

Foundations of Network and Computer Security

John Black

Lecture #2
Aug 25th 2005

CSCI 6268/TLEN 5831, Fall 2005

Economist Survey

- Please read it
- Main points
 - Security is a MUCH broader topic than just SSL and viruses
 - Firewalls don't always work
 - Economics are a factor
 - And more...

Economist.com

What IS Computer Security?

- Cryptography
 - Mostly based in mathematics
- Network Services
 - Offense: Overflows, SQL injection, format strings, etc
 - Defense: Firewalls, IDSes, Sandboxing, Honeypots
- Software Engineering
 - You have to find all flaws, they only have to find one
- Policy
 - Laws affect profoundly our security and privacy, as we have already seen

What IS Computer Security?

- Soft Science
 - Trust Models (Bell-LaPadula, Insider Threat, etc)
 - Economics, Game Theory
 - Social Engineering
- Education
 - Students become our programmers
 - Insufficient training in security issues
- Various
 - Credit Card Scanners
 - Should you trust your CC# on the Internet?
 - ATM story

Cryptography

- Introduction to cryptography
 - Why?
 - We're doing things bottom-up
 - Crypto is a fundamental building block for securing networks, but by NO MEANS a panacea
 - Often done well
 - Breaking the crypto is often not the easiest way in
 - Instead exploit some of those other holes!
 - Long history
 - Based on lots of math

In the Beginning...

- “Classical” cryptography
 - Caesar cipher aka shift cipher
 - $A \rightarrow Z, B \rightarrow A, C \rightarrow B, \text{ etc...}$
 - We are shifting by -1 or, equivalently, by 25
 - Here the “domain” is A...Z and shifts are done modulo 26
 - Ex: What happens to “IBM” with a shift of 25?

Kerckhoff

– Kerckhoff's principal

- Assume algorithm is public and all security rests with the “key”
- The opposite philosophy is sometimes called “security thru obscurity”
 - Often dubious
 - » Experts say you *want* people to see the algorithm... the more analysis it sees, the better!
 - Used in military settings however
 - » Why give them *any* information??
 - » Skipjack was this way

What is a “key”?

- We have a basic enciphering mechanism
 - We just saw the Caesar cipher
- The “key” is the “variable” part of the algorithm
 - What was our key with the Caesar cipher?
 - How many keys were possible?
 - Why is this cipher insecure?

Digression on Blocksize

- The “blocksize” of a cipher is the size, in bits, of its domain
 - Caesar took 26 inputs, so about 4.7 bit blocksize
 - We take $\lg(|D|)$ to compute blocksize
 - Often it’s already specified in bits
 - Keysize is analogous
 - What was the keysize of the Caesar cipher?
- A “blockcipher” always outputs the same number of bits as it takes in
 - Ciphers induce a “permutation” on their domain
 - This means they are 1-to-1
 - Without this, we couldn’t decipher!

Improving Caesar

- Substitution Cipher
 - We allow each $\{A, \dots, Z\}$ to map to any other character in this set
 - Ex: $A \rightarrow Q, B \rightarrow S, C \rightarrow A$, etc...
 - We must still ensure a 1-to-1 mapping!
 - How many mappings are possible?
 - What is the key here?
 - What is the keysize?
 - Is exhaustive key-search feasible here?

Exhaustive Key-search

- We had 403291461126605635584000000 possible keys
 - Keysize is \lg of this, or about 88.4 bits
 - Infeasible to exhaustively search even with a lot of money and resources!
- Rule of Thumb
 - 2^{30} quite easily
 - 2^{40} takes a while, but doable (exportable keysize!)
 - 2^{50} special hardware, parallelism important
 - 2^{60} only large government organizations
 - 2^{70} approaching the (current) limits of imagination

So Substitution Cipher is Secure?

- Nope
 - Ever do the Sunday Cryptograms?
 - Attacks:
 - Frequency analysis
 - etaoinsrhdlu...
 - Diphthongs, triphthongs
 - ST, TH, not QX
 - Word lengths
 - A and I are only 1-letter words
 - Other statistical measures
 - Index of Coincidence

What did we just Implicitly Assume?

- What assumption was made in these attacks?
- What was a central feature of the Substitution Cipher which permitted these attacks? (hard)
- How can we repair these problems?

Small Blocksizes are Bad

- Ok, we had a blocksize of < 5 bits
 - So fix it!
 - Try 64 bits instead
 - All is well?
 - How many permutations are there now?
 - $2^{64}! \approx 2^{2^{70}}$
 - Stirling's formula: $n! \approx (n/e)^n \sqrt{2\pi n}$
 - What is the keysize (in bits)?
 - About 2^{70} bits! Yow!
 - 64 GB is $2^6 * 2^{30} * 2^3 = 2^{39}$

Key is too Large

- We can't store 2^{70} bit keys
 - What can we do then?
 - Idea: instead of representing ALL $2^{64}!$ permutations we select a “random looking” subset of them!
 - We will implement the map via an algorithm
 - Our subset will be MUCH smaller than the set of all permutations

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:
 - $A \oplus A = 0$ $A \oplus B = B \oplus A$
 - $A \oplus 0 = A$ $A \oplus (B \oplus C) = (A \oplus B) \oplus C$
 - 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

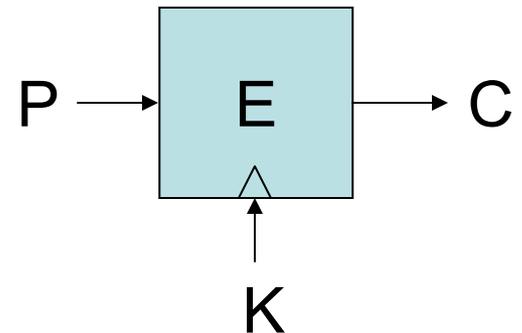
- What about its security?
 - It's terrible, but before we can really look more closely at it we need to learn more about what "secure" means
 - A second problem is that we still haven't said how to "encrypt," only to "encipher"
 - Encryption handles a bunch of variable-length messages
 - Enciphering 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 , $E(K, \cdot)$ 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?
 - I.e., 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