

TLS Security Training



HACKMANIT

Prof. Dr. Juraj Somorovsky | @jurajsomorovsky

Overview of the Training

1. **Crypto Basics**
2. **Transport Layer Security**
3. **Certificates**
4. **Attacks on PKI**
5. **Attacks on TLS**
6. **TLS Evaluation Tools**
7. **TLS Implementations**
8. **Outlook**

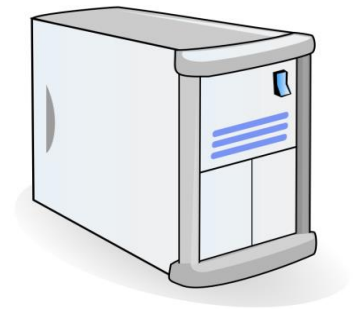
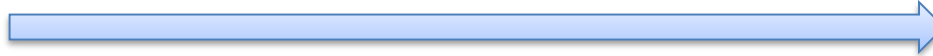
How to Distribute Symmetric Keys?

- Public-key (asymmetric-key) crypto



**Server Public
key: pub
Symmetric key: k**

$$C = \text{Enc}(\text{pub}, k)$$



**Public key: pub
Private key: priv**

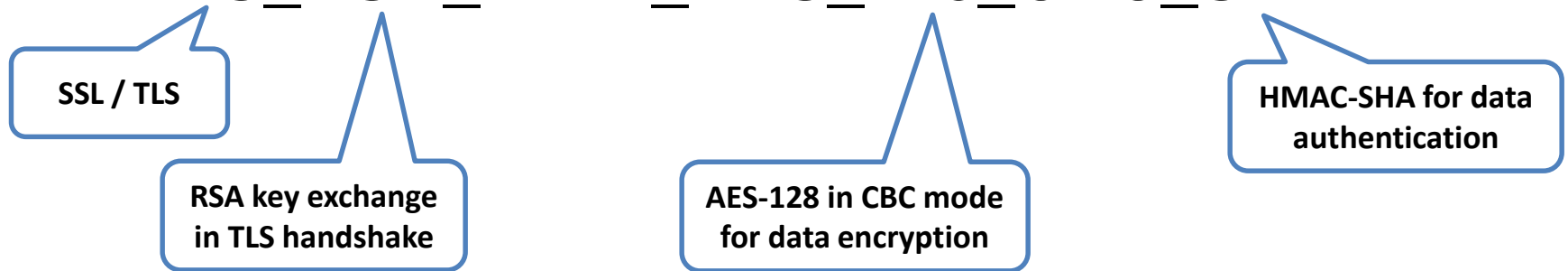
- Algorithms: RSA

