

# Cryptography Intro

## CS642: Computer Security



Professor Ristenpart

<http://www.cs.wisc.edu/~rist/>

rist at cs dot wisc dot edu

## How a Google Headhunter's E-Mail Revealed Massive Misuse of DKIM

Posted by **Unknown Lamer** on Wednesday October 24, @01:16PM  
from the insecurity-through-blatant-idiocy dept.



[concealment](#) writes with a tale of how an email sent to a mathematician led to him discovering that dozens of high profile companies were using easily crackable keys to authenticate mail sent from their domains. From the article:

## PS3 Encryption Keys Leaked

Posted by **Soulskill** on Tuesday October 23, @07:02PM  
from the andrei,-you've-lost-another-submarine? dept.



An anonymous reader writes

"PS3 security [has been compromised again](#). The holy grail of the PS3 security encryption keys — [LV0 keys](#) — [have been found and leaked into the wild](#). For the homebrew community, this means deeper access into the PS3: the possibility of custom (or modified) firmware up to the most recent version, the possibility of

# Cryptography



Basic goals and setting

TLS (HTTPS)

Provable security

One time pad

Block ciphers

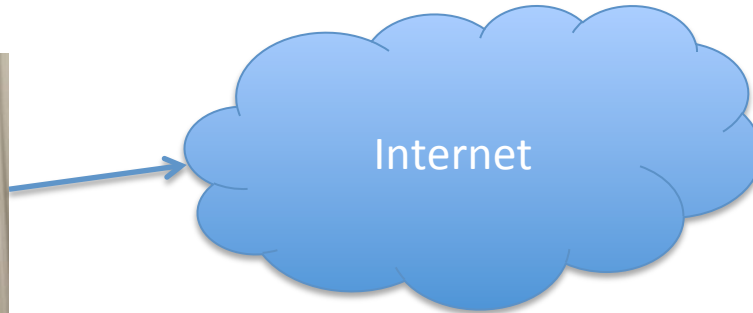
# Cryptography: “Hidden writing”

- Study and practice of building security protocols that resist adversarial behavior
- Blend of mathematics, engineering, computer science



