Software Security Foundations:
Crypto concepts

Dan Boneh, Stanford University

STANFORD UNIVERSITY
Stanford Center for Professional Development

Stanford Center for Professional Development

Cryptography

Is:

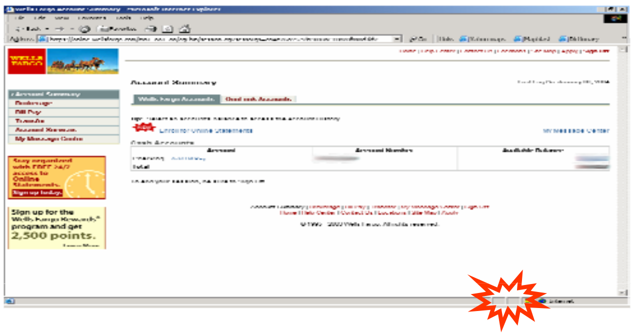
- A tremendous tool
- The basis for many security mechanisms

Is not:

- The solution to all security problems
- Reliable unless implemented and used properly
- Something you should try to invent yourself

Dan Boneh

Goal 1: Secure communication



no eavesdropping
no tampering

Dan Boneh

Secure Sockets Layer / TLS

Standard for Internet security

- Goal: "... provide privacy and reliability between two communicating applications"

<u>Two main parts</u>

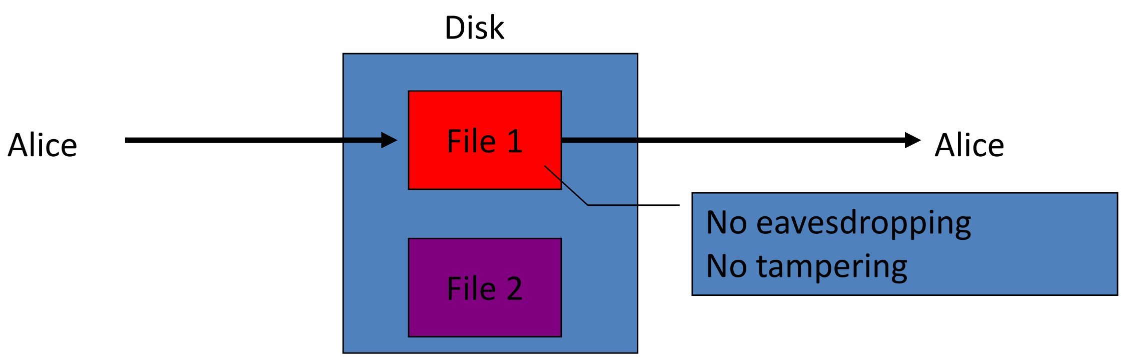
1. Handshake Protocol: **Establish shared secret key using public-key cryptography**

2. Record Layer: **Transmit data using negotiated key**

   Our starting point: Using a key for encryption and integrity

Dan Boneh

Goal 2: protected files

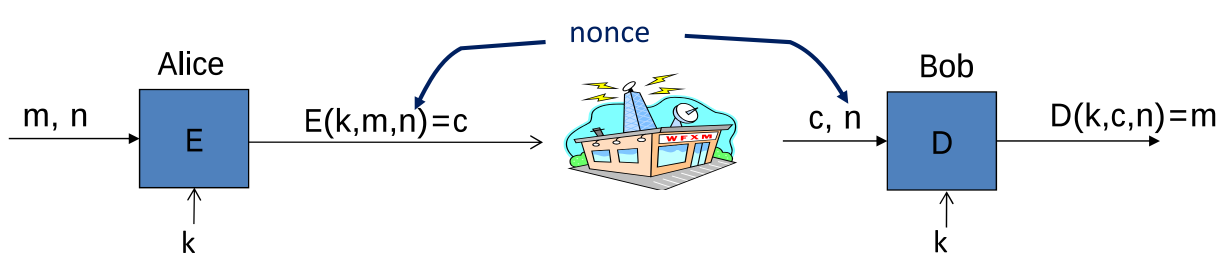


Analogous to secure communication:

