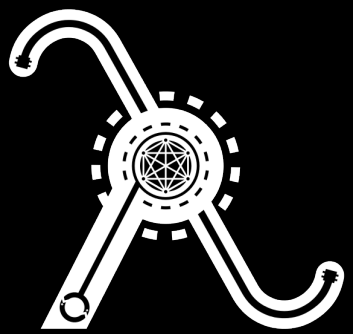
Introduction to and survey of TLS Security

Aaron Zauner
azet@azet.org



lambda.co.at:
Highly-Available, Scalable & Secure Distributed Systems

DevOps/Security Meetup Vienna - 19/5/2014

Table of Contents

Motivation

Background
- Information Security and Cryptography
- Transport Layer Security
- Extensions and Trust

History & Attacks

Mitigation

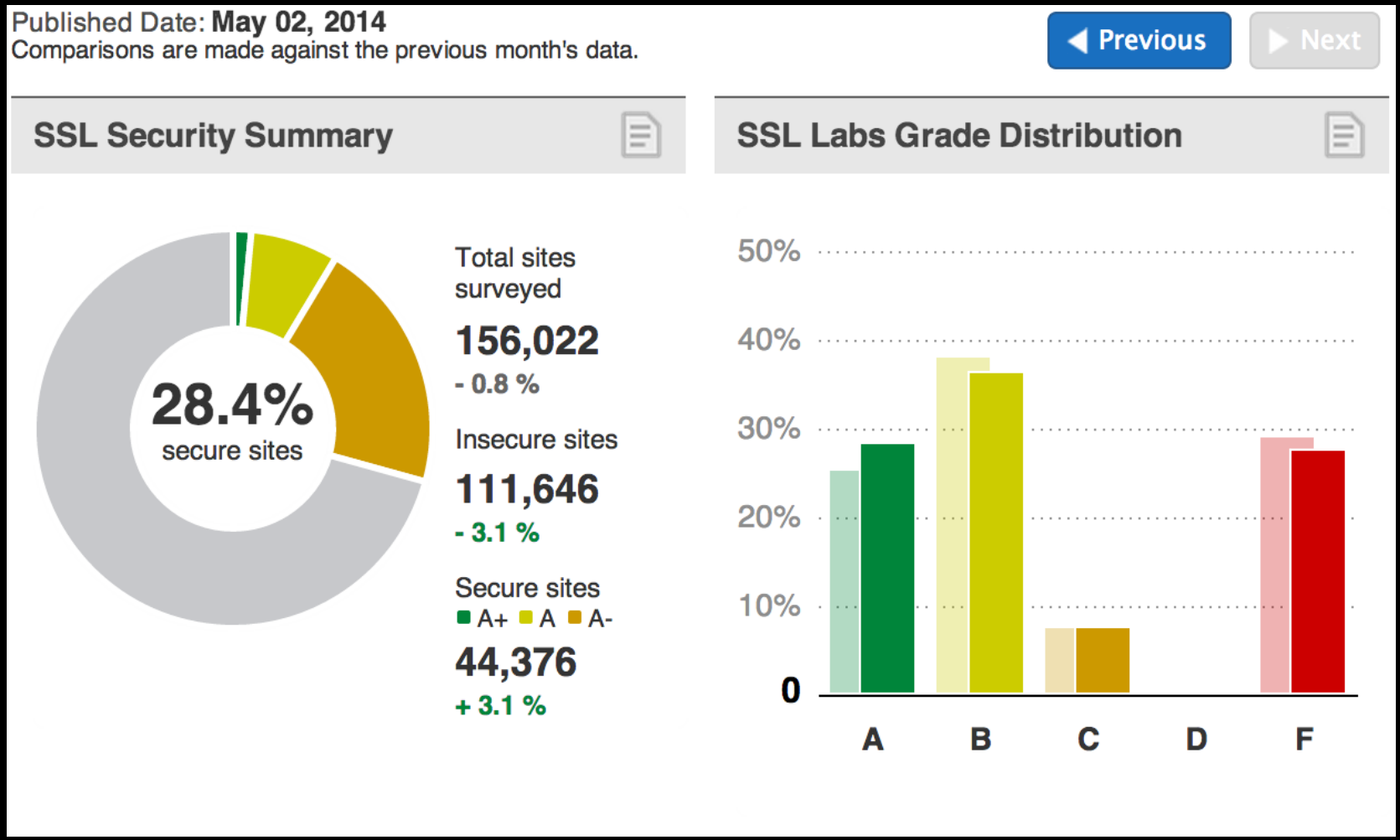
Conclusion

Motivation

TLS is something we deal with on a daily basis, so this is an obvious topic for DevOps education.

Keep in mind that these attacks are not only possible for nation-state actors, some of them I can mount on this very laptop.

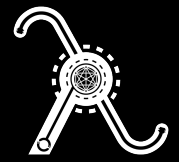
Motivation



https://www.trustworthyinternet.org/ssl-pulse

Background

Disclaimer

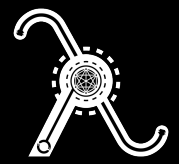


Unfortunately this is not a talk about InfoSec nor cryptography, so I will only cover the very basics needed to understand the topic properly. Background information on Information Security and Cryptography herein is fitted to TLS only.

I will recommend appropriate resources for those who want to gain a deeper understanding at the end of my talk.

Background

Basics



Information Security mandates at minimum the following three properties in a secure protocol:

- **C**onfidentiality
- **I**ntegrity
- **A**vailability (discussed later)

..this is commonly known as the “CIA triad” (seriously). You will see later on why these are paramount. The triad is usually extended by:

- Authenticity & Non-repudiation

Background

Confidentiallity

- Prevent unauthorized disclosure of information
- *Encryption* and *Decryption* of confidential data

Example: Rijndael cipher. Later renamed AES (Advanced Encryption Standard) after winning a NIST challenge by the same name.

- Symmetric 128, 192 or 256 bit **block cipher**
- Stick figure guide to AES: `http://www.moserware.com/2009/09/stick-figure-guide-to-advanced.html`

Background

Confidentiallity: Block cipher modes

Block ciphers operate on fixed size *blocks of data*.
Online communication is defined by *streams of data*.

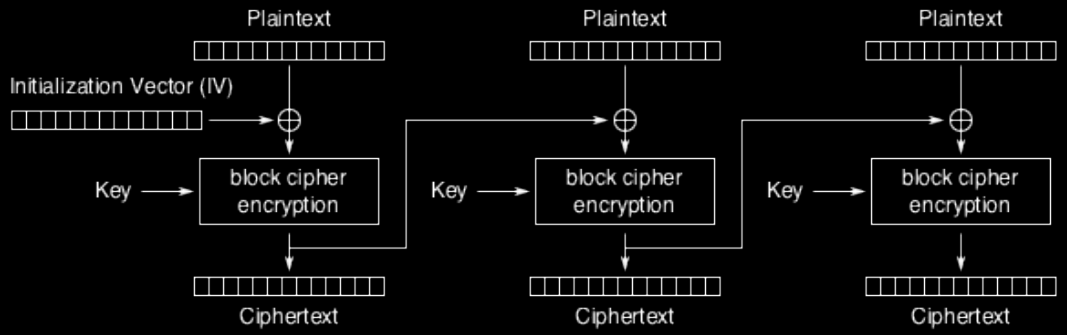
Block cipher modes repeatedly apply a given cipher to a stream of data. Examples include:

