



Authenticated Encryption



Authenticated Encryption

Case study: TLS

The TLS Record Protocol (TLS 1.2)



Unidirectional keys: $k_{b \rightarrow s}$ and $k_{s \rightarrow b}$

Stateful encryption:

- Each side maintains two 64-bit counters: $\text{ctr}_{b \rightarrow s}$, $\text{ctr}_{s \rightarrow b}$
- Init. to 0 when session started. $\text{ctr}++$ for every record.
- Purpose: replay defense

TLS record: encryption (CBC AES-128, HMAC-SHA1)

$$k_{b \rightarrow s} = (k_{\text{mac}}, k_{\text{enc}})$$



Browser side $\text{enc}(k_{b \rightarrow s}, \text{data}, \text{ctr}_{b \rightarrow s})$:

not transmitted in packet

step 1: $\text{tag} \leftarrow S(k_{\text{mac}}, [++\text{ctr}_{b \rightarrow s} \parallel \text{header} \parallel \text{data}])$

step 2: pad $[\text{header} \parallel \text{data} \parallel \text{tag}]$ to AES block size

step 3: CBC encrypt with k_{enc} and new random IV

step 4: prepend header

TLS record: decryption (CBC AES-128, HMAC-SHA1)

Server side **dec($k_{b \rightarrow s}$, record, $ctr_{b \rightarrow s}$) :**

step 1: CBC decrypt record using k_{enc}

step 2: check pad format: send **bad_record_mac** if invalid

step 3: check tag on [++ $ctr_{b \rightarrow s}$ || header || data]

send **bad_record_mac** if invalid

Provides authenticated encryption

(provided no other info. is leaked during decryption)

Bugs in older versions (prior to TLS 1.1)

IV for CBC is predictable: (chained IV)

IV for next record is last ciphertext block of current record.

Not CPA secure. (a practical exploit: BEAST attack)

Padding oracle: during decryption

if pad is invalid send **decryption failed** alert

if mac is invalid send **bad_record_mac** alert

⇒ attacker learns info. about plaintext (attack in next segment)

Lesson: when decryption fails, do not explain why

Leaking the length

The TLS header leaks the length of TLS records

- Lengths can also be inferred by observing network traffic

For many web applications, leaking lengths reveals sensitive info:

- In tax preparation sites, lengths indicate the type of return being filed which leaks information about the user's income
- In healthcare sites, lengths leaks what page the user is viewing
- In Google maps, lengths leaks the location being requested

No easy solution

End of Segment



Authenticated Encryption

CBC paddings attacks

Recap

Authenticated encryption: CPA security + ciphertext integrity

- Confidentiality in presence of **active** adversary
- Prevents chosen-ciphertext attacks

Limitation: cannot help bad implementations ... (this segment)

Authenticated encryption modes:

- Standards: GCM, CCM, EAX
- General construction: encrypt-then-MAC

The TLS record protocol (CBC encryption)

Decryption: $\text{dec}(k_{b \rightarrow s}, \text{record}, \text{ctr}_{b \rightarrow s})$:

step 1: CBC decrypt record using k_{enc}

step 2: check pad format: abort if invalid

step 3: check tag on $[++\text{ctr}_{b \rightarrow s} \parallel \text{header} \parallel \text{data}]$
abort if invalid

Two types of error:

- padding error
- MAC error



Padding oracle

Suppose attacker can differentiate the two errors
(pad error, MAC error):

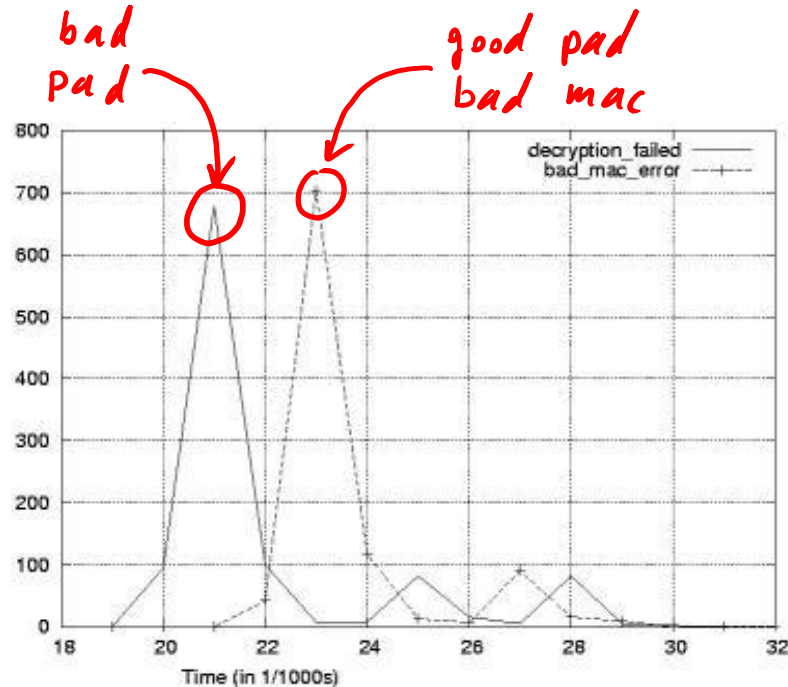
⇒ **Padding oracle:**

attacker submits ciphertext and learns if
last bytes of plaintext are a valid pad

Nice example of a
chosen ciphertext attack



Padding oracle via timing OpenSSL



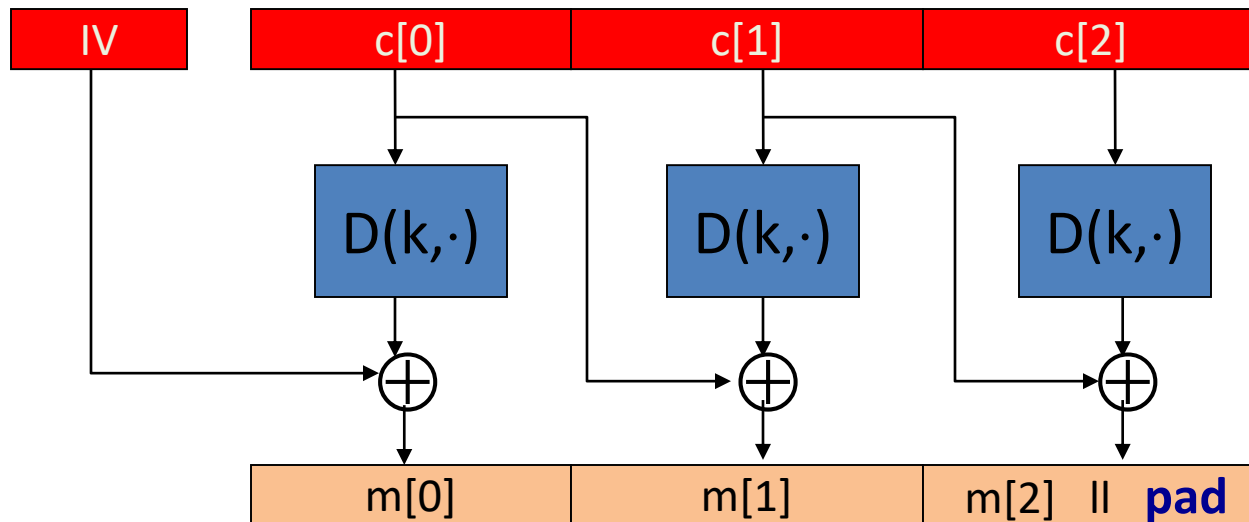
Credit: Brice Canvel

(fixed in OpenSSL 0.9.7a)

In older TLS 1.0: padding oracle due to different alert messages.

