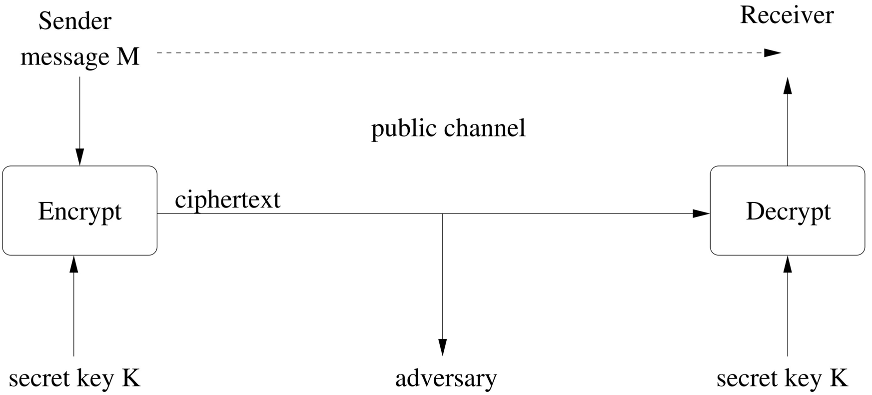
Symmetric Key Encryption

Some Basics

Symmetric Key Encryption



Definition

- A symmetric-key encryption (SKE) scheme consists of:
  - $\mathcal{M}$: Set of possible plaintexts.
  - $\mathcal{C}$: Set of possible ciphertexts.
  - $\mathcal{K}$: Set of possible keys.
  - A family of encryption functions, $E_k : \mathcal{M} \to \mathcal{C}$, $\forall k \in \mathcal{K}$.
  - A family of decryption functions, $D_k : \mathcal{C} \to \mathcal{M}$, $\forall k \in \mathcal{K}$, such that $D_k(E_k(m)) = m$ for all $m \in \mathcal{M}$ and $k \in \mathcal{K}$.

Shift Cipher

- Define $\mathcal{M} = \mathcal{C} = \mathcal{K} = \mathbb{Z}_{26}$.
- For $0 \leq k \leq 25$, define:
  - $E_k(x) = (x + k) \bmod 26$.
  - $D_k(y) = (y - k) \bmod 26$.
- How "secure" is the Shift Cipher if you use it in communication?

Shift Cipher

- Define $\mathcal{M} = \mathcal{C} = \mathcal{K} = \mathbb{Z}_{26}$.
- For $0 \leq k \leq 25$, define:
  - $E_k(x) = (x + k) \bmod 26$.
  - $D_k(y) = (y - k) \bmod 26$.
- How "secure" is the Shift Cipher if you use it in communication?
- Lesson 1: The key-space should be sufficiently large so that exhaustive key search is infeasible.

What is (in)feasible?

Current state of affairs:

- 2^{56} bit operations: easy.
- 2^{64} bit operations: feasible.
- 2^{80} bit operations: considered infeasible around 2010.
  - 2022: Antminer S17 performs 2^{86} bit operations in 65 days.
  - In the same year Bitcoin network as a whole performed 2^{111} bit operations!
- 2^{128} bit operations is considered infeasible.

Substitution Cipher

- $\mathcal{M}$: set of *all* English messages.
- $\mathcal{C}$: set of all *encrypted* messages.
- $\mathcal{K}$: all permutations of the English alphabet.
- $E_k(m)$: Apply permutation k to m, one letter at a time.
- $D_k(c)$: Apply the inverse permutation k^{-1} to c, one letter at a time.

