ELEC5616 COMPUTER & NETWORK SECURITY

Lecture 14:
Cryptographic Protocols II

CRYPTO BULLETIN

- **US goes on offensive with new cyber security policy**
  http://www.itnews.com.au/News/403232,us-goes-on-offensive-with-new-cyber-security-policy.aspx
  http://www.defense.gov/home/features/2015/0415_cyber-strategy/

- **China Censors Facebook.net, Blocks Sites With “Like” Buttons**
  http://krebsonsecurity.com/2015/04/china-censors-facebook-net-blocks-sites-with-like-buttons/

- **Microsoft Windows MS15-034 – Critical**
  Vulnerability in HTTP.sys Could Allow Remote Code Execution (3042553)
  https://technet.microsoft.com/en-us/library/security/ms15-034.aspx

SECRET SPLITTING

Problem:

You are the CEO of Coca-Cola. You’re responsible for keeping the formula secret from Pepsi’s industrial spies.

You could tell your most trusted employees, but...

– They could defect to the opposition

– They could fall to rubber hose cryptanalysis

How can we split a secret among two or more parties where each piece by itself is useless?

SECRET SPLITTING

Simple XOR Algorithm:

Assume Trent wishes to protect message **m**:

1. Trent generates a random bit string **r**, the same length **m**
2. Trent computes **s** = **m** $\oplus$ **r**
3. Trent gives Alice **r**
4. Trent gives Bob **s**

Each piece is called a *shadow*

To reconstruct m, Alice and Bob XOR their shadows together

If **r** is truly random, the system is perfectly secure (OTP)

To extend the scheme to **n** people, generate **n** random bit strings
(e.g. **m** $\oplus$ **r** $\oplus$ **s** $\oplus$ **t** = **u**)

SECRET SPLITTING

Secret splitting aims to enhance reliability without increasing risk through distributing trust

Issues:

The system is adjudicated by Trent

Trent can hand out rubbish and say it's part of the secret

He can hand out a piece to Alice, Bob, Carol and Dave, and later tell everyone that only the first three pieces are needed and Dave is fired

All parties know the length of the message

It's the same length as their piece of message

The message is malleable

Alice can manipulate her shadow to "blind" it or alter bits in a known way (like flipping)

All parties are required to recover message (bus factor = 1)

SECRET SHARING

Problem:

You are responsible for a small third-world country’s nuclear weapons program.

You want to ensure that no single lunatic can launch a missile.

You want to ensure that no two lunatics can collude to launch a missile.

You want at least three of five officers to be lunatics before a missile can be launched
(bus factor = 3)

We call this a **(3,5)-threshold scheme**

SHAMIR’S [T,N]-THRESHOLD SCHEME

Based on polynomial interpolation, and the fact that a polynomial y=f(**x**) of degree **t-1** is uniquely defined by **t** points (x,y)

Trent wishes to distribute message **m** amongst **n** users, where any group of **t** users can recover **m**

(bus factor = **n**-**t**+1)

<u>**Setup**</u>

– **Trent chooses a prime p > max(m, n)**

– **Trent sets a_0 = m**

– **Trent selects t-1 random, independent coefficients**

$(a_1 \ldots a_{t-1}$ $(0 \leq a_j \leq p-1)$, defining the polynomial $f(x) = \Sigma_{j=0}^{t-1} a_j x_j$)

– **Trent computes $y_i = f(x_i) \bmod p$ $(1 \leq x_i \leq p-1)$**

(just any random points on the curve)

– **Trent sends share (x_i,y_i) to user i**

SHAMIR’S [T,N]-THRESHOLD SCHEME

Pooling of Shares:

Any t users can get together and pool their distinct points

– Each party’s ($\mathbf{x_i}$, $\mathbf{y_i}$)

Since any **t** points are enough to define the polynomial, the coefficients $\mathbf{a_i}$ can be computed using Lagrange interpolation

The message **m** can be found by the fact that f(0) = $\mathbf{a_0}$ = **m**

BIT COMMITMENT

Problem:

Alice wants to sell Bob information regarding police informants within his Mafia empire

Alice doesn’t trust Bob enough to tell him the rats without getting paid first (they might suddenly disappear)

