Class Notes for November 22

Name of Author

December 8, 2016

Recall basic goals of cryptography:

- Confidentiality (Eve can't get information about a message)
- Data Integrity (Eve can't modify a message)
- Authentication (Bob knows that Alice sent a message)
- Non-Repudiation (Bob can prove that Alice sent the message)

Hash Functions

A hash function $h(x) \to m$ is a function that takes a long string x and shortens it to a much shorter string m. The output has a fixed length but size of x may be variable. We call the output of a hash function the digest or hash or hashcode of x. One thing we can use a hash function for is data integrity.

Rather than just sending a message m, Alice sends (m, h(m)).

Bob checks if the m he recieves is equal to the hash Alice sent.

If it's not equal, then the message was wrong.

Example: Let h(x) be the sum of the binary digits of x reduced mod 2 (parity bit function). Test to see if a bit was flipped in transmission. Works for accidental mistakes, not for malicious ones.

Ideal Properties of a Hash Function:

- 1. Fast to compute
- 2. Produce a fixed length (short) output

Say a hash function has a collision if $h(x_1) = h(x_2)$ for $x_1 \neq x_2$.

Ideal Collision Properties of a Hash Function

- 1. Preimage resistant: given y it is hard to find x such that y = h(x) (hard to undo the hash function)
- 2. Weak collision resistant: given a message x_1 it should be hard to find another message x_2 where $h(x_1) = h(x_2)$

3. Strong collision resistant: It is hard to find any two inputs x_1 and x_2 where $h(x_1) = h(x_2)$.

These properties are increasingly stronger $3) \Rightarrow 2) \Rightarrow 1$). For use in encryption, we'd like a hash function with property 3.

Simple example: suppose we want outputs in the range 0 to n-1. Then we would choose $h(x) \equiv x \pmod{n}$.

This is fast, but not preimage resistant. To find x with h(x) = x, let x = y or y + n or $y + \ell n$.

Better example: Use discrete logarithms

Discrete Log Hash

Choose a prime q such that 2q + 1 = p is also prime. Pick different primitive roots (mod p). Call them α, