6.857 Computer and Network Security
Lecture 10

Admin:

- Problem Set #2 due
- Problem Set #3 out

Project Ideas:

- Where did Mt. Gox bitcoins go?
- Attack reduced round “Simon” or “Speck” with SAT solver?

Today:

- Message Authentication Codes
    - HMAC
    - CBC-MAC
    - PRF-MAC
    - One-time MAC
- Combined mode
    - AEAD (Authenticated encryption with associated data)
    - EAX mode (ref. pages 1-10 of paper only)
- Finite fields and number theory

TOPIC:

DATE:

FILE UNDER:

PAGE: L12.2

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

$$\text{Alice} \xrightarrow{M,\ MAC_K(M)} \text{Bob}$$

Alice: M, K

Bob: K

[Here M is message to be authenticated, which could be ciphertext resulting from encryption.]

Note if MAC has t bits, then Adv can forge with prob 2^{-t}, just by guessing. $t = 32$ might be ok in practice...

- Alice computes $MAC_K(M)$ & appends it to M.
- Bob recomputes $MAC_K(M)$ & verifies it agrees which what is received. If ≠, reject message.

Adversary (Eve) wants to forge $M', \mathrm{MAC}_K(M')$ pair that Bob accepts, without Eve knowing K.

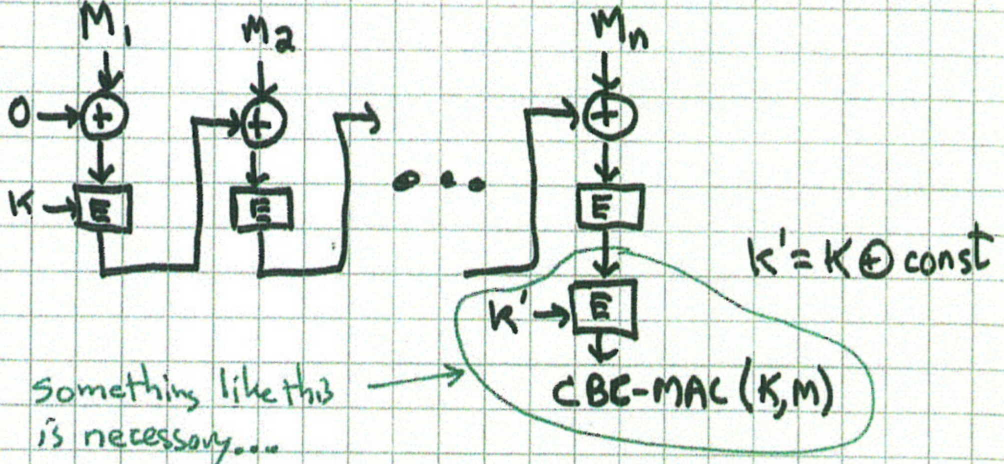
- She may hear a number of valid $(M, \mathrm{MAC}_K(M))$ pairs first, possibly even with M's of her choice (chosen msg attacks).
- She wants to forge for M' for which she hasn't seen $(M', \mathrm{MAC}_K(M'))$ valid pair.

Two common methods:

HMAC (K, M) $= h(K_1 \,\|\, h(K_2 \,\|\, M))$

where $K_1 = K \oplus \text{opad}$
$K_2 = K \oplus \text{ipad}$ $\Big\}$ opad, ipad are fixed constants

CBC-MAC (K, M) $\cong$ last block of CBC enc. of M



$K' = K \oplus \text{const}$

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 fns, that last block may need special treatment.)

One-Time MAC (problem stmt):

> Can we achieve security against <u>unbounded</u> Eve, as we did for confidentiality with OTP, except here for integrity?

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

$$\underset{K}{A} \xrightarrow{\quad M,T \quad} \underset{K}{B}$$

$$T = MAC_K(M) \qquad (\text{"tag"})$$

- Eve can learn M, T then try to replace M, T with M′, T′ (where $M' \neq M$) that Bob accepts.
- Eve is computationally <u>unbounded</u>.