$$K = \text{Decrypt}(\text{priv}, C)$$

TLS Cipher Suite

- Collection of crypto algorithms used in a TLS session

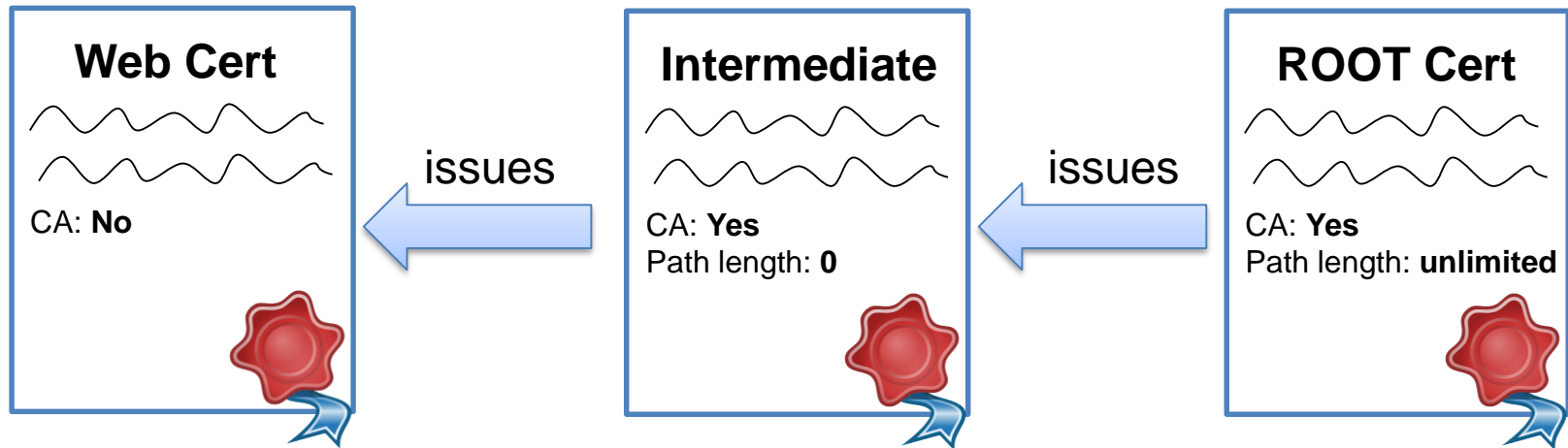
TLS_RSA_WITH_AES_128_CBC_SHA



TLS_DHE_RSA_WITH_AES_128_CBC_SHA

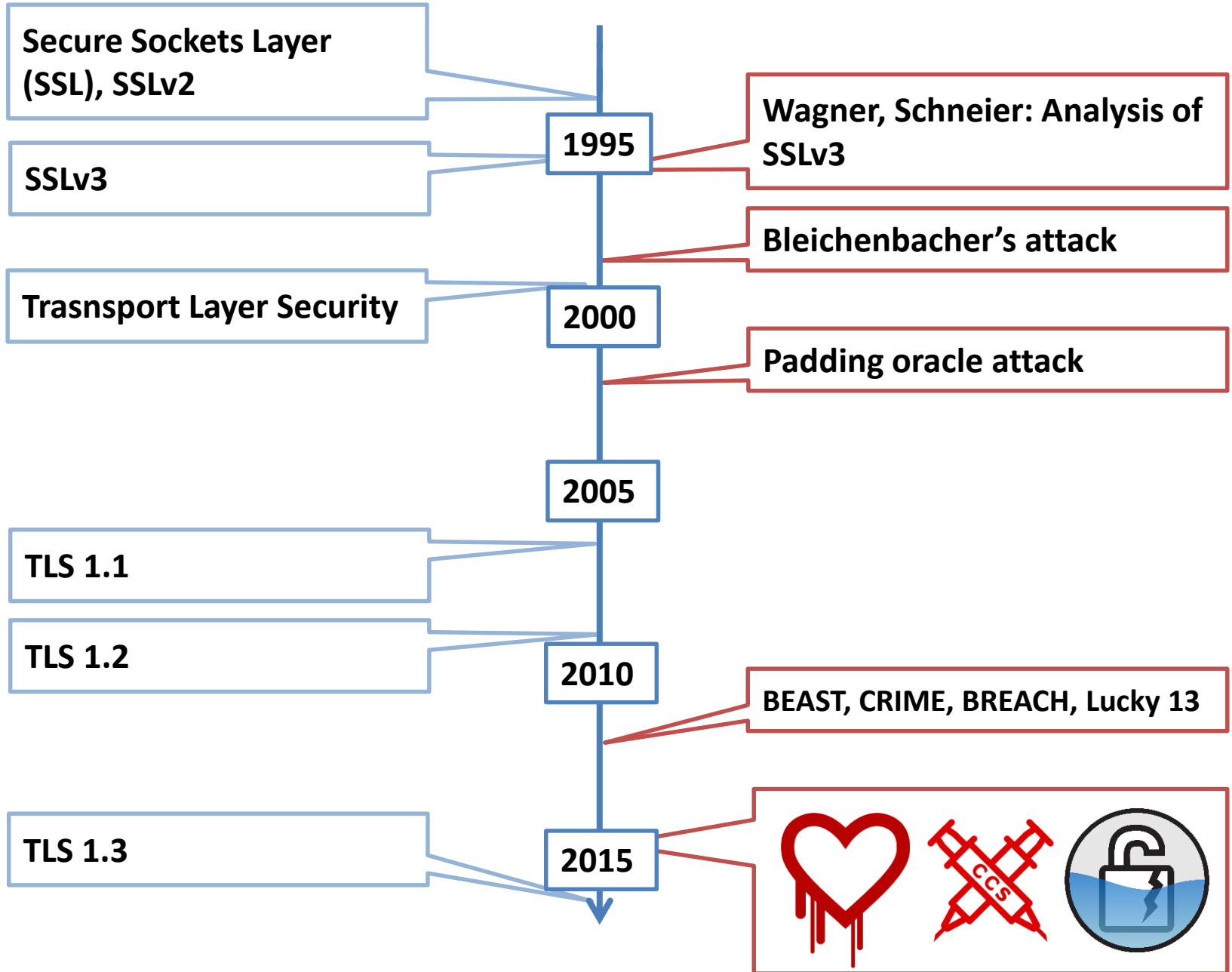


Certificate Chains



- Root CAs issue intermediate certs
- Intermediate CAs issue certificates for subscribers

TLS History



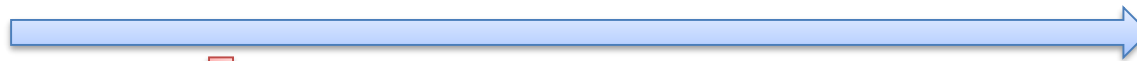
TLS Server as an Oracle

- Attacker can query the server
- The server decrypts and responds with **valid/invalid**
- Possible side channels:
 - Direct messages
 - Timing

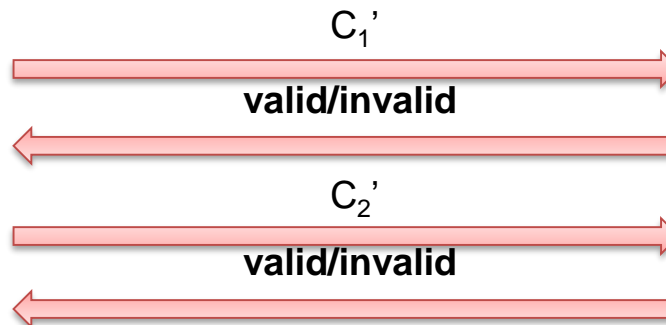
Million message attack