Alice today sends a message to Alice tomorrow

Dan Boneh

Building block: sym. encryption



E, D: cipher k: secret key (e.g. 128 bits)

m, c: plaintext, ciphertext n: nonce

Encryption algorithm is publicly known

⇒ never use a proprietary cipher

Dan Boneh

Use Cases

<u>Single use key</u>: (one time key)

- Key is only used to encrypt one message
    - encrypted email: new key generated for every email
- No need for nonce (set to 0)

<u>Multi use key</u>: (many time key)

- Key used to encrypt multiple messages
    - SSL: same key used to encrypt many packets
- Need either *unique* nonce or *random* nonce

Dan Boneh

First example: One Time Pad (single use key)

Vernam (1917)

Key:	0	1	0	1	1	1	0	0	1	0	$\oplus$
Plaintext:	1	1	0	0	0	1	1	0	0	0	
Ciphertext:	1	0	0	1	1	0	1	0	1	0	

Encryption: $c = E(k, m) = m \oplus k$

Decryption: $D(k, c) = c \oplus k = (m \oplus k) \oplus k = m$

Dan Boneh

One Time Pad (OTP) Security

Shannon (1949):

- OTP is "secure" against one-time eavesdropping
- without key, ciphertext reveals no "information" about plaintext

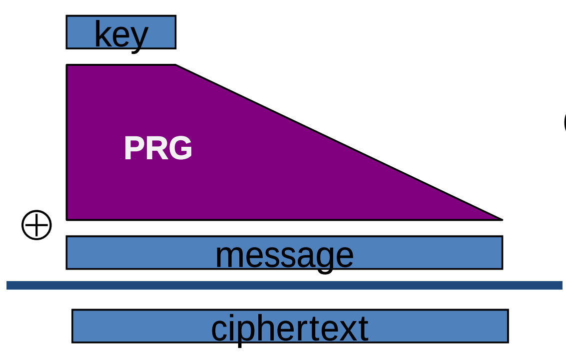
Problem: OTP key is as long the message

Dan Boneh

Stream ciphers (single use key)

Problem: OTP key is as long the message

<u>Solution</u>: Pseudo random key -- stream ciphers



$$c \leftarrow \mathbf{PRG}(k) \oplus m$$

Examples: **Salsa20/12** (643MB/s) , **Sosemanuk** (727MB/s), **RC4** (126MB/s)

Dan Boneh

Dangers in using stream ciphers

One time key !! "Two time pad" is insecure:

$$C_1 \leftarrow m_1 \oplus PRG(k)$$
$$C_2 \leftarrow m_2 \oplus PRG(k)$$

Eavesdropper does:

$$C_1 \oplus C_2 \rightarrow m_1 \oplus m_2$$

Enough redundant information in English that:

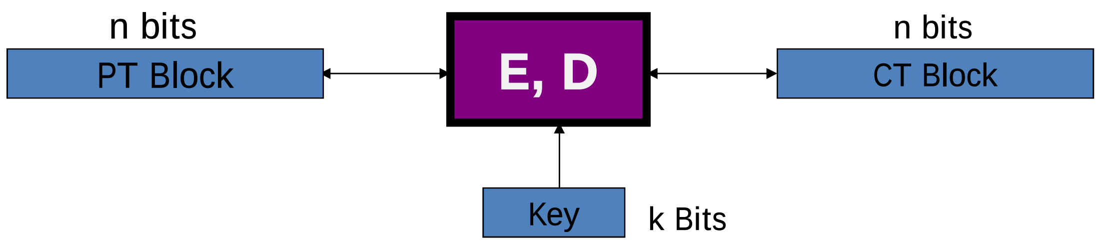
$$m_1 \oplus m_2 \rightarrow m_1 , m_2$$

