EECS 388 Discussion 2

Project 1: Length Extension & Hash Collision

Length Extension Attack

- Why use a MAC instead of a hash? (e.g. HMAC-SHA256 vs. SHA256)
  - Merkle-Damgård construction
- Given an MD5 hash for a message *m*, we can calculate a valid hash of a longer message
  - We don’t even need to know m - just it’s length
  - SHA-1 and SHA-256 also vulnerable

Length Extension: Padding

- MD5 processes 512-bit blocks, will pad messages to a multiple of that length
- The bit 1, followed by zeros, then a 64 bit integer indicating the amount of padding
- If the 1 and the number don’t fit, adds an extra block

Length Extension (cont.)

- MD5(m) => MD5(m + padding + suffix)
  - Remember Merkel-Damgård construction?
  - Initialize MD5 algorithm with MD5(m), add blocks
- Try it out: Crypto Project Part 1.1
  - Download:

    https://www.eecs.umich.edu/courses/eecs388/static/pymd5.py

Hash Collision: Background

- MD5 used to be widely used on the web
  - Broken in 2004 - efficient collision algorithm
  - Now dangerously insecure
- Why are Hash Collisions bad?
  - How could a malicious user use hash collisions?

Hash Collision Attack

- MD5 lets us construct 2 different messages with the same hash
  - prefix || $\text{blob}_{A/B}$ || suffix
  - prefix and suffix are the same, binary blob is different
  - Why is this possible?

BYOC (Build Your Own Collision)

- Crypto Project Part 2.1

Download:

http://www.win.tue.nl/hashclash/fastcoll_v1.0.0.5-1_source.zip

https://www.eecs.umich.edu/courses/eecs388/static/project1/Makefile

```
apt-get install libboost-all-dev
time fastcoll -o file1 file2
```