

CSE 227 Computer Security

Stefan Savage, Fall 2021

Transport Layer Security (TLS) II papers

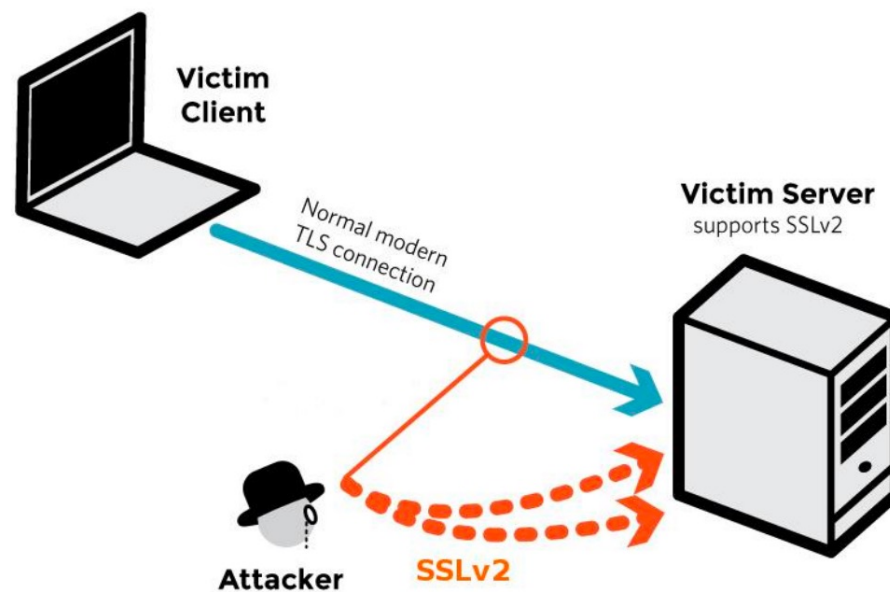
DROWN paper

- What problem are they attacking? Why now?
- Who are these people and why are they attacking this problem in this way?
- What is the goal? (i.e., how should you evaluate the paper?)

Context

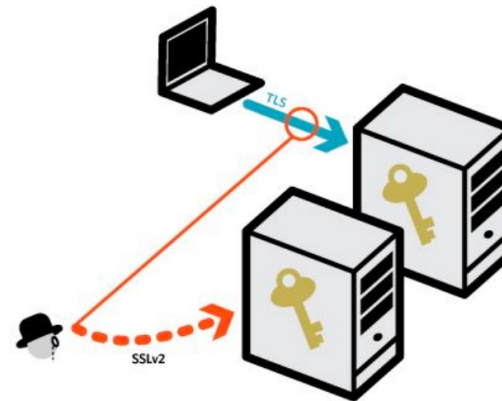
- SSLv2 published in 1995, broken immediately
 - Significant vulnerabilities to MITM attacks
 - Length extension attacks
- But, still enabled as an option (esp for email) on millions of servers (how do we know?)
- Downgrade attacks
 - What are they? What is the risk?
 - Is that what we're dealing with here? (i.e., is DROWN a downgrade attack?)

Big result: can break TLS using SSLv2

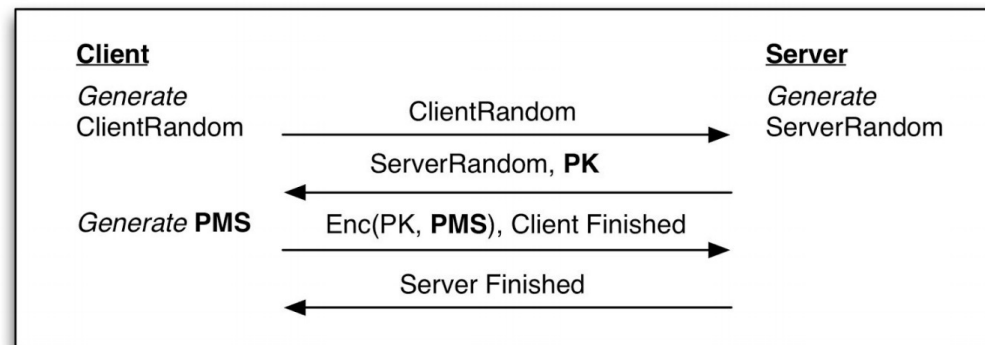


Drown overview

- Attacker decrypts intercepted TLS traffic
- Cross-protocol attack
 - **Attack TLS server using SSLv2 server**
 - Attack HTTPS server using email server - SSLv2 much more prevalent on email ports
- **22% of trusted HTTPS hosts vulnerable** with cross-protocol use

