Admin:

Pset #2 due Mon 3/13
Project proposal due Fri 3/24

News:

WikiLeaks dump of CIA hacking tools.

Today:

Encryption: CBC mode
IND-CCA secure def
"UFE" mode

Authentication (MACS):
HMAC, CBC-MAC
(One-time MAC?)
EAX mode, Encrypt then MAC (AEAD modes)

Next: Finite fields & number theory

Readings:

Katz Ch. 4
Paar Ch. 12
CBC (Cipher-block chaining):

Choose IV ("initialization value") randomly, then use each $C_i$ is "IV" for $M_i$. Transmit IV with ciphertext:

$$ IV, C_1, C_2, ..., C_n $$

Decryption easy, and parallelizable (i.e., little error propagation)

Lookup "ciphertext stealing" for cute way of handling messages that are not a multiple of 64 bits in length. This method gives ciphertext length = message length.

Last block $C_n$ is the "CBC-MAC" (CBC Message Authentication Code) for message $M$. [A fixed IV is used here.] The MAC is a "cryptographic checksum" (more later...) (IF messages have variable length then key for last block should be different.)

MAC should use a different key than that used for encryption! Requires 2 passes to do CBC-Enc, then CBC-MAC over ciphertext.

IV might be Enc($k$, msg #) or Enc($k$, nonce) saves space if msg # does not need to be transmitted, or is short.
CFB (Cipher Feedback mode)

Similar to CBC mode. Uses random IV transmitted with ciphertext.

If M is not a multiple of b bits in length, can just transmit shortened ciphertext. (No need for ciphertext stealing.)
Are these modes good ones? What do we want?

If block cipher is indistinguishable from ideal block cipher then mode provides indistinguishability based on chosen ciphertext attack (IND-CCA):

- Define as game with adversary.
- Mode is IND-CCA secure if adversary can win with probability at most $\frac{1}{2} + \epsilon$ for "negligible" $\epsilon$.

Let $K$ be randomly chosen key.
Let $E_K$ denote encryption (using mode) with key $K$.
Let $D_K$ denote decryption.

**Phase I ("Find"):**

- Adversary given black-box access to $E_K$, $D_K$ (can encrypt/decrypt whatever it likes).
- Adversary outputs two messages $m_0$, $m_1$, of same length, plus state information $s$.

**Phase II ("Guess"):**

- Examiner secretly picks $d \leftarrow \{0,1\}$.
  Examiner computes $y = E_K(m_d)$.
- Adversary given $y$, $s$, access to $E_K$, and access to $D_K$ (except on $y$).
- Adversary computes for a while, then must produce bit $\hat{d}$ as its guess for $d$.
- Adversary's advantage is $|P(\hat{d} = d) - \frac{1}{2}|$.

Encryption secure against CCA attack if advantage is negligible.
Theorem: Modes ECB, CTR, CBC, CFB are not IND-CCA secure.

Proof: Adversary picks \( m_0 = 0^x, m_1 = 1^x \) for large \( x \).

Then \( y = E_k(m_d) \)

Let \( z = 1^{st} \text{ half of } y \).

Since \( z \neq y \), Adversary allowed in phase II to ask for \( D_k(z) \).

This gives first half of \( m_d \), revealing \( d \).

Adversary always wins. \( \square \)

Can one design a IND-CCA scheme?
Here is a sketch of one IND-CCA secure method, due to Desai. UFE = "Unbalanced Feistel encryption".

\( M = \) long message, sequence \( M_1, M_2, \ldots, M_n \) of \( b \)-bit blocks.

\( K = (K_1, K_2, K_3) \)

Three indep. keys for block ciphers

\( r \leftarrow \{0,1\}^b \)

starting counter value

\( P = P_1, P_2, \ldots, P_n \) where \( P_i = E_{K_i}(r + i) \leftarrow (CTR mode) \)

ciphertext \( C = C_1, C_2, \ldots, C_n \) where \( C_i = M_i \oplus P_i \)

CBC-MAC: \( X_0 = 0^b \)

\( X_i = E_{K_2}(X_{i-1} \oplus C_i) \quad 1 \leq i < n \)

\( X_n = E_{K_3}(X_{n-1} \oplus C_n) \) (MAC)

Output: \( C_1, C_2, \ldots, C_n, \sigma - \)

\( \sigma = r \oplus X_n \)

(no message authentication)

\( V_0 \rightarrow \text{PRF, not as MAC!} \)

\( V_1 \rightarrow \text{PRF} \)

---

**VO-PRF**

**VI-PRF**

\[ M \]

\[ P \]

\[ K_1 \]

\[ \text{CTR} \]

\[ X \]

\[ C \]

\[ \sigma \]

\[ r \leftarrow \{0,1\}^b \]

(actual proposal encrypt \( r \) with fourth key, to get starting counter value)

