but is generally required for at least one of the peers.
- The negotiation of a shared secret is secure (TLS Handshake). The negotiated secret is unavailable to eavesdroppers, and for any authenticated connection, the secret cannot be obtained, even by an attacker who can place himself in the middle of the connection.
- The negotiation is reliable (TLS Handshake). No attacker can modify the negotiation communication without being detected by the parties to the communication.

B. Versions of The TLS Protocol

TLS 1.0 was released in 1999 and published as RFC 2246. This version of TLS was very similar to SSL 3.0. TLS 1.0 does include a means by which a TLS implementation can downgrade the connection to SSL 3.0, thus weakening security [18, 30].

TLS 1.1 was released in 2006 (RFC 4346) and it is the second version of TLS. The major differences between TLS 1.1 and TLS 1.0 include the following [19, 31]:

- The implicit Initialization Vector (IV) is replaced with an explicit Initialization Vector for protection against CBC attacks.
- Padding error handling is modified to use "bad_record_mac" alert rather than "decryption_failed" alert to protect against CBC attacks.
- Internet Assigned Numbers Authority (IANA) registries are defined for protocol parameters.
- A premature close no longer causes a session to be non-resumable.

TLS 1.2 (RFC 5246) is currently the most used version of TLS and has made several improvements in security compared to TLS 1.1. The major differences between TLS 1.2 and previous version are presented below [20, 31]:

- The MD5/SHA-1 combination in the pseudorandom function (PRF) has been replaced with cipher-suite-specified PRFs.
- The MD5/SHA-1 combination in the digitally signed element has been replaced with a single hash. Signed elements now include a field that explicitly specifies the hash algorithm used.
- Substantial cleanup to the client's and server's ability to specify which hash and signature algorithms they will accept.
- Addition of support for authenticated encryption with additional data modes.
- Added AES and HMAC-SHA256 cipher suites.

Thereby, the greater enhancement in encryption of TLS 1.2 is using secure hash algorithms such as SHA-256 as well as advanced cipher suites that support elliptical curve cryptography [20].

TLS 1.3 aims to further improve upon the security protocol. This newest version of the protocol is represented by a working draft. The major differences from TLS 1.2 include the following [27, 32]:

- A Zero Round Trip Time (0-RTT) mode was added, saving a round-trip at connection setup for some application data at the cost of certain security properties.
- Reworking handshake to provide 1-RTT mode. All handshake messages after the message "Server Hello" are now encrypted.
- The handshake state machine has been restructured to be more consistent and remove superfluous messages.
- Removing support for weak and lesser-used named elliptical curves.
- Removing support for MD5 and SHA-224.

The main improvement of the new version is the TLS 1.3 Handshake that is involves only one round-trip as opposed to three in TLS 1.2. In result, latency is reduced. The TLS 1.3 Handshake consist of three steps [27, 32]:

- Step 1. The TLS 1.3 handshake commences with the "Client Hello" message. The client sends the list of supported cypher suites and guesses which key agreement protocol the server is likely to select. The client also sends its key share for that particular key agreement protocol.
- Step 2. In reply to the "Client Hello" message, the server replies with the key agreement protocol that it has chosen. The "Server Hello" message also comprises of the server's key share, its certificate as well as the "Server Finished" message. The "Server Finished" message, which was sent in the sixth step in TLS 1.2 handshake, is sent in the second step.
- Step 3. Now, the client checks the server certificate, generates keys as it has the key share of the server, and sends the "Client Finished" message. Hence, the encryption of the data begins.

III. Network Attacks on TLS

A. The BEAST Attack

BEAST is a "browser exploit against SSL/TLS" that was revealed in September 2011. This attack leverages weaknesses in CBC to exploit the SSL/TLS protocol [21, 33].

There are five conditions that must be carried out for this attack to take place [21]:

- The Web server must have SSL 3.0/TLS 1.0 (or older versions of the SSL protocol).
- JavaScript (JS) or applet injection into the same origin of the web site.
- Network sniffing of the connection must be possible.
- A vulnerable version of SSL must be a block cipher with CBC.
- An attacker should be able to inject his own messages into an SSL channel.