- Cipher-Block Chaining Mode (CBC)
- Counter Mode (CTR)
- Galois-Counter Mode (GCM) [**authenticated**]
- Counter with CBC-MAC (CCM) [**authenticated**]
- Offset Codebook Mode (OCB) [**authenticated**]

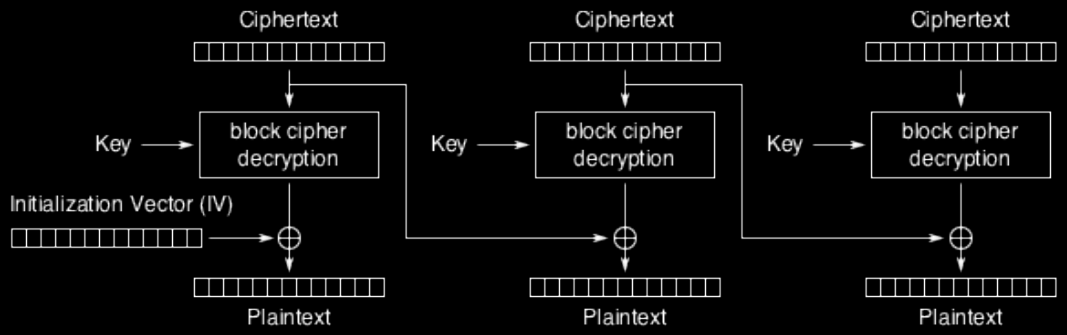
Background

Confidentiallity: Block cipher modes

- CBC Mode (sequential):



Cipher Block Chaining (CBC) mode encryption

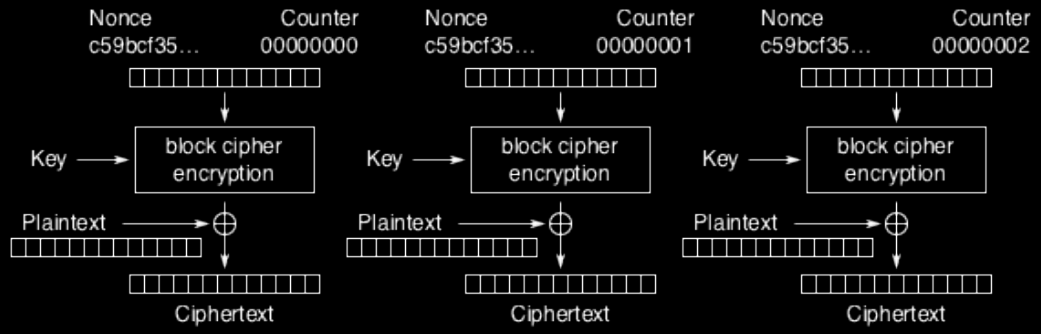


Cipher Block Chaining (CBC) mode decryption

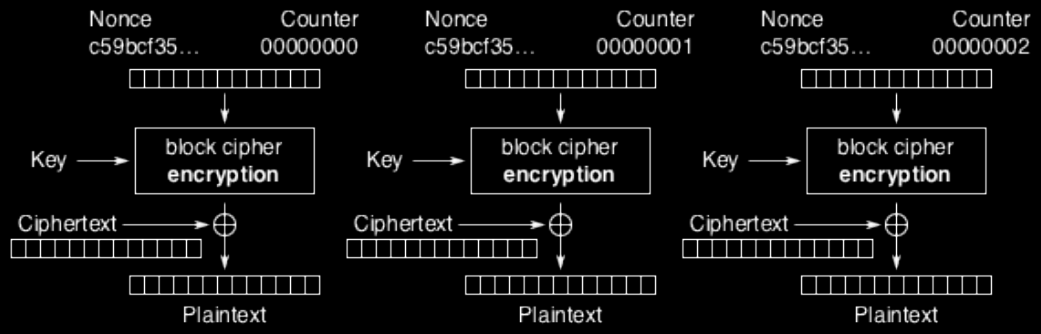
Background

Confidentiallity: Block cipher modes

- CTR Mode (parallel):



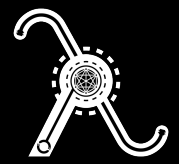
Counter (CTR) mode encryption



Counter (CTR) mode decryption

Background

Integrity



- Assures consistency of data
- e.g. tampered data will be detected (and may be discarded)
- Cryptographic hash functions like SHA can provide integrity by acting as a Hash based Message Authentication Code (HMAC) on the message to be sent and recieved
- Hash-functions need to be collission-resistant: Two different inputs should never produce the same hash!
- Ideally messages should be encrypted first then MACed (encrypt-then-mac, ETM) to prevent against attacks (CCA, CPA) and to provide for integrity of ciphertexts and plaintexts. See: `http://bit.ly/1kZA6WR`
- Authenticated Encryption with Associated Data (AEAD) can be used instead (e.g. GCM, CCM, OCB)

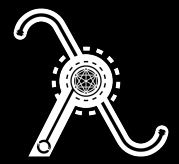
Background

Authenticity & Non-repudiation

- Assures that the involved parties are genuine
- i.e. are who they say they are
  - Injection attacks
  - Man-In-The-Middle (MITM) attacks
  - Various cryptographic attack vectors

Examples:

- RSA (Rivest, Shamir, Adleman)
- DSA (Digital Signature Algorithm)
- ECDSA (Elliptic Curve DSA)
- Ed25519 (`http://ed25519.cr.yp.to`)
- ElGamal Signature System

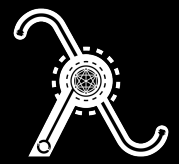


RSA

- Used for:
  - Signatures
  - Certificates
  - Authenticated Key-Exchanges (don't)
  - Encryption
- Security based on the difficulty of factoring integers:
  - p, q are large prime numbers (e.g. ≈ 1024bits)
  - $N = p \times q$
  - find factors of N
- Best known algorithm (~1994): General Number Field Sieve
- Computational complexity for n-bits: $O\left(\exp\left((\frac{64}{9}n)^{\frac{1}{3}}(\log n)^{\frac{2}{3}}\right)\right)$

Background

Authenticity & Non-repudiation: RSA



RSA Key Generation

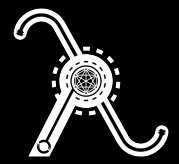
- integer N is the product of two large primes p and q
- $\phi(N) = (p-1)(q-1)$
- choose an integer e (usually *65537*), such that
  - $1 < e < \phi(N)$
  - $\gcd(e, \phi(N)) = 1$
- Public Key: N *(modulus)*, e *(public exponent)*
- Private Key: $d \equiv e^{-1} \pmod{\phi(N)}$

gcd and ϕ denote the *Greatest Common Divisor (Euclidian algorithm)* and *Euler totient function*, respectively.

Math background: *Modular arithmetic*.
Proof: *Euler's theorem* and *Fermat's little theorem*.

Background

Authenticity & Non-repudiation: RSA



Alice sends her public key (modulus N and exponent e) to Bob
Bob sends his public key (modulus N and exponent e) to Alice

RSA Encryption:

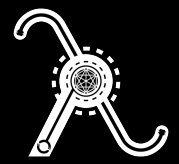
- Bob wants to send a message M to Alice, turns M into an integer m ($0 \leq m < N$) using a common padding-scheme
- ..computes ciphertext $c \equiv m^e \pmod{N}$

RSA Decryption:

- Bob sends ciphertext to Alice
- Alice computes $m \equiv c^d \pmod{N}$
- ..recoveres Bob's message M by reversing the common padding-scheme

Background

Authenticity & Non-repudiation: RSA padding



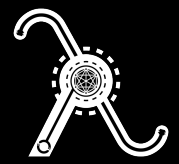
..a secure padding scheme is really important.