CBC MAC uses \( K_2 \) except uses \( K_3 \) on last block
• Encryption with UFE can be done in single pass over data, but decryption requires two passes:
  - first to compute $m = X_0$, then to get $r$
  - second to decrypt $C$ to get $M$
• Only designed for confidentiality (there is no way provided for receiver to tell if ciphertext has been tampered with.) (Need to use MAC on top of all of this, or some “combined mode” providing both confidentiality & integrity.)
• Note “unbalanced Feistel structure”.
• Length of ciphertext $(C, o) = |M| + |r|$; expansion only as needed for randomization. No need for “ciphertext stealing” since we use CTR mode.
MAC (Message Authentication Code)

- Not confidentiality, but integrity (recall "CIA")
- Alice wants to send messages to Bob, such that Bob can verify that messages originated with Alice & arrive unmodified.
- Alice & Bob share a secret key K
- Orthogonal to confidentiality; typically do both (e.g. encrypt, then append MAC for integrity)
- Need additional methods (e.g. counters) to protect against replay attacks

\[
\text{Alice} \quad \rightarrow \quad M, \text{MAC}_k(M) \quad \rightarrow \quad \text{Bob}
\]

[Here M is message to be authenticated, which could be ciphertext resulting from encryption.]
- Alice computes MAC\(_k(M)\) & appends it to M.
- Bob recomputes MAC\(_k(M)\) & verifies it agrees with what is received. If not, reject message.
Adversary (Eve) wants to forge \( M', \text{MAC}_k(M') \) pair that Bob accepts, without Eve knowing \( k \).

- She may hear a number of valid \((M, \text{MAC}_k(M))\) pairs first, possibly even with \( M \)'s of her choice (chosen msg attacks).

- She wants to forge for \( M' \) for which she hasn't seen \((M', \text{MAC}_k(M'))\) valid pair.

**Two common methods:**

\[
\text{HMAC}(k, M) = h(k_1 \| h(k_2 \| M))
\]

where \( k_1 = k \oplus \text{opad} \) \{opad, ipad are fixed constants\}

\( k_2 = k \oplus \text{ipad} \)

\[
\text{CBC-MAC}(k, M) \equiv \text{last block of CBC enc. of } M
\]

\( \text{note } IV=0 \)

Something like this is necessary...
MAC using random oracle (PRF):
\[ MAC_k(M) = h(k \| M) \]
(OK, if \( h \) is indistinguishable from RO, which means, as we saw, for sequential hash fn's, that last block may need special treatment.)

One-Time MAC (problem stmt):
Can we achieve security against unbounded Eve, as we did for confidentiality, with OTP, except here for integrity?

Here key \( K \) may be "use-once" [as it was for OTP].

\[ A \xrightarrow{K, M, T} B \]
\[ T = MAC_k(M) \] ("tag")

- Eve can learn \( M, T \) then try to replace \( M, T \) with \( M', T' \) (where \( M' \neq M \)) that Bob accepts.

- Eve is computationally unbounded.
<table>
<thead>
<tr>
<th></th>
<th>Confidentiality</th>
<th>Integrity</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Unconditional</strong></td>
<td>OTP ✅</td>
<td>One-time MAC?</td>
</tr>
<tr>
<td><strong>Conventional</strong></td>
<td>Block ciphers (AES) ✅</td>
<td>MAC (HMAC) ✅</td>
</tr>
<tr>
<td>(symmetric key)</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Public-key</strong></td>
<td>PK enc.</td>
<td>Digital signature</td>
</tr>
<tr>
<td>(asymmetric)</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Note: digital signatures are unforgeable, but also have nonrepudiation since only one copy of signing key exists.
EAX mode

See pgs 1-10 of
The EAX Mode of Operation
by Bellare, Rogaway, & Wagner

Figure 3 of EAX paper

Encrypt-then-MAC

\[ C = \text{Enc}(K_1, M) \]
\[ T = \text{MAC}(K_2, H||C) \]

\( C \), not \( M \)!

xmit: \( H, C, T \) [not encrypted, but authenticated]

Two passes
Two keys
Algorithm $\text{CBC}_K(M)$

10 Let $M_1 \cdots M_m \leftarrow M$ where $|M_i| = n$
11 $C_0 \leftarrow 0^n$
12 for $i \leftarrow 1$ to $m$
do
13 $C_i \leftarrow E_K(M_i \oplus C_{i-1})$
14 return $C_m$

Algorithm $\text{CTR}_K^N(M)$

20 $m \leftarrow \left\lceil |M|/n \right\rceil$
21 $S \leftarrow E_K(N) \parallel E_K(N+1) \parallel \cdots \parallel E_K(N+m-1)$
22 $C \leftarrow M \oplus S$ [first $|M|$ bits]
23 return $C$

Algorithm $\text{pad}(M; B, P)$

30 if $|M| \in \{n, 2n, 3n, \ldots\}$
31 then return $M \oplus B$
32 else return $(M \parallel 10^{n-1-\left\lceil |M| \mod n \right\rceil}) \oplus P$

