Section 3: Symmetric Encryption

How can two parties use a common secret to communicate privately in the presence of eavesdroppers?

Slides by current and former TAs

Administrivia

HW 1 (Threat Modeling) due 4/23 @ 11:59 pm

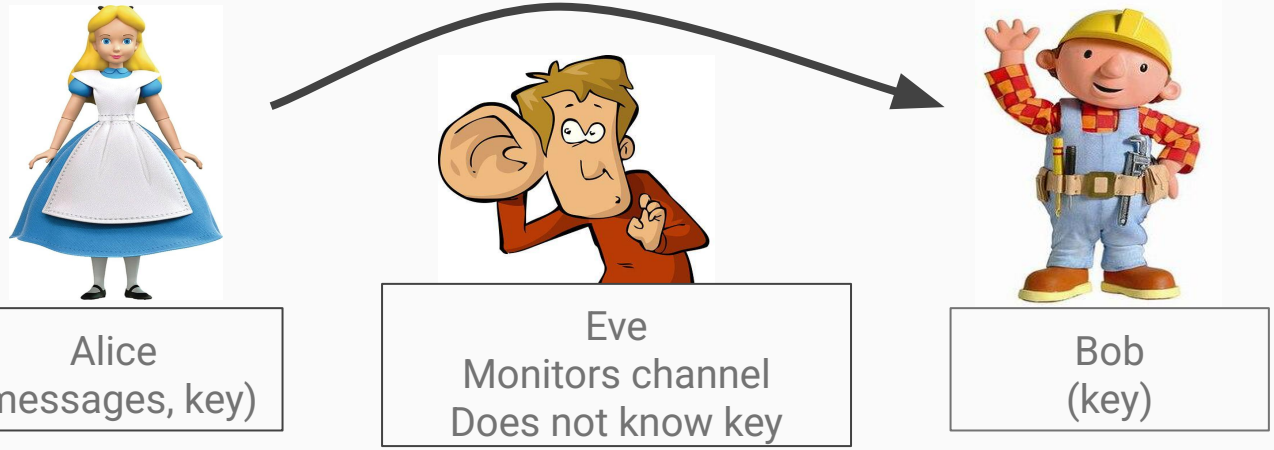
Lab 2 out!

- 3 required programming problems + 1 extra credit (not much harder)
- Wait until you learn about RSA for goldmine-of-ps-and-qs
- Individual write-ups
- Individual written problems (separate from the lab)

It is a new lab, please be patient with us!

Symmetric Encryption: Goal

Setting:



Goal: Bob wants to learn Alice's messages.

Security (informally): Eve does not learn anything about the messages, even when seeing many ciphertexts (and seeing ciphertexts for messages of his choice).

Symmetric Encryption: scheme syntax

Alice
(message, key)

Eve
Monitors channel
Does not know key

Bob
(key)

```
Key generation():
# How Alice and Bob
pick their key.
Usually:
    Pick random key k
    Return k
```

```
Encrypt(k, pt):
# How Alice encrypts her
message (pt) using the
key (k).
# Returns a ciphertext.
```

```
Decrypt(k, ct):
# How Bob decrypts
ciphertext (ct) using
the key (k).
# Returns a plaintext.
```

Important Tool: Block Ciphers

“The workhorse of cryptography”

Definition: a function B that takes a key and a single chunk (“block”) of plaintext as input and outputs a “cipher block.”

- Block sizes on the order of 64 bits, 128 bits, 256 bits.
- Key size does not have to equal the block size.
- Each key defines a “random” invertible permutation of inputs to possible outputs.

Security: Without knowledge of the key, f(x) = B(Key, x) looks like a random permutation on the plaintext blocks.

Important Tool: Block Ciphers

Building a secure block cipher: Not covered in this class

- Not much theoretical understanding of the block ciphers in use
- In Lab2 you break an insecure block cipher (2DES)

This section: building a secure encryption scheme, assuming you have a secure block cipher.

- Related to Lab2 “don’t count on counters” and “confession by compression” problems.

