Admin:
(final projects!)

Today:
Message Authentication Codes (MACs)

- HMAC
- CBC-MAC
- PRF-MAC
- One-time MAC (problem stmt)

AEAD (Authenticated Encryption with Associated Data)

- EAX mode
- Encrypt-then-MAC

Readings:
Katz: Chapter 4, Chapter 7 (7.1-7.3)
Paar: Ch. 12, Ch. 6.3
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

Alice $\xrightarrow{M, MAC_k(M)}$ Bob $^K$

If MAC has $t$ bits, then Adv has prob $2^{-t}$ of successful forgery. Good MAC is (keyed) PRF.

- 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 \( K_1 = K \oplus \text{opad} \)

\( K_2 = K \oplus \text{iopad} \)

\( \text{opad, iopad are fixed constants} \)

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

\( \text{note } IV=0 \)
MAC using random oracle (PRF):
\[ \text{MAC}_k(M) = h (K \| M) \]
(OK if \( h \) is indistinguishable from \( \text{RO} \), which means, as we saw, for sequential hash \( \text{fn} \), that last block may need special \( \text{treatment} \).)

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].

\[
\begin{array}{ccc}
A & \text{K} & M,T \\
\hline
K & \text{T = MAC}_k(M) \quad \text{("tag")} \\
\end{array}
\]
- 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>Unconditional</td>
<td>OTP ✓</td>
<td>One-time MAC?</td>
</tr>
<tr>
<td>Conventional</td>
<td>Block ciphers (AES) ✓</td>
<td>MAC (HMAC) ✓</td>
</tr>
<tr>
<td>(symmetric key)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Public-key</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 non-repudiation, since only one copy of signing key exists.
Authenticated Encryption

EAX mode

[See pgs 1-10 of

The EAX Mode of Operation

by Bellare, Rogaway, & Wagner

Figure 3

Encrypt-then-MAC

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

\[ \overbrace{\text{Header}}^{c, \text{not } M} \]

\text{xmit: } H, C, T

Two passes
Two keys
Finite fields: System \((S, +, \cdot)\) s.t.

- \(S\) is a finite set containing "0" & "1"
- \((S, +)\) is an abelian (commutative) group with identity 0
  
  \[
  \begin{align*}
  (a+b)+c &= (a+(b+c)) \quad \text{associative} \\
  a+0 &= 0+a = a \quad \text{identity 0} \\
  (\forall a \in S) (\exists b) \quad a+b=0 \quad \text{(additive inverses \(b=-a\)} \\
  a+b &= b+a \quad \text{commutative}
  \end{align*}
  \]
- \((S^*, \cdot)\) is an abelian group with identity 1
  
  \[
  S^* = \text{nonzero elements of } S
  \]
  
  \[
  \begin{align*}
  (a \cdot b) \cdot c &= a \cdot (b \cdot c) \quad \text{associative} \\
  a \cdot 1 &= 1 \cdot a = a \quad \text{identity 1} \\
  (\forall a \in S^*) (\exists b \in S^*) \quad a \cdot b = 1 \quad \text{(multiplicative inverses \(b=a^{-1}\)} \\
  a \cdot b &= b \cdot a \quad \text{commutative}
  \end{align*}
  \]
- Distributive laws: \(a \cdot (b+c) = a \cdot b + a \cdot c\) 
  
  \[
  (b+c) \cdot a = b \cdot a + c \cdot a \quad \text{(follows)}
  \]

Familiar fields: \(\mathbb{R}\) (reals) are infinite 

\(\mathbb{C}\) (complex)

For crypto, we're usually interested in finite fields, such as \(\mathbb{Z}_p\) (integers mod prime \(p\))
Over field, usual algorithms work (mostly).

E.g. solving linear eqns:

\[ ax + b = 0 \pmod{\rho} \]

\[ \Rightarrow x = a^{-1} \cdot (-b) \pmod{\rho} \text{ is soln.} \]

\[ 3x + 5 = 6 \pmod{7} \]

\[ 3x = 1 \pmod{7} \]

\[ x = 5 \pmod{7} \]
**Notation:** GF(q) is the finite field ("Galois field") with q elements

**Theorem:** GF(q) exists whenever

\[ q = p^k, \text{ } p \text{ prime, } k \geq 1 \]

**Two cases:**

1. **GF(p)** - work modulo prime p
   
   \[ \mathbb{Z}_p = \text{integers mod } p = \{0, 1, \ldots, p-1\} \]
   
   \[ \mathbb{Z}_p^* = \mathbb{Z}_p - \{0\} = \{1, 2, \ldots, p-1\} \]

2. **GF(p^k)** : \( k > 1 \)
   
   Work with polynomials of degree < k with coefficients from GF(p)
   
   modulo fixed irreducible polynomial of degree k

Common case is **GF(2^k)**

**Note:** all operations can be performed efficiently

(inverses to be demonstrated)
Construction of $\mathbb{GF}(2^3) = \mathbb{GF}(8)$

Has 4 elements.

Is not arithmetic mod 4, (where 2 has no multi. inverse)

Elements are polynomials of degree 2 with coefficients
mod 2 (i.e. in $\mathbb{GF}(2)$):

\[
\begin{array}{c}
0 \\
1 \\
x \\
x+1
\end{array}
\]

Addition is component-wise according to powers, as usual

\[
(x) + (x+1) = (2x+1) = 1 \quad (\text{coefs, mod 2})
\]

Multiplication is modulo $x^2 + x + 1$

which is irreducible (doesn't factor)

\[
\begin{array}{cccc}
0 & 1 & x & x+1 \\
0 & 0 & 0 & 0 \\
0 & 1 & x & x+1 \\
0 & x & x+1 & 1 \\
0 & x+1 & 1 & x \\
\end{array}
\]

$x^2 \mod (x^2 + x + 1)$ is $x+1$ \text{(note that $x \equiv -x$ coefs mod 2)}