Attacker can insert his data into an SSL/TLS channel using following methods [21]:

- SSL VPN protocol (in Cisco systems).
- Java-plugin that bypasses Same Origin Policy (SOP).
- JS code that initiates connection to the target server with SSL/TLS protocol.

User can be protected from the BEAST attack using the following ways [21, 33]:

- Use newer versions of the TLS protocol (1.1 or 1.2).
- If you have an older version of TLS or SSL on the server, you must use stream ciphers (e.g., RC4).

B. The CRIME Attack

CRIME stands for "Compression Ratio Info-leak Made Easy". It's a security exploit against secret web cookies over connections using the HyperText Transfer Protocol Secure (HTTPS) and SPDY (speedy) protocols that also use data compression. [22, 34].

CRIME is also a MITM. Its algorithm is represented below [22, 34]:

- The attacker generated HTTP requests using Deflate compression to the target website with SSL/TLS support.
- With each new request, the attacker increases the extra data in the HTTP request (session token).
- The attacker then analyzes the length of HTTP requests after compression and tries to find the correct cookie value for the victim session.
- If the value is selected correctly, the redundancy of the HTTP request will increase, and the length of the HTTP response will decrease accordingly.
- Thus, the attacker will be able to guess the first character of the session token. Then the same procedure is repeated until all token are found.

The CRIME attack can be defeated by preventing the use of compression at the client side (disabling the compression in browser) or by the website preventing the use of data compression [22, 35].

C. The BREACH Attack

BREACH (Browser Reconnaissance and Exfiltration via Adaptive Compression of Hypertext) attacks exploit HTTP responses. To be vulnerable to this attack, a web application must [24, 36]:

- Be served from a server with HTTP-level compression.
- Reflect user-input in HTTP response bodies.
- Reflect a CSRF (Cross Site Request Forgery) token in HTTP response bodies.

The attack is agnostic to the version of TLS/SSL, and does not require TLS-layer compression. Additionally, the attack works against any cipher suite. Against a stream cipher, the attack is simpler (the difference in sizes across response bodies is much more granular) [24].

The following methods can be used as a defense against the BREACH attacks [24]:

- Disable the HTTP compression.
- Use another compression when working with the user.
- Changing the value of the token for each new request.
- Masking of CSRF tokens (use shifts, XOR with randomly generated key).
- Masking the length of the message (adding a random number of bytes to the HTTP response).
- Increase the time of the request/response.

D. The DROWN Attack

The DROWN attack stands for "Decrypting RSA using Obsolete and Weakened eNcryption". It allows an attacker to decrypt intercepted TLS connections by making specially crafted connections to an SSL 2.0 server that uses the same private key [25, 37].

The attack can be represented in this way: the attacker observes for several hundred connections between the victim client and Web server. Collecting these connections might involve intercepting traffic for a long time or tricking the user into visiting a website that quickly makes many connections to another site in the background. The connections can use any version of the SSL/TLS protocol, including TLS 1.2 with RSA key exchange method [25, 37].

Next, the attacker repeatedly connects to the SSL 2.0 server and sends special handshake messages with modifications to the RSA ciphertext from the victim's connections. The way the server responds to each of these probes depends on whether the modified ciphertext

decrypts to a plaintext message with the right form. The way, that the server responds, helps attacker to reveal secret keys and use them for the victim's TLS connections [25, 37].

So, for modern servers and users that are supporting the SSL 2.0 protocol, DROWN is real issue. The server can be vulnerable to DROWN attack for following reasons [25]:

- Using SSL 2.0 connections.
- Its private key is used on any other server that allows SSL 2.0 connections, even for another protocol.

To protect against DROWN, it need to be ensure that private keys are not used anywhere with server software, that allows SSL 2.0 connections: web servers, SMTP (Simple Mail Transport Protocol) servers, IMAP (Internet Message Access Protocol) and any other software that supports SSL/TLS [25].