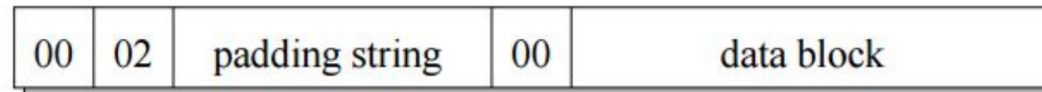


TLS RSA Handshake



PKCS#1 v1.5

- Textbook RSA: $k^e \bmod N$
 - Problem: No randomization
- In real-life:
 - PKCS #1 v1.5: pad k to length of N with random padding



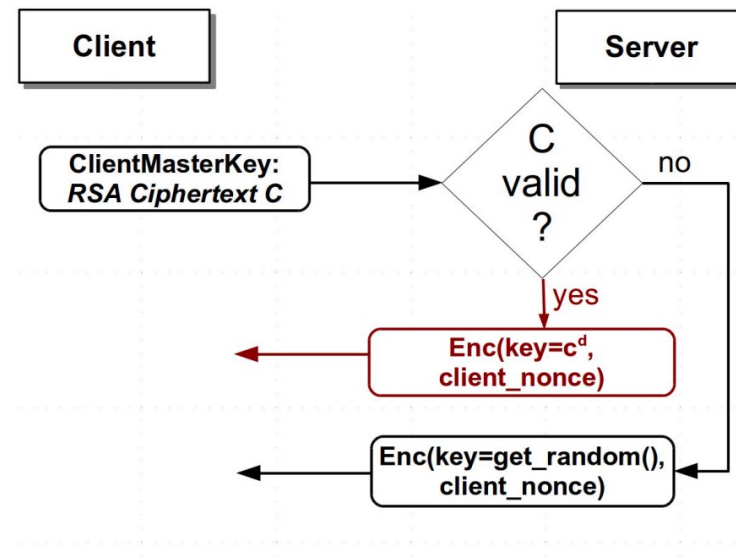
Bleichenbacher's attack

- If padding is incorrect after decryption, then...
 - ~~Send an error message~~
 - Attacker can deduce if padding was correct.

This “padding oracle” is sufficient to break RSA

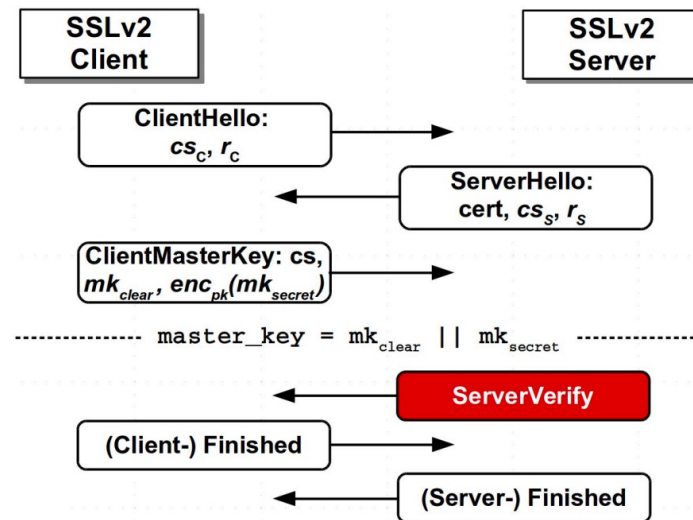
Bleichenbacher's attack

- If padding is incorrect after decryption, then...
The server has to behave as if the padding was valid!
- Solution: Server generates a random “replacement” plaintext, continues as usual.



Key (for this paper) differences between SSLv2 and TLS

- Server authenticates **first** (sends first message encrypted with symmetric key)
- Short secrets for export grade crypto:
 - SSLv2: 40 bit key.