WikiLeaks encrypted cables

# Cryptography

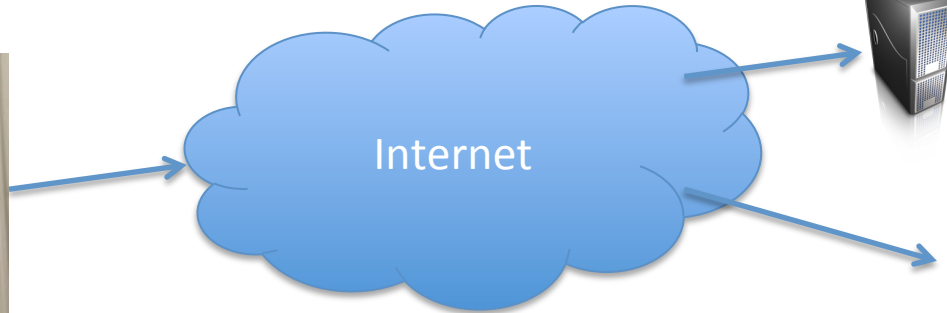


US  
diplomatic  
cables

Don't want to reveal data early

Want to store it in way that it  
can quickly be revealed later

# Cryptography



```
01101010
10101010
10101010
11111101
```

```
01101010
10101010
10101010
11111101
```

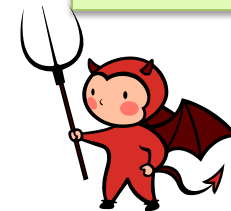
Don't want to reveal data early

Want to store it in way that it  
can quickly be revealed later

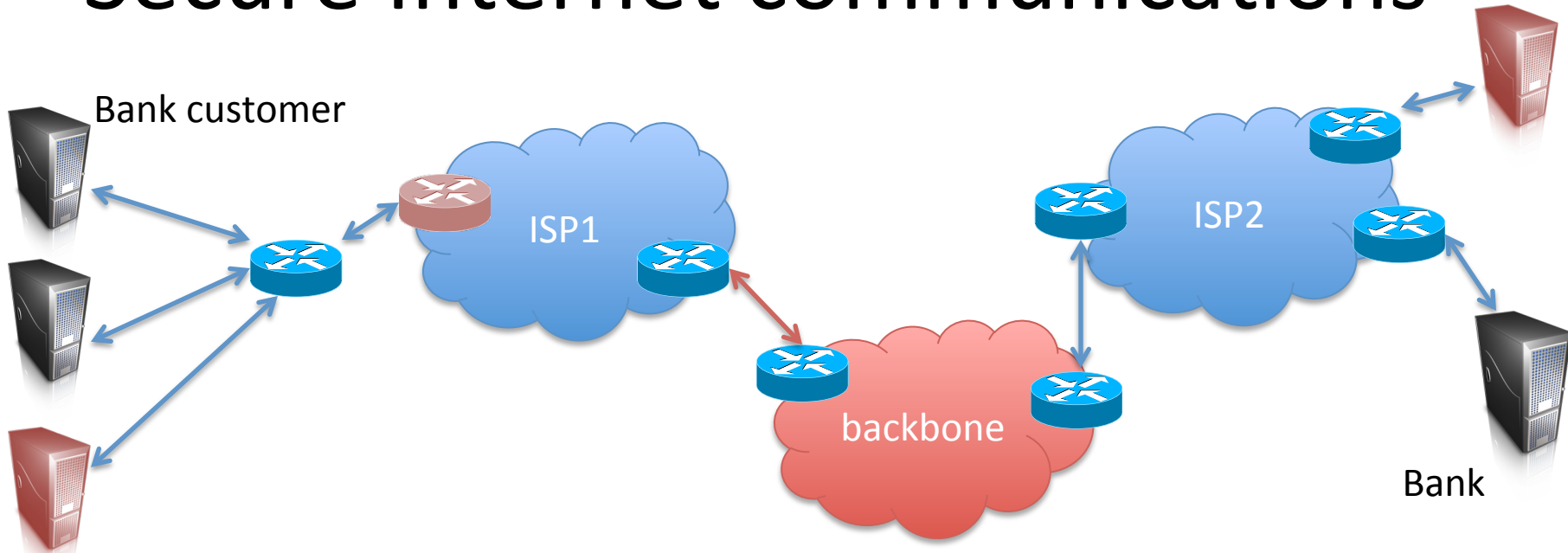
```
01101010
10101010
10101010
11111101
```

Modern cryptography enables this:

- Encrypt file
- Store key in secure place ←



# Secure Internet communications

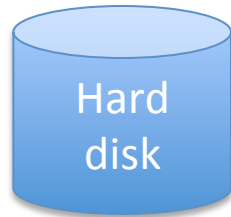


Customer and bank want to communicate securely:

- **Confidentiality** (messages are private)
- **Integrity** (accepted messages are as sent)
- **Authenticity** (is it the bank? is this the customer?)
- Non-goal: anonymity (hide identities)
- Non-goal: steganography (hide that communication took place)

TLS, SSH, IPsec, PGP

# Encrypted hard disks



Company's intellectual property  
Customer records  
Your personal diary

Encrypt hard drives (or volumes):

- Confidentiality of data
- Attacker has physical access to device

Bitlocker, Truecrypt, Seagate



# Crypto

- Powerful tool for confidentiality, authenticity, and more
- But:
  - must design securely
  - must implement designs securely
  - must use properly (e.g., key management)

# Auguste Kerckhoffs' (Second) Principle

“The system must not require secrecy and can be stolen by the enemy without causing trouble”

