Foundations of Network and Computer Security

John Black

Lecture #3
Aug 30st 2005

CSCI 6268/TLEN 5831, Fall 2005

Assignment #0

- Please add yourself to the class mailing list
  - Send mail to listproc@lists.colorado.edu
  - Subject is ignored
  - In body of message write

    "subscribe CSCI-6268 *Your Name*"
- Due by September 6th (Tuesday)

Review

- Summing up last lecture on blockciphers:
  - Blockciphers have a fixed-size input
    - Called “blocksize”
  - Blockciphers have a fixed-size key
    - Called the “keysize”
  - Small keysize bad (exhaustive search)
  - Small blocksize bad (frequency analysis)

Example Blockcipher

- Suppose we have 64-bit blocksize
- Suppose we have 64-bit keys
  - Notice this is **FAR** smaller than 2^{70}-bit keys, so we will be representing a *vastly* smaller set of permutations
  - Select a key K at random from $\{0,1\}^{64}$
    - $\{0,1\}^{64}$ is the set of all length-64 binary strings
- Let $C = P \oplus K$
  - Here $\oplus$ means XOR

Digression on Terminology

- Note that we used specific letters in our formula $C = P \oplus K$
  - P is the “plaintext”
  - C is the “ciphertext”
  - K is usually used for “key”
- Call this blockcipher X
  - $X : \{0,1\}^{64} \times \{0,1\}^{64} \rightarrow \{0,1\}^{64}$
  - This means E takes two 64-bit strings and produces a 64-bit output

Looking at Blockcipher X

- First, is it even a valid cipher?
  - Is it 1-to-1?
    - Basic facts on xor's:
      - $A \oplus A = 0$ $\quad$ $A \oplus B = B \oplus A$
      - $A \oplus 0 = A$ $\quad$ $A \oplus (B \oplus C) = (A \oplus B) \oplus C$
    - So prove 1-to-1:
      - Suppose $P \neq P'$ but $C = C$
      - Then $P \oplus K = P' \oplus K$
      - so $P \oplus P' = K \oplus K$
      - and $P \oplus P' = 0$
      - so $P = P'$, contradiction

So it's *Syntactically* Valid

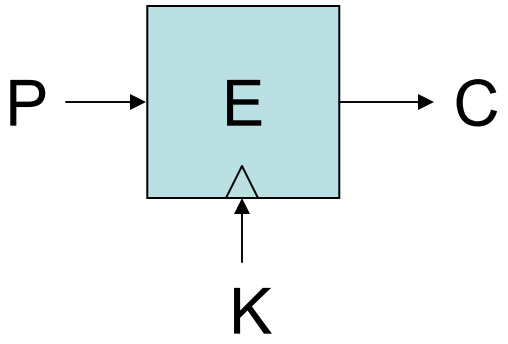
- What about its security?
  - It's terrible, but before we can really look more closely at it we need to learn more about what "secure" means
  - A second problem is that we still haven't said how to "encrypt," only to "encipher"
    - Encryption handles a bunch of variable-length messages
    - Enciphering handles inputs of one fixed size; ergo the term "blockcipher"

Background

- So really we've been talking about things like encryption and security without proper definitions!
  - Although it may be a pain, definitions are a central (and often ignored) part of doing "science"
  - You will see textbooks teach cryptography without defining the terms they use
  - We have an intuitive sense of these things, but we can't do science without writing down precise meanings for the terms we're using
  - The network security part of the course won't be much like this

Blockciphers

- One of the most basic components
  - Used EVERYWHERE in cryptography
  - Blockcipher E maps a k-bit key K and an n-bit plaintext P to an n-bit ciphertext C
  - Requirement: for any fixed K, E(K, ·) is a permutation (ie, is 1-to-1)



Security

- Intuition:
  - A “secure” blockcipher under a (uniformly-chosen) random key should “look random”
- More precisely (but still informal):
  - Suppose you are given a black-box which contains blockcipher E with a secret, random, fixed key K embedded within it
  - Suppose you are also given another black-box (looks identical) which has a permutation π from n-bits to n-bits embedded within it, and π was chosen uniformly at random from the set of all $2^n!$ possible permutations
  - You are allowed to submit arbitrary plaintexts and ciphertexts of your choice to either box
  - Could you tell which was which using a “reasonable” amount of computation?