TOPIC:

FILE UNDER:

DATE:

PAGE: L12.5

	Confidentiality	Integrity
Unconditional	OTP ✓	One-time MAC ?
Conventional (symmetric key)	Block ciphers (AES) ✓	MAC (HMAC) ✓
Public-key (asymmetric)	PK enc.	Digital signature

TOPIC: 6.857

DATE: 3/11/13

FILE UNDER:

PAGE:

EAX mode

[See pgs 1-10 of

The EAX Mode of Operation

by Bellare, Rogaway, & Wagner

]

TOPIC:	DATE:
FILE UNDER:	PAGE: L12.6

Finite fields:

System $(S, +, \cdot)$ s.t.

- S is a <u>finite</u> set containing "0" & "1"
- $(S, +)$ is an abelian (commutative) group with identity 0

group laws
$$\left[\begin{array}{ll} ((a+b)+c) = (a+(b+c)) & \text{associative} \\ a+0 = 0+a = a & \text{identity } 0 \\ (\forall a)(\exists b)\ a+b=0 & \text{(additive) inverses } b=-a \end{array}\right.$$

$a+b = b+a$ commutative

- $(S^*, \cdot)$ is an abelian group with identity 1

S^* = nonzero elements of S

group laws
$$\left[\begin{array}{ll} (a \cdot b) \cdot c = a \cdot (b \cdot c) & \text{associative} \\ a \cdot 1 = 1 \cdot a = a & \text{identity } 1 \\ (\forall a \in S^*)(\exists b \in S^*)\ a \cdot b = 1 & \text{(multiplicative inverses) } b = a^{-1} \end{array}\right.$$

$a \cdot b = b \cdot a$ commutative

- Distributive laws: $a \cdot (b+c) = a \cdot b + a \cdot c$

$(b+c) \cdot a = b \cdot a + c \cdot a$ (follows)

Familiar fields: $\mathbb{R}$ (reals) are <u>infinite</u>

$\mathbb{C}$ (complex)

For crypto, we're usually interested in <u>finite</u> fields,

such as $\mathbb{Z}_p$ (integers mod prime p)

Over field, usual algorithms work (mostly).

E.g. solving linear eqns:

$$ax + b = 0 \pmod{p}$$

$$\Rightarrow \quad x = a^{-1} \cdot (-b) \pmod{p} \quad \text{is soln.}$$

$$3x + 5 = 6 \pmod{7}$$

$$3x = 1 \pmod{7}$$

$$x = 5 \pmod{7}$$

TOPIC:

DATE:

FILE UNDER:

PAGE: L12.8

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

Z_p = integers mod p = $\{0, 1, \ldots, p-1\}$

$Z_p^* = Z_p - \{0\} = \{1, 2, \ldots, p-1\}$

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)

TOPIC:

DATE:

FILE UNDER:

PAGE:

Construction of $GF(2^2) = GF(4)$

Has 4 elements.

Is *not* arithmetic mod 4. (where 2 has no mult. inverse)

elements are polynomials of degree < 2 with coefficients mod 2 (i.e. in $GF(2)$):

0
1
x
$x+1$

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)

	0	1	x	$x+1$
0	0	0	0	0
1	0	1	x	$x+1$
x	0	x	$x+1$	1
$x+1$	0	$x+1$	1	x

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

"Repeated squaring" to compute a^b in field

(Here b is a non-negative integer)

$$a^b = \begin{cases} 1 & \text{if } b=0 \\ (a^{b/2})^2 & \text{if } b>0,\ b \text{ even} \\ a \cdot a^{b-1} & \text{if } b \text{ odd} \end{cases}$$

Requires ≤ 2 · lg(b) multiplications in field (efficient)

≈ a few milliseconds for $a^b \pmod p$ 1024-bit integers

≈ $\Theta(k^3)$ time for k-bit inputs

Computing (multiplicative) inverses:

Theorem: (For GF(p) called "Fermat's Little Theorem")

In GF(q) $(\forall a \in GF(q)^*)\ a^{q-1} = 1$

Corollary: $(\forall a \in GF(q))\ a^q = a$

Corollary: $(\forall a \in GF(q)^*)\ a^{-1} = a^{q-2}$

Example: $3^{-1} \pmod 7$

$= 3^5 \pmod 7$

$= 5 \pmod 7$

6.857 Network and Computer Security
Spring 2014