Side-Channel Attacks and Chosen Ciphertext Security
Mike Reiter

Chosen Ciphertext Attacks

Recall that in a chosen ciphertext attack, the adversary is given:
- An encryption oracle $E_K$
- A decryption oracle $D_K$
- A test oracle $T_K$
  - If $c \leftarrow T_K(m_0, m_1)$ then adversary is not permitted to invoke $D_K(c)$

Arguably having otherwise unfettered access to $D_K$ is unrealistic, and so variations on this model have been explored:
- Lunchtime attack: Adversary can query $D_K$ only before querying $T_K$
- Side-channel attack: Instead of having access to $D_K$, adversary is given access to a “side channel” oracle $P_K$
  - $P_K(c)$ returns $f(D_K(c))$ for a particular function $f$

We will explore a frequently practical side channel in this lecture.
Recall CBC Mode Encryption

- Let \( f: \text{Keys} \times \{0,1\}^L \rightarrow \{0,1\}^L \) be a pseudorandom permutation

Algorithm \( E_K(m) \):
- let \( m_1 | \ldots | m_n = m : m_i \in \{0,1\}^L \)
- \( c_0 \leftarrow_R \{0,1\}^L \)
- for \( i = 1 \ldots n \) do \( c_i \leftarrow f_K(c_{i-1} \oplus m_i) \)
  - return \( c_0 | c_1 | \ldots | c_n \)

Algorithm \( D_K(c) \):
- let \( c_0 | c_1 | \ldots | c_n = c : c_i \in \{0,1\}^L \)
- for \( i = 1 \ldots n \) do \( m_i \leftarrow f_K^{-1}(c_i) \oplus c_{i-1} \)
  - return \( m_1 | \ldots | m_n \)

- Above description assumes that length of \( m \) is a multiple of \( L \)
  - If not, padding is required

Padding

- A padding function is a function \( \text{PAD}: \{0,1\}^* \rightarrow (\{0,1\}^L)^+ \)
  - Most applications require \( \text{PAD} \) to be reversible

- Two types of padding functions
  - Byte-oriented, where \( \text{PAD}: (\{0,1\}^8)^+ \rightarrow (\{0,1\}^L)^+ \) and \( L = 8b \)
  - Bit-oriented, where domain of \( \text{PAD} \) is unrestricted

- Example: CBCPAD is a byte-oriented padding function

Algorithm CBCPAD(\( m \)):
- let \( m_1 | \ldots | m_n = m : m_i \in \{0,1\}^8 \)
- \( p \leftarrow b - (n \mod b) \)
  - return \( m | pp \ldots p \) \( p \) times

- Padding is “01”, “02 02”, “03 03 03”, “04 04 04 04” ...
  - Denote this by “\( p \times p \)”
Processing Padding

- What if the padding in a ciphertext is not valid?
  - tear down the session (as in SSL/TLS)?
  - log the error (as in ESP)?
  - return an error message (as in WTLS)?
- Either way, typically will leak whether the padding was valid
- Abstract this as an oracle \( P_K \)

Algorithm \( P_K(c) \):

\[
\begin{align*}
\text{let } c_0 | c_1 | \ldots | c_n = c : c_i \in \{0,1\}^L \\
\text{for } i = 1 \ldots n \text{ do } m_i \leftarrow f_K^{-1}(c_i) \oplus c_{i-1} \\
\text{if } m_n \text{ ends in } p \times p \text{ for some } p > 0 \text{ return } 1 \\
\text{else} \text{ return } 0
\end{align*}
\]

Last Byte Decryption

[Vaudenay 2002]

- Consider a two-block ciphertext \( c_0 | c_1 \)
- We know that decryption is performed as follows

\[
\begin{align*}
m_1 & \leftarrow f_K^{-1} \\
& \quad \oplus c_0 \quad \text{---} \quad c_1
\end{align*}
\]

- Consider any \( c_0' \neq c_0 \)

\[
\begin{align*}
m_1' & \leftarrow f_K^{-1} \\
& \quad \oplus c_0' \quad \text{---} \quad c_1
\end{align*}
\]

- Since \( c_0 \oplus m_1 = c_0' \oplus m_1' = f_K^{-1}(c_1) \), we get \( m_1 = (c_0 \oplus c_0') \oplus m_1' \)
  - We know \( (c_0 \oplus c_0') \) but not \( m_1' \)
Last Byte Decryption (cont.)