Blockcipher Security (cont.)

- A “good” blockcipher requires that, on average, you must use a TON of computational resources to distinguish these two black-boxes from one another
  - A good blockcipher is therefore called “computationally indistinguishable” from a random permutation
  - If we had 2^{70}-bit keys, we could have *perfect* 64-bit blockciphers
  - Since we are implementing only a small fraction, we had better try and ensure there is no computationally-simple way to recognize this subset

Blockcipher Security (cont.)

- If we can distinguish between black-boxes quickly, we say there is a “distinguishing attack”
  - Practical uses?
  - Notice that we might succeed here even without getting the key!
    - Certainly getting the key is sufficient since we assume we know the underlying algorithm
    - What is the attack if we know the key?

Theme to Note

- Note that our notion of security asks for MORE than we often need in practice
  - This is a common theme in cryptography: if it is reasonable and seemingly achievable to efficiently get more than you might need in practice, then require that your algorithms meet these higher requirements.

Our Blockcipher X

- So is X secure under this definition?
  - No, simple distinguishing attack:
    - Select one black-box arbitrarily (doesn't matter which one)
    - Submit plaintext $P=0^{64}$ receiving ciphertext C
    - Submit plaintext $P'=1^{64}$ receiving ciphertext C'
    - If black-box is our friend X (under key K) then we will have
      - $C = K$ and $C' = K \oplus 1^{64}$
      - So if $C \oplus C' = 1^{64}$ we guess that this box is blockcipher X
      - If not, we guess that this box is the random permutation

Analysis of X (cont.)

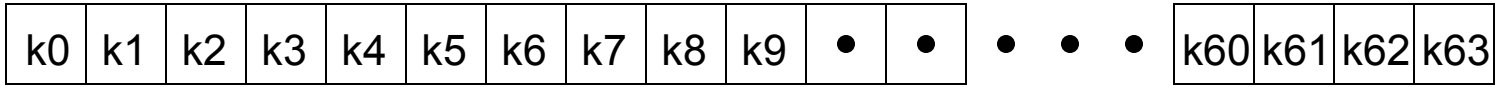
- What is the probability that we guess wrong?
  - Ie, what is the chance that two random distinct 64-bit strings are 1's complements of each other?
  - $1/(2^{64}-1)$ … about 1 in 10^{20}
- Note that this method does not depend on the key K

Let's build a Better Blockcipher

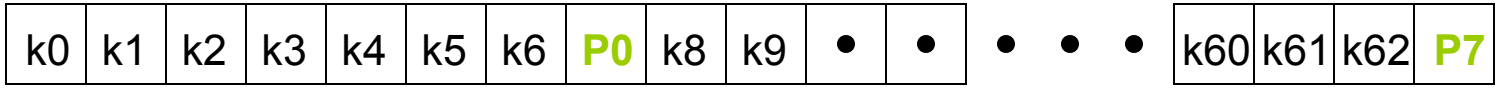
- DES – The Data Encryption Standard
  - 64-bit blocksize, 56 bit key
  - Formerly called "Lucifer"
    - Developed by Horst Feistel at IBM in early 70's
  - Tweaked by the NSA
    - No explanation given for tweaks
    - Some people worried that NSA was adding backdoors/weaknesses to allow it to be cracked!
    - NSA shortened key from 64 bits to 56 bits (definite added weakness)
  - Adopted by NIST (then called NBS) as a Federal Information Processing Standard (FIPS 46-3)
    - NIST is retiring it as a standard this year after nearly 30 years

The DES Key

- Was 64 bits



- But NSA added 8 parity bits



- Key is effectively only 56 bits!

