

CS574

Computer Security

San Diego State University

Spring 2008

Lecture #4

Today's Structure

- Administrivia
- Questions
- Recent News
- Lectures
- Crashers

Administrivia

- Crashing
 - Most add codes distributed by email
 - Some people didn't request add codes by email
 - Some people didn't provide transcripts
 - Open University / add codes at end of class
- Roll

Questions?

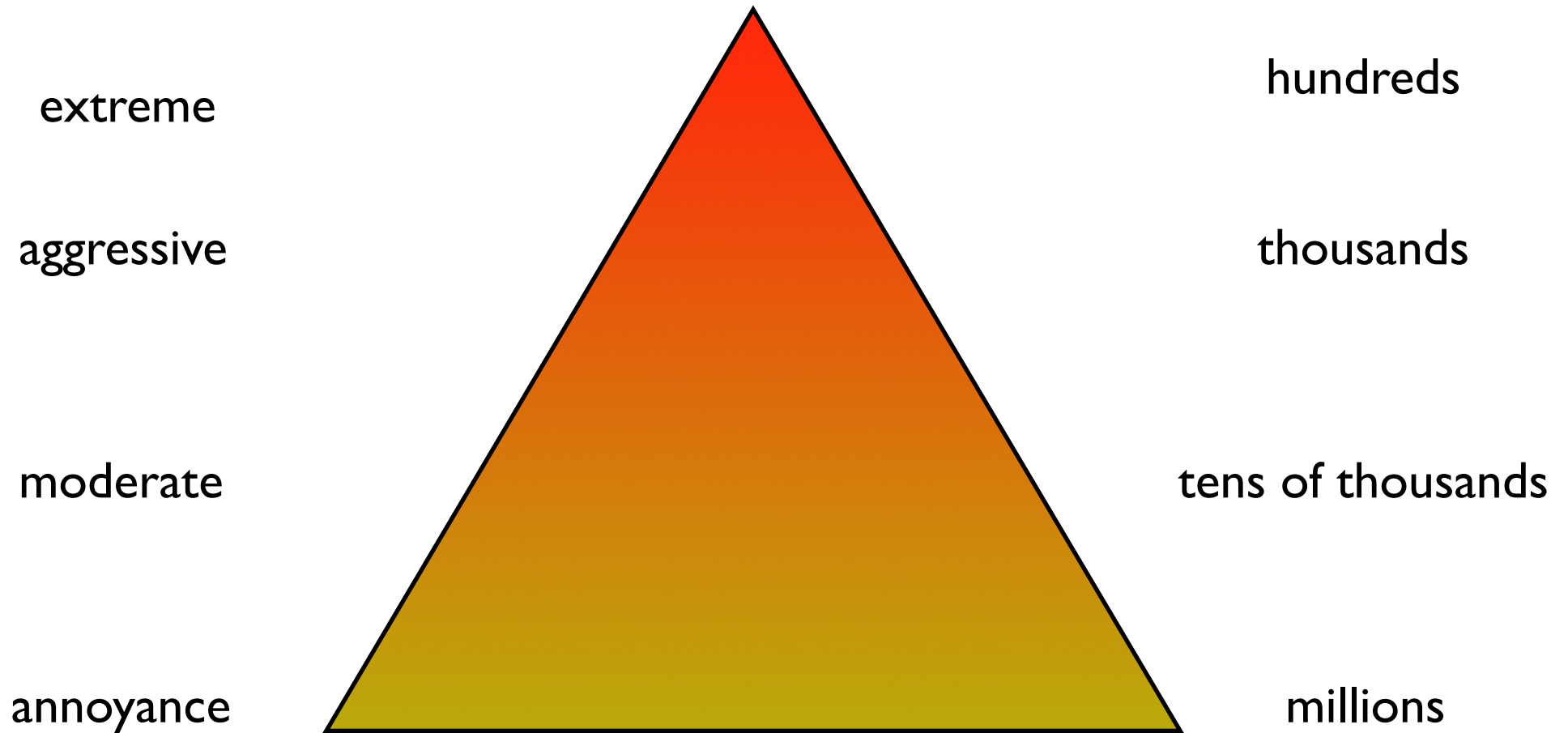
Recent News

- *CAPTCHAs*
- *DTrace*
- *Security vs. Privacy*

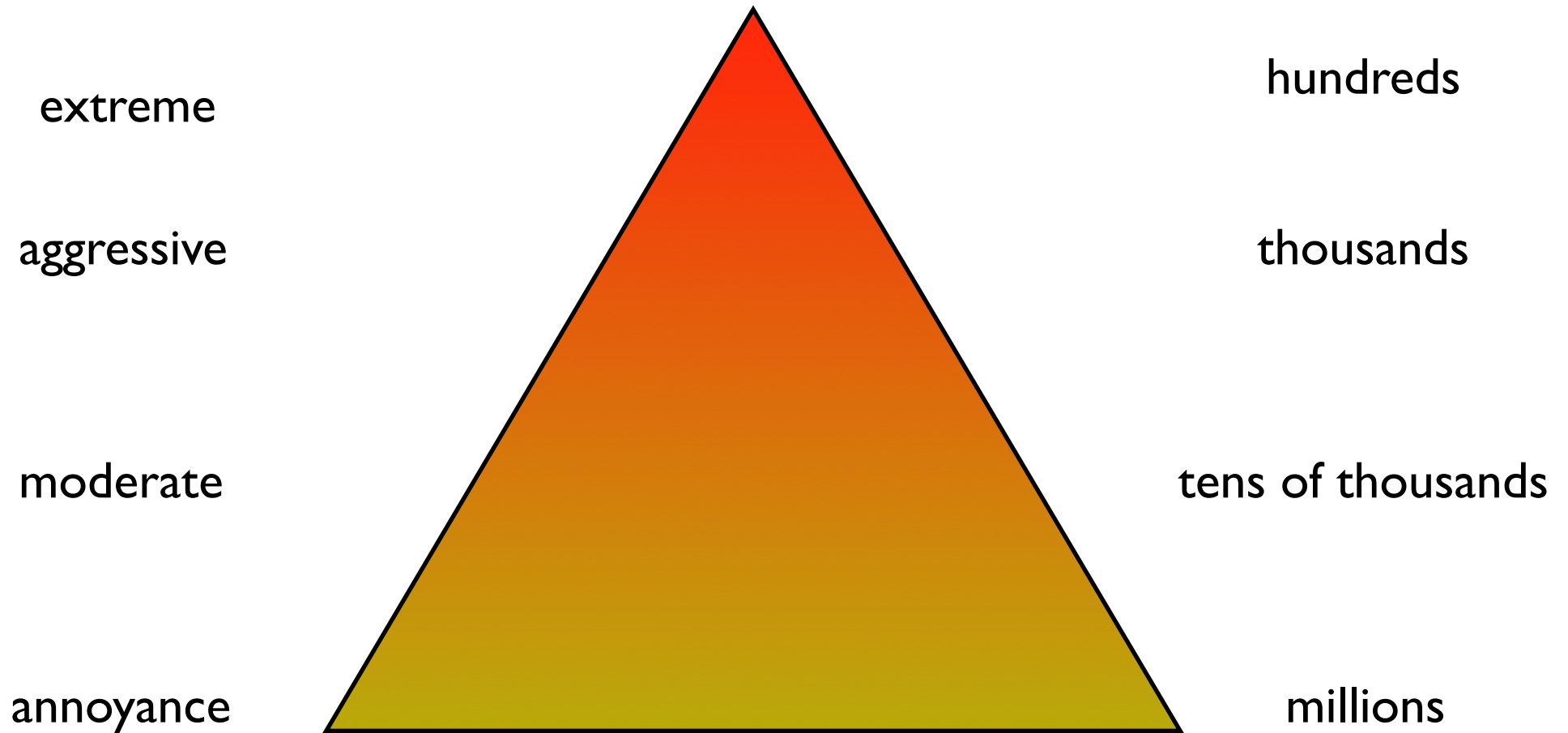
Lecture

Previous Topics

Threat Pyramid



Tom Perrine's Threat Pyramid



Tom Perrine, (*Unpublished*)

Information Theory

- Why bother with this?
- Entropy = disorder = randomness
- How does entropy measure information?
- I looked, but this isn't in my book! Wah!

Ciphers

Symmetric Ciphers

- private-key cryptosystems
- “classic” cryptosystems
- Same key used to encrypt and decrypt

“The security of a symmetric cryptosystem is a function of two things - the strength of the algorithm and the length of the key.”