E. Others Attacks

There are many more attacks that exploit TLS vulnerabilities: SLOTH, ROBOT, Lucky 13 [23] etc. We briefly review some of this attacks due to they expand possibilities of previous reviewed attacks or require special software on the victim machine (e.g., OpenSSL, NSS etc.) that do not full apply to The TLS protocol:

- The SLOTH attack. This attack use downgrade vulnerability of the TLS 1.2 protocol. Thus it can force clients/servers to downgrade to a weaker hash algorithm (like MD5), lowering the amount of computing power needed for a successful attack. There are two possible downgrades: when the TLS client accepts a weak hash algorithm from the server (field "SignatureAndHashAlgorithm" in message "ServerKeyExchange"); when server accepts a weak hash algorithm from the client (message "ClientCertificateVerify") [26].
- The ROBOT attack. It's attack that allows performing RSA decryption and signing operations with the private key of a TLS server. Unlike Bleichenbacher's original attack used an oracle based on different TLS alerts, the ROBOT attack use its own scanning methodology with various different signals to distinguish between error types like timeouts, connection resets, duplicate TLS alerts (e.g., different PKCS#1 v1.5 messages) [28, 38].
- The POODLE attack. This attack is also using downgrade vulnerability in the TLS 1.2 protocol (and older versions). The POODLE attack uses the feature that when a secure connection attempt fails, servers will fall back to older protocols such as SSL 3.0. Besides, the server must support SSL 3.0 with CBC-mode encrypt. Attacker can trigger a connection failure and then force the use of SSL 3.0 [23, 39].
- The Lucky 13 attacks. They use the vulnerability of CBC-mode encryption in TLS 1.2 (or older) with Datagram TLS (DTLS) protocol and incorporate padding oracle attack countermeasures. The attacks come in various plaintext recovery flavors. For the plaintext recovery attacks, no chosen-plaintext capability is needed, unlike the BEAST attack: a standard MITM attacker who sees only ciphertext and can inject ciphertexts of his own composition into the network can mount the attacks [29].
- The Heartbleed attack. This attack exploit critical vulnerability in the heartbeat extension of the cryptography library OpenSSL. Attacker takes advantage of the TLS heartbeat extension, which is primarily used as a keep-alive method between two parties to and ensure that the connection is not closed if they are both still there. If the client sent false data length, the server would respond with the data received by the "client + random data" from its memory to meet the length requirements specified by the sender. Using this feature attacker can extract the secret from server messages [40].

IV. COMPARISON OF ATTACKS ON THE TLS

For summarizing all information, we compare the reviewed attacks that is presented in Table 1. The following criteria were used to make clear view to understand attacks:

- The ability to run JS code in victim's browser. One of the most popular realization of network attacks on the TLS, and particularly the MITM type attack, is inject malicious JS code in victim browser [21, 22, 23]. Thus, it need to consider that feature.
- Security measures for the TLS protocol. It possible to use special security options or features in the actual versions of TLS to protect from the TLS attacks [20].
- Possible security options in the TLS 1.3. The newest version of the TLS might fix of previous versions vulnerabilities, so considering security features of the TLS 1.3 is necessary. Since the TLS 1.3 is represented in working draft [27], this security options is only potential.

For comparing we chose the four main reviewed attacks (BEAST, CRIME, BREACH, DROWN) and two additional (ROBOT and Lucky 13) due to the ROBOT attack allows to use different scanning methodology of obtaining secret information [28], and the Lucky 13 attack is represented by a family of attacks that apply CBC-mode encryption [29]. Thus, they provide to attacker a broad suite to exploit the TLS vulnerability.

In future works we are going to test the vulnerability of current version of the TLS protocol (1.2) against from considered attacks. For this purpose, a protection system will be developed as well as one of the attacks that is presented in the Table below will be implemented. The system will be able to detect the threat and use countermeasures to protect the target network host from this attack.