Exhaustive Key Search -- DES

- This meant that instead of 2^{64} keys there were only 2^{56} keys
  - Expected number of keys to search before finding correct value is 2^{55}
    - Note that we need a handful of plaintext-ciphertext pairs to test candidate keys
  - NSA surely could do this in a reasonable amount of time, even in the 70's

Exhaustive Key Search -- DES

- In 1994, Michael Wiener showed that you could build a DES-cracking machine for $1,000,000 that would find the key in an expected 3.5 hours
  - In 1998 he revised this to 35 minutes for the same cost
  - In 1997, Rocke Verser used 10,000+ PCs to solve DES Challenge I to win $10,000 (Loveland, CO!)
  - distributed.net solved the DES Challenge II in 41 days with 50,000 processors covering 85% of the keyspace
  - Later the same year the EFF built the DES Cracker machine which found the same key in 56 hours
    - $210,000 for the machine
    - 92 billion key trials per second

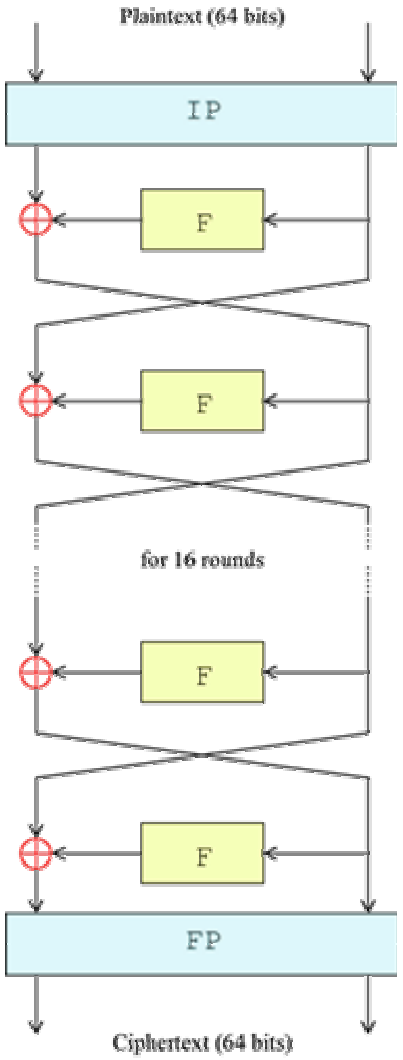
No Better Attack has Ever Been Found against DES

- This is saying something:
  - Despite lots of cryptanalysis, exhaustive key search is still the best known attack!

- Let's have a look at (roughly) how DES works and see in what ways it's still in use

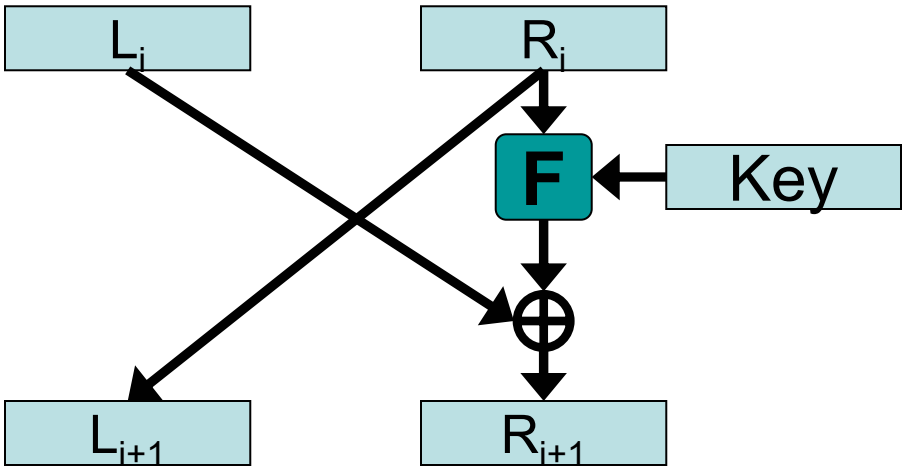
DES -- Feistel Construction

- IP – Initial permutation swaps bits around for hardware purposes
  - Adds no cryptographic strength; same for FP
- Each inner application of F and the XOR is called a “round”
- F is called the “round function”
  - The cryptographic strength of DES lies in F
- DES uses 16 rounds