Bob thinks that the deal is a police setup, and won’t give her the money until she commits to names

BIT COMMITMENT

Commitment:

Bob → Alice: random **r**

Alice → Bob: $\{\mathbf{r}|\mathbf{m}\}_k$

Revelation:

Alice → Bob: **k**

Bob decrypts the message and verifies **r**

Discussion:

The random value **r** is used for freshness and to stop Alice from finding two messages where $\{m\}_{k1} == \{m'\}_{k2}$

i.e. forcing Alice to commit

Bob does not know k until revelation so cannot brute force the message space

BIT COMMITMENT WITH HASH FUNCTIONS

Commitment:

Alice: generates random **r1**, **r2**

Alice → Bob: **r1** and **x** = h(**r1**, **r2**, **m**) [**x** is called a blob]

Revelation:

Alice → Bob: **r1**, **r2**, **m**

Bob hashes (**r1**, **r2**, **m**) and compares it to **x**

Discussion:

Bob does not have to send any messages
Alice sends a message to commit and a message to reveal

Alice cannot find $\mathbf{r_3}$ such that h($\mathbf{r_1}$, $\mathbf{r_3}$, **m**) == h($\mathbf{r_1}$, $\mathbf{r_2}$, **m**)

The value $\mathbf{r_2}$ is kept secret so Bob can't brute force the message space.

FAIR COIN FLIPPING

Problem:

Alice and Bob are arguing on the Internet over who will be white in a game of online chess

They agree to flip a coin to resolve the situation

Alice doesn’t trust Bob to flip the coin

Bob doesn’t trust Alice to flip the coin

How can we flip a coin fairly?

FAIR COIN FLIPPING

Solution:

1. Alice commits to a random bit **b** using a bit commitment scheme and sends the blob **y** = **f(b)** to Bob.
2. Bob tries to guess the bit.
3. If Bob guesses correctly then Bob wins the toss.
   If Bob guesses incorrectly then Alice wins the toss.

Discussion:

The security of the algorithm rests in the security of the function **f(x)** to generate the blob.

The least significant bit of **f(x)** cannot correlate with x.

– Similar to the reason why we hash the result of the Diffie-Hellman exchange to obtain a session key rather than taking the last **n** bits

FAIR COIN FLIPPING USING PUBLIC KEY CRYPTO

Requires that the algorithm commutes e.g. RSA with identical moduli

$$E_B(E_A(\mathbf{m})) = E_A(E_B(\mathbf{m}))$$
$$D_A(E_B(E_A(\mathbf{m}))) = E_B(\mathbf{m})$$

Algorithm:

1. Alice and Bob generate public/private key pairs.
2. Alice generates two random numbers $\mathbf{r_T}$, $\mathbf{r_H}$
3. Alice → Bob: $\mathbf{m_1} = E_A(\text{“heads”}, \mathbf{r_H})$, $\mathbf{m_2} = E_A(\text{“tails”}, \mathbf{r_T})$
4. Bob selects one message **x** at random.
5. Bob → Alice: $E_B(E_A(\mathbf{x}))$
6. Alice → Bob: $D_A(E_B(E_A(\mathbf{x}))) = E_B(\mathbf{x})$
7. Bob → Alice: **x**

FAIR COIN FLIPPING USING PUBLIC KEY CRYPTO

Alice verifies that **x** is one of the two random strings.

Alice and Bob reveal to each other their keypairs to ensure that neither cheated.

Discussion:

- The algorithm is self-enforcing. Either party can detect cheating by the other without a TTP.
- Note: Bob learns of the result of the coin flip before Alice. Although he can't change it, he may delay the result on purpose to take advantage of the situation.
  - Otherwise known as "flipping the coin into a well".
- Coin flipping has use in session key generation as neither party can influence the result of each flip (i.e. bit)
  - e.g. in Diffie-Hellman one party selects an exponent after the first.

MENTAL POKER

Problem:

Alice and Bob want to play poker over email.

Alice doesn’t trust Bob.

Bob doesn’t trust Alice.

How can Alice and Bob be deal hands fairly?