Algorithm $\text{OMAC}_K(M)$

40 $L \leftarrow E_K(0^n)$; $B \leftarrow 2L$; $P \leftarrow 4L$
41 return $\text{CBC}_K(\text{pad}(M; B, P))$

Algorithm $\text{OMAC}_K^f(M)$

50 return $\text{OMAC}_K(|t|_n \parallel M)$

Figure 1: Basic building blocks. The block cipher $E: \text{Key} \times \{0,1\}^n \rightarrow \{0,1\}^n$ is fixed and $K \in \text{Key}$. For $\text{CBC}$, $M \in \{(0,1)^n\}^1$. For $\text{CTR}$, $M \in \{0,1\}^*$ and $N \in \{0,1\}^n$. For $\text{pad}$, $M \in \{0,1\}^*$ and $B, P \in \{0,1\}^n$ and the operation $\oplus$ xors the shorter string into the end of longer one. For $\text{OMAC}$, $M \in \{0,1\}^*$ and $t \in [0..2^n - 1]$ and the multiplication of a number by a string $L$ is done in GF($2^n$).

We have made a small modification to the OMAC algorithm as it was originally presented, changing one of its two constants. Specifically, the constant 4 at line 40 was the constant 1/2 (the multiplicative inverse of 2) in the original definition of OMAC [14]. The OMAC authors indicate that they will promulgate this modification [15], which slightly simplifies implementations.

3 The EAX Algorithm

Algorithm. Fix a block cipher $E: \text{Key} \times \{0,1\}^n \rightarrow \{0,1\}^n$ and a tag length $\tau \in [0..n]$. These parameters should be fixed at the beginning of a particular session that will use EAX mode. Typically, the parameters would be agreed to in an authenticated manner between the sender and the receiver, or they would be fixed for all time for some particular application. Given these parameters, EAX provides a nonce-based AEAD scheme EAX($E, \tau$) whose encryption algorithm has signature Key × Nonce × Header × Plaintext → Ciphertext and whose decryption algorithm has signature Key × Nonce × Header × Ciphertext → Plaintext ∪ \{INVALID\} where Nonce, Header, Plaintext, and Ciphertext are all $\{0,1\}^*$. The EAX algorithm is specified in Figure 2 and a picture illustrating EAX encryption is given in Figure 3. We now discuss various features of our algorithm and choices underlying the design.

No Encodings. We have avoided any nontrivial encoding of multiple strings into a single one.\footnote{One could view the prefixing of $[t]_n$ to $M$ in the definition of OMAC$_K^f(M)$ as an encoding, but $[t]_n$ is a constant, fixed-length string, and the aim here is just to "tweak" the PRF. This is very different from needing to encode arbitrary-length strings into a single string.} Some other approaches that we considered required a PRF to be applied to what was logically a tuple, like $(N, H, C)$. Doing this raises encoding issues we did not want to deal with because, ultimately, there would seem to be no simple, efficient, compelling, on-line way to encode multiple strings into a single one. Alternatively, one could avoid encodings and consider a new kind of primitive, a multi-argument PRF. But this would be a non-standard tool and we didn't want to use any non-standard tools. All in all, it seemed best to find a way to sidestep the need to do encodings.
### Algorithm EAX.Encrypt$^N_H$ ($M$)

10 $N \leftarrow \text{OMAC}_K^0(N)$
11 $K \leftarrow \text{OMAC}_K^1(H)$
12 $C \leftarrow \text{CTR}_K^N(N)$
13 $C \leftarrow \text{OMAC}_K^2(C)$
14 $\text{Tag} \leftarrow N \oplus C \oplus K$
15 $T \leftarrow \text{Tag}$ [first $\tau$ bits]
16 **return** $CT \leftarrow C \parallel T$

### Algorithm EAX.Decrypt$^N_H$ ($CT$)

20 **if** $|CT| < \tau$ **then return** INVALID
21 **Let** $C \parallel T \leftarrow CT$ where $|T| = \tau$
22 $N \leftarrow \text{OMAC}_K^0(N)$
23 $K \leftarrow \text{OMAC}_K^1(H)$
24 $C \leftarrow \text{OMAC}_K^2(C)$
25 $\text{Tag'} \leftarrow N \oplus C \oplus K$
26 $T' \leftarrow \text{Tag'}$ [first $\tau$ bits]
27 **if** $T \neq T'$ **then return** INVALID
28 $M \leftarrow \text{CTR}_K^N(C)$
29 **return** $M$

Figure 2: Encryption and decryption under EAX mode. The plaintext is $M$, the ciphertext is $CT$, the key is $K$, the nonce is $N$, and the header is $H$. The mode depends on a block cipher $E$ (that CTR and OMAC implicitly use) and a tag length $\tau$.

![Diagram](image)

Figure 3: Encryption under EAX. The message is $M$, the key is $K$, and the header is $H$. The ciphertext is $CT = C \parallel T$.

**WHY NOT GENERIC COMPOSITION?** Why have we specified a block-cipher based (BC-based) AEAD scheme instead of following the generic-composition approach of combining a (privacy-only) encryption method and