Example: Bleichenbacher attack on PKCS#1
`http://tinyurl.com/bleichenbacher` (real world SSLv3 attack)

Not going into that here, see: `http://tinyurl.com/rsa-padding`

Background

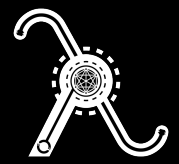
Key Agreement



A key exchange algorithm exchanges cryptographic keys (i.e. shared secrets) among parties which want to communicate confidentialy.

Background

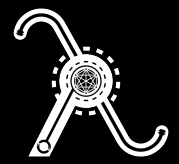
Key Agreement: Diffie-Hellman



The Diffie–Hellman key exchange algorithm (1976) was the first scheme devised to exchange cryptographic keys among multiple parties. It is the most widely used key exchange algorithm to this date.

Background

Key Agreement: Diffie-Hellman



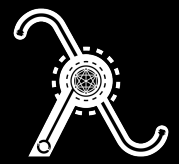
Alice and Bob want to communicate with each other and thus agree on a large prime number p and a generator g ($0 < g < p$)

- Alice choses a secret integer x (private key)
  ..and calculates $g^x \pmod{p}$ as her public key
- Bob choses a secret integer y (private key)
  ..and calculates $g^y \pmod{p}$ as his public key

Math Background: *Multiplicative group of integers modulo p*
`https://en.wikipedia.org/wiki/Multiplicative_group_of_integers_modulo_n`
`http://tinyurl.com/multiplicativegroupmodp`

Background

Key Agreement: Diffie-Hellman



Alice and Bob exchange their public keys

- Alice doesn't know Bob's y
- Bob doesn't know Alice's x

But,..

- Alice however knows x and g^y
  ..therefore calculates $(g^y)^x \pmod p = g^{yx} \pmod p$
- Bob however knows y and g^x
  ..therefore calculates $(g^x)^y \pmod p = g^{xy} \pmod p$

They now have established a *shared secret*: $g^{xy} \pmod p$

Background

Key Agreement: Forward secrecy

Forward secrecy (aka Perfect Forward Secrecy – PFS) ensures that only ephemeral session keys are used.

Even if a key is compromised in the future, not all communication that may have been recorded is compromised.

Simple, at the end of a session:

- Alice discards her public-key x
- Bob discards his public-key y

Hence: *Ephemeral* Diffie-Hellman (DH**E** and ECDH**E**).

Background

Ciphersuites

In SSL/TLS terminology; a *ciphersuite* combines the previously mentioned cryptographic techniques to work together and forms part of a secure (online) communication protocol

Background

Ciphersuites

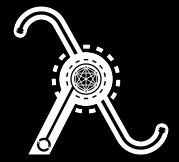
Example:

- Elliptic Curve Diffie-Hellman (Ephemeral - PFS)
- RSA
- AES128
- in Galois Counter Mode (GCM)
- SHA256

IANA standardized TLS parameters:
TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256

Background

Ciphersuites



OpenSSL:
ECDHE-RSA-AES128-GCM-SHA256

PolarSSL:
TLS-ECDHE-RSA-WITH-AES-128-GCM-SHA256

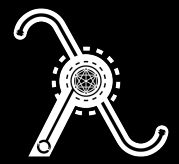
GnuTLS:
TLS_ECDHE_RSA_AES_128_GCM_SHA256

Apple Crypto Framework:
TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256

Microsoft SChannel:
TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256_P256
(No ECDHE with RSA and GCM for Microsoft users, you get crappy ECDSA instead.)

Background

Transport Layer Security



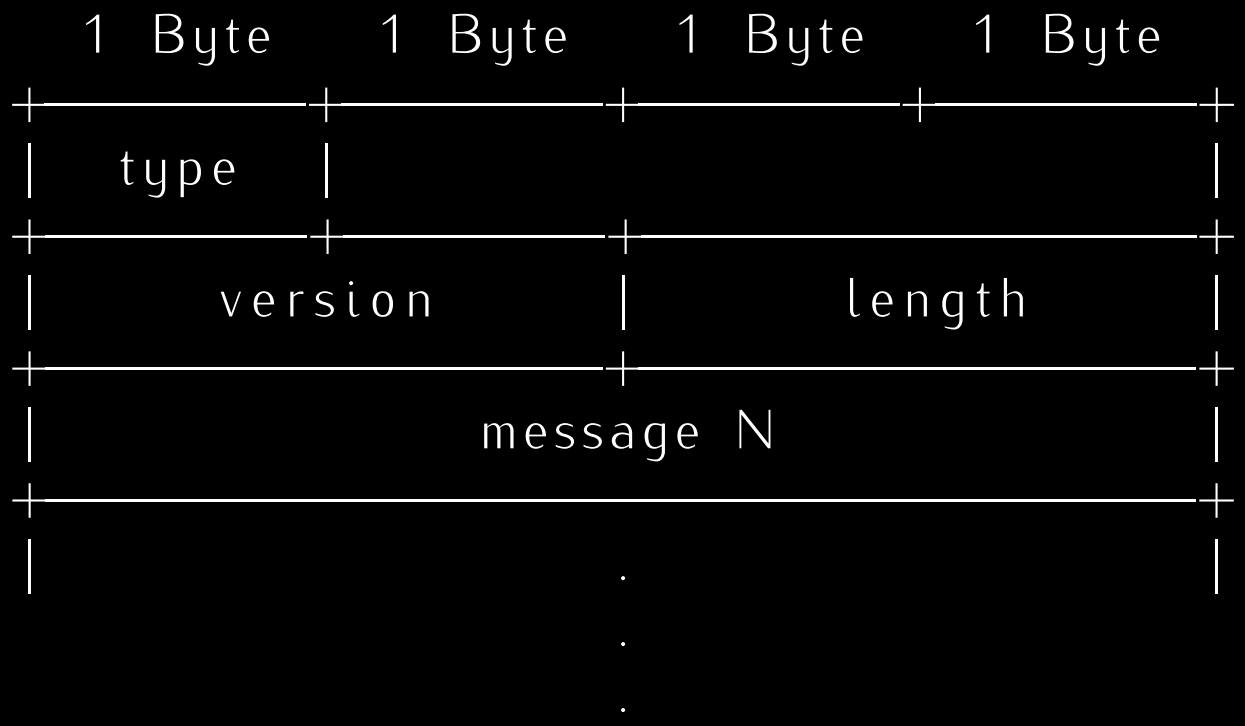
Before talking about the history and attacks of TLS it just makes sense to point out how TLS actually works (TLS 1.2).

Background

Transport Layer Security: TLS records

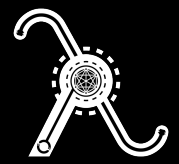
TLS deals in “records”. Different types of records exist: Handshake, ChangeCipherSpec, Application data and Alert.

In general TLS record looks like this:



Background

Transport Layer Security: TLS records



- Handshake records do have a different message structure
- Alert records basically just send an error code (1 byte) and description (1 byte) instead of a full message.

Besides Handshake and ChangeCipherSpec records – any records may optionally contain a MAC and padding (up to 4 bytes each) at the end, depending on the previously negotiated ciphersuite.