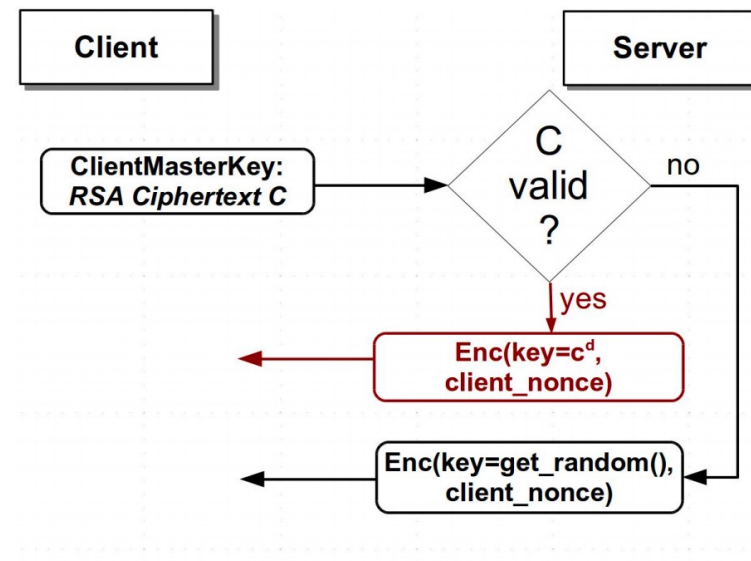


Quick aside: abbreviated US crypto wars

- Since cold war crypto classified as a munition for the purposes of export control (i.e., required a license from the DoD)
- Challenges from commercial use by PCs
 - **Deal struck in 1992 to allow export of 40bit crypto with low-level of review**
- Clipper chip (1993)
 - US govt proposes to mandate use of NSA-developed encryption alg (skipjack) with law enforcement escrow (LEAF)
 - Huge bipartisan backlash (also crypto moves to commerce)
- 2000
 - Commerce liberalizes rules (key length restrictions removed – export license still required)

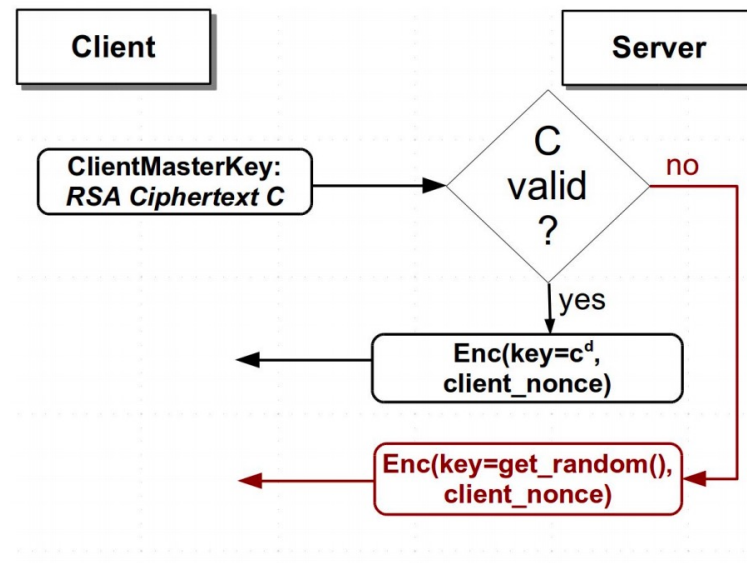
Important observation

- Attacker connects twice with same RSA ciphertext.
- Ciphertext **valid**:
- 2 server replies encrypted with same **key**.



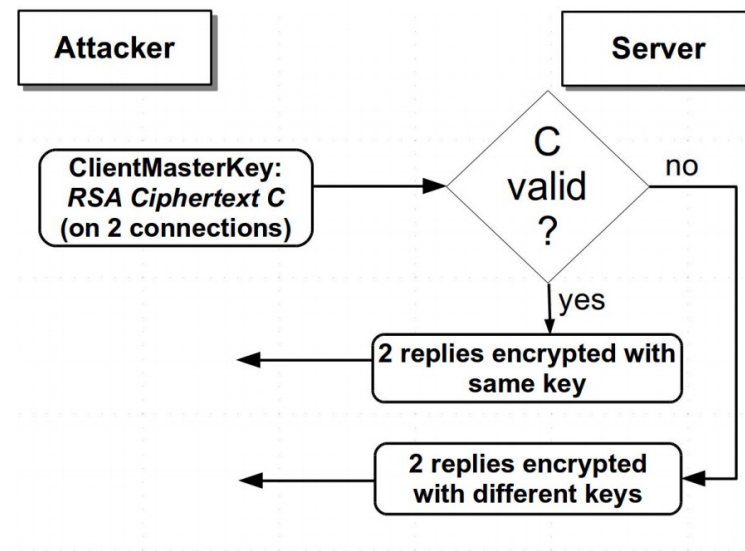
Important observation

- Attacker connects twice with same RSA ciphertext.
- Ciphertext not valid:
- 2 server replies encrypted with different keys.