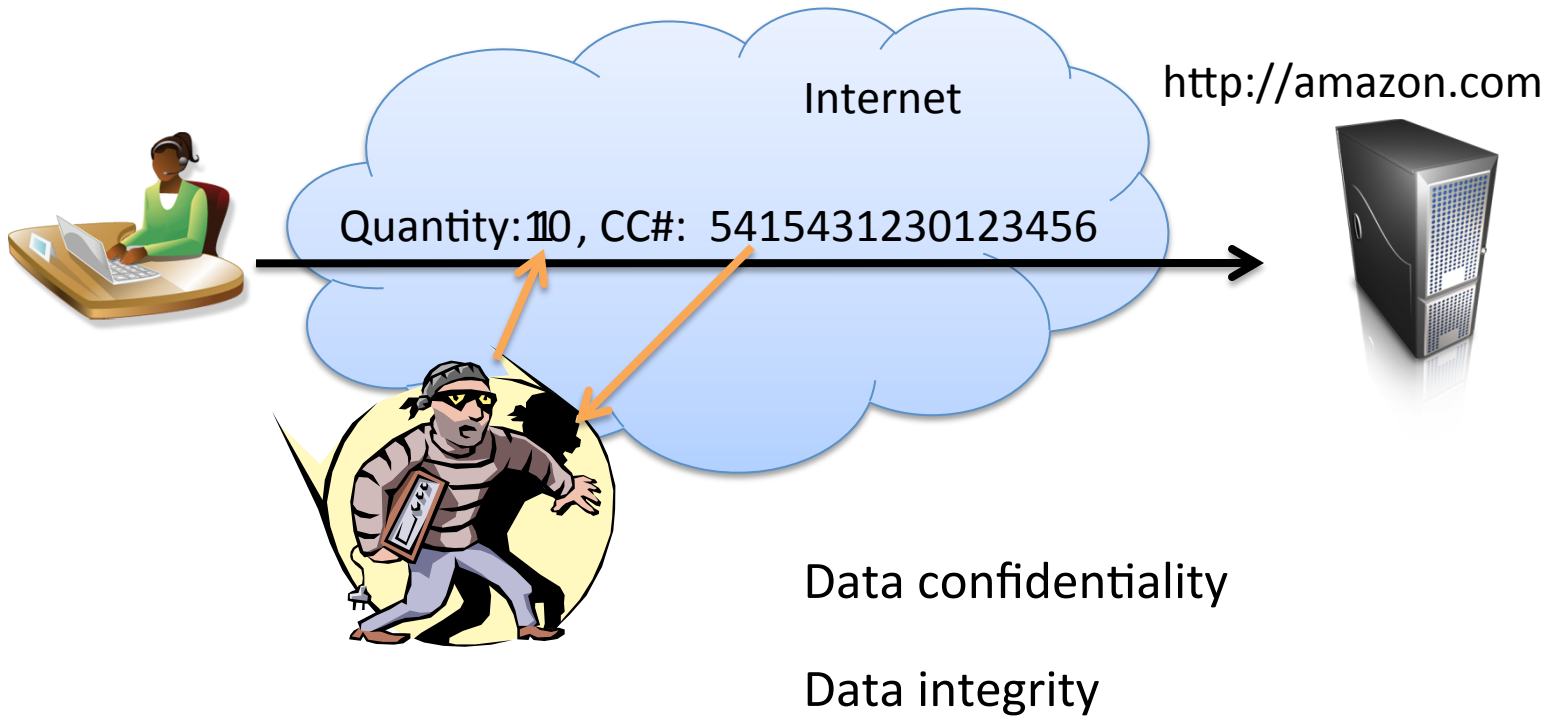
A cryptosystem should be secure even if its algorithms, implementations, configuration, etc. is made public --- the only secret should be a key

Why?

# Some basic primitives

- Symmetric cryptography (shared key  $K$ )
  - encryption & decryption using  $K$
  - message authentication using  $K$
  - pseudorandom functions (PRF)
- Public-key cryptography (public key  $pk$ , secret key  $sk$ )
  - encrypt with  $pk$  and decrypt with  $sk$
  - digitally sign using  $sk$  and verify with  $pk$
- Hash functions (no keys)
  - used to “compress” messages in a secure way