Scheme 1: Block Ciphers Only

Assume we have a secure block-cipher B with block size 128. Consider the following encryption scheme for 128-bit plaintext:

```
Key generation:
    Pick random key k
    Return k
```

```
Encrypt(k, pt):
    ct = B(k, pt)
    Return ct
```

```
Decrypt(k, ct):
Activity 1: How to decrypt?
```

Scheme 1: Block Ciphers Only

Assume we have a secure block-cipher B with block size 128. Consider the following encryption scheme for 128-bit plaintext:

```
Key generation:
    Pick random key k
    Return k
```

```
Encrypt(k, pt):
    ct = B(k, pt)
    Return ct
```

```
Decrypt(k, ct):
Activity 1: How to decrypt?
    pt = inverse_B(k, ct)
    return pt
```

Is this scheme secure?

NO! Encrypting the same plaintext twice results in the same ciphertext.

Other problems:

Can only encrypt one block at a time.

Scheme 1: Conclusion

A secure encryption scheme MUST encrypts the same message to a different ciphertext each time (with high probability).

How?

Include randomness in the encryption algorithm

Scheme 2: Block cipher with randomization

Assume we have a secure block-cipher B with block size 128.
Consider the following encryption scheme for 128-bit plaintext:

```
Key generation:
    Pick random key k
    Return k
```

```
Encrypt(k, pt):
    Pick random 128bit IV
    ct = IV || (B(k, IV) XOR pt)
    Return ct
```

```
Decrypt(k, ct):
Activity 2: How to decrypt?
```

Scheme 2: Block cipher with randomization

Assume we have a secure block-cipher B with block size 128.
Consider the following encryption scheme for 128-bit plaintext:

```
Key generation:
    Pick random key k
    Return k
```

```
Encrypt(k, pt):
    Pick random 128bit IV
    ct = IV || (B(k, IV) XOR pt)
    Return ct
```

```
Decrypt(k, ct):
Activity 2: How to decrypt?
    Parse ct as (IV || ct_block)
    pt = ct_block XOR B(k, IV)
    return pt
```

Is this scheme secure?

Yes (can be proved assuming security of B)

Problems:

Can only encrypt one block plaintexts.

Scheme 3: Same thing, more blocks!

Assume we have a secure block-cipher B with block size 128.

Consider the following encryption scheme for plaintext blocks $pt_1, \ldots, pt_n$:

```
Encrypt(k, pt_1, …, pt_n):
    For i from 1 to n:
        pick random IV_i
        ct_i = IV_i || (B(k, IV_i) XOR pt_i)
    ct = ct_1 || … || ct_n
    Return ct
```

```
Decrypt(k, ct):
Activity 3: How to decrypt?
```

Scheme 3: Same thing, more blocks!

Assume we have a secure block-cipher B with block size 128.

Consider the following encryption scheme for plaintext blocks $pt_1, \ldots, pt_n$:

```
Encrypt(k, pt_1, …, pt_n):
    For i from 1 to n:
        pick random IV_i
        ct_i = IV_i || (B(k, IV_i) XOR pt_i)
    ct = ct_1 || … || ct_n
    Return ct
```

```
Decrypt(k, ct):
    Parse ct as ct_1 || … || ct_n
    For i from 1 to n:
        Parse ct_i as IV_i || ct_block_i
        pt_i = ct_block_i XOR B(k, IV_i)
    Return pt_1, …, pt_n
```

Is this scheme secure?

Yes (can be proved assuming security of B)

Problems:

Ciphertext is twice the length of the plaintext

Scheme 4: Insecure Shortcut

Assume we have a secure block-cipher B with block size 128.

Consider the following encryption scheme for plaintext blocks $pt_1, \ldots, pt_n$:

```
Encrypt(k, pt_1, …, pt_n):
    pick random IV
    For i from 1 to n:
        ct_i = B(k, IV) XOR pt_i
    ct = IV || ct_1 || … || ct_n
    Return ct
```

```
Decrypt(k, ct):
Activity 4: How to decrypt?
```

Scheme 4: Insecure Shortcut

Assume we have a secure block-cipher B with block size 128.
Consider the following encryption scheme for plaintext blocks $pt_1, \ldots, pt_n$:

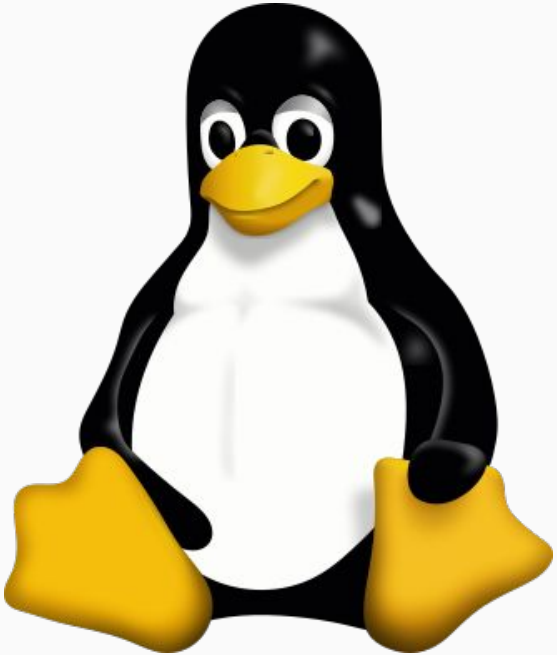
```
Encrypt(k, pt_1, …, pt_n):
    pick random IV
    For i from 1 to n:
        ct_i = B(k, IV) XOR pt_i
    ct = IV || ct_1 || … || ct_n
    Return ct
```

```
Decrypt(k, ct):
    Parse ct as IV || ct_1 || … || ct_n
    For i from 1 to n:
        pt_i = ct_i XOR B(k, IV)
    Return pt_1, …, pt_n
```

Why is this scheme insecure?
Repeated plaintext blocks result in repeated ciphertext blocks

Advantage:
ct length = 128 + (pt length)

Scheme 4: Insecure Shortcut



“Encryption”

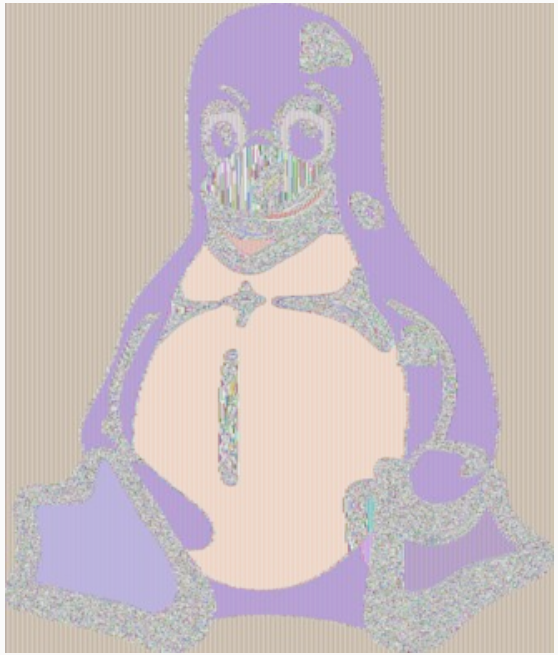


Photo credit: Wikipedia

Scheme 4: Conclusion

A secure encryption scheme MUST encrypt a repeated block to different ciphertext blocks (with high probability).

How?

Scheme 5: Best we got (CTR mode)

Assume we have a secure block-cipher B with block size 128.

Consider the following encryption scheme for plaintext blocks $pt_1, \ldots, pt_n$:

```
Encrypt(k, pt_1, …, pt_n):
    pick random IV
    For i from 1 to n:
        IV_i = (IV + i - 1) % 2^{128}
        ct_i = B(k, IV_i) XOR pt_i
    ct = IV || ct_1 || … || ct_n
    Return ct
```

```
Decrypt(k, ct):
Activity 5: How to decrypt?
```

Scheme 5: Best we got (CTR mode)

Assume we have a secure block-cipher B with block size 128.