One Round

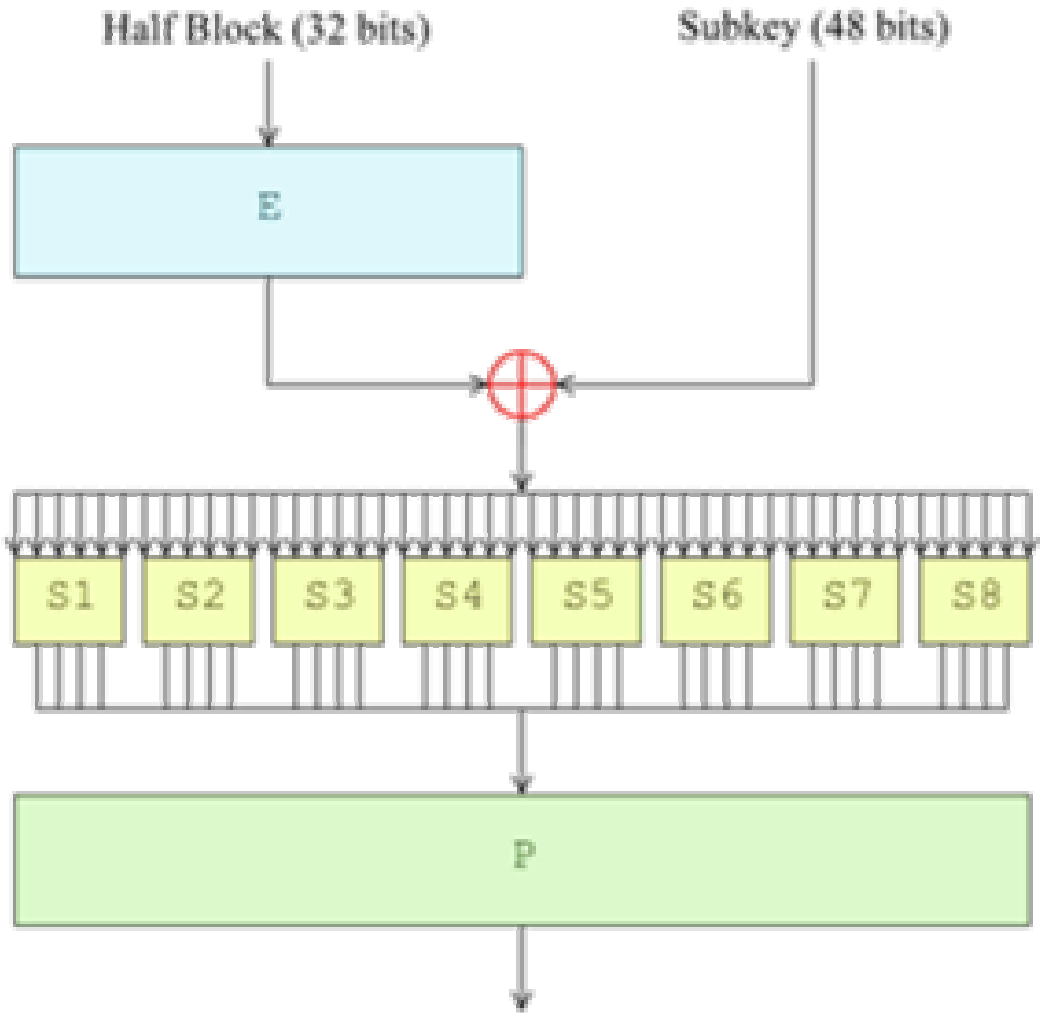
- Each half is 32 bits
- Round key is 48 bits
- Is this a permutation (as required)?
    - How do we invert?
- Note that F need not be invertible with the round key fixed



Why so many Rounds?