Example

The key (k) is the following permutation:

a	b	c	d	e	f	g	h	i	j	k	l	m
T	H	P	I	D	A	V	L	J	E	M	C	R

n	o	p	q	r	s	t	u	v	w	x	y	z
F	B	W	Q	Z	O	X	Y	G	S	K	N	U

m = crypto is fun
E_k(crypto is fun) = PZNWXB JO AYF

Example

The key (k) is the following permutation:

a	b	c	d	e	f	g	h	i	j	k	l	m
T	H	P	I	D	A	V	L	J	E	M	C	R
n	o	p	q	r	s	t	u	v	w	x	y	z
F	B	W	Q	Z	O	X	Y	G	S	K	N	U

m = crypto is fun

E_k(crypto is fun) = PZNWXB JO AYF

Attack Strategy: Decrypt the ciphertext with one key at a time and see whether the resulting plaintext "makes sense".

Question: Is exhaustive key search feasible?

Exhaustive search

- Number of keys: $26! \geq 4 \times 10^{26} \approx 2^{88}$.
  - Suppose the adversary uses one million computers,
  - Each capable of trying one trillion keys per second,
- Exhaustive key search will take about 10 thousand years!

Exhaustive search

- Number of keys: $26! \geq 4 \times 10^{26} \approx 2^{88}$.
  - Suppose the adversary uses one million computers,
  - Each capable of trying one trillion keys per second,
- Exhaustive key search will take about 10 thousand years!
- Is the Substitution Cipher secure?
  - What does it mean for a crypto-system to be secure?

Security by obscurity!

Kerckhoff's Principle [1883]: *Compromise of the system details should not inconvenience the correspondents.*

- The adversary knows the cryptosystem being used.
- For Symmetric Key Encryption: attacker knows the encryption and the decryption algorithm, but not the secret key.

Modeling the Adversary

- What is the adversary's (computational) power?
- How does the adversary interact with the crypto-system or the communicating parties?
- What is the adversary's goal?

Attack Models for Encryption System

- Passive attacks:
  1. Ciphertext only attack: The adversary can only see ciphertexts.
  2. Known plaintext attack: The adversary also knows some plaintext and the corresponding ciphertext.

Attack Models for Encryption System

- Passive attacks:
  1. Ciphertext only attack: The adversary can only see ciphertexts.
  2. Known plaintext attack: The adversary also knows some plaintext and the corresponding ciphertext.
- Active attacks:
  1. Chosen plaintext attack: The adversary can choose any plaintext and obtains the corresponding ciphertext. Adversary is given access to an "Encryption Oracle".
  2. Chosen ciphertext attack: The adversary can choose any ciphertext and obtains the corresponding plaintext. Adversary is given access to a "Decryption Oracle".

Power of the Adversary

- Information-theoretic: Adversary has infinite computational resource.
- Complexity-theoretic: Adversary is a polynomial-time (Quantum) Turing Machine.
- Computational: Adversary is computationally bounded (has access to x GPUs, y desktop PCs etcetera).

Adversary's Goal

1. Recover the secret key.
2. Recover plaintext from ciphertext (without necessarily learning the secret key).
3. Learn some partial information about the plaintext from the ciphertext.

- If the adversary can achieve Goal 1 or Goal 2, then the encryption scheme is completely broken.
- If the adversary cannot learn any partial information about the plaintext from the ciphertext (except possibly its length), the encryption scheme is said to be semantically secure.

Semantic Security of Symmetric Key Encryption

Consider the following game between an adversary $\mathcal{A}$ and a challenger $\mathcal{C}$.

1. The adversary is given a challenge ciphertext c (generated by Alice or Bob using their secret key k).
2. The adversary can select certain number of plaintexts and obtains (from Alice or Bob) the corresponding ciphertext.
3. At the end, the adversary obtains some information about the plaintext corresponding to c (other than the length of m).

Semantic Security of Symmetric Key Encryption

Consider the following game between an adversary $\mathcal{A}$ and a challenger $\mathcal{C}$.

1. The adversary is given a challenge ciphertext c (generated by Alice or Bob using their secret key k).
2. The adversary can select certain number of plaintexts and obtains (from Alice or Bob) the corresponding ciphertext.
3. At the end, the adversary obtains some information about the plaintext corresponding to c (other than the length of m).