Dan Boneh

Crypto concepts

Block ciphers

Block ciphers: crypto work horse

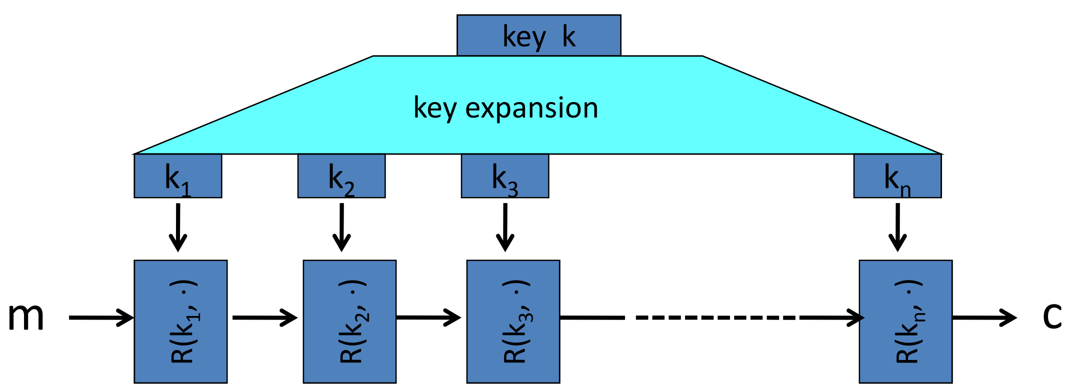


Canonical examples:

1. 3DES: n= 64 bits, k = 168 bits
2. AES: n=128 bits, k = 128, 192, 256 bits

Dan Boneh

Block Ciphers Built by Iteration



R(k,m): round function

for 3DES (n=48), for AES-128 (n=10)

Dan Boneh

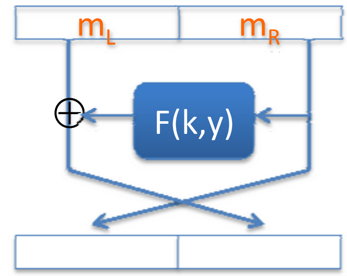
Standard Block Ciphers

Input: m, k

Repeat simple mixing operation several times

- 3**DES**: Repeat 48 times:

$$\begin{cases} x_L \leftarrow m_R \\ x_R \leftarrow m_L \oplus F(k_i, m_R) \end{cases}$$



- **AES-128**: Mixing step repeated 10 times (x86 HW support)

Difficult to design: must resist subtle attacks

- differential attacks, linear attacks, brute-force, …

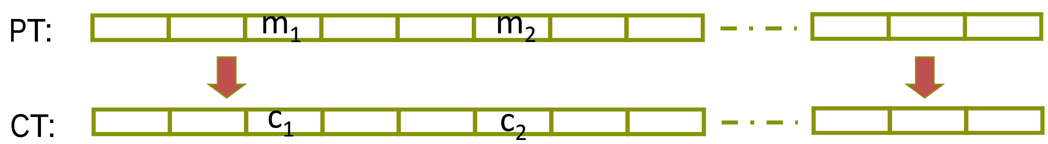
Dan Boneh

Crypto concepts

Using block ciphers

Incorrect use of block ciphers

Electronic Code Book (ECB):