Block and Stream

- Block ciphers operate on blocks all-at-once
 - same plaintext block encrypts to same ciphertext block
- Stream ciphers operate on bytes/words
 - same plaintext 'word' encrypts to different ciphertext 'word'
- Block ciphers seem more “general”, stream ciphers seem easier to analyze

Block / Stream

“Block ciphers operate with fixed transformations on large blocks of plaintext data. Stream ciphers operate with time-varying transformations on individual plaintext digits.”

-- Rainer Reuppel

Some Ciphers

- DES
- AES
- IDEA
- Skipjack

DES

DES

- DES = Data Encryption Standard
- Block cipher
- Based on IBM's Lucifer from the 1970s (with a smaller key and more rounds)
- Operates on 64-bit chunks
- “Designed” for hardware implementations
- Input from the NSA

DES Algorithm

- a product cipher
- 16 rounds
- splits block
- expand one half block and combine with subkey
- condense & confuse with S-boxes
- permute with P-boxes
- combine and swap

DES Psuedo-Code Example

Next Two Slides

```

f( block, key ):
    b48 <- ebox( block ) // expanding permutation
    b48 <- b48 xor key
    b32 <- sbox( b48 ) // substitution
    b32 <- pbox( b32 ) // permutation
    answer b32

end

key( r, key ):
    if r == 1 do
        remember key as K
        remove_parity( K ) // 64 bits -> 56 bits
    fi

    subkeyL <- shift( left( K ), shiftby( r ) )
    subkeyR <- shift( right( K ), shiftby( r ) )

    K = join( subkeyL, subkeyR )
    answer permuted_choice( K, r ) // 48 bits

end

```

```
des( plaintext, K ):
  block <- permute( plaintext )
  L <- left( block )
  R <- right( block )

  for round = 1 to 16 do
    Rn <- f( R, key( round, K ) )
    Ln <- L xor Rn
    R <- Ln
    L <- Rn
  rof

  block <- join( L, R )
  answer unpermute( block )
end
```

DES Questions

There were lot of questions at the time...

- Why did the NSA change the S-boxes?
- Why did the NSA reduce the key size?
- Why did the NSA increase the number of rounds?
(8 were considered sufficient for diffusion)
- Did the NSA put in “trapdoors”?

DES Weaknesses

- 64 “weak”, “semi-weak”, or “potentially weak” keys.
- Chosen plaintext only needs to test 2^{55} keys.
- Differential Cryptanalysis...
- Linear Cryptanalysis...

Differential Cryptanalysis

- (re)invented in 1990 by Eli Biham and Adi Shamir
- looks at differences in ciphertext/plaintext pairs
- lots of results against lots of algorithms
- Lucifer falls in 30 ciphertext pairs
- DES weak when reduced to 8 rounds
- DES better than exhaustive search with 16 rounds - appears optimized against DC.

Linear Cryptanalysis

- Invented by Mitsuru Matsui
- Collect plaintexts and the associated ciphertexts to get a guess at the key.
- Dependent on the structure of S-Boxes.
- DES S-boxes *not* optimized against linear cryptanalysis.

Multiple DES

- Single-DES (brute force checks 2^{56} keys)

$$C = E(k_1, P)$$

- Double-DES (brute force checks 2^{57} keys)

$$C = E(k_1, E(k_2, P))$$

- Triple-DES (brute force checks 2^{112} keys)

$$C = E(k_1, E(k_2, E(k_3, P)))$$

$$C = E(k_1, D(k_2, E(k_1, P)))$$

AES

AES

- AES = Advanced Encryption Standard
- Block cipher like DES
- Successor to DES
- Open Competition
(Rijndael, Serpent, Twofish, RC6, MARS)
- Rijndael won - selected October 2000

AES Features

- 16 byte (128 bits) block length*
- Variable key length
- Number of rounds vary with key sizes
- Round keys derived from master keys and not reused in other rounds
- A product cipher

**Rijndael can vary block sizes*

AES Description

- Multiple rounds with lots of table lookups
- Mixing
 - byte substitution
 - shift row
- Mix columns
- XOR with round key

AES Psuedocode

```
// from descriptions, not source, not reliable
aes( block, key ):
  b <- block
  rounds <- get_round_count( key )
  for r = 1 to rounds - 1 do
    k <- round_key( r, key )
    b <- sbox( b , r )
    b <- shift_rows( b, r )
    b <- mix_columns( b, r )
    b <- b xor k
  rof
  b <- sbox( b, rounds )
  b <- shift_rows( b, rounds )
  b <- b xor round_key( rounds, key )
  answer b
end
```