A symmetric-key encryption scheme is said to be semantically secure against chosen-plaintext attack if *no* computationally bounded adversary can win the above game.

Security of Substitution Cipher

- Completely insecure against a chosen plaintext attack.

Security of Substitution Cipher

- Completely insecure against a chosen plaintext attack.
- Is it secure against ciphertext only attack?
  - We've seen that exhaustive key search is not possible.
  - Is there any other *tool*?

Security of Substitution Cipher

- Completely insecure against a chosen plaintext attack.
- Is it secure against ciphertext only attack?
  - We've seen that exhaustive key search is not possible.
  - Is there any other *tool*?
    - Have you read The Adventure of the Dancing Men!

Frequency Analysis

One way to solve an encrypted message is to find a different plain-text of the same language and then count the occurrences of each letter. Call the most frequently occurring letter the first, the next most occurring letter the second...and so on, until we account for all the different letters in the plaintext sample.
Then look at the ciphertext and also classify its symbols. We find the most occurring symbol and change it to the form of the first letter of the plaintext sample, the next most common symbol is changed to the form of the second letter...and so on, until we account for all symbols of the cryptogram we want to solve.

al-Kandi: The Philosopher of the Arabs

The (slightly abridged) text in the previous slide is from
A Manuscript on Deciphering Cryptographic Messages
A classic on cryptanalysis by
Abu Yusuf Yaqub ibn Ishaq ibn as-Sabbah ibn omran ibn Ismail al-Kandi
Written in the 9th century (modernity came to know about the text only in 1987)!

Statistical Properties of an English Text

- Relative frequencies of the 26 letters are known.
  - The letters can be clustered into five groups.
  - Letters in each group have approximately the same frequency.
- Here letters in each group is arranged in order of decreasing frequency.

Group 1 E
Group 2 T, A, O, I, N, S, H, R
Group 3 D, L
Group 4 C, U, M, W, F, G, Y, P, B
Group 5 V, K, J, X, Q, Z

- One can also consider the frequent digrams and trigrams.
- Turned out to be a very effective tool for cryptanalysis!

Perfect Secrecy

- Suppose Alice and Bob selected one of the 26 letters uniformly at random as the secret key.
- Alice uses the key to encrypt only one of the two letters y or n to Bob.
- The key is never re-used.
- Can Eve learn any information about the plaintext from the ciphertext?
- Even unlimited computational resources will not be of any help to Eve.
  - Simply because there is not enough information available in the ciphertext about the underlying plaintext.

Encrypting a Stream of Data

- Suppose you want to encrypt some message bit by bit.
- For unconditional security use One Time Pad.
  - Invented by Frank Miller [1882], re-invented by Gilbert Vernam [1917] (also called Vernam cipher).
  - Joseph Mauborgne of US Army (WW-I) suggested only one-time use of a random key.

The key is:

1. Random.
2. Never re-used.
3. As large as the plaintext.

Beyond OTP

- OTP is useless for almost all practical purposes.
- Question: What will be a reasonable approximation of OTP for an efficient and secure encryption mechanism?

Beyond OTP

- OTP is useless for almost all practical purposes.
- Question: What will be a reasonable approximation of OTP for an efficient and secure encryption mechanism?
- Intuitive answer: Use a key string that *"appears as random"* but not truly random.

Stream Cipher

- Stream ciphers encrypt individual bits of the plaintext one at a time.
- Extremely fast in hardware.
- Suitable in situations where
  1. The device has no memory or scope of buffering is limited.
  2. Plaintext characters must be individually processed as they are received.
- Advantage: No error propagation.