# An example: On-line shopping



We need secure channels for transmitting data

# An example: On-line shopping **with TLS**

https://amazon.com



**K**



$Enc(K, \text{"Quantity: 1 , CC#: 5415431230123456"})$



**K**

Step 1:  
Key exchange  
protocol to  
share secret **K**

Step 2:  
Send data via  
secure  
channel

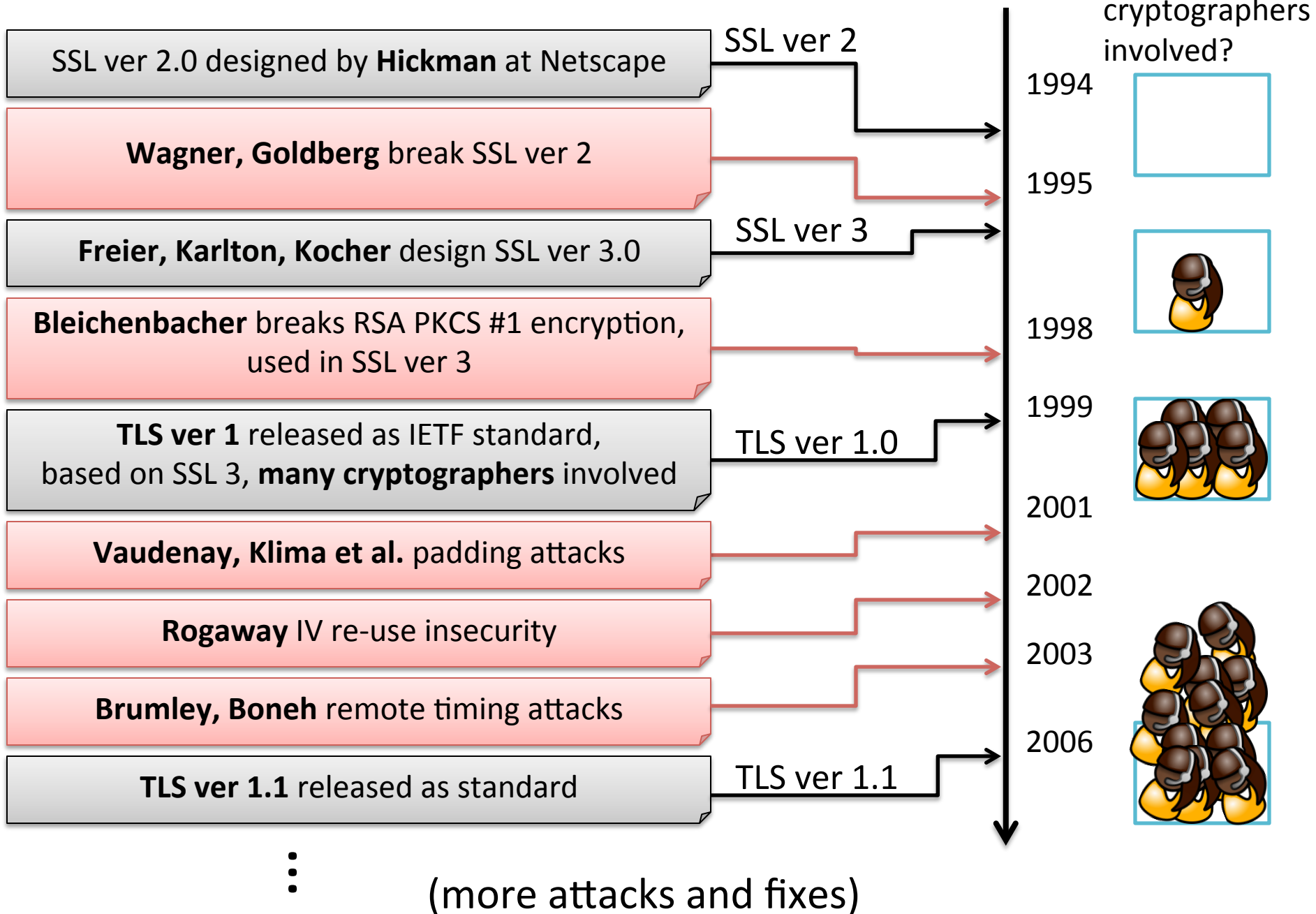
TLS uses many **cryptographic primitives**:

**key exchange:** hash functions, digital signatures, public key encryption

**secure channel:** symmetric encryption, message authentication

Mechanisms to resist **replay attacks**, **man-in-the-middle attacks**,  
**truncation attacks**, etc...

# A short history of TLS up to 2009





Bank customer

# TLS handshake for RSA transport



Bank

Pick random  $N_c$

ClientHello, MaxVer,  $N_c$ , Ciphers/CompMethods

Pick random  $N_s$

ServerHello, Ver,  $N_s$ , SessionID, Cipher/CompMethod

Check CERT  
using CA public  
verification key

CERT = (pk of bank, signature over it)

Pick random PMS  
 $C \leftarrow E(pk, PMS)$

C

$PMS \leftarrow D(sk, C)$

ChangeCipherSpec,  
{ Finished, PRF( $MS$ , "Client finished" ||  $H(\text{transcript})$ ) }

ChangeCipherSpec,  
{ Finished, PRF( $MS$ , "Server finished" ||  $H(\text{transcript}')$ ) }

Bracket notation  
means contents  
encrypted

$MS \leftarrow \text{PRF}(PMS, \text{"master secret"} || N_c || N_s)$



Bank customer

# TLS Record layer



Bank

$$MS \leftarrow \text{PRF}(PS, \text{"master secret"} \parallel N_c \parallel N_s)$$
$$K_1, K_2 \leftarrow \text{PRF}(MS, \text{"key expansion"} \parallel N_s \parallel N_c)$$
$$C_1 \leftarrow E(K_1, \text{Message})$$

C1


$$\text{Message} \leftarrow D(K_1, C_1)$$

C2


$$C_2 \leftarrow E(K_2, \text{Message}')$$
$$\text{Message}' \leftarrow D(K_2, C_2)$$



# Primitives used by TLS

← CERT = (pk of bank, signature over it)

Digital signatures

C →

Public-key encryption  
(RSA)

ChangeCipherSpec,  
{ Finished, PRF(MS, "Client finished" || H(transcript)) }

PRF  
Hash function

C1 →

← C2

Symmetric encryption

TLS was built via “design-**break**-redesign-**break**...”

We're now at TLS ver 1.2

~~No (publicly) known attacks~~

Did the TLS designers get it right?

We recently showed some new attacks against TLS 1.2 record layer  
(Paterson, Ristenpart, Shrimpton 2011)

Even for “simple” applications (secure channels), secure cryptography is **really hard to design**. The problems are rarely in primitives.