TABLE I. COMPARISON OF NETWORK ATTACKS ON THE TLS PROTOCOL

Criteria	Reviewed attacks on the TLS protocol					
	BEAST	***CRIME***	***BREACH***	***DROWN***	***ROBOT***	***Lucky 13***
Type of the attack	CBC attack	Compression attack	Compression attack	Padding oracle attack	Padding oracle attack	CBC attack
Exploited vulnerability	Using CBC-mode encryption with block cypher	HTTP request with Deflate compression	HTTP-level compression	Supporting SSL 2.0 with RSA key exchange	TLS cipher modes with RSA key exchange	TLS CBC-mode with MAC checking
Running JS code in victim's browser	Yes	Yes	Yes	Not necessary	Not necessary	When combining Lucky 13 with BEAST
Vulnerable version of the TLS protocol	SSL 3.0/TLS 1.0	TLS 1.0, TLS 1.1, TLS 1.2 with SPDY extension	TLS 1.0, TLS 1.1, TLS 1.2	TLS 1.0, TLS 1.1, TLS 1.2	TLS 1.0, TLS 1.1, TLS 1.2	TLS 1.0, TLS 1.1, TLS 1.2
Security measures for the TLS protocol	Use newer versions of TLS (1.1 and 1.2) and stream ciphers	Disable HTTP Deflate compression or prevent to use this compression on client side	Disable HTTP compression, use another compression, use masking methods or increase of message time	Disable supporting SSL 2.0 (if possible) or use the newest version of software (e.g., OpenSSL, NSS)	Disable RSA encryption suites, using ciphers with RSA signature is allowed	Add Random Time Delays, use RC4 and authenticated encryptions
Possible security options in the TLS 1.3 protocol	Removed CBC encryption modes and weak stream ciphers (e.g. RC4)	Removed compression and custom DHE (Diffie-Hellman Ephemeral) groups	Using record padding mechanism to avoid leaking message length during the Handshake	Removed static RSA handshake; it necessary not use an TLS 1.2 old certificate base	Removed static RSA handshake; PSK[a] with DHE key exchange mode are used	Removed CBC-MAC-Encrypt modes; GCM[b]-AES mode is used

a. Pre-shared key

b. Galois/Counter Mode

V. CONCLUSIONS

Despite the fact that TLS is one of the most studied protocols now, there are quite a lot of attacks that use various vulnerabilities [23], even the already obsolete ones like the ROBOT attack [28, 38]. Thus, it need to examine the TLS and find new ways to secure the protocol software.

Although there are a huge number of attacks, they can be classified, because they have similarities and some of them are the successors of old attacks [23, 28]. Based on reviewed stuff, all considered threats can be divided into attacks that using the following vulnerabilities:

- CBC-mode encryption (BEAST, Lucky 13).
- Compression of data (CRIME, BREACH).
- Data padding (DROWN, ROBOT, Heartbleed).
- Using downgrades (SLOTH, POODLE).

Therefore, to protect from all of this threats, for the TLS protocol we can use such security options:

- Using the newest version of special software (OpenSSL, NSS).
- Using special security options (e.g., data masking methods, changing the time of request/response, authenticated encryption).
- Disable unreliable protocols (e.g., RSA encryption suites, data compression protocols).

The TLS 1.3 represents a great potential for protection against all existing attacks. It will include the algorithms and ciphers suites that based on elliptic cryptography, which greatly complicates the conduct of attacks for intruders.

REFERENCES

[1] I.D. Gorbenko, V.I. Dolgov, V.I. Rublinetskii and K.V. Korovkin. "Methods of Information Protection in Communications Systems and Methods of Their Cryptoanalysis". *Telecommunications and Radio Engineering*, volume 52, issue 4, pages 89-96, 1998.

[2] N.I.Naumenko, Yu.V. Stasev and A.A. Kuznetsov. "Methods of synthesis of signals with prescribed properties". *Cybernetics and Systems Analysis*, volume 43, issue 3, pp. 321-326, May 2007. DOI: 10.1007/s10559-007-0052-8

[3] T. Lavrovska and S. Rassomahin."Physical model of pseudorandom codes in multidimensional Euclidean space." *2016 Third International Scientific-Practical Conference Problems of Infocommunications Science and Technology (PIC S&T)*, Kharkiv, 2016, pp. 67-70. DOI: 10.1109/INFOCOMMST.2016.7905337

[4] A.A. Kuznetsov, A.A. Smirnov, D.A. Danilenko and A. Berezovsky. "The statistical analysis of a network traffic for the intrusion detection and prevention systems" *Telecommunications and Radio Engineering*, volume 74, issue 1, pages 61-78, 2015. DOI: 10.1615/TelecomRadEng.v74.i1.60

[5] I.D. Gorbenko, A.A. Zamula and Ye.A. Semenko. "Ensemble and correlation properties of cryptographic signals for telecommunication system and network applications" *Telecommunications and Radio Engineering*, volume 75, issue 2, pp. 169-178, 2016.