Consider the following encryption scheme for plaintext blocks $pt_1, \ldots, pt_n$:

```
Encrypt(k, pt_1, …, pt_n):
    pick random IV
    For i from 1 to n:
        IV_i = (IV + i - 1) % 2^{128}
        ct_i = B(k, IV_i) XOR pt_i
    ct = IV || ct_1 || … || ct_n
    Return ct
```

```
Decrypt(k, ct):
    Parse ct as IV || ct_1 ||…|| ct_n
    For i from 1 to n:
        IV_i = (IV + i - 1) % 2^{128}
        pt_i = B(k, IV_i) XOR ct_i
    pt = pt_1 || … || pt_n
    Return pt
```

Is this scheme secure?

Yes! Can be proven, assuming B is a psuedo-random permutation

Advantage: ct length = 128 + (pt length)

Scheme 5 (CTR Mode): Diagram

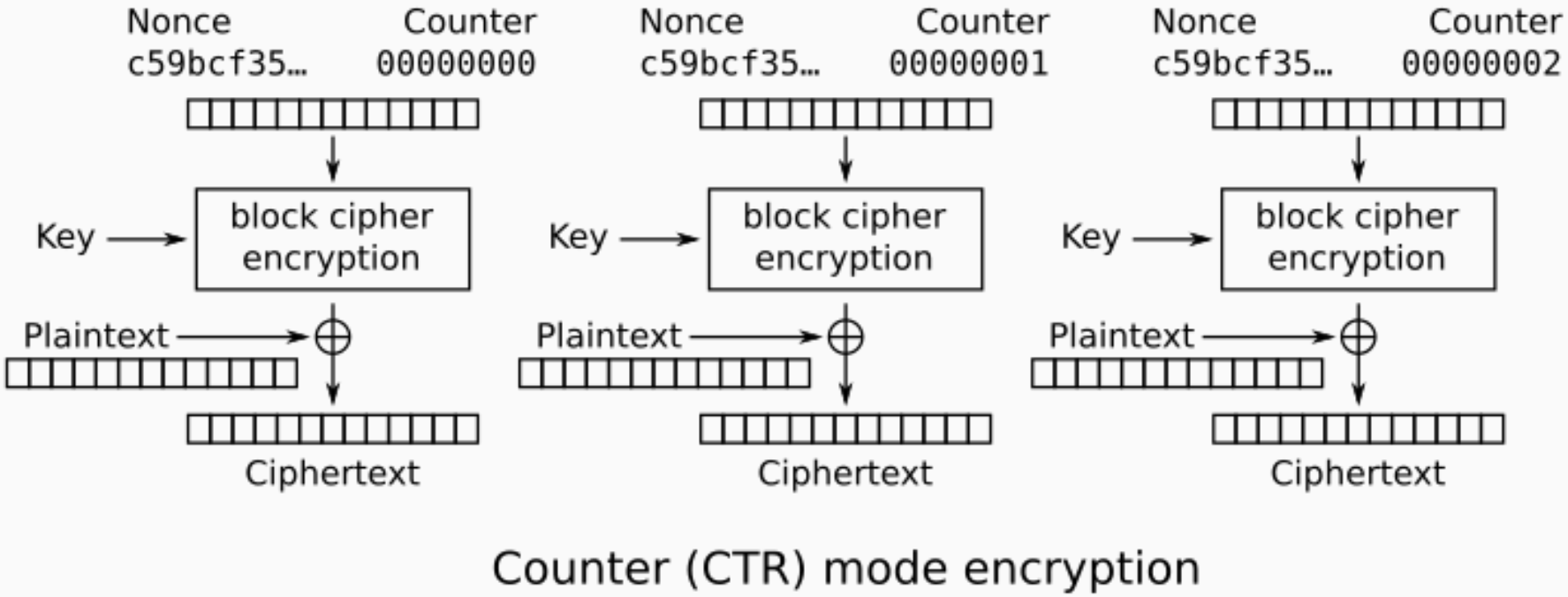


Diagram credit: Wikipedia

CTR Mode: Considerations

Other good things:

- decryption does not require inverting the block cipher
- plaintext padding often used but is not required
- IV is public, can be generated deterministically (but CAN'T use IV twice)

Bad things:

- UNAUTHENTICATED!
  - If I see a ciphertext, I can modify it to decrypt to a different plaintext without knowing the key
- Requires non-colliding IVs
- For provable security guarantees, you have to restart with a new IV often.