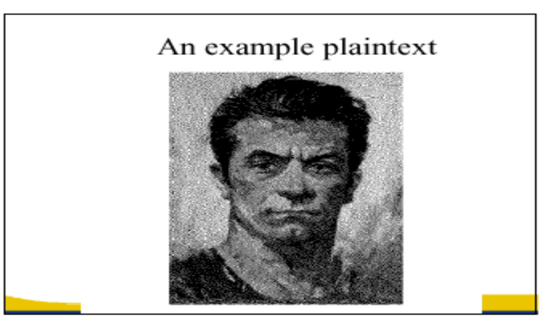


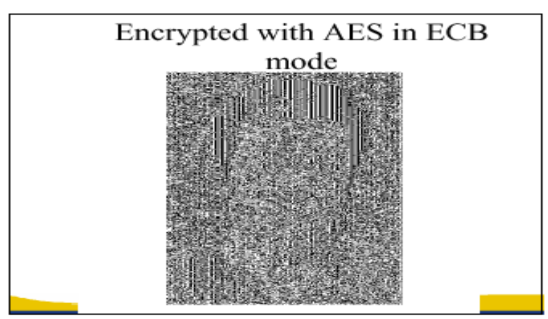
Problem:

– if $m_1=m_2$ then $c_1=c_2$

Dan Boneh

In pictures

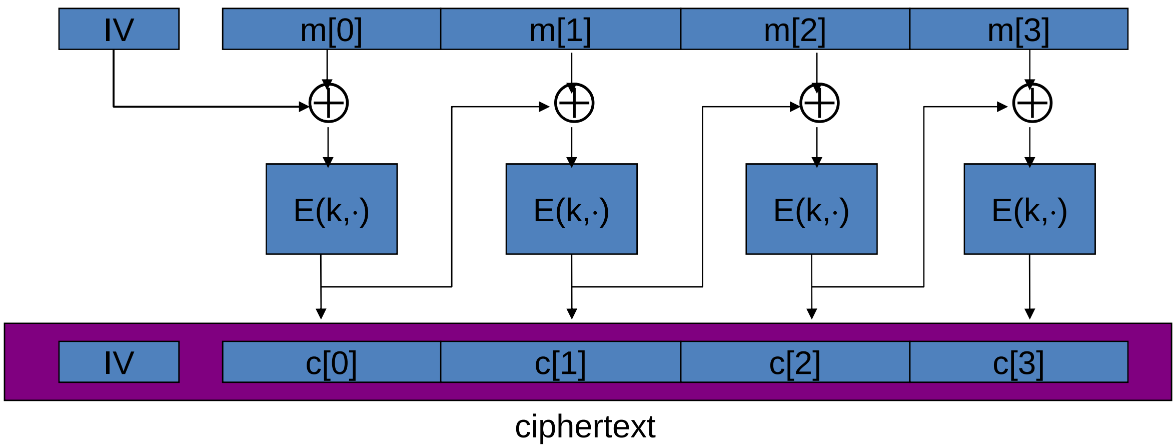




Dan Boneh

Eavesdropping security 1: CBC mode

E: block cipher. <u>Cipher Block Chaining</u> with random IV :



Dan Boneh

Use cases: how to choose an IV

Single use key: no IV needed (IV=0)

Multi use key: (a.k.a chosen plaintext security)

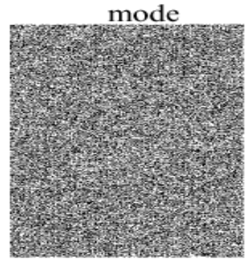
- Best: use a fresh ***<u>random</u>*** IV for every message

Dan Boneh

In pictures

An example plaintext

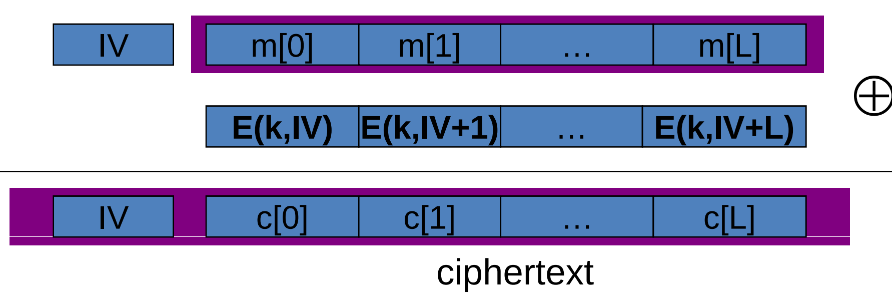
Encrypted with AES in CBC mode



Dan Boneh

Eavesdropping security 2: CTR mode

Counter mode with a random IV: (parallel encryption)



Why are these modes secure? See the crypto course.