Padding oracles

C



M = Dec(C)



...
(repeated several times)



CRIME: Compression

- TLS offers compression
- Deflate compression (used in ZIP, GZIP...)
- Compression:

compression is **complex**  compression is **(-15,4)lex**

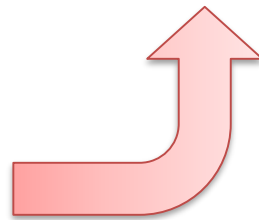
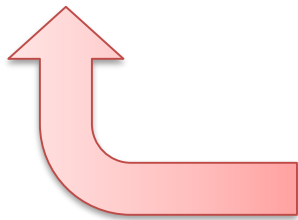
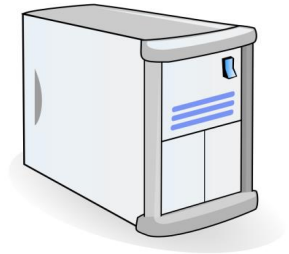
compression is easy  compression is easy

- More redundancy -> more compression
- Less redundancy -> less (no) compression

CRIME: Scenario

- Assumption: Streaming-based cipher

GET /SID=4 HTTP/1.1
Cookie: SID=48024820404804



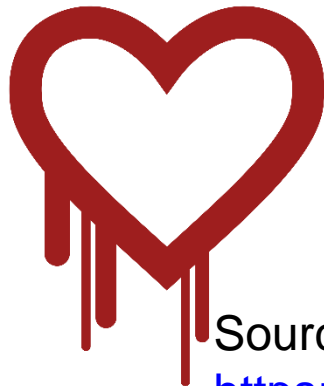
Length = 51

5-byte Compression!