[6] R. Oliynykov, I. Gorbenko, O. Kazymyrov, V. Ruzhentsev, O. Kuznetsov, Yu. Gorbenko, O. Dyrda, V. Dolgov, A. Pushkaryov, R. Mordvinov and D. Kaidalov. "Informatsiyni tekhnolohiyi. Kryptohrafichnyy zakhyst informatsiyi. Alhorytm symetrychnoho blokovoho peretvorennya [DSTU 7624:2014. National Standard of Ukraine. Information technologies. Cryptographic Data Security. Symmetric block transformation algorithm]." *DSTU 7624*:2014. (In Ukrainian).

[7] O. Kuznetsov, M. Lutsenko and D. Ivanenko. "Strumok stream cipher: Specification and basic properties" *2016 Third International Scientific-Practical Conference Problems of Infocommunications Science and Technology (PIC S&T)*, Kharkiv, 2016, pp. 59-62. DOI: 10.1109/INFOCOMMST.2016.7905335

[8] V.A. Krasnobayev, S.A. Koshman and M.A. Mavrina. "A Method for Increasing the Reliability of Verification of Data Represented in a Residue Number System". *Cybernetics and Systems Analysis*, volume 50, issue 6, pp 969–976, November 2014.

[9] R. Brumnik, V. Kovtun, A. Okhrimenko and S. Kavun. (2014). "Techniques for Performance Improvement of Integer Multiplication in Cryptographic Applications". *Mathematical Problems in Engineering*, vol. 2014, Article ID 863617, pp.1-7, 2014. DOI:10.1155/2014/863617

[10] V.M. Grachev, V.I. Esin, N.G. Polukhina and S.G. Rassomakhin. "Technology for developing databases of information systems" *Bulletin of the Lebedev Physics Institute*, vol.41(5), pp.119-122, May 2014.

[11] V.M. Grachev, V.I. Esin, N.G.Polukhina and S.G. Rassomakhin. "Data security mechanisms implemented in the database with universal model". *Bulletin of the Lebedev Physics Institute*., volume 41, issue 5, pp 123-126, May 2014.

[12] O. Trydid, S. Kavun and M. Goykhman. (2014). "Synthesis concept of information and analytical support for bank security system". *Actual Problems of Economics,* 11(161), pp.449-461. [Online] Available: http://eco-science.net/archive2014/336--11161.html [3 August 2017].

[13] I. Gorbenko and V. Ponomar. "Examining a possibility to use and the benefits of post-quantum algorithms dependent on the conditions of their application". *EasternEuropean Journal of Enterprise Technologies*, vol 2, no 9 (86), pp. 21-32, 2017. DOI: http://dx.doi.org/10.15587/1729-4061.2017.96321

[14] A. Andrushkevych, T. Kuznetsova, I. Bilozertsev and S. Bohucharskyi. "The block symmetric ciphers in the post-quantum period". *2016 Third International Scientific-Practical Conference Problems of Infocommunications Science and Technology (PIC S&T)*, Kharkiv, 2016, pp. 43-46. DOI: 10.1109/INFOCOMMST.2016.7905331

[15] O. Kuznetsov, Y. Gorbenko and I. Kolovanova. "Combinatorial properties of block symmetric ciphers key schedule". *2016 Third International Scientific-Practical Conference Problems of Infocommunications Science and Technology (PIC S&T),* Kharkiv, 2016, pp. 55-58. DOI: 10.1109/INFOCOMMST.2016.7905334

[16] A.V. Potii and A.K. Pesterev. "A System Approach to Certification of Pseudorandom Numbers Generators Used in Information Protection Systems". *Telecommunications and Radio Engineering,* volume 52, issue 4, pp. 97-102, 1998. DOI: 10.1615/TelecomRadEng.v52.i4.220

[17] "What Is Transport Layer Security (TLS)?" [Online]. Available: https://www.cloudflare.com/learning/security/glossary/transport-layer-security-tls/

[18] T. Dierks and C. Allen. The TLS Protocol Version 1.0. *RFC 2246, Internet Engineering Task Force*, 1999.