For reference, see: `https://en.wikipedia.org/wiki/Transport_Layer_Security#TLS_record`

Background

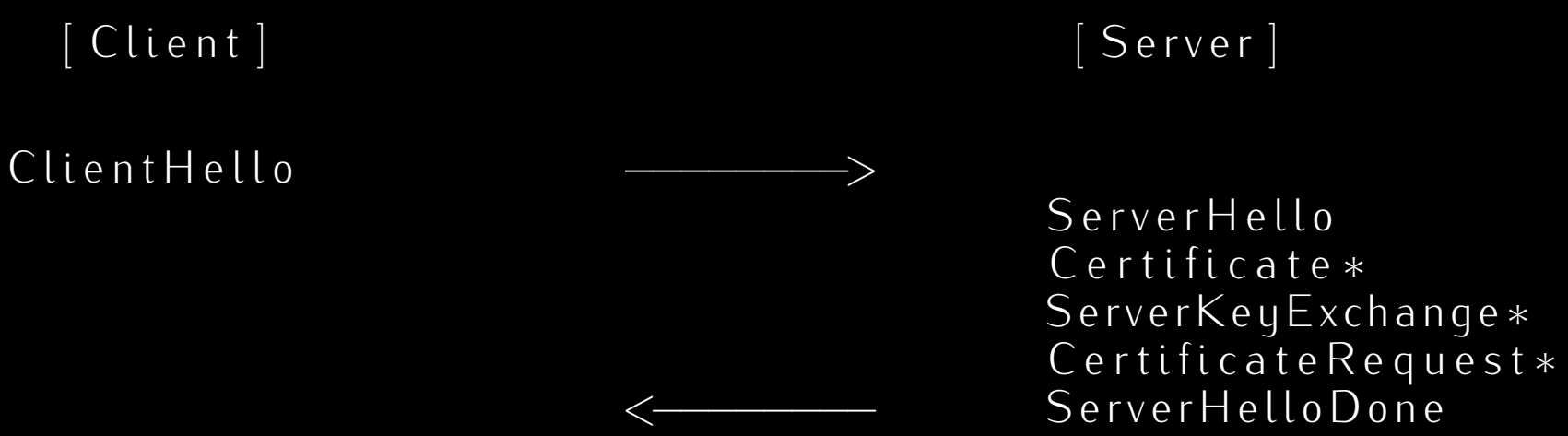
Transport Layer Security: Handshake

[Client] [Server]