Javascript:
Send(example.com/SID=4)

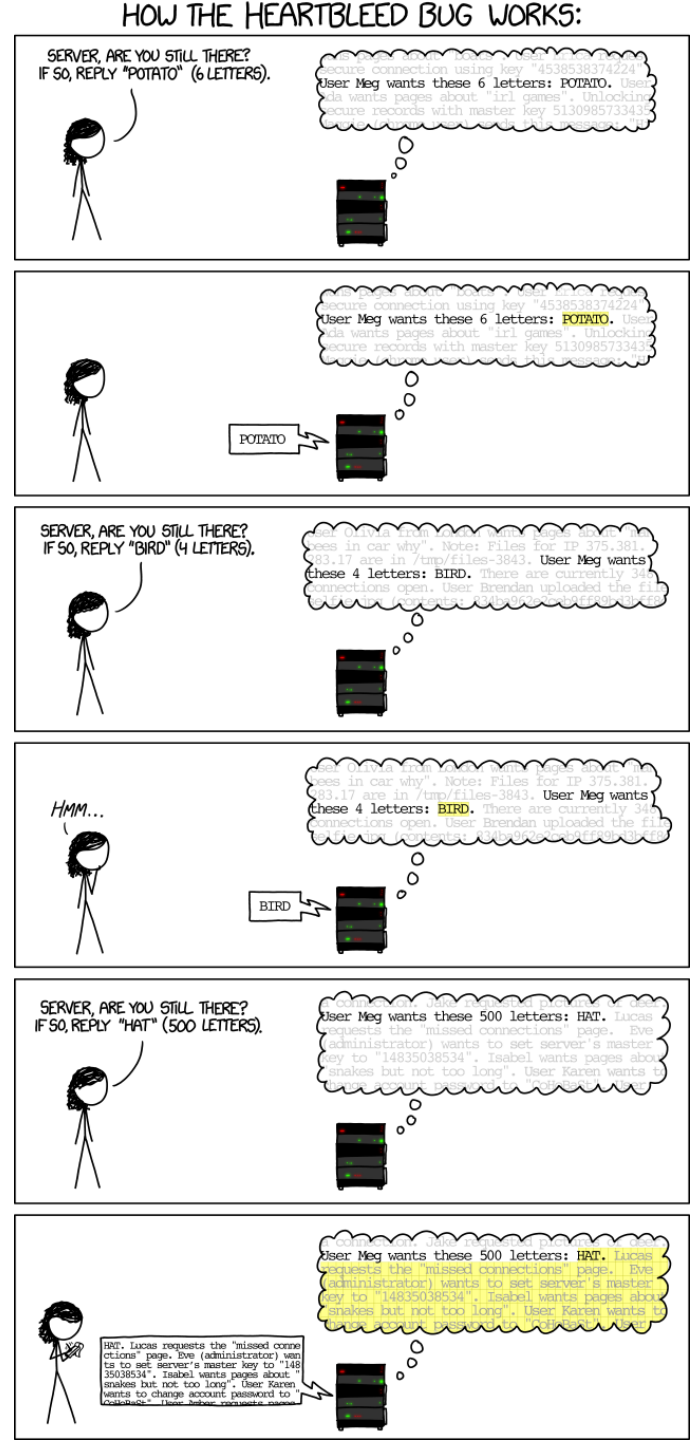
Heartbleed

- Discovered in April 2014
- Riku, Antti and Matti (Codenomicon) and Mehta (Google Security)
- Buffer Overread
- Improper validation of a Heartbeat request



Source:

<https://xkcd.com/1354/>



mbed TLS

- Formerly known as PolarSSL
- <https://tls.mbed.org/>
- Especially for hardware devices
- Running:
 - Compile the library
 - Dummy test server located in:
`mbedtls-2.0.0/programs/ssl/ssl_server2`
 - Run with the following parameters:
`ssl_server2 server_port=54001 key_file=[rsakey.pem]
crt_file=[rsacert.pem]`

Complex Cipher Suite Configuration

- Only SHA1 cipher suites:
 - openssl ciphers 'SHA'
- No SHA1 cipher suites:
 - openssl ciphers '!SHA'
- ECDH and DH cipher suites, no RSA:
 - openssl ciphers 'ECDH:DH:!RSA'
- “Whitebox” configuration example:
 - openssl ciphers 'ECDHE-ECDSA-AES256-GCM-SHA384,ECDHE-ECDSA-AES256-SHA384,ECDHE-ECDSA-AES128-GCM-SHA256'

Does not mean your server will offer them all. You have to provide appropriate keys!