Basic Principle

- One-Time Pad uses a random key which is as long as the plaintext message.
- A stream cipher uses a pseudorandom key and XOR it with the plaintext message.
- The key is the output of a pseudorandom (bit) generator (PRG).
  - A **deterministic** algorithm that takes a small random seed as input and outputs a much longer "pseudorandom" bit sequence.
  - The seed is the secret key shared between the communicating parties.
- Pros: The key can be much smaller than the plaintext message.
- Cons: No question of perfect secrecy – security depends on the strength of the PRG.

Security of PRG

- Unpredictability: Given a keystream of length ℓ, no polynomial-time adversary should be able to gain any information about the rest of the keystream.
  - Next Bit Predictor: Given a keystream of length ℓ, no poly-time adversary should be able to predict the next bit with probability significantly greater than $\frac{1}{2}$.
- Indistinguishability: No polynomial-time adversary should be able to distinguish the keystream generated by a PRG from a truly random sequence.

Some Questions

1. Recall the structure of rand() and srand(). Do they satisfy the PRG security requirement?
2. Which Stream Cipher will you recommend for use in practice?
3. Suppose a steam cipher is used to encrypt data. How will the receiver detect whether the ciphertext has been modified in transit or not?

RC4 Stream Cipher

- Designed by Ron Rivest in the late 80's.
  - RC stands for Ron's Code (or Rivest's Cipher?).
  - It's a trade secret of RSA Security.
- A description of RC4 was anonymously posted on the web in 1994.
- Extremely simple and fast with variable key length.
- RC4 has been widely used in many security protocols.
  1. SSL/TLS (prohibited by IETF).
  2. Wired Equivalent Privacy (WEP).
- RC4 has two components:
  1. Key Scheduling Algorithm (KSA).
  2. Pseudo-Random Generation Algorithm (PRGA).

Key Scheduling Algorithm

Input: Secret key: $\text{Key}[0], \text{Key}[1], \ldots, \text{Key}[d-1]$.
(Keysize $= 8d$ bits, typically 40–128 bits.)
Output: An array: $S[0], S[1], \ldots, S[255]$.
Each $\text{Key}[i]$ and $S[i]$ are of size 1-byte.

For i from 0 to 255 do:
 $S[i] \leftarrow i$;
$j \leftarrow 0$;
For i from 0 to 255 do:
 $j \leftarrow (\text{Key}[i \bmod d] + S[i] + j) \bmod 256$;
 Swap $(S[i], S[j])$;

So, finally what is S?

Key Scheduling Algorithm

Input: Secret key: $\text{Key}[0], \text{Key}[1], \ldots, \text{Key}[d-1]$.
(Keysize $= 8d$ bits, typically 40–128 bits.)
Output: An array: $S[0], S[1], \ldots, S[255]$.
Each $\text{Key}[i]$ and $S[i]$ are of size 1-byte.

For i from 0 to 255 do:
  $S[i] \leftarrow i$;
$j \leftarrow 0$;
For i from 0 to 255 do:
  $j \leftarrow (\text{Key}[i \bmod d] + S[i] + j) \bmod 256$;
  Swap $(S[i], S[j])$;

So, finally what is S?
A random(-looking) permutation of $\{0, 1, 2, \ldots, 255\}$ which is generated based on the secret key.

Pseudo-Random Generation Algorithm

Input: $S[0], S[1], \ldots, S[255]$
Output: Keystream bits
$i \leftarrow 0;$
$j \leftarrow 0;$
While keystream bytes are required do:
    $i \leftarrow (i + 1) \bmod 256;$
    $j \leftarrow (S[i] + j) \bmod 256;$
    Swap$(S[i], S[j]);$
    $t \leftarrow (S[i] + S[j]) \bmod 256;$
    Output$(S[t]);$

Note: The keystream bits are XOR-ed with the plaintext bits to generate the ciphertext.