ClientHello ——————>

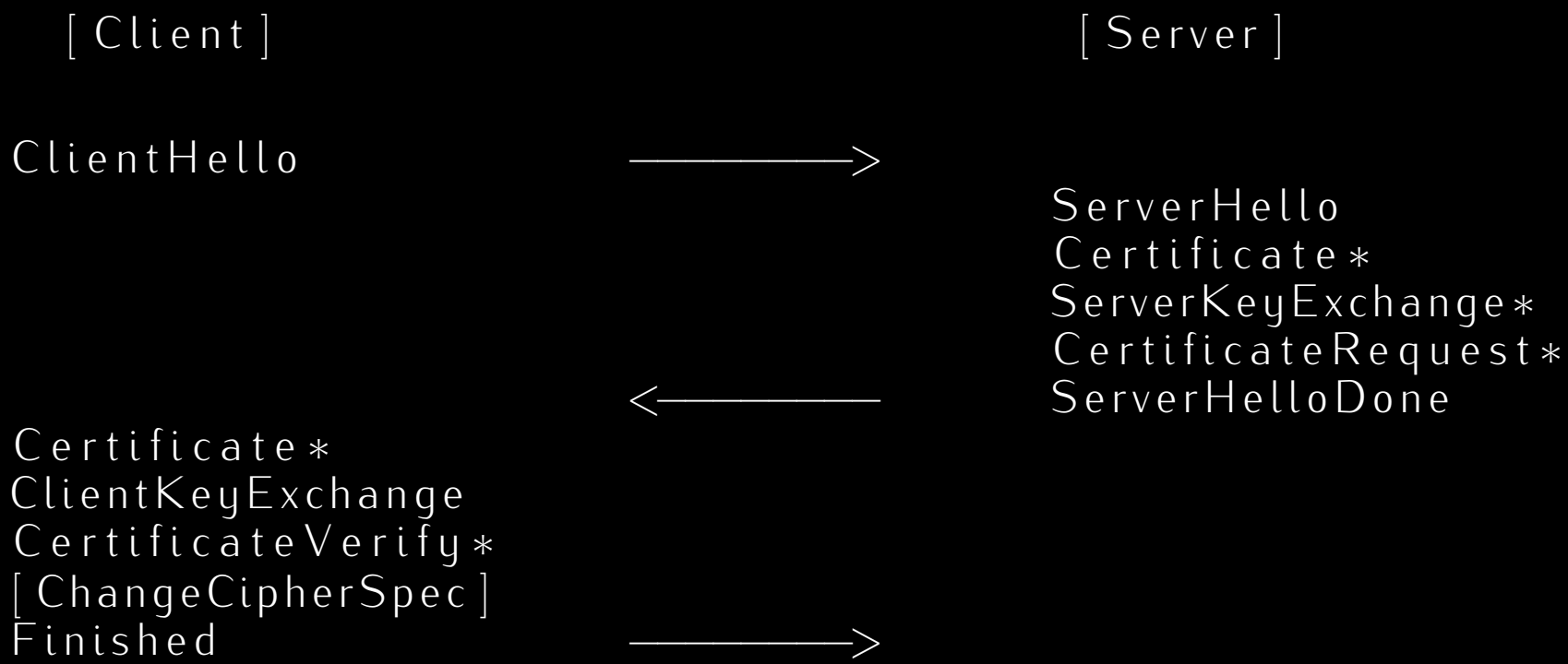
Background

Transport Layer Security: Handshake



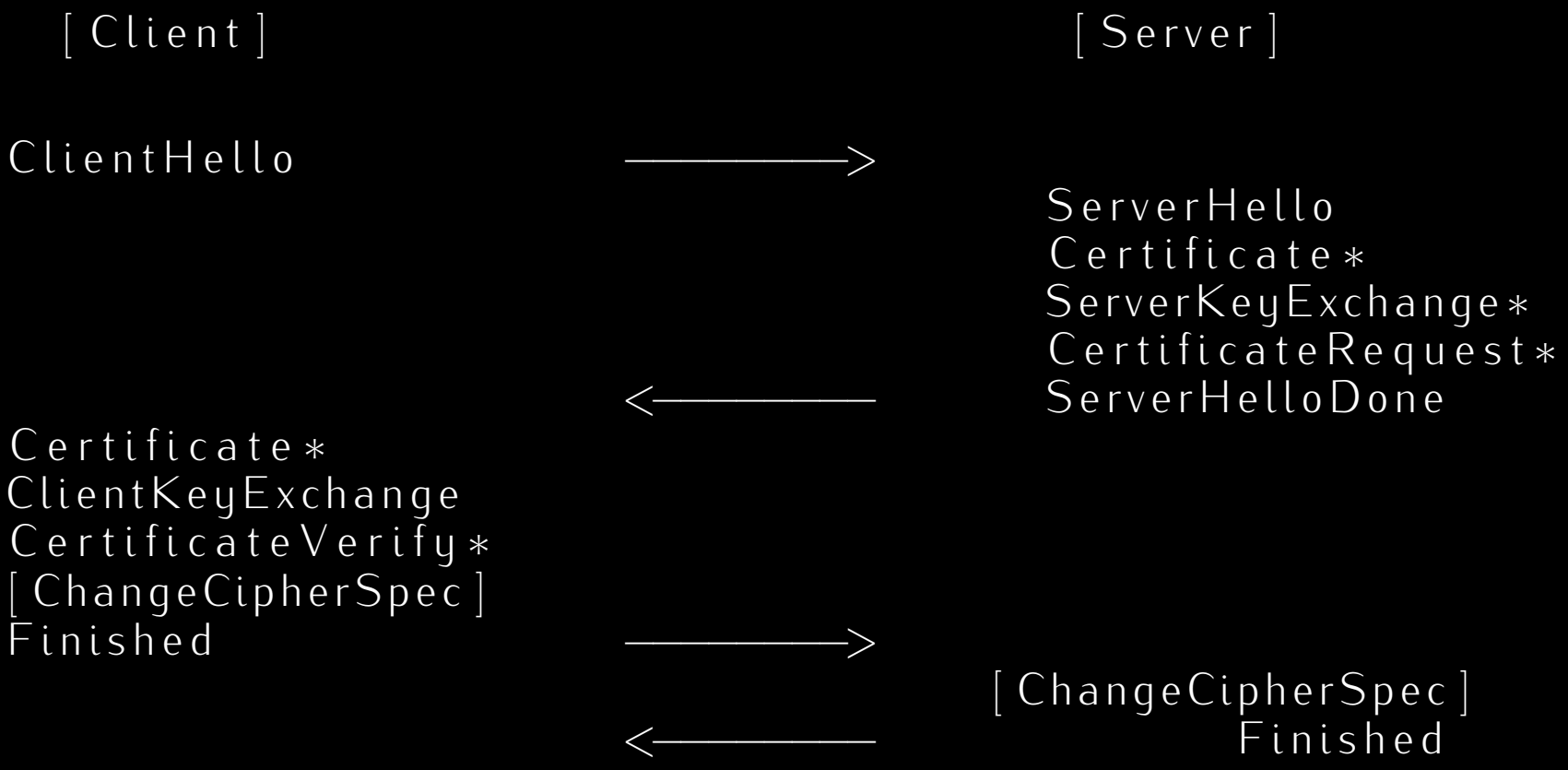
Background

Transport Layer Security: Handshake



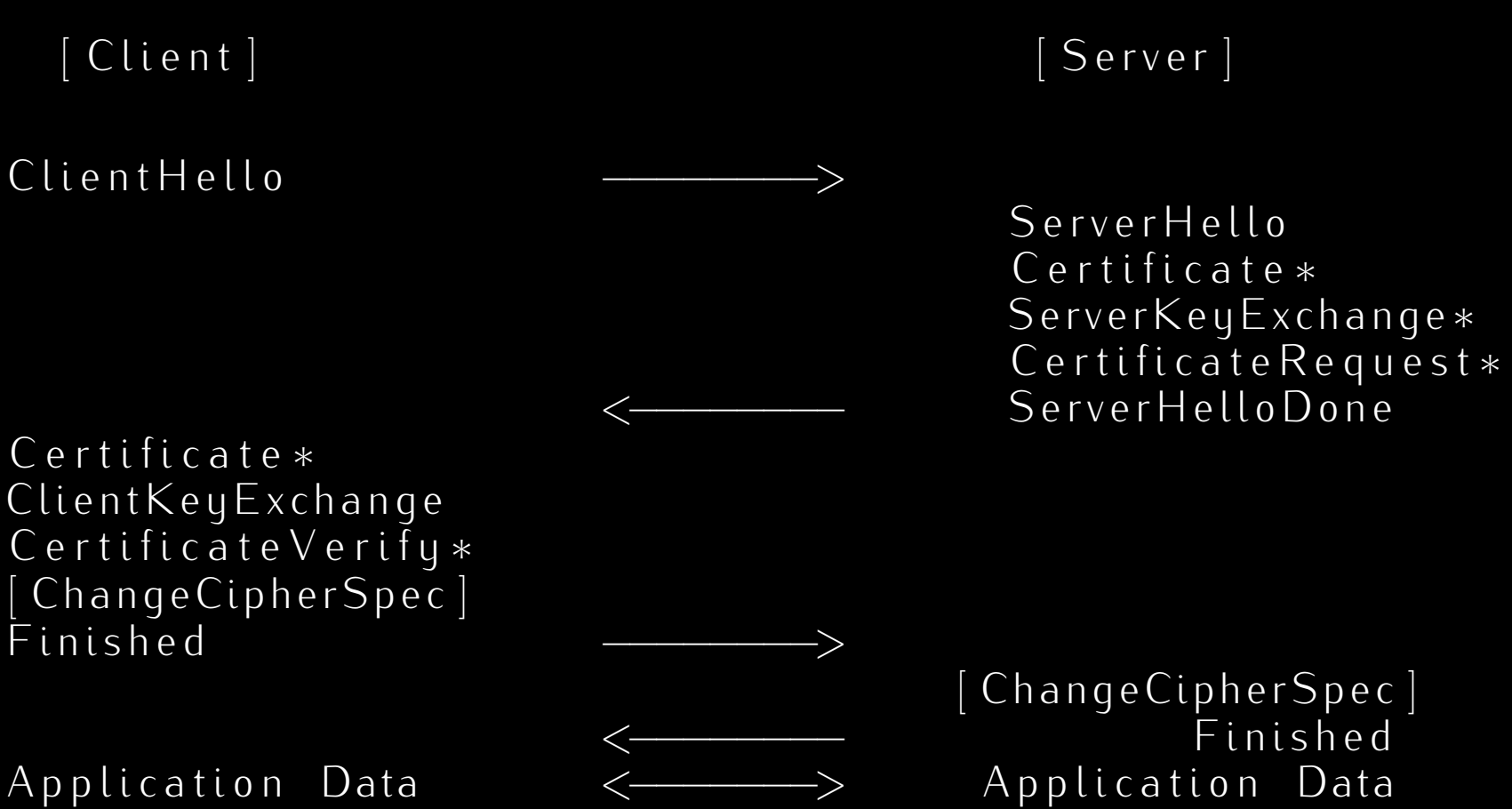
Background

Transport Layer Security: Handshake



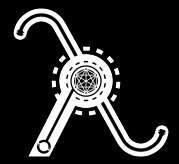
Background

Transport Layer Security: Handshake



Background

Session Resumption and False-start



Session-resumption and False-start are TLS extensions minimizing round-trip time and CPU cost.

Session-resumption:

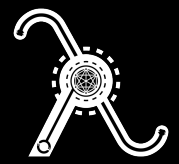
- An “abbreviated handshake” is used
- Previously negotiated Handshake parameters are reused

False-start (somewhat deprecated):

- Optional extension to send data before the Handshake is completed
- After ChangeCipherSpec and Finished messages Client or Server data may be sent
- Even if the other side has not acknowledged yet

For a good description see: `http://chimera.labs.oreilly.com/books/1230000000545/ch04.html`
`http://blog.cryptographyengineering.com/2012/04/so-long-false-start-we-hardly-knew-ya.html`

Background
OCSP



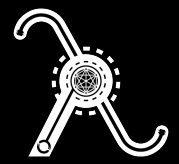
Online Certificate Status Protocol (OCSP) is a mechanism to check for the validity and revocation of certificates.