Dan Boneh

Performance:

Crypto++ 5.6.0 [Wei Dai]

AMD Opteron, 2.2 GHz (Linux)

	Cipher	Block/key size	Speed (MB/sec)
stream	Salsa20/12		643
	Sosemanuk		727
block	3DES	64/168	13
	AES	128/128	109

Dan Boneh

A Warning

eavesdropping security is insufficient for most applications

Need also to defend against active attacks.

CBC and CTR modes are insecure against active attacks

Next: methods to ensure message integrity

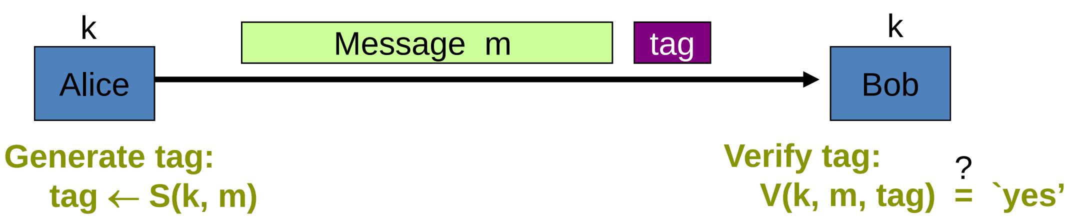
Dan Boneh

Crypto concepts

Message Integrity

Message Integrity: MACs

- Goal: provide message integrity. No confidentiality.
  - ex: Protecting public binaries on disk.



note: non-keyed checksum (CRC) is an insecure MAC !!

Dan Boneh

Secure MACs

Attacker's power: **chosen message attack**

- for $m_1, m_2, \ldots, m_q$ attacker is given $t_i \leftarrow S(k, m_i)$

Attacker's goal: **existential forgery**

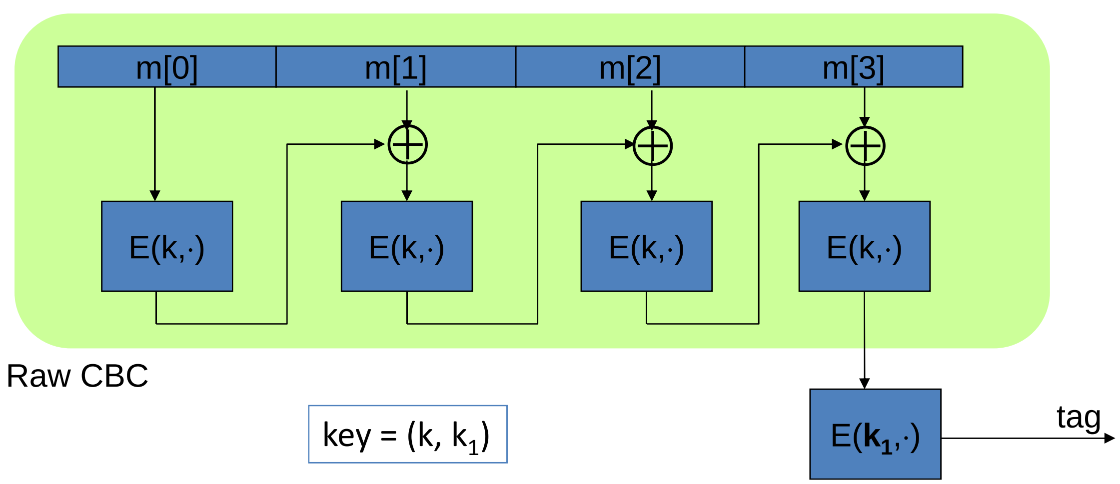
- produce some <u>**new**</u> valid message/tag pair (m,t).

$$(m,t) \notin \{ (m_1,t_1), \ldots, (m_q,t_q) \}$$

Dan Boneh

Construction 1: ECBC

E: a block cipher



Dan Boneh

Construction 2: HMAC (Hash-MAC)

Most widely used MAC on the Internet.

