Three Attacks

1. ECB: The Penguin, and Worse (CPA)
2. Compression + CTR = Insecure (CPA)
3. Next time: Padding Oracles vs CBC (not a CPA)
**AES-ECB**

**AES-ECB(K, M)**
- Parse M into blocks \( M_1, M_2, \ldots, M_t \)
  - // all blocks except \( M_t \) are 16 bytes
- Pad \( M_t \) up to 16 bytes
- For \( i=1 \ldots t \):
  - \( C_i \leftarrow AES(K, M_i) \)
- Return \( C_1, \ldots, C_t \)
The ECB Penguin

Plaintext  ECB Ciphertext  CTR Ciphertext

Looks random
Detecting ECB in Practice

Repeated ciphertext blocks (when plaintext repeats blocks)

- Good modes like CTR and CBC basically never repeat a block
CPAs and Partial Plaintext Recovery

Picks user data (e.g. username)

 userdata=… ; token=uchic4gos3cr34

Encrypt()

3F C2 12 FF 32 12 99 D2 3E 42 4A 2C 27 DA 04 07

Observe ciphertext

- Goal: Recover token
- Capability: Submit user data, observe ciphertext (many times)
PKCS7 Padding

- Need to pad a byte string up to a multiple of 16 bytes
- First look at how many bytes are missing. Here, need 10 bytes
- Fill missing k bytes with value k (k = 10 = 0x0A in example)

- If data is already a multiple of 16 bytes long, add an entire block of 0x10 bytes

Can’t leave data unchanged;
Bytes might be interpreted as padding.

- Un-padding is easy
ECB and Partial Plaintext Recovery: One Byte

Plan: Learn AES(K, 0F0F0F0F0F0F0F0F0F0F0F0F0F0F0F) for all 256 values of .
- Easily read off value of last byte
Learning the Padded Blocks

```
username = 0x??0F0F0F0F0F0F0F0F0F0F0F0F0F0F0F
```

```
userdata=... ; token=uchic4gos3cr34
```

- 256 Queries
  - Easy if username is block-aligned.
  - Harder but still possible if not aligned, or if alignment is random
  - Need to pre-pend username with bytes to shift it around
ECB and Partial Plaintext Recovery: Second Byte

(username one char longer)

- Plan: Learn AES(K, ??770E0E0E0E0E0E0E0E0E0E0E0E0E) for all 256 values of ??.
- Easily read off value of next byte
- Continue for each byte, then move onto next block
Compress-then-CTR-Encrypt

AES-CTR()

Compress()
Compression Attack Setting: Browser Data

Real-world attacks against:
- TLS (2012)
- HTTPS (2013)

Adversary can see ciphertext length.

More overlap with secret means better compression, i.e., shorter ciphertext.

PPS 2: Seeing several ciphertexts enables full plaintext recovery.

Secret info

Username, etc (Adversary controlled)
The End