MENTAL POKER

Solution:

1. Alice and Bob use a commutative public key cryptosystem
$$D_A(E_B(E_A(\mathbf{m}))) = E_B(\mathbf{m})$$
2. Alice encrypts 52 messages $\mathbf{m_1}$ = (“Ace of Spades”, $\mathbf{r_1}$) … using her public key.
3. Alice sends the 52 blobs to Bob.
4. Bob picks 5 of these at random, encrypts with his public key and sends them back to Alice.
5. Alice decrypts the messages with her public key and sends back to Bob.
6. Bob decrypts the messages to determine his hand.
7. At the end of the game, Alice and Bob reveal their key pairs to ensure neither cheated.

ATTACKS AGAINST POKER SCHEME

Since some cryptographic algorithms are not truly random processes, they tend to leak small amounts of information.

In RSA, for example, if the binary representation of the card is a quadratic residue, then the encryption of the card is also a quadratic residue.

This could be used by a malicious dealer to “mark” some cards (e.g. the Aces)

OBLIVIOUS TRANSFER

Problem (Kilian):

Bob is trying to factor a 2000-bit number, n.

Alice wants to sell Bob a 1000-bit factor for $1000 (at a very reasonable $1/bit)

Bob only has $500 and offers to buy half the bits- but only if Alice proves that the number is a factor of n, and Alice won’t know which bits Bob bought.

How can the deal be done given, Alice cannot prove that her number is a factor of n without telling it to Bob?

OBLIVIOUS TRANSFER

Algorithm:

1. Alice generates two public/private key pairs $\mathbf{E_{A1}}$, $\mathbf{D_{A1}}$ and $\mathbf{E_{A2}}$, $\mathbf{D_{A2}}$
2. Alice → Bob: $\mathbf{E_{A1}}$, $\mathbf{E_{A2}}$
3. Bob generates a symmetric cypher key, **k**
4. Bob picks one of Alice’s public keys randomly and encrypts **k**
5. Bob → Alice: $\{\mathbf{k}\}_{\mathbf{EX}}$
6. Alice decrypts the key twice $D_{A1}\{k\}_{EX}$ $D_{A2}\{k\}_{EX}$ resulting in **k** and garbage $D_Y\{k\}_{EX}$ (Alice does not know which is the real key).
7. Alice sends Bob two messages, half the bits each: {“first 500 bits”}, {“second 500 bits”}; each encrypted with one of these keys.
8. Bob decrypts both with **k**. One message will make sense to him.
9. Bob now has one of the messages. Alice has no idea which one.

OBLIVIOUS TRANSFER

Discussion:

Alice still needs to convince Bob that the message is a factor of **n**. She does that using a zero-knowledge proof (remember: a way of Alice telling Bob that she knows **x** without revealing any information about **x**).

Obvious transfer is a way Alice can send a bit to Bob in such a way that Bob receives the bit with probability 0.5 and Alice does not know if it is received or not. (i.e. “I have one secret and you get it with probability 0.5”).

This can be extended to “I have two secrets and you get one” “I have **n** secrets and you get one”, etc.

Oblivious transfer not used alone. It’s used as a building block in other protocols.

ELEC5616 COMPUTER & NETWORK SECURITY

Lecture 14b:

Covert Channels & Steganography

SUBLIMINAL CHANNELS

Problem:

Alice and Bob have been arrested for conspiracy to factor large numbers by the government.

Alice has been sent to a woman’s jail, Bob to a men’s jail.

The warden, Walter, is willing to let them communicate on the condition that messages are not encrypted.

How can Alice and Bob communicate secretly given Walter can read their messages and might attempt to deceive them by planting false messages?

SUBLIMINAL CHANNELS

Alice and Bob can set up a subliminal (or covert) channel

Simplest level: Alice and Bob can use ***steganography*** (information hiding) to place in hidden messages in places no-ones suspects a message exists

Steganography is not cryptography: it's "security through obscurity"

Steganography is usually used together with cryptography

EXAMPLE OF STEGANOGRAPHY (TEXT)