- Can we just have one round of Feistel?
  - Clearly this is insecure
- How about two rounds?
  - Expect to be asked a related question on the first quiz
- DES has 16 rounds
  - It's easily broken with 8 rounds using differential cryptanalysis

The DES Round Function



DES Round Function (cont)

- F takes two inputs
  - 32 bit round value
  - 48 bits of key taken from 56 bit DES key
    - A different subset of 48 bits selected in each round
  - E is the “expansion” box
    - Turns each set of 4 bits into 6, by merely repeating some bits
  - S boxes take 6 bits back to 4 bits
    - Non-linear functions and they are the cryptographic heart of DES
    - S-boxes were tweaked by NSA back in the 70’s
    - It is believed that they IMPROVED DES by doing this

Full Description of DES

- If you want all the gory details

  http://en.wikipedia.org/wiki/DES

- Challenge Problem:
  - Alter the S-boxes of DES any way you like so that with ONE plaintext-ciphertext pair you can recover all 56 key bits
  - (Warning: you need some linear algebra here)

So if not DES, then what?

- Double DES?
- Let's write DES(K, P) as $DES_K(P)$
- Double DES (DDES) is a 64-bit blockcipher with a 112 bit key K = (K1, K2) and is

$$DDES_K(P) = DES_{K2}(DES_{K1}(P))$$

- We know 112 bits is out of exhaustive search range… are we now secure?

Meet in the Middle Attack

- With enough memory, DDES isn't much better than single DES!
- Attack (assume we have a handful of pt-ct pairs P1,C1; P2, C2; …)
  - Encipher P1 under all 2^{56} possible keys and store the ciphertexts in a hash table
  - Decipher C1 under all 2^{56} possible keys and look for a match
  - Any match gives a candidate 112-bit DDES key
  - Use P2, C2 and more pairs to validate candidate DDES key until found

Meet in the Middle (cont)

- Complexity
  - $2^{56} + 2^{56} = 2^{57}$ DES operations
  - Not much better than the 2^{55} expected DES operations for exhaustive search!
  - Memory requirements are quite high, but there are techniques to reduce them at only a slightly higher cost
  - End result: no one uses DDES

How about Triple-DES!

- Triple DES uses a 168-bit key K=(K1, K2, K3)

$$TDES_K(P) = DES_{K3}(DES_{K2}(DES_{K1}(P)))$$

- No known attacks against TDES
  - Provides 112-bits of security against key-search
  - Widely used, standardized, etc
  - More often used in “two-key triple-DES” mode with EDE format (K is 112 bits like DDES):

$$TDES_K(P) = DES_{K1}(DES^{-1}_{K2}(DES_{K1}(P)))$$

  - Why is the middle operation a decipherment?

AES – The Advanced Encryption Standard

- If TDES is secure, why do we need something else?
  - DES was slow
  - DES times 3 is three times slower
  - 64-bit blocksize could be bigger without adding much cost
  - DES had other annoying weakness which were inherited by TDES
  - We know a lot more about blockcipher design, so time to make something really cool!

AES Competition

- NIST sponsored a competition
  - Individuals and groups submitted entries
    - Goals: fast, portable, secure, constrained environments, elegant, hardware-friendly, patent-free, thoroughly analyzed, etc
  - Five finalists selected (Aug 1999)
    - Rijndael (Belgium), MARS (IBM), Serpent (Israel), TwoFish (Counterpane), RC6 (RSA, Inc)
  - Rijndael selected (Dec 2001)
    - Designed by two Belgians

AES – Rijndael

- Not a Feistel construction!
  - 128 bit blocksize
  - 128, 192, 256-bit keysize
  - SP network
    - Series of invertible (non-linear) substitutions and permutations
  - Much faster than DES
    - About 300 cycles on a Pentium III
  - A somewhat risky choice for NIST

Security of the AES

- Some close calls last year (XL attack)
  - Can be represented as an overdetermined set of very sparse equations
  - Computer-methods of solving these systems would yield the key
  - Turns out there are fewer equations than previously thought
  - Seems like nothing to worry about yet

Block Ciphers – Conclusion

- There are a bunch out there besides AES and DES
  - Some are pretty good (IDEA, TwoFish, etc)
  - Some are pretty lousy
    - LOKI, FEAL, TEA, Magenta, Bass-O-Matic
- If you try and design your own, it will probably be really really bad
  - Plenty of examples, yet it still keeps happening