OCSP has recieved a lot of critique:

- MITM attackers may also interfere with OCSP requests
- OCSP stapling can be used to mitigate this problem
- OCSP latency for large CAs is usually in the hundreds of milliseconds
- OCSP infrastructure completely broke down during Heartbleed

Background

HTTP Strict Transport Security



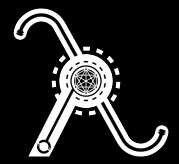
HTTP Strict Transport Security (HSTS) is a security policy that enforces HTTPS connections on following requests (e.g. upgrade every HTTP requests to HTTPS).

A small downside: the first HSTS header must be sent over HTTPS to ensure it cannot be tampered with.

Effectively disables ssl-stripping attacks.

Background

TLS Renegotiation Indication Extension

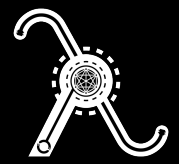


RFC5746 defines a TLS extension that prevents for TLS Handshake renegotiaton attacks by sending a special Signaling Cipher Suite Value (SCSV) in the ClientHello which ties a renegotiation request to a TLS connection.

Called “Secure Renegotiation”.

Background

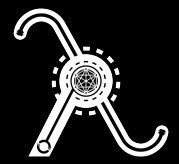
SPDY, NPN, ALPN and so forth



Google drafted specifications for protocol upgrade to SPDY including NPN and ALPN which SPDY effectively relies on. SPDY is the basis for the work in the IETF HTTPBIS-WG that will standardize HTTP2.

Background

Web-of-Trust and X.509



X.509, and Web-of-(mis)trust and ASN.1 would require a two hour talk on their own.

Certificate Authorities should and can not be trusted, they are known to behave maliciously at times, give away sub-CAs for DPI to large companies and nations and regularly fuck up their own security processes.

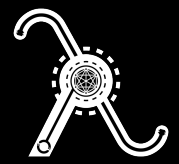
Read for example:
`http://www.certificate-transparency.org/what-is-ct`

History & Attacks

Now that you have an idea of the necessary background, let's take a look at the history of TLS (in)security.

History & Attacks

Internet Dark Ages



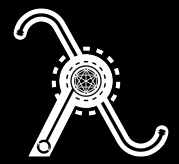
- SSLv1 engineered at Netscape, never released to the public
- Kipp Hickman of Netscape introduces SSLv2 as an IETF draft back in 1995:

> The SSL Protocol is designed to provide privacy between two communicating applications (a client and a server). Second, the protocol is designed to authenticate the server, and optionally the client. [...]

http://tools.ietf.org/html/draft-hickman-netscape-ssl-00

History & Attacks

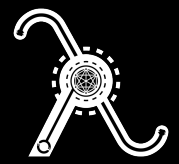
Internet Dark Ages



- SSLv2 was fundamentally broken and badly designed. Basically full loss of Confidentiallity and integrity of on-wire data thus susceptible to MITM attacks, see: `http://osvdb.org/56387`
- CipherSpec is sent in the clear
- Size of Block-cipher padding is sent in the clear

History & Attacks

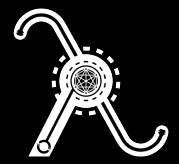
Internet Dark Ages



- SSLv3 was introduced in 1996 by Paul Kocher, Phil Karlton and Alan Freier, utilizing an algoritm by Taher ElGamal, a known cryptographer and Chief Scientist at Netscape at the time: `https://tools.ietf.org/html/rfc6101`

History & Attacks

Internet Dark Ages



On a side note; back then the choice algorithms was limited and export ciphers (low security) common as recommended by NSA and mandated by US law. Google: “Bernstein vs. United States”

- encryption algorithms (Confidentiality): NULL, FORTEZZA-CBC (NSA), IDEA-CBC, RC2-CBC-40 (40bit security), RC4-128, DES40-CBC (40bit security), DES-CBC (56bit security), Triple-DES-EDE-CBC
- hash functions (integrity): NULL, MD5 and SHA

History & Attacks

Internet Dark Ages

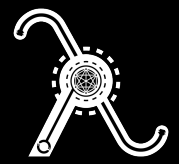
David Wagner and Bruce Schneier publish a paper entitled "Analysis of the SSL 3.0 protocol":

- Keyexchange algorithm rollback
- Protocol fallback to SSLv2
- Protocol leaks known plaintexts – may be used in cryptanalysis
- Replay attacks on Anonymous DH (don't use it anyway!)

`https://www.schneier.com/paper-ssl.pdf`

History & Attacks

TLS appears



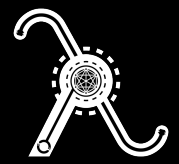
1999. The SSL protocol is renamed to TLS (version 1) with little improvements over SSLv3. The spec. is almost identical.

- Diffie-Hellman, DSS and Triple-DES are now required by implementors
- most SSLv3 security issues are still present in TLS 1.0

(RFC2246)

History & Attacks

TLS gets padding attacks



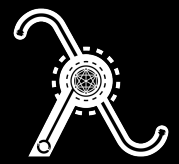
2002. Vaudenay publishes a paper entitled “Security Flaws Induced by CBC Padding Applications to SSL, IPSEC, WTLS...”

- Side-channel attack on CBC mode padding
- valid/invalid padding causes different reactions
- can be used to influence decryption operations
- introduces “padding oracle attacks” in SSL

`http://www.iacr.org/cryptodb/archive/2002/EUROCRYPT/2850/2850.pdf`

History & Attacks

TLS gets extended

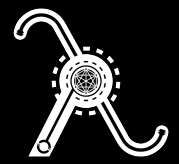


2003. TLS extensions get specified in RFC3546.

- General: Extended Handshake, ClientHello and ServerHello
- Server Name Indication (SNI) for virtual hosting
  (SNI leaks metadata!)
- Certificate Status Request (CSR) support via OCSP
- (...)

History & Attacks

TLS gets timing attacks



2003. Brumley and Boneh publish a paper entitled “Remote timing attacks are practical”.

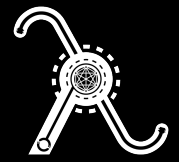
Timing attack on RSA in SSL/TLS implementations (OpenSSL):

- Send specially crafted ClientKeyExchange message
- Mesure time between ClienyKeyExchange and Alert response
- do a bit of statistics
- retrieve Private Key

`http://dl.acm.org/citation.cfm?id=1251354`

History & Attacks

TLS gets padding oracle password retrieval



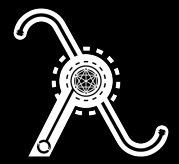
2003. Canvel, Hiltgen, Vaudenay, Vuagnoux publish “Password Interception in a SSL/TLS Channel”.

Extend earlier work of Vaudenay and successfully intercept IMAP passwords in TLS channels.

`http://www.iacr.org/cryptodb/archive/2003/CRYPTO/1069/1069.pdf`

History & Attacks

TLS gets chosen plaintext attacks



2004 & 2006. Bard demonstrates Chosen-Plaintext Attacks against SSL and TLS1.0

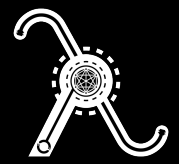
Attack on CBC:

- CBC exchanges an Initialization Vector (IV) during Handshake
- these IVs turn out to be predictable
- PINs and Passwords can be decrypted
- VPNs/Proxies can also be used to accomplish this task

`https://eprint.iacr.org/2004/111`
`https://eprint.iacr.org/2006/136`

History & Attacks

TLS gets updated



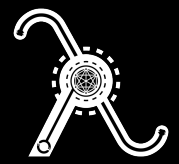
2006. A new TLS protocol version is standardized: TLS 1.1

- EXPORT ciphers removed
- Session resumption
- Protection against the CBC attacks by Bard
- IANA TLS parameters standardized
- (...)