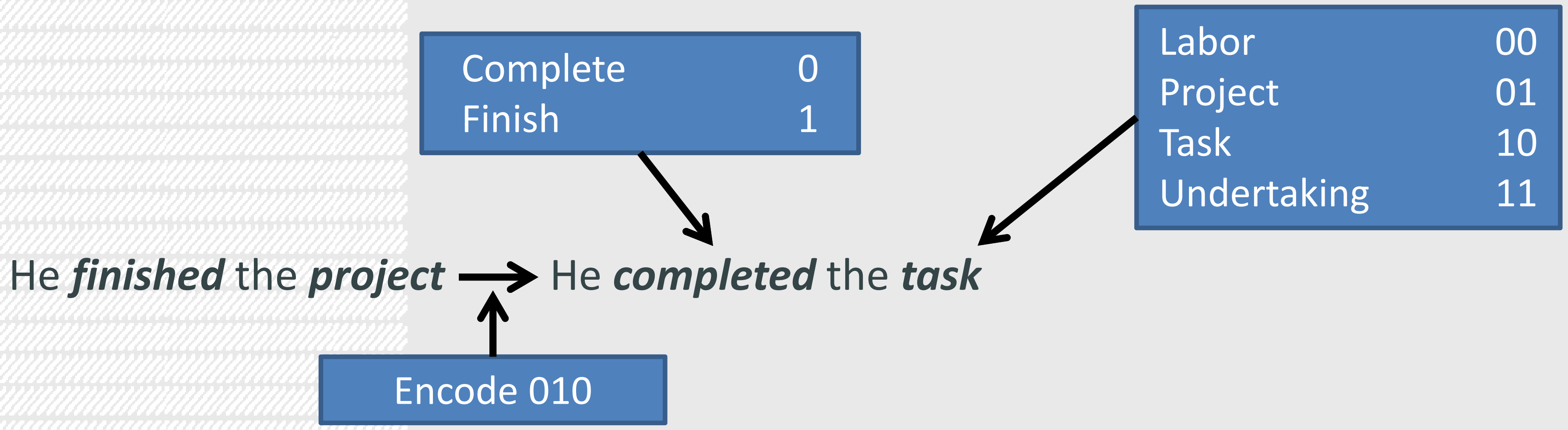
Naive, obvious and bloated: count(words in sent) => if even = 0, if odd = 1

Synonym substitution (0.67 bits per sentence) [Topkara et al.]

Syntactic substitution (0.5 bits per sentence) [Atallah et al., Topkara et al.]

Contextual Synonym Substitution and Vertex Color Coding [Chang & Clark]

Slightly higher than 1 bit per newspaper sentence



EXAMPLE OF STEGANOGRAPHY (IMAGE)

Imagine a grayscale (8 bitsof gray) lossless 32x32 pixel icon

128	64	32	16	8	4	2	1

128 + 64 + 32 + 16 + 8 + 4 + 2 + 1 = 255 + (zero) => your 256 different grays

Imagine if you were happy with a slight trade off – get rid of one bit

128	64	32	16	8	4	2	Msg

Original and modified image will be imperceptibly different to the naked eye

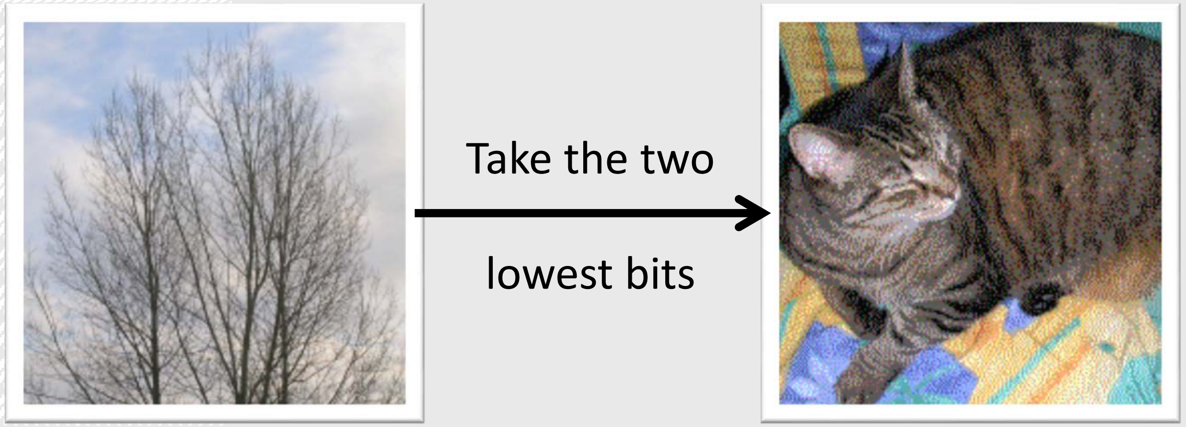
Yet you can now encode a secret message 1024 bits (128 characters) long!