[19] T. Dierks and E. Rescorla. The Transport Layer Security (TLS) Protocol Version 1.1. *RFC 4346, Internet Engineering Task Force*, 2006.

[20] T. Dierks and E. Rescorla. The Transport Layer Security (TLS) Protocol Version 1.2. *RFC 5246, Internet Engineering Task Force,* 2008.

[21] T. Duong and J. Rizzo. *"Here come the ⊕ Ninjas".* Ekoparty (2011).

[22] T. Duong and J. Rizzo. *"The CRIME attack".* Ekoparty (2012).

[23] Y. Sheffer, R. Holz and P. Saint-Andre. Summarizing Known Attacks on Transport Layer Security (TLS) and Datagram TLS (DTLS) *RFC 7457, Internet Engineering Task Force*, 2015.

[24] Y. Gluck, N. Harris and Angelo Prado. *"BREACH: Reviving The CRIME Attack"*. Blackhat, USA (2013).

[25] Nimrod Aviram, Sebastian Schinzel, Juraj Somorovsky, Nadia Heninger, Maik Dankel, Jens Steube, Luke Valenta, David Adrian, J. Alex Halderman, Viktor Dukhovni, Emilia Käsper, Shaanan Cohney, Susanne Engels, Christof Paar and Yuval Shavitt. "DROWN: Breaking TLS using SSLv2". *Proceedings of the 25th USENIX Security Symposium*, August 2016.

[26] Karthikeyan Bhargavan and Gaetan Leurent. "Transcript Collision Attacks: Breaking Authentication in TLS, IKE, and SSH". *Network and Distributed System Security Symposium – NDSS 2016*, Feb 2016, San Diego, United States.

[27] E. Rescorla. The Transport Layer Security (TLS) Protocol Version 1.3, *Network Working Group Internet-Draft*, March 20, 2018.

[28] Hanno Bock , Juraj Somorovsky, and Craig Young. "Return Of Bleichenbacher's Oracle Threat (ROBOT)". *Cryptology ePrint Archive: Report 2017/1189*.

[29] Nadhem J. AlFardan and Kenneth G. Paterson. "Lucky Thirteen: Breaking the TLS and DTLS Record Protocols". *2013 IEEE Symposium on Security and Privacy.*

[30] Tim Polk, Terry McKay, and Santosh Chokhani. "Guidelines for the Selection, Configuration, and Use of Transport Layer Security (TLS) Implementations". *National Institute of Standards and Technology*. p. 67, April 2014.

[31] "TLS 1.2 vs TLS 1.1", July 27, 2016. [Online]. Available: https://www.keycdn.com/support/tls-1-2-vs-tls-1-1/

[32] Jay Thakkar. "TLS 1.3 Handshake: Taking a Closer Look", 2018. [Online]. Available: https://www.thesslstore.com/blog/tls-1-3-handshake-tls-1-2/

[33] "SSL/TLS Information Disclosure (BEAST) Vulnerability", 2016. [Online]. Available: https://docs.secureauth.com/pages/viewpage.action?pageId=14778519

[34] Asif Balasinor. "SSL/TLS attacks: Part 2 – CRIME Attack". [Online]. Available: http://niiconsulting.com/checkmate/2013/12/ssltls-attacks-part-2-crime-attack/

[35] "CRIME SSL/TLS attack". [Online]. Available: https://www.acunetix.com/vulnerabilities/web/crime-ssl-tls-attack

[36] [34] Asif Balasinor. "SSL/TLS attacks: Part 3 – BREACH Attack" [Online]. Available: http://niiconsulting.com/checkmate/2013/12/ssltls-attacks-part-3-breach-attack/

[37] "The DROWN Attack" [Online]. Available: https://drownattack.com

[38] "The ROBOT Attack". [Online]. Available: https://robotattack.org

[39] Bodo Möller, Thai Duong and Krzysztof Kotowicz. "This POODLE Bites: Exploiting The SSL 3.0 Fallback". Google. September 2014.

[40] Agathoklis Prodromou. "Features TLS/SSL Explained – Examples of a TLS Vulnerability and Attack, Final Part", 2017. [Online]. Available: https://www.acunetix.com/blog/articles/tls-vulnerabilities-attacks-final-part/