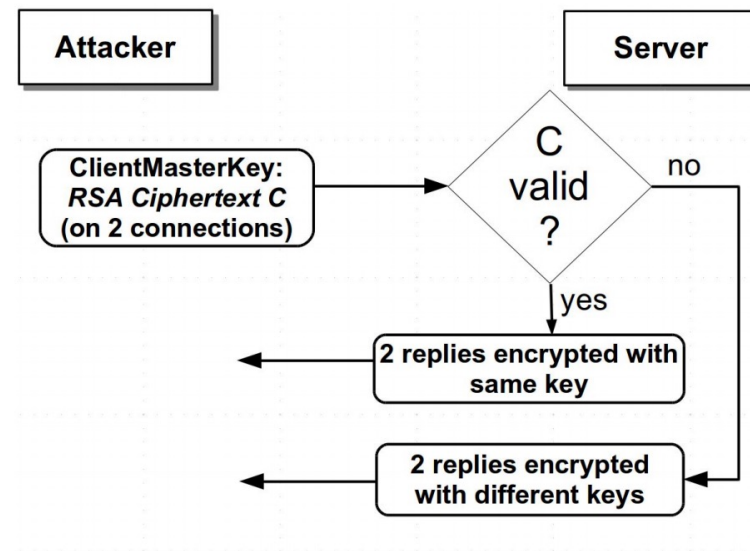
SSLv2 RSA Decryption Oracle

- Attacker breaks 40 bit key for both messages.
- Ciphertext valid:
 - Both keys will be the unpadded RSA plaintext -> keys will be identical.



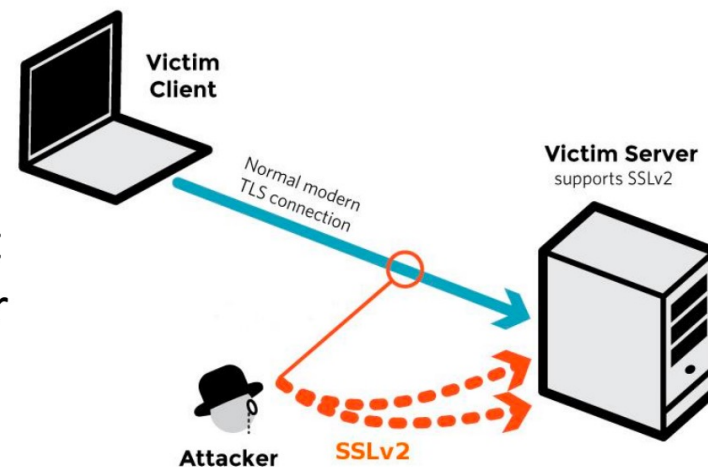
SSLv2 RSA Decryption Oracle

- Attacker breaks 40 bit key for both messages.
- If ciphertext is invalid:
 - Both keys will be randomly generated -> keys will be different.
- Reminder: If attacker can distinguish between valid/invalid RSA message, attacker can decrypt TLS!



Basic DROWN Attack

- Attacker records ~1,000 modern TLS connections.
- Attacker morphs TLS RSA ciphertext to SSLv2 ciphertext
 - Uses SSLv2 Bleichenbacher oracle to decrypt.
- Client never makes an SSLv2 connection.



Offline work

- Attacker executes ~10K queries, breaks 40-bit key for each Bleichenbacher query.
 - 2^{50} keys tested overall.
- Feasible on modern hardware:
 - Naive CPU implementation: \$21K of CPU, 114 days.
- Highly optimized GPU implementation:
 - \$18K of GPUs, 18 hours, or \$440 on AWS, 8 hours.
- Special DROWN: Implementation vulnerability in OpenSSL
 - 22% of trusted HTTPS servers are vulnerable

Takeways

- A problem in real life?
 - How to decide?
- Lessons for future implementations?

HTTPS interception paper

- What problem are they attacking? Why now?
- Who are these people and why are they attacking this problem in this way?
- What is the goal? (i.e., how should you evaluate the paper?)

HTTPS

- What properties does it provide?
- What problems does it present for those trying to implement network security?

Lots of products intercept HTTPS

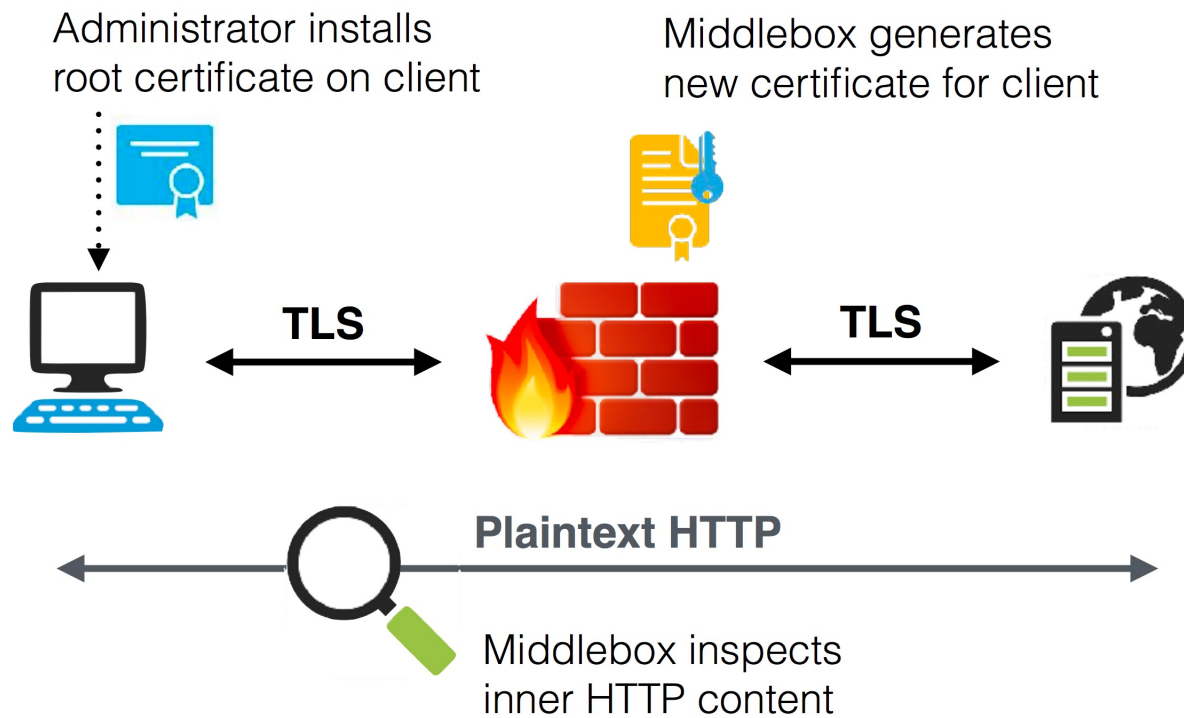


AVG



MITM HTTPS Interception

How it typically works



Key idea

- How might you measure if HTTPS sessions are being intercepted?
 - TLS data stream is transformed, but valid
 - None of the HTTP elements are modified
- But can look for *impossibilities*
 - i.e., what browsers could generate what TLS requests?

What do you see?

HTTP



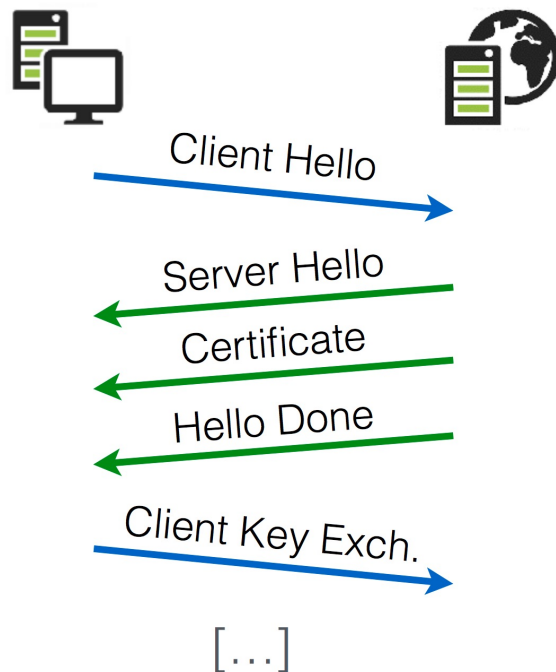
Parse HTTP User Agent Header:

Mozilla/5.0 (Macintosh; Intel [Mac OS X 10_12_2](#)) AppleWebKit/
537.36 (KHTML, like Gecko) [Chrome/55.0.2883.95](#) Safari/537.36

TLS



What the TLS handshake looks like



```
Secure Sockets Layer
  TLSv1.2 Record Layer: Handshake Protocol: Client Hello
    Content Type: Handshake (22)
    Version: TLS 1.0 (0x0301)
    Length: 217
  Handshake Protocol: Client Hello
    Handshake Type: Client Hello (1)
    Length: 213
    Version: TLS 1.2 (0x0303)
    Random
    Session ID Length: 0
    Cipher Suites Length: 36
    Cipher Suites (18 suites)
    Compression Methods Length: 1
    Compression Methods (1 method)
    Extensions Length: 136
    Extension: Unknown 35466
    Extension: renegotiation_info
    Extension: server_name
    Extension: Extended Master Secret
    Extension: SessionTicket TLS
    Extension: signature_algorithms
    Extension: status_request
    Extension: signed_certificate_timestamp
    Extension: Application Layer Protocol Negotiation
    Extension: channel_id
    Extension: ec_point_formats
      Type: ec_point_formats (0x000b)
      Length: 2
      EC point formats Length: 1
      EC point format: uncompressed (0)
    Extension: elliptic_curves
    Extension: Unknown 43690
```

Firefox vs GnuTLS ClientHello

Extensions

Server Name (SNI)
Extended Master Secret
Renegotiation Info
Elliptic Curves
[...]



Ciphers

ECDHE_ECDSA_AES128_GCM_SHA256
ECDHE_RSA_AES128_GCM_SHA256
ECDHE_RSA_CHACHA20_SHA256
ECDHE_ECDSA_AES256_GCM_SHA384
[...]

Curves

secp256r1
secp384r1
secp521r1

Extensions

Extended Master Secret
Encrypt then MAC
OCSP Status Request
Server Name (SNI)
[...]



Ciphers

ECDHE_ECDSA_AES128_GCM_SHA256
ECDHE_ECDSA_AES128_GCM_SHA386
ECDHE_RSA_AES128_GCM_SHA256
ECDHE_RSA_CHACHA20_SHA256
ECDHE_ECDSA_AES256_GCM_SHA384
[...]

Curves

secp256r1
secp384r1
secp521r1
secp224r1
secp192r1

Interception oracle

- Painstakingly measure TLS/HTTP handshake data for each browser and browser version
- Make impossibility maps
 - Combinations of User-Agent, TLSHello that should not be possible

Magic bit of measurement

- Gets three large data hosters to give him HTTP User-Agent and TLS Client data
 - CloudFlare
 - Firefox Update Server
 - Set of E-commerce Web sites

Findings

	No Interception	Likely Interception	Confirmed Interception
Cloudflare	88.6%	0.5%	10.9%
Firefox	96.0%	0.0%	4.0%
E-Commerce	92.9%	0.9%	6.2%

Next question

- Assuming the interceptor is “benign”...
- What is the impact on session security?
- How would you decide this?

Impact on security

	Increased Security	Decreased Security	Severely Broken
E-Commerce	4%	27%	18%
Cloudflare	14%	45%	16%
Firefox Updates	0%	66%	37%

End-system vs network interception products

TLS Security	Increased Security	Same Security	Decreased Security	Severely Broken
Client Security Products	0/20	2/20	18/20	10/20
Middleboxes	0/12	1/12	6/12	5/12

No products implemented new HTTPS features beyond the TLS specification (e.g., HPKP)

Discussion

- Methodology?
- What next?

Next classes

- Wednesday class cancelled
- Next week: Usable security and human factors