Pseudo-Random Generation Algorithm

Input: $S[0], S[1], \ldots, S[255]$
Output: Keystream bits
$i \leftarrow 0$;
$j \leftarrow 0$;
While keystream bytes are required do:
    $i \leftarrow (i + 1) \bmod 256$;
    $j \leftarrow (S[i] + j) \bmod 256$;
    Swap($S[i], S[j]$);
    $t \leftarrow (S[i] + S[j]) \bmod 256$;
    Output($S[t]$);

Note: The keystream bits are XOR-ed with the plaintext bits to generate the ciphertext.
Question: Why Swap($S[i], S[j]$)?

Steam Cipher: Use Case

Security Protocol for Wireless Network

Wireless Network

- Popular standards:
  1. Bluetooth: short range, low speed, published in 1994.
  2. IEEE 802.11: Long range, high speed, widely used for wireless LANs, published in 1999.

Wireless Network

- Popular standards:
  1. Bluetooth: short range, low speed, published in 1994.
  2. IEEE 802.11: Long range, high speed, widely used for wireless LANs, published in 1999.
- Security risks
  - No need of physical access – attack from a distance (if you have a good antenna).
  - No physical evidence.

Wireless Network

- Popular standards:
  1. Bluetooth: short range, low speed, published in 1994.
  2. IEEE 802.11: Long range, high speed, widely used for wireless LANs, published in 1999.
- Security risks
  - No need of physical access – attack from a distance (if you have a good antenna).
  - No physical evidence.
- Wired Equivalent Privacy (WEP): A security protocol specified in the IEEE 802.11 standard for wireless LAN communications.
- WEP's goal is to protect the data at the link-level during wireless transmission between mobile stations and access points.
  - Supposed to provide security equivalent to a wired connection.

Wireless Network Security

Security goals of WEP:

1. Confidentiality: Fundamental goal is to prevent casual eavesdropping.
   - RC4 is used to encrypt data.
2. Access Control: Prevent unauthorised access.
   - Discard data that are not properly encrypted using WEP.
3. Data Integrity: Prevent tampering of transmitted message.
   - An integrity checksum field is included.

WEP Protocol

- Key: A secret key k is shared between the communicating parties (i.e., clients (C) and the access points (AP)).
  - k is either 40-bits or 104-bits. (Why 40?)
  - The standard does not specify any key distribution mechanism.
  - In practice a single key is often used for the entire network.
  - k is changed very rarely.

WEP Protocol

- Key: A secret key k is shared between the communicating parties (i.e., clients (C) and the access points (AP)).
  - k is either 40-bits or 104-bits. (Why 40?)
  - The standard does not specify any key distribution mechanism.
  - In practice a single key is often used for the entire network.
  - k is changed very rarely.
- Message: divided into packets of some fixed length.

WEP Protocol

- Key: A secret key k is shared between the communicating parties (i.e., clients (C) and the access points (AP)).
  - k is either 40-bits or 104-bits. (Why 40?)
  - The standard does not specify any key distribution mechanism.
  - In practice a single key is often used for the entire network.
  - k is changed very rarely.
- Message: divided into packets of some fixed length.
- Initialization Vector (IV): a 24-bit IV v is appended with k.
  - WEP recommends to change the IV after each packet but does not say how to select the IVs.
  - In practice IVs are generated
    1. sequentially - starting from 0 and then incremented by 1.
    2. Or randomly for each packet.

WEP Protocol: Packet Sending

To send a packet m do the following:

WEP Protocol: Packet Sending

To send a packet m do the following:

1. Compute an *integrity checksum* $s = \mathrm{IC}(m)$.
   - 802.11 uses CRC-32 checksum which is linear:

$$\mathrm{IC}(m_1 \oplus m_2) = \mathrm{IC}(m_1) \oplus \mathrm{IC}(m_2).$$

WEP Protocol: Packet Sending

To send a packet m do the following:

1. Compute an *integrity checksum* $s = \mathrm{IC}(m)$.
   - 802.11 uses CRC-32 checksum which is linear:
   $$\mathrm{IC}(m_1 \oplus m_2) = \mathrm{IC}(m_1) \oplus \mathrm{IC}(m_2).$$
   - The plaintext is $P = m||s$.