Many other tools have similar story:

SSH, IPsec, Kerberos, WEP/WPA (WiFi security), GSM (cell phone networks), ...

“Those who cannot remember the past are condemned to repeat it”

[Santayana 1905]

# Provable security cryptography

Supplement “design-**break**-redesign-**break**...” with a more **mathematical approach**

1. Design a cryptographic scheme
2. Provide **proof** that no one is able to break it



Shannon 1949

Formal definitions

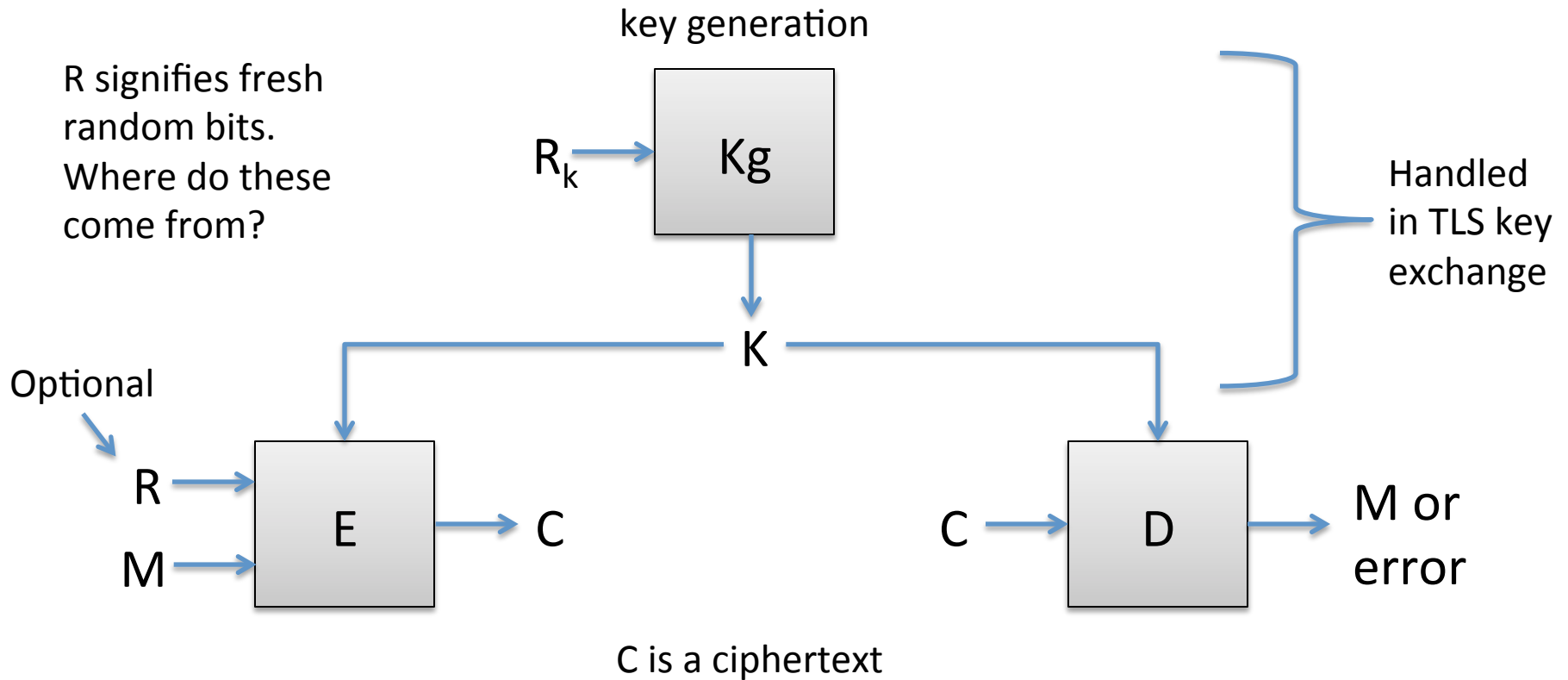
Scheme semantics

Security

Security proofs

Show it is mathematically impossible to break security

# Symmetric encryption



Correctness:  $D(K, E(K, M, R)) = M$  with probability 1 over randomness used

Kerckhoffs' principle: what parts are public and which are secret?