Apache httpd Installation

- Better do not compile yourself ... but you can try it
- Ubuntu 14.04:

```
$ sudo apt-get install apache2
$ apache2 -version
Server version: Apache/2.4.7 (Ubuntu)
Server built:   Oct 14 2015 14:20:21
```
- Install mod_ssl

```
$ sudo a2enmod ssl
```

Apache httpd: Client Authentication

- Given we want to protect “/secure” directory
- Access given only to specific certificates from the company Hackmanit

```
SSLVerifyClient none
```

```
SSLVerifyDepth 10
```

```
<Location "/secure">
```

```
    SSLVerifyClient require
```

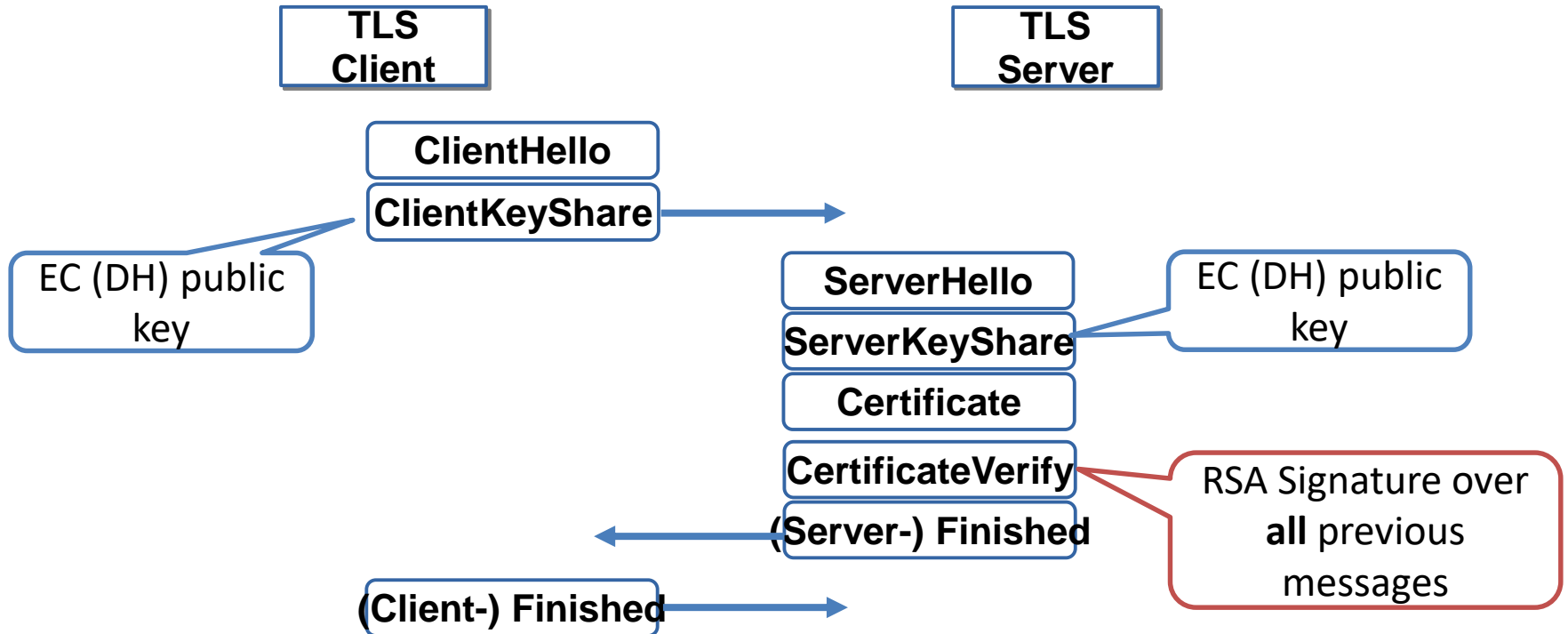
```
    SSLVerifyDepth 10
```

```
    SSLRequireSSL
```

```
    SSLRequire %{SSL_CLIENT_S_DN_O} eq „hackmanit“
```

```
</Location>
```

TLS 1.3





HACKMANIT

THREAT ANALYSIS | TRAINING | PENETRATION TESTS

Prof. Dr. Juraj Somorovsky: juraj.somorovsky@hackmanit.de

www.hackmanit.de | [@hackmanit](#)