AES Details

- Designed to be resistant to all known attacks, including linear & differential cryptanalysis
- High diffusion and non-linear diffusion
- Designed for hardware implementation (that is, for 8-bit CPUs on up, and for >32-bit CPUs, each round is a single table lookup)
- Algorithm is *not* symmetrical

AES Questions

- Are there any trapdoors?
- Open design, selection, and review
- Theoretical attacks, but not officially broken just yet

IDEA

IDEA

- IDEA = International Data Encryption Algorithm
- Block cipher like DES and AES
- PES → IPES → IDEA
- Also a product cipher
- When implemented in software, is approximately twice as fast as DES

IDEA Features

- 64-bit blocks
- 128-bit key
- symmetric algorithm
- 8 rounds
- confusion and diffusion
- operations:
 - XOR
 - multiplication modulo $2^{16} + 1$
 - addition modulo 2^{16}

IDEA Overview

- 64-bit data-block divided into 4 16-bit sub-blocks
- multiply or add sub-blocks with sub-keys
- xor alternating sub-blocks
- multiply with sub-key and add
- several additional XORs

IDEA Psuedo-code?

Next three slides

```
do_round( r, key, x ):
    k[1,2,3,4,5,6] <- derive_key( key, r )

    a[1] <- x[1] * k[1] mod ( 2^16 + 1 )
    a[2] <- x[2] + k[2] mod ( 2^16 )
    a[3] <- x[3] + k[3] mod ( 2^16 )
    a[4] <- x[4] * k[4] mod ( 2^16 + 1 )

    a[5] <- a[1] xor a[3]
    a[6] <- a[2] xor a[4]

    a[7] <- a[5] * k[5] mod ( 2^16 + 1 )
    a[8] <- a[6] + a[7] mod ( 2^16 )
    a[9] <- a[8] * k[6] mod ( 2^16 + 1 )
    a[10] <- a[7] + a[9] mod ( 2^16 )
```

continued...

...continued

```
a[11] <- a[1] xor a[9]
a[12] <- a[3] xor a[9]
a[13] <- a[2] xor a[10]
a[14] <- a[4] xor a[10]
```

```
if ( r < 8 ) do
  x[1] <- a[11]
  x[2] <- a[13]
  x[3] <- a[12]
  x[4] <- a[14]
```

```
else
```

```
  x[1] <- a[11]
  x[2] <- x[12]
  x[3] <- x[13]
  x[4] <- x[14]
```

```
fi
```

```
end
```

```
idea( block, key ):  
  x[1,2,3,4] <- split( block )  
  
  for r = 1 to 8 do  
    do_round( r, key, x )  
  rof  
  
  k[1,2,3,4] <- derive_key( key )  
  
  x[1] <- x[1] * k[1] mod ( 2^16 + 1 )  
  x[2] <- x[2] + k[2] mod ( 2^16 )  
  x[3] <- x[3] + k[3] mod ( 2^16 )  
  x[4] <- x[4] * k[4] mod ( 2^16 + 1 )  
  
  answer join( x[1] , x[2], x[3], x[4] )  
end
```

IDEA Cryptanalysis

- Apparently immune to differential cryptanalysis
- No known linear cryptanalysis attack
- A class of weak-keys for chosen-plaintext (1 in 2^{96} chance)
- Some progress for
 - attacks on the key schedule
 - for reduce-round versions

Skipjack

Clipper / Skipjack

- The Clipper Chip
- US Government (NSA) designed
- No software implementation
- Uses Key Escrow

What is Key Escrow?