H: hash function.
example: SHA-256 ; output is 256 bits

Building a MAC out of a hash function:

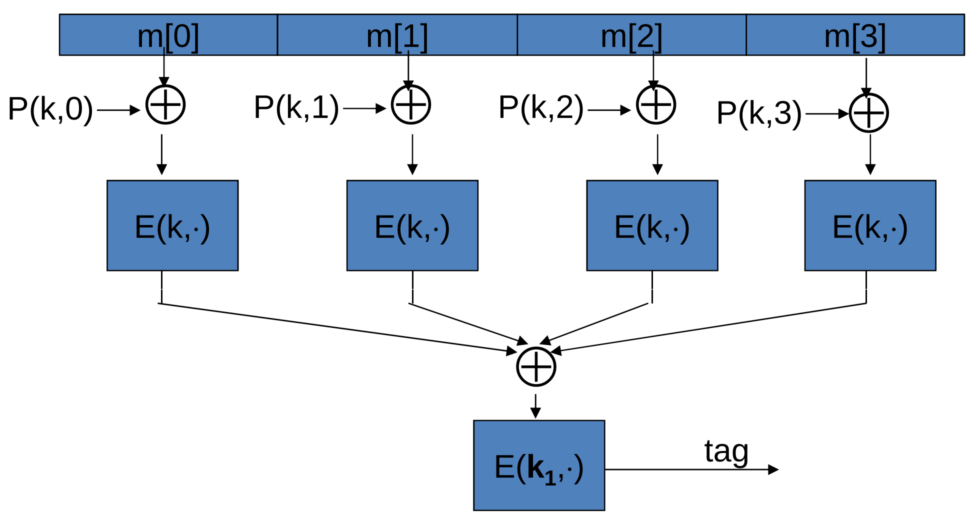
– Standardized method: HMAC

$$S(k, m) = H\big(\ k \oplus opad\ ,\ \mathbf{H(\ k \oplus ipad\ ,\ m\ )}\ \big)$$

Dan Boneh

Construction 3: PMAC -- a parallel MAC

ECBC and HMAC are sequential. PMAC:



Dan Boneh

Why are these MAC constructions secure?

… take the crypto course

Why the last encryption step in ECBC?

- CBC (aka raw-CBC) is not a secure MAC:
  - Given tag on a message m, attacker can deduce tag for some other message m'
  - How: good exercise

Dan Boneh

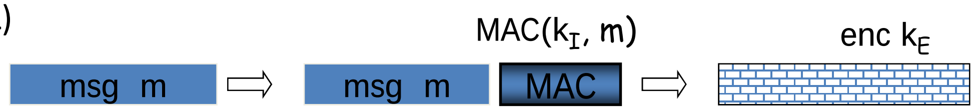
Crypto concepts

Authenticated Encryption

Combining MAC and ENC (CCA)

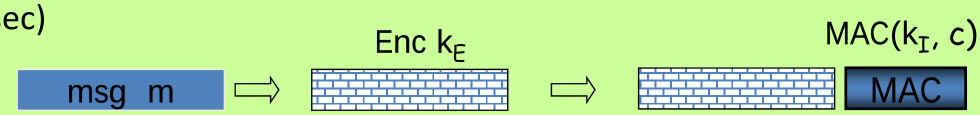
Encryption key k_E. MAC key = k_I

Option 1: (SSL)

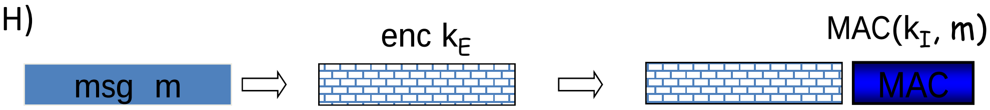


Option 2: (IPsec)

always correct



Option 3: (SSH)