WEP Protocol: Packet Sending

To send a packet m do the following:

1. Compute an *integrity checksum* $s = \mathrm{IC}(m)$.
   - 802.11 uses CRC-32 checksum which is linear:

$$\mathrm{IC}(m_1 \oplus m_2) = \mathrm{IC}(m_1) \oplus \mathrm{IC}(m_2).$$

   - The plaintext is $P = m || s$.
2. Select 24-bit IV v.

WEP Protocol: Packet Sending

To send a packet m do the following:

1. Compute an *integrity checksum* $s = \mathrm{IC}(m)$.
   - 802.11 uses CRC-32 checksum which is linear:
   $$\mathrm{IC}(m_1 \oplus m_2) = \mathrm{IC}(m_1) \oplus \mathrm{IC}(m_2).$$
   - The plaintext is $P = m||s$.
2. Select 24-bit IV v.
3. Compute the ciphertext $c = P \oplus \mathrm{RC4}(\mathrm{v}||\mathrm{k})$.

WEP Protocol: Packet Sending

To send a packet m do the following:

1. Compute an *integrity checksum* $s = \mathrm{IC}(m)$.
   - 802.11 uses CRC-32 checksum which is linear:

$$\mathrm{IC}(m_1 \oplus m_2) = \mathrm{IC}(m_1) \oplus \mathrm{IC}(m_2).$$

   - The plaintext is $P = m || s$.
2. Select 24-bit IV v.
3. Compute the ciphertext $c = P \oplus \mathrm{RC4}(\mathrm{v}||\mathrm{k})$.
4. Send (v, c) over the wireless channel.

WEP Protocol: Packet Receiving

Receiver of a packet (v, c) does the following:

1. Compute $P' = c \oplus \text{RC4}(\text{v}||\text{k}) = \text{m}||\text{s}$.

WEP Protocol: Packet Receiving

Receiver of a packet (v, c) does the following:

1. Compute $P' = c \oplus \mathrm{RC4}(\mathrm{v}||\mathrm{k}) = \mathrm{m}||\mathrm{s}$.
2. Compute $s' = \mathrm{IC}(\mathrm{m})$.

WEP Protocol: Packet Receiving

Receiver of a packet (v, c) does the following:

1. Compute $P' = c \oplus \mathrm{RC4(v||k)} = \mathrm{m||s}$.
2. Compute $s' = \mathrm{IC(m)}$.
3. Accept m as valid if $s' = s$; reject if $s' \neq s$.

WEP Security

Security: confidentiality, access control, data integrity

Question: Are the security goals achieved in 802.11?

IV Collision

- Two encrypted packets (v_1, c_1) and (v_2, c_2) have IV collision if $v_1 = v_2$.
- Recall that the IV v is
  1. sent in the clear
  2. and 24-bits (how many possibilities?)

IV Collision

- Two encrypted packets (v_1, c_1) and (v_2, c_2) have IV collision if $v_1 = v_2$.
- Recall that the IV v is
  1. sent in the clear
  2. and 24-bits (how many possibilities?)
- If v is generated sequentially then in a network with average 5Mbps bandwidth an IV collision occurs within a few days.

IV Collision

- Two encrypted packets (v_1, c_1) and (v_2, c_2) have IV collision if $v_1 = v_2$.
- Recall that the IV v is
  1. sent in the clear
  2. and 24-bits (how many possibilities?)
- If v is generated sequentially then in a network with average 5Mbps bandwidth an IV collision occurs within a few days.
- If IVs are chosen randomly, one can expect to see a collision after around 5000 packets are transmitted (due to birthday paradox)!