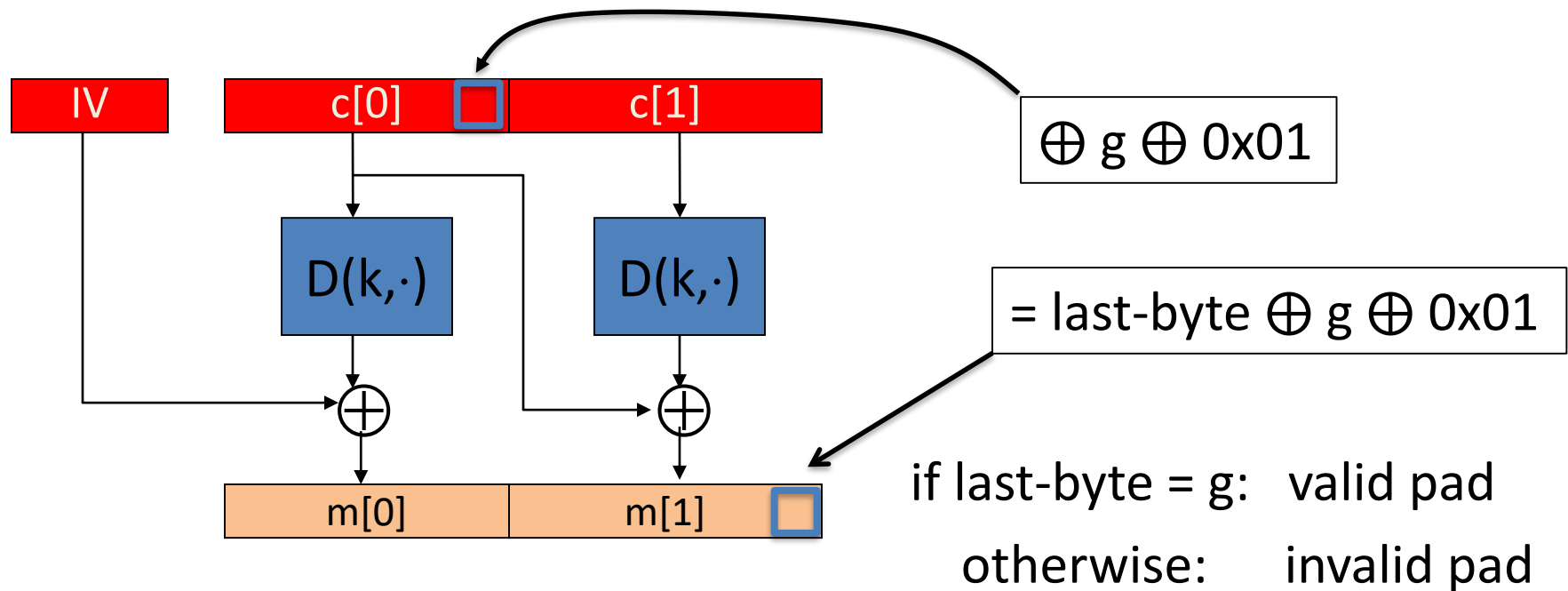
Using a padding oracle (CBC encryption)

Attacker has ciphertext $\mathbf{c} = (c[0], c[1], c[2])$ and it wants $\mathbf{m}[1]$



Using a padding oracle (CBC encryption)

step 1: let \mathbf{g} be a guess for the last byte of $m[1]$



Using a padding oracle (CBC encryption)

Attack: submit $(IV, c'[0], c[1])$ to padding oracle

\Rightarrow attacker learns if last-byte = g

Repeat with $g = 0, 1, \dots, 255$ to learn last byte of $m[1]$

Then use a $(02, 02)$ pad to learn the next byte and so on ...

IMAP over TLS

Problem: TLS renegotiates key when an invalid record is received

Enter IMAP over TLS: (protocol for reading email)

- Every five minutes client sends login message to server:
LOGIN "username" "password"
- Exact same attack works, despite new keys
⇒ recovers password in a few hours.


Lesson

1. Encrypt-then-MAC would completely avoid this problem:

MAC is checked first and ciphertext discarded if invalid

2. MAC-then-CBC provides A.E., but padding oracle destroys it

Will this attack work if TLS used counter mode instead of CBC?
(i.e. use MAC-then-CTR)

- ☐ Yes, padding oracles affect all encryption schemes
- ☐ It depends on what block cipher is used
- ☐ No, counter mode need not use padding 
- ☐

End of Segment

Further reading

- The Order of Encryption and Authentication for Protecting Communications, H. Krawczyk, Crypto 2001.
- Authenticated-Encryption with Associated-Data, P. Rogaway, Proc. of CCS 2002.
- Password Interception in a SSL/TLS Channel, B. Canvel, A. Hiltgen, S. Vaudenay, M. Vuagnoux, Crypto 2003.
- Plaintext Recovery Attacks Against SSH, M. Albrecht, K. Paterson and G. Watson, IEEE S&P 2009
- Problem areas for the IP security protocols, S. Bellare, Usenix Security 1996.