# Some attack settings (not security definitions)

- Unknown plaintext
  - attacker only sees ciphertexts
- Known plaintext
  - attacker knows some plaintext-ciphertext pairs
- Chosen plaintext
  - attacker can choose some plaintexts and receive encryptions of them





THE  
STRONG  
WIRING  
ENIGMA  
MACHINE

# One-time pads

Fix some message length  $L$

$K_g$ : output random bit string  $K$  of length  $L$

$$E(K, M) = M \oplus K$$

$$D(K, C) = C \oplus K$$



# Shannon's security notion

Def. A symmetric encryption scheme is **perfectly secure** if for all messages  $M, M'$  and ciphertexts  $C$

$$\Pr[ E(K, M) = C ] = \Pr[ E(K, M') = C ]$$

where probabilities are over choice of  $K$

In words:

each message is equally likely to map to a given ciphertext

In other words:

seeing a ciphertext leaks nothing about what message was encrypted

Does a substitution cipher meet this definition?

# Shannon's security notion

Def. A symmetric encryption scheme is **perfectly secure** if for all messages  $M, M'$  and ciphertexts  $C$

$$\Pr[ E(K, M) = C ] = \Pr[ E(K, M') = C ]$$

where probabilities are over choice of  $K$

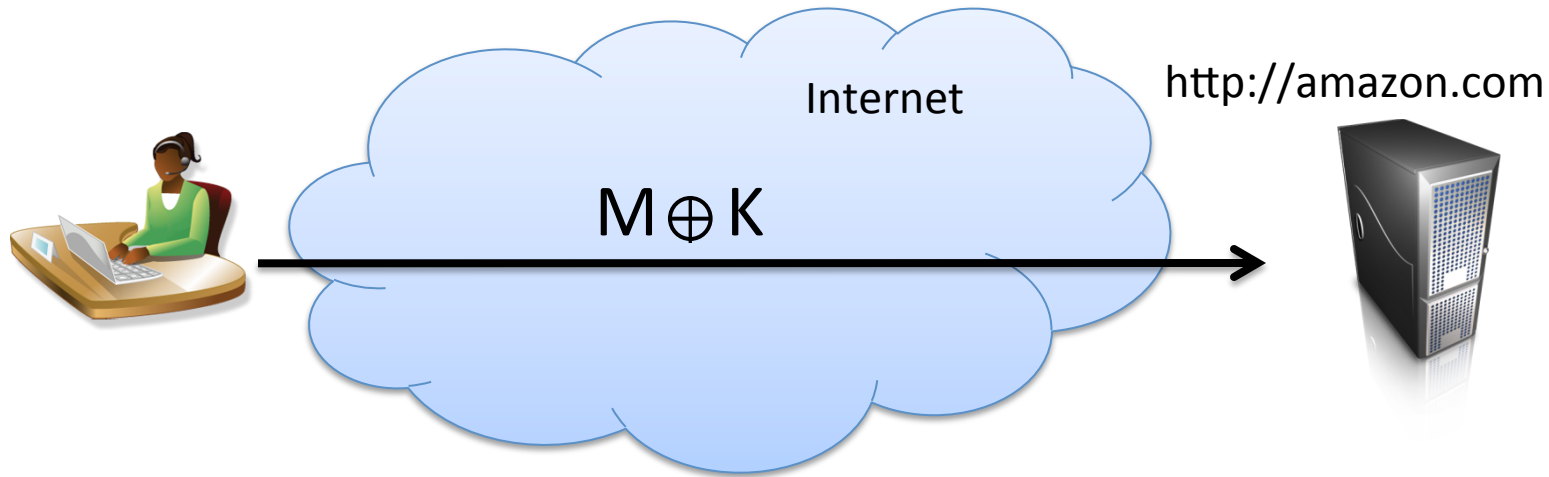
Thm. OTP is **perfectly secure**

For any  $C$  and  $M$  of length  $L$  bits

$$\Pr[ K \oplus M = C ] = 1 / 2^L$$

$$\Pr[ K \oplus M = C ] = \Pr[ K \oplus M' = C ]$$

## Back to our application



Does OTP provide a secure channel?

Integrity easily violated

Reuse of  $K$  for messages  $M, M'$  leaks  $M \oplus M'$

Encrypting same message twice leaks the fact

$K$  must be as large as message

Message length revealed

# Cryptography as **computational science**

Use computational intractability as basis for confidence in systems

1. Design a cryptographic scheme
2. Provide **proof** that no attacker with limited computational resources can break it



Goldwasser, Micali and Blum circa 1980's

Formal definitions  
Scheme semantics  
Security

Security proofs (reductions)

Breaking scheme



Breaking assumptions

Example:

**Attacker can **not** recover credit card**



Can **not** factor large composite numbers

As long as assumptions holds we believe in security of scheme!

Provable security yields

- 1) **well-defined assumptions and security goals**
- 2) **attackers (cryptanalysts) can focus on assumptions**

But no one knows how to do this. It's been studied for a very long time!