(RFC4346)

History & Attacks

TLS gets modern crypto



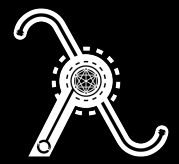
2008. A new TLS protocol version is standardized: TLS 1.2

- MD5/SHA1 removed as pseudorandom function (PRF)
- configurable PRFs in ciphersuites (e.g. SHA256)
- Authenticated encryption: CCM, GCM
- AES ciphersuites
- (...)

(RFC5246)

History & Attacks

Rouge CA Certificates



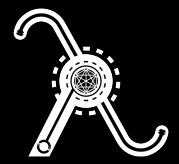
2008. Sotirov, Stevens, Appelbaum, Lenstra, Molnar, Osvik and de Weger present a paper based on earlier work by Lenstra et al. at 25c3 entitled “MD5 considered harmful today”

- MD5 Hash-collision of a CA Certificate
- Create colliding (rouge) CA Certificates
- Generate any Certificate for MITM you want

`http://www.win.tue.nl/hashclash/rogue-ca/`
`https://www.youtube.com/watch?v=PQcWyDgGUVg`

History & Attacks

sslstrip



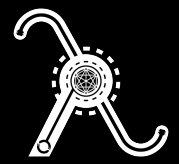
2009. Moxie Marlinspike releases *sslstrip* at BlackHat DC 2009.

- Client connects to server
- Attacker intercepts session via MITM
- Attacker sends HTTP 301 (moved permanently)
- Attacker forwards requests to/from server via SSL/TLS
- Client receives data via unencrypted channel
- Attacker reads plaintext

`http://www.thoughtcrime.org/software/sslstrip`
`http://vimeo.com/50018478`

History & Attacks

Null-prefix attacks against Certificates



2009. Moxie Marlinspike publishes “Null prefix Attacks against SSL/TLS Certificates”.

- Specially crafted domain strings trick CA checking
- null-terminate stuff in a domain name
- ex.: `www.paypal.com\0.thoughtcrime.org` is valid
- ex.: `*\0.thoughtcrime.org` is valid
- CA ignores prefix
- Client does not -> Certificate valid for prefix

Moxie updated his *sslsniff* project to carry out this attack.

`http://www.thoughtcrime.org/papers/null-prefix-attacks.pdf`
`http://thoughtcrime.org/software/sslsniff`

History & Attacks

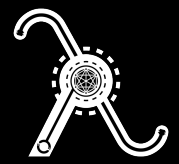
SSLv2 Forbidden

2011. IETF publishes and standardized a RFC to prohibit negotiation and thus compatibility of SSLv2 in TLS1.0–1.2 entirely.

`https://tools.ietf.org/html/rfc6176`

History & Attacks

Comodo



2011. Comodo CA: Attacker issues 9 certificates via reseller account for popular domains (google.com, yahoo.com, live.com, skype.com [...])

`https://www.comodo.com/Comodo-Fraud-Incident-2011-03-23.html`

History & Attacks
BEAST

2011. Doung and Rizzo publish the BEAST attack at ekoparty and demo a live attack on PayPal. Based on Bards earlier work on predictable IVs in CBC:

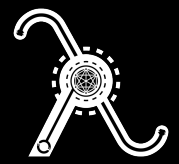
- Phishing gets victim to visit a certain website
- Script on said website makes request to genuine site
- Attacker records encrypted cookie information
- Tries to guess session-cookie with known CBC attack

Same Origin Policy (SOP) forbids this attack in client software. If SOP can be bypassed (as shown by the authors with Java's SOP) this attack is still practical.

`http://vnhacker.blogspot.co.at/2011/09/beast.html`

History & Attacks

Trustwave



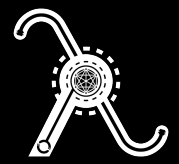
2012. Trustwave CA: Trustwave sells subordinate CAs to big corporations to be used for Deep Packet Inspection.

A sub-CA can issue and fake any certificate for MITM attacks.

http://blog.spiderlabs.com/2012/02/clarifying-the-trustwave-ca-policy-update.html
http://arstechnica.com/business/2012/02/critics-slam-ssl-authority-for-minting-cert-used-to-impersonate-sites/

History & Attacks

DigiNotar



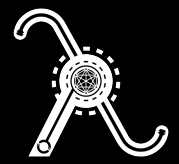
2012. DigiNotar CA: Attackers compromise DigiNotar in it's entirety.

- attackers generate tons of certificates
- Google Chromes certificate store detects mismatches
- DigiNotar acknowledges breach
- DigiNotar files for bankrupcy
- FOX-IT never gets paid for the investigation

`https://en.wikipedia.org/wiki/DigiNotar`
`http://cryptome.org/0005/diginotar-insec.pdf`
`http://nakedsecurity.sophos.com/2011/09/05/`
`operation-black-tulip-fox-its-report-on-the-diginotar-breach`

History & Attacks

Certificate validation in non-browser software



2012. Georgiev, Iyengar, Jana, Anubhai, Boneh and Shmatikov publish a paper entitled “The most dangerous code in the world: validating SSL certificates in non-browser software”

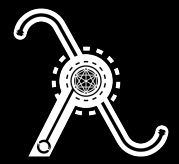
Certificate validation vulnerabilities in:

- OpenSSL
- GnuTLS
- JSSE
- EC2 Java libraries & Amazon SDKs
- PayPal SDKs
- eCommerce/WebShop software
- ..cURL, PHP, Python, tons of Java middleware

`https://crypto.stanford.edu/~dabo/pubs/abstracts/ssl-client-bugs.html`

History & Attacks

CRIME



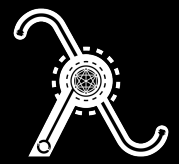
2012. Doung and Rizzo publish an attack against TLS Compression and SPDY titled CRIME.

- MITM attacker sees length of compressed ciphertext
- compression has direct affect on the length
- attacker makes client compress/encrypt data (or uses known data) with secret data
- attacker compares
- correct guesses yield shorter messages due to compression
- repeat until done

This is only feasible for small amounts of data, e.g. session strings, cookies and so forth.

`https://isecpartners.com/blog/2012/september/details-on-the-crime-attack.aspx`

History & Attacks
TIME



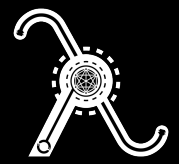
2013. Be'ery and Shulman present TIME at BlackHat Europe.
Extend on the CRIME Attack:

- Attacker generates HTTP requests (XSS, injection,..)
- Attacker exploits SOP design flaw and measures RTT differences
- determines correct or failed guesses by SOP timing leak

`https://media.blackhat.com/eu-13/briefings/Beery/bh-eu-13-a-perfect-crime-beery-wp.pdf`
`https://www.youtube.com/watch?v=rTIpFfTp3-w`

History & Attacks

Lucky13



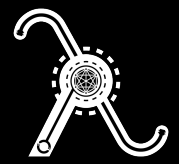
2013. AlFardan and Paterson present a novel attack against CBC for TLS and DTLS based on timing analysis.