- However, if we can find $c_0'$ such that $P_K(c_0' \mid c_1) = 1$, then we know that $m_1'$ is correctly padded.

- Moreover, if $c_0'$ is chosen randomly from $\{0,1\}^L$, then
  - $m_1'$ ends in 01 with probability $1/2^8$,
  - $m_1'$ ends in 02 02 with probability $1/2^{16}$,
  - $m_1'$ ends in 03 03 03 with probability $1/2^{24}$,
  - ...

- So, we could just assume that $m_1'$ ends in 01, and would usually be right
  - If correct, then last byte of $m_1$ is last byte of $(c_0 \oplus c_0') \oplus 01$

---

Last Byte Decryption (cont.)

- To get it right in all cases, start from $c_0'$ where $P_K(c_0' \mid c_1) = 1$ and do the following
  - If $P_K((c_0' \oplus 01(00)^{b-1}) \mid c_1) = 0$ then $m_1'$ ends in $b \times b$, else
  - If $P_K((c_0' \oplus 01(00)^{b-2}) \mid c_1) = 0$ then $m_1'$ ends in $b-1 \times b-1$, else
  - ...
  - If $P_K((c_0' \oplus 01(00)^{1}) \mid c_1) = 0$ then $m_1'$ ends in 02 02, else
  - $m_1'$ ends in 01

$$
\begin{array}{c}
\oplus \\
\end{array}
\begin{array}{cccc}
\cdots & \cdots & \cdots & f_K^{-1}(c_1) \\
\end{array}
\begin{array}{c}
b \\
\end{array}
\begin{array}{c}
\oplus \\
\end{array}
\begin{array}{cccc}
\cdots & \cdots & \cdots & c_0' \\
\end{array}
= \begin{array}{cccc}
\cdots & \cdots & 03 & 03 & 03 & m_1' \\
\end{array}
$$
To get it right in all cases, start from $c_0'$ where $P_K(c_0' | c_1) = 1$ and do the following:

- If $P_K((c_0' \oplus 01(00)^{b-1}) | c_1) = 0$ then $m_1'$ ends in $b \times b$, else
- If $P_K((c_0' \oplus 01(00)^{b-2}) | c_1) = 0$ then $m_1'$ ends in $b-1 \times b-1$, else
- ... 
- If $P_K((c_0' \oplus 01(00)) | c_1) = 0$ then $m_1'$ ends in 02 02, else
- $m_1'$ ends in 01
Now we can use this to find all of $m_1$

$\oplus \quad c_0' \quad \oplus \quad c_0 \oplus c_0' \quad = \quad m_1$

$\oplus \quad c_0' \quad \oplus \quad f_K^{-1}(c_1) \quad = \quad 01 \oplus 02 \quad m_1'$

$\oplus \quad c_0' \quad \oplus \quad c_0 \oplus c_0' \quad = \quad m_1$

Copyright © 2019 by Michael Reiter
All rights reserved.
Block Decryption

Now we can use this to find all of $m_1$

\[
\begin{align*}
    & f_K^{-1}(c_1) \quad c_0' \\
    \oplus & \quad c_0' \\
    & 02 \quad 02 \\
\end{align*}
\]

Find $x$ such that $P_K = 1$.

\[
\begin{align*}
    & c_0 \oplus c_0' \\
    \oplus & \quad x \\
    & \quad 01 \oplus 02 \\
\end{align*}
\]