# Typical assumptions

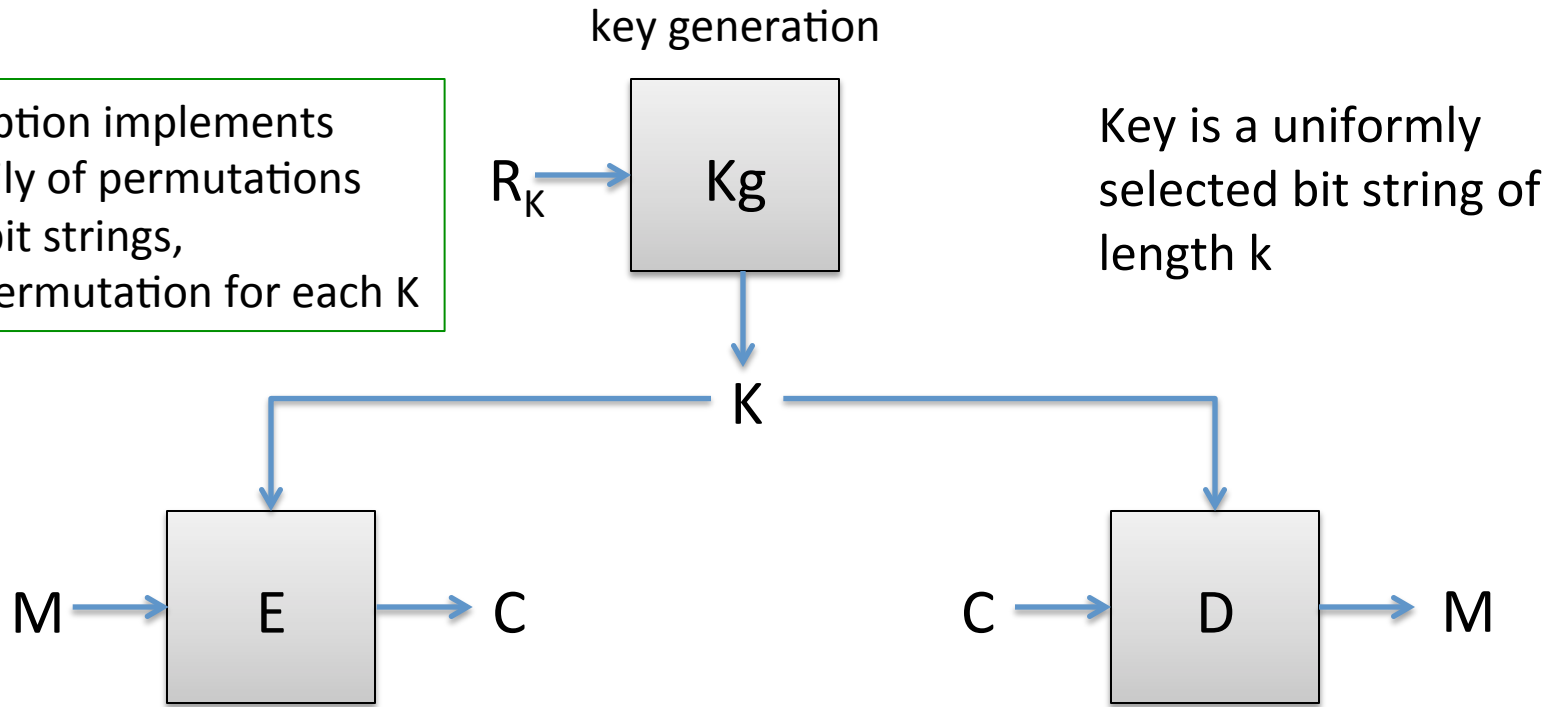
- Basic atomic primitives are hard to break:
  - Factoring of large composites intractable
  - RSA permutation hard-to-invert
  - Block ciphers (AES, DES) are good pseudorandom permutations (PRPs)
  - Hash functions are collision resistant

Confidence in atomic primitives is gained by cryptanalysis,  
public design competitions

SHA-3 just selected!

# Block ciphers

Encryption implements a family of permutations on  $n$  bit strings, one permutation for each  $K$