- Attacker intercepts and modifies a message including padding
- Attacker tempers with the padding of the message
- MAC computation takes longer during decryption process
- Attacker repeats and measures
- Attacker performs padding oracle attack described earlier
- (Extremely latency sensitive attack)

`http://www.isg.rhul.ac.uk/tls/Lucky13.html`
`http://www.isg.rhul.ac.uk/tls/TLStiming.pdf`

History & Attacks

RC4 Biases



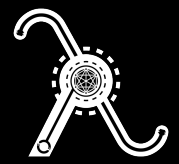
2013. AlFardan, Bernstein, Paterson, Poettering and Schuldt publish a generic attack on the RC4 cipher for TLS and WPA.

- Statistical biases in the first 257 bytes of ciphertext
- Recovery of the first 200 bytes after 2^{28} to 2^{32} encryption operations of the same plaintext
- A broadcast attack: mounted on unique keys
- May also be mounted with a single key with repeating target plaintexts
- Only feasible for large amounts of data and very time consuming

`http://www.isg.rhul.ac.uk/tls`
`http://www.isg.rhul.ac.uk/tls/RC4biases.pdf`

History & Attacks

NIST curves

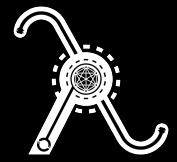


2013 & 2014. Daniel J. Bernstein and Tanja Lange voice concern about the NIST Elliptic Cuves that are widely implemented and used in TLS for ECDH and ECDSA

- NIST curves defined on recommendations by NSA's Jerry Solinas
- Unclear why these curves and their parameters were chosen
- NIST cites efficiency: more efficient and secure curves available
- Possible mathematical backdoor through previous analysis and carefully chosen and unexplained parameters
- Start SafeCurves project (ongoing)

`http://www.hyperelliptic.org/tanja/vortraege/20130531.pdf`
`http://cr.yp.to/talks/2013.09.16/slides-djb-20130916-a4.pdf`
`http://safecurves.cr.yp.to`
`https://archive.org/details/ShmooCon2014_SafeCurves`

History & Attacks
BREACH



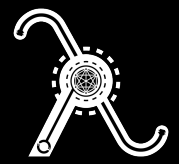
2013. Gluck, Harris and Prado demonstrate yet another attack based on CRIME at BlackHat USA.

Very similar to CRIME but the attack works based on information leaks from HTTP compression instead of TLS compression.

`http://breachattack.com`
`https://www.youtube.com/watch?v=CoNKarq1IYA`

History & Attacks

Unused Certificates in Truststores



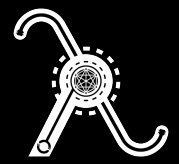
2014. Perl, Fahl, Smith publish a paper entitled "You Won't Be Needing These Any More: On Removing Unused Certificates From Trust Stores"

- Compared 48 mio. HTTP certificates
- 140 CA Certificates are unused in all major trust stores
- Of 426 trusted root certificates only 66% are even used

`http://fc14.ifca.ai/papers/fc14_submission_100.pdf`

History & Attacks

Triple Handshakes Considered Harmful



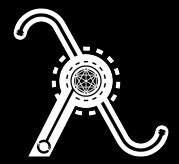
2014. Bhargavan, Delignat-Lavaud, Pironti, Langley and Ray present an attack one day before the IETF'89 meeting in London.

- Limited to client-certificate authentication with renegotiation
- MITM attack on renegotiation with a three-way handshake
- Variations of the attack also discussed on their website
- Can't possibly fit this into one slide, homework: understand the attack by reading their excellent description on the website

`https://secure-resumption.com`
`https://secure-resumption.com/IETF-triple-handshakes.pdf`

History & Attacks

Frankencerts



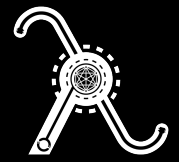
2014. Brubaker, Jana, Ray, Khurshid and Shmatikovy publish a paper entitled “Using Frankencerts for Automated Adversarial Testing of Certificate Validation in SSL/TLS Implementations”

- Fuzzing of X.509 related code in all major implementations shows serious weaknesses in certificate validation and handling
- OpenSSL, NSS, GnuTLS, MatrixSSL, PolarSSL, CyaSSL, cyptlib [...]

`https://www.cs.utexas.edu/~shmat/shmat_oak14.pdf`
`https://github.com/sumanj/frankencert`

History & Attacks

Heartbleed



2014. Heartbleed is independently discovered by Codenomicon and a Google Security engineer.

Faulty implementation in OpenSSL of the TLS Heartbleed extension leaks memory content over the wire. This has been all over the media and discussed in detail all over the internet. People have successfully extracted sensitive information (password files et cetera) from victim memory.

I wrote an nmap plugin to scan for Heartbleed:
`https://github.com/azet/nmap-heartbleed`

`http://heartbleed.com`

Mitigation

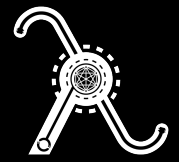
Get informed!

Luckily Mitigation does not require as much slides. Because it's rather simple:

- Use current software and update regularly:
  Most of these attacks are fixed upstream
- Use peer-reviewed and solid configurations:
  Check out `https://bettercrypto.org`
- Listen to recommendations by security experts
- Audit your infrastructure (possibly even pay external contractors to take a look)
- Keep track of TLS security

Mitigation

Future work



IETF TLS-WG is currently working on TLS 1.3 to mitigate some of the issues and concerns raised.

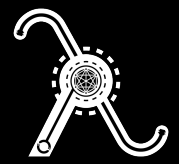
Drafts like new SCSVs would effectively prevent protocol downgrade attacks. (constant-time) ciphers and curves and DH parameter negotiation have been proposed along with protocol improvements for efficiency, security and privacy. It's still in the making and worth to keep track of.

Cryptography libraries seem to be more actively audited now, which has resulted in some of the disclosures in the paths months.

`http://www.ietf.org/proceedings/89/slides/slides-89-tls-5.pdf`
`https://tools.ietf.org/html/draft-bmoeller-tls-downgrade-scsv`
`http://tools.ietf.org/html/draft-agl-tls-chacha20poly1305`
`https://datatracker.ietf.org/doc/draft-gillmor-tls-negotiated-dl-dhe/?include_text=1`

Mitigation

Distribute Trust



Certificate Transparency, TACK and HTKP are only a few examples of proposed solutions to tackle the issue of malicious certificate authorities, certificate forgery and MITM attacks.

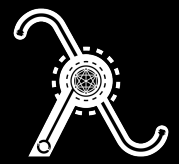
- Certificate Transparency: distributed consent, monitoring and auditability of certificates and CAs in the wild
- TACK and HTKP pin keys to certificates to prevent MITM and certificate based attacks

`http://www.certificate-transparency.org`
`http://tack.io`
`https://datatracker.ietf.org/doc/draft-ietf-websec-key-pinning/`

Conclusion

- “Trust the Math” (Schneier)
- Implementation is still a big issue
- Software bugs are a big issue
- Protocol design is hard and longsome
- We’re going to see many more attacks on TLS
- TLS and Crypto improvements are being constantly worked on

Conclusion



Thanks for your patience. Are there any questions?

Twitter:

@a_z_e_t

Mail:

azet@azet.org

XMPP:

azet@jabber.ccc.de

GPG Fingerprint:

7CB6 197E 385A 02DC 15D8 E223 E4DB 6492 FDB9 B5D5