Confidentiality after Collision

- Suppose we detect a collision in two encrypted packets: (v, c_1) and (v, c_2).
- Suppose P_1 and P_2 are the underlying plaintext messages: $c_1 = P_1 \oplus \mathrm{RC4(v, k)}$, $c_2 = P_2 \oplus \mathrm{RC4(v, k)}$.

Confidentiality after Collision

- Suppose we detect a collision in two encrypted packets: (v, c_1) and (v, c_2).
- Suppose P_1 and P_2 are the underlying plaintext messages: $c_1 = P_1 \oplus \mathrm{RC4(v, k)}$, $c_2 = P_2 \oplus \mathrm{RC4(v, k)}$.
- Then $c_1 \oplus c_2 = P_1 \oplus P_2$.
  - If one of them (say P_1) is known then the other (P_2) is immediately revealed.
  - More generally, one can utilize the expected distribution of P_1 and P_2 to extract information about them.
- Conclusion: WEP does not provide a high degree of confidentiality.

Access Control

- Suppose the attacker obtains a single message m corresponding to IV-ciphertext pair (v, c).
  - Attacker can compute the corresponding part of RC4 keystream: $\mathrm{RC4(v,k) = c \oplus (m||IC(m))}$.
- Attacker can now create the encryption of *any* message m' as

$$c' = (m'||\mathrm{IC(m')}) \oplus \mathrm{RC4(v,k)}$$

- Attacker transmits (v, c') which will be accepted as valid.

Access Control

- Suppose the attacker obtains a single message m corresponding to IV-ciphertext pair (v, c).
  - Attacker can compute the corresponding part of RC4 keystream: $\mathrm{RC4(v, k)} = \mathrm{c} \oplus (\mathrm{m}||\mathrm{IC(m)})$.
- Attacker can now create the encryption of *any* message m' as

$$c' = (m'||\mathrm{IC(m')}) \oplus \mathrm{RC4(v, k)}$$

- Attacker transmits (v, c') which will be accepted as valid.
- WEP allows to repeat old IV values without triggering any alarms at the receiver.
  - So the attacker can go on sending as many packets as s/he wishes.
- Conclusion: Access control is violated!

Data Integrity

- Recall that WEP checksum IC is a linear function of the message.
  - Works fine for random errors.
  - Fails when an adversary deliberately modifies the message.
- Exercise: Show that WEP fails to provide data integrity.

Reading Material

- The above attacks on WEP were discovered by Borisov, Goldberg and Wagner in 2001.
  - Intercepting Mobile Communications: The Insecurity of 802.11
  - Read the paper, specially Sections 1, 2, 3, 4.1, 4.2 and 6.
- Another attack due to key stream reuse was discovered by Schneier and Mudge:
  Cryptanalysis of Microsoft's Point to Point Tunneling Protocol (PPTP)
- An easy to read article on insecurity due to stream reuse:
  https://cryptosmith.com/2008/05/31/stream-reuse/

Salsa/ChaCha Stream Cipher

- Dan Bernstein proposed Salsa in 2005 and its variant ChaCha in 2008.
  - Extremely fast
  - No effective attack
  - Widely deployed
- Study the ChaCha20 stream cipher.
- Reference: https://cryptography101.ca/crypto101-building-blocks/
  - Watch the video lecture on Stream Ciphers that discusses ChaCha20.

Take Home

- RC4 was believed to be "secure". However, the improper use of RC4 resulted in insecure security protocol.
- A good door-lock is necessary but may not be sufficient for the security of your home – security is more like a chain.
  - A system is as secure as its weakest link.
- Defining the security goals and designing a secure system are difficult problems.
  - Hire experts – very very costly :(
  - Make protocols available for public scrutiny (cryptanalysis).

Adi Shamir: Turing Award lecture

Three Laws of Security:

1. Absolutely secure systems do not exist.
2. To halve your vulnerability, you have to double your expenditure.
3. Cryptography is typically bypassed, not penetrated.

And, a prediction:

Crypto research will remain vigorous, but only its simplest ideas will become practically useful.