$$E: \{0,1\}^k \times \{0,1\}^n \rightarrow \{0,1\}^n$$

# Data encryption standard (DES)

Originally called Lucifer

- team at IBM
- input from NSA
- standardized by NIST in 1976

$n = 64$

$k = 56$

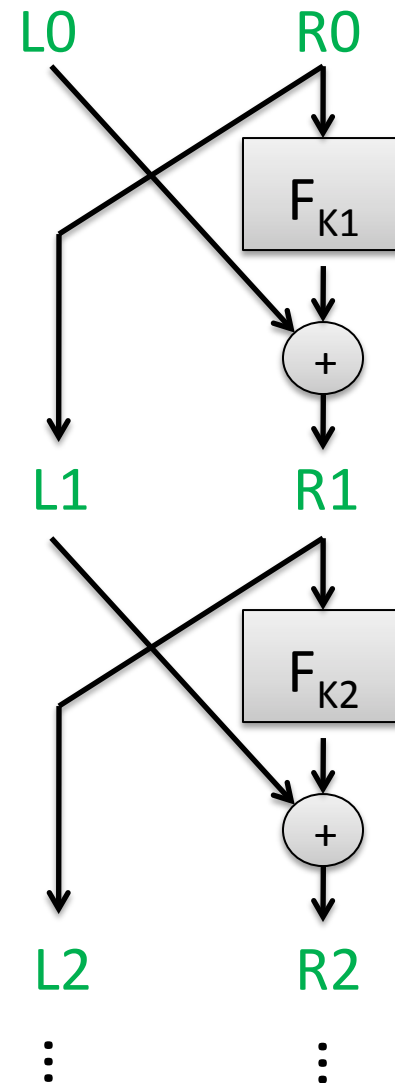
Number of keys:

72,057,594,037,927,936

Split 64-bit input into  $L_0, R_0$  of 32 bits each

Repeat Feistel round 16 times

Each round applies function  $F$  using separate round key



# Best attacks against DES

Attack	Attack type	Complexity	Year
Biham, Shamir	Chosen plaintexts, recovers key	$2^{47}$ plaintext, ciphertext pairs	1992
DESCHALL	Unknown plaintext, recovers key	$2^{56/4}$ DES computations 41 days	1997
EFF Deepcrack	Unknown plaintext, recovers key	~4.5 days	1998
Deepcrack + DESCHALL	Unknown plaintext, recovers key	22 hours	1999

- DES is still used in some places
- 3DES (use DES 3 times in a row with more keys) expands keyspace and still used widely in practice



# Advanced Encryption Standard (AES)

Response to 1999 attacks:

- NIST has design competition for new block cipher standard
- 5 year design competition
- 15 designs, Rijndael design chosen

# Advanced Encryption Standard (AES)

Rijndael (Rijmen and Daemen)

$n = 128$

$k = 128, 192, 256$

Number of keys for  $k=128$ :

340,282,366,920,938,463,463,374,607,431,768,211,456

Substitution-permutation design.

$k=128$  has 10 rounds of:

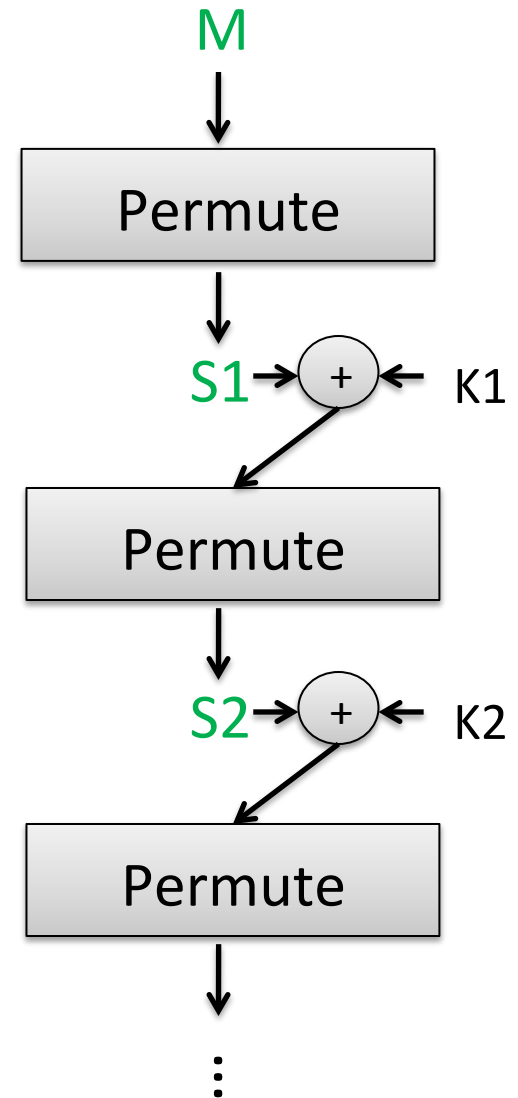
1) Permute:

SubBytes (non-linear S-boxes)

ShiftRows + MixCols (invertible linear transform)

2) XOR in a round key derived from  $K$

(Actually last round skips MixCols)



# Best attacks against AES

Attack	Attack type	Complexity	Year
Bogdanov, Khovratovich, Rechberger	chosen ciphertext, recovers key	$2^{126.1}$ time + some data overheads	2011

- Brute force requires time  $2^{128}$
- Approximately factor 4 speedup