Find $x$:

\[
\begin{align*}
    & m_1' \\
    \oplus & \quad m_1 \\
\end{align*}
\]
### Block Decryption

- Now we can use this to find all of \( m_1 \)

\[
\begin{align*}
\oplus & \quad \ldots \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad f_K^{-1}(c_1) \\
\oplus & \quad \ldots \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad c_0' \\
\oplus & \quad \ldots \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad x \quad 01 \oplus 02 \\
\oplus & \quad \ldots \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad 02 \quad 02 \quad m_1' \\
\oplus & \quad \ldots \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad c_0 \oplus c_0' \\
\oplus & \quad \ldots \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad x \\
\oplus & \quad \ldots \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad m_1
\end{align*}
\]

### Full Decryption

- Once we’ve implemented block decryption, full decryption of multi-block messages is straightforward
  - Do each block separately
  - Use preceding ciphertext block as its initialization vector

- Block decryption can be sped up using binary search instead of linear search to find padding length
Other Symmetric Encryption Schemes

- CBC is not the only encryption mode where padding is used
- Recall *counter mode*

**Algorithm** $E_K(m)$:

```
let $m_1|\ldots|m_n = m : m_i \in \{0,1\}^L$
let $c_0 \leftarrow_R \{0,1\}^L$
for $i = 1 \ldots n$ do $c_i \leftarrow f_K(c_0 + i \mod 2^L) \oplus m_i$
return $c_0 | c_1 | \ldots | c_n$
```

**Algorithm** $D_K(c)$:

```
let $c_0 | c_1 | \ldots | c_n = c : c_i \in \{0,1\}^L$
for $i = 1 \ldots n$ do $m_i \leftarrow f_K(c_0 + i \mod 2^L) \oplus c_i$
return $m_1 | \ldots | m_n$
```

- Padding here is similarly tricky

---

Counter Mode Encryption

- Suppose CBCPAD were used with counter mode
- A ciphertext $c_0 | c_1$ is decrypted as follows

```
c_1 \quad \oplus \quad m_1
```

- For any $c_1' \neq c_1$

```
c_1' \quad \oplus \quad m_1'
```

- Since $c_1 \oplus m_1 = c_1' \oplus m_1' = f_K(c_0 + 1)$, we get $m_1 = (c_1 \oplus c_1') \oplus m_1'$
- Once again, padding oracle enables $m_1$ to be recovered
Counter Mode Encryption (cont.)

- Note, however, that unlike with CBC, padding is not necessary with counter mode encryption

\[
\begin{array}{ccc}
m_1 & \cdots & m_{n-2} & m_{n-1} & m_n & m \\
\oplus & f_K(c_0+1) & \cdots & f_K(c_0+n-2) & f_K(c_0+n-1) & f_K(c_0+n) \\
c_0 & c_1 & \cdots & c_{n-2} & c_{n-1} & c_n & c
\end{array}
\]

- Blue portion can be discarded, rather than padding to utilize it
  - Advantage: eliminates any padding oracle
  - Disadvantage: exposes exact bit length of plaintext

Aborts

- Some protocols (notably SSL/TLS) abort if they encounter a padding error
  - If ciphertext is not authenticated, this is denial-of-service vulnerability
  - If ciphertext is authenticated, then padding oracle is unavailable

- Aborts limit the attacker to one guess

- If the receiver does not abort, then attacker learns last byte of plaintext for whatever ciphertext he submitted
  - Succeeds with probability \( \approx 1/2^8 \)
Padding Oracles in Public Key Systems
[Bleichenbacher 1998]

- Public key systems are equally vulnerable to attacks using padding oracles

- Recall RSA cryptosystem
  - Public key $K = <e, N>$, where $N = pq$ for primes $p, q$
  - Private key $K^{-1} = <d, N>$, where $ed \equiv 1 \mod (p-1)(q-1)$
  - $E_K(m) = (\text{pad}(m))^e \mod N$
  - $D_{K^{-1}}(c) = \text{pad}^{-1}(c^d \mod N)$