- Allows for recovery of plaintext by legal means
- Session keys encrypted by a master key
- Master key split into (at least) 2 pieces, each piece held by a different agency

Clipper Chip / Skipjack Algorithm

- Designed prior to 1995
- Partial details published in 1999/2000
 - e.g., 32 rounds
- Not Revealed:
 - anti-tampering technology for chip
 - LEAF details
 - details of algorithm analysis

LEAF

- LEAF = Law Enforcement Access Field
- 80-bit unit key “U”
- 80-bit family key “F”
- 80-bit session key “K”
- Details are classified

LEAF Use

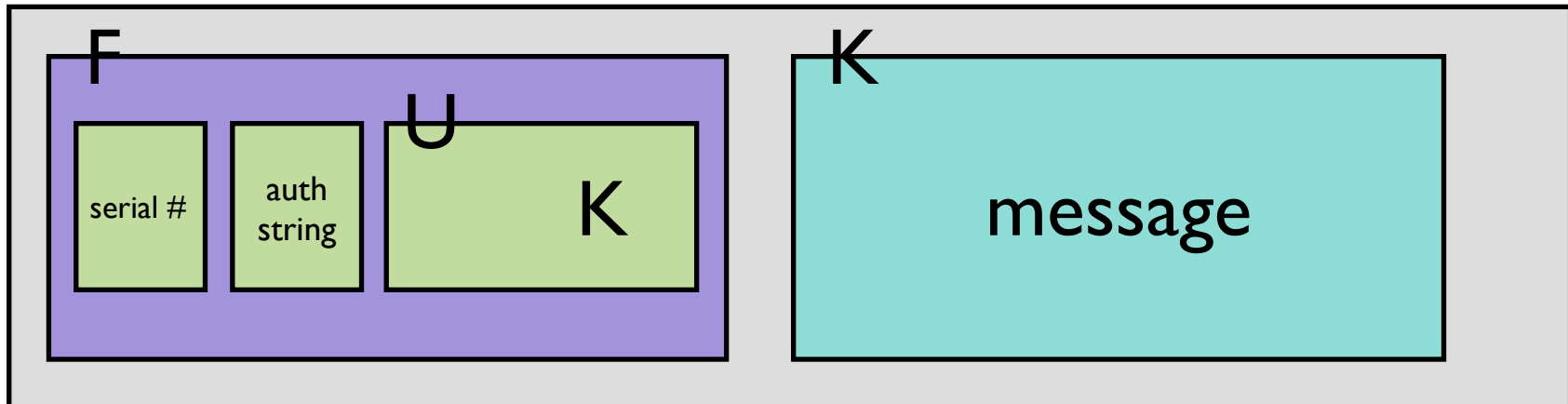
- LEAF sent with message. Encrypt with F:
 - session key K encrypted with unit key U
 - serial # of the chip
 - authentication string
- To get session key K :
 - law enforcement obtains a warrant
 - decrypt LEAF with F
 - present warrant + serial # to escrow agencies
 - combine pieces of U and decrypt to get K
- With K , can decrypt message

A Pretty Picture

K is the session key used to encrypt the message.

U is the unit key used to encrypt the session key.

F is the family key used to encrypt the LEAF



Clipper Controversy

- Matt Blaze of AT&T showed that you could tweak LEAF so that law enforcement couldn't tell where the message originally came from (save & reuse the session key)
- Can create bogus LEAF (2^{16} checksums)
- Can forge message if have plaintext and ciphertext: XOR to get the keystream, encrypt new message with stream, and send through CLIPPER.

Clipper Controversy

- Key escrow agencies obvious targets.
- Chip factory a reasonable target.
- Too few key escrow agencies.
- Keys on clipper chips not generated in a secure fashion.
- Insufficient competition among implementors.
- Software implementation “not possible”
- Key size is fixed.

Key Escrow

“The people involved in the crypto debate are all intelligent, honorable, and pro-escrow, but they never possess more than two of these qualities at once.”

-- Kenneth Neil Cukier

Lecture References

- Pfleeger, *Security in Computing*, 3rd & 4th edition
- Bishop, *Introduction to Computer Security*
- Schneier, *Applied Cryptography*
- Schneier, *Practical Cryptography*
- Schneier, *The Crypto-Gram Newsletter*, Sep. 15, 2002
- Singh, *The Code Book*
- RSA Laboratories, *RSA Crypto FAQ*
- Gutman, *Lessons Learned in Implementing and Deploying Crypto Software*
- Rothenburg, et al., *The Snake-Oil FAQ*
- *They Cryptography FAQ*

End Lecture

Reading

(For the next few classes)

- Pfleeger, Chapter 2
- Pfleeger, Chapter 12
- Next time:
 - block cipher modes
 - the key management problem
 - public key cryptography

(Keep Reading)

Crashers

- Do I owe you an add code?
- Do you owe me a transcript?
- Do I need to sign your open university form?

Finis