EXAMPLE OF STEGANOGRAPHY (IMAGE)

If you were using a colour image (RGB) you can steal 3 bits per pixel

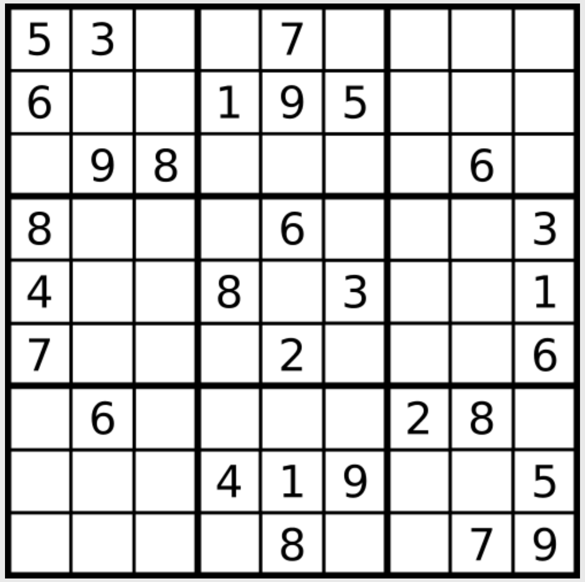
Double the resolution and you quadruple the amount of data you can hide

Settle for some lost quality and you can go to 6 bits per pixel (2 bits per color)



Techniques exist that work with lossy recompression and even resizing

EXAMPLE OF STEGANOGRAPHY (CRAZY)



5	3			7				
6			1	9	5			
	9	8					6	
8				6				3
4			8		3			1
7				2				6
	6					2	8	
			4	1	9			5
				8			7	9

Sudoku has approx 2^{70} different possible solutions, longer than DES key

A standard shuffled deck of cards is one of 52! or 2^{230} different combinations (see *Solitaire cipher* from Cryptonomicon & Bruce Schneier)

NETWORK STEGANOGRAPHY

Loki

- Daemon9, Alhambra (phrack/the guild)
- Bidirectional covert UNIX shell client using the data field in ICMP type 0 (Echo Reply) and type 8 (Echo Request) packets.

Daemonshell-UDP

- ICMP Echo Reply only (more stealthy)

ICMP Backdoor

- Reusable tunnel library
- Messages fragmented to look more like ping packets (multiples of 64 bytes)

Rwwwshell

- Backdoor emits requests as HTTP Response packets
- Output from commands return from the slave as cgi script HTTP GETs

B0CK

- IGMP multicast messages used as transport

AckCmd

- TCP ACK packets for request (port 80), TCP RESET packets for response (high port)

FIRESMITHING

The head of IT at XYZ tells you they're entirely leak safe:
"I disabled SSL, have a limited white-list of websites the users can access and monitor all their outgoing email for sensitive documents..."

Can data still leave this network? Yes, of course!

Ask Google Translate to access mattbarrie.com/?Firesmithing
Google Translate fetches the page in order to retrieve the content to translate

```
 [Google's IP] - - mattbarrie.com 193.239.120.148:80 [date]\
"GET /?Firesmithing HTTP/1.0" 200 7863 "-"\ "browser (via
translate.google.com)"
```

Use DNS: send data out by making requests to turn domain names into Ips
facebook.com => 173.252.110.27
answers-to-exam.smerity.com
Want SSH? You can get SSH over DNS. Not fast but entirely doable.

HOW DO WE PROTECT AGAINST THIS?

Start to see the problems with content filtering?

Consider national content filters

- Great firewall of China (and other middle eastern countries)
- Australian Government NetAlert

Corporate content filters

Net-nannies

What if malware used these techniques to communicate?

- Answer: they do

Could other internets be layered onto the Internet

REFERENCES

Handbook of Applied Cryptography

– read § 12.7 - 12.7.2, 12.9