PKCS #1 v1.5 Padding for Encryption

- If $|N| = k$ bytes, then $256^{k-1} < N < 256^k$
- PKCS #1 (v1.5) padding for encryption is correct if
  - 1st byte is 00
  - 2nd byte is 02
  - next 8 bytes different from 00
  - at least one more 00 byte

<table>
<thead>
<tr>
<th>most significant byte</th>
<th>least significant byte</th>
</tr>
</thead>
<tbody>
<tr>
<td>00 02 padding string</td>
<td>00 message</td>
</tr>
</tbody>
</table>

at least 8 bytes

$k$ bytes
Properties of PKCS #1 v1.5 Padding

- Probability \( \Pr(\text{PKCS}) \) that a random message is correctly padded is
  \[ 0.18 \cdot 2^{-16} < \Pr(\text{PKCS}) < 0.97 \cdot 2^{-8} \]

- \( 1/\Pr(\text{PKCS}) < 360,000 \)
  - PKCS conforming messages can be found by trial and error

- Given a target ciphertext \( c = m^e \mod N \), attacker can submit \( c_i = c(s_i)^e \mod N \) to the padding oracle
  - If \( c_i \) is PKCS conforming, then \( 2 \cdot 256^{k-2} \leq ms_i \mod N < 3 \cdot 256^{k-2} \)

- This fact can be leveraged to decrypt \( c \)

---

Cost of Attack

- Number of queries needed
  - \( \Pr(\text{PKCS}) \) = probability that a random message is PKCS conforming
    - \( 0.18 \cdot 2^{-16} < \Pr(\text{PKCS}) < 0.97 \cdot 2^{-8} \)
  - \( \Pr(\text{PKCS}|A) \) = probability that a message with leading bytes 00 and 02 is PKCS conforming
    - \( 0.18 < \Pr(\text{PKCS}|A) < 0.97 \)
  - Number of oracle queries is \( \approx 3/\Pr(\text{PKCS}) + 16k/\Pr(\text{PKCS}|A) \)

- For example, if \( N \) is 1024 bits then roughly 1,000,000 queries are needed
Chosen Ciphertext Security

- These various side-channel attacks motivate the need for chosen ciphertext security

- Any adversary that can succeed using a side-channel attack can succeed using a chosen-ciphertext attack
  - simply uses the decryption oracle to implement the side channel

- Conversely, encryption that is invulnerable to chosen ciphertext attacks is also invulnerable to side channel attacks (based on the output from the decrypting party)

Recall a Chosen Ciphertext Attack

- The adversary is given three oracles
  - An encryption oracle $E_K$
  - A test oracle $T_K(m_0, m_1)$ that can be called only once

  Oracle $T_K(m_0, m_1)$:
  - if $|m_1| \neq |m_2|$ then return ⊥
  - $b \leftarrow R \{0,1\}$
  - return $E_K(m_b)$

  - A decryption oracle $D_K$

- The adversary must guess whether $b = 0$ or $b = 1$, but if
  - $c \leftarrow T_K(m_0, m_1)$

then adversary cannot query $D_K(c)$
Definition of CCA Security

- A CCA-secure encryption scheme is a triple $\langle \text{Gen}, E, D \rangle$
  such that for every PPT $A$ there is a negligible $\nu_A$ where
  $$\Pr[A^{E_K, D_K, T_K} = 0 : b \leftarrow 0] - \Pr[A^{E_K, D_K, T_K} = 0 : b \leftarrow 1] \leq \nu_A(\lambda)$$
  for all sufficiently large $\lambda$, where
  - the probabilities are taken over $K \leftarrow \text{Gen}(1^\lambda)$
  - $A^{E_K, D_K, T_K}$ is not permitted to query $D_K(c)$ if $c \leftarrow T_K(m_0, m_1)$