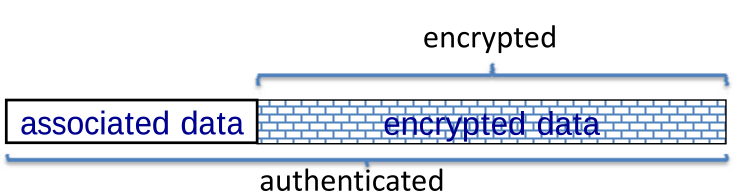


Dan Boneh

Standards (at a high level)

- CCM: CBC-MAC then CTR mode encryption
- GCM: CTR mode encryption then MAC (x86 HW support)
- EAX: CTR mode encryption then CBC-MAC

All support AEAD: (auth. enc. with associated data)



Dan Boneh

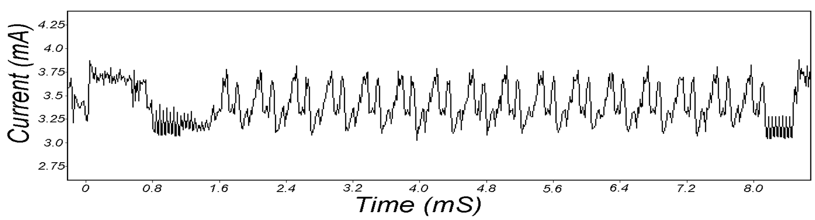
Implementation problems: side channels

Power analysis (Kocher-Jaffe-Jun 99)

– Power consumption depends on instruction and data

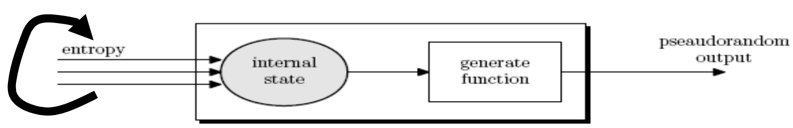
– Measure power consumption during block cipher operation

– About 1000 ciphertexts suffice to expose secret key.



Dan Boneh

Generating Randomness (e.g. keys, nonces)



Pseudo random generators in practice: (e.g. /dev/random)

- Continuously add entropy to internal state
- Entropy sources:
  - Hardware RNG: Intel **RdRand** inst. (Ivy Bridge). 3Gb/sec.
  - Timing: hardware interrupts (keyboard, mouse)

NIST SP 800-90: NIST approved generators

Dan Boneh

Summary

Shared secret key:

- Used for secure communication and document encryption

Encryption: (eavesdropping security) **[should not be used standalone]**

- One-time key: stream ciphers, CBC or CTR with fixed IV
- Many-time key: CBC or CTR with random IV

Integrity: ECBC or HMAC or PMAC

Authenticated encryption: encrypt-then-MAC

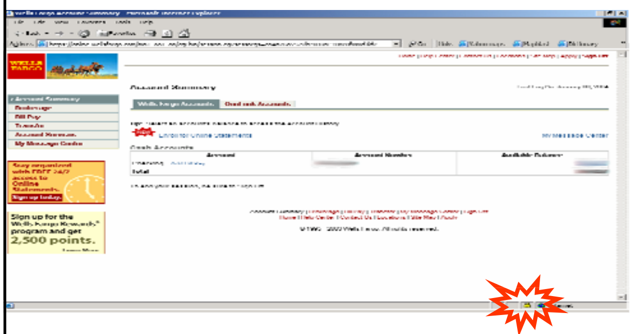
Dan Boneh

Software Security Foundations: *Crypto concepts II*

Dan Boneh, Stanford University

STANFORD UNIVERSITY
Stanford Center for Professional Development

Secure communication



Authenticated channel
privacy + integrity

This segment: how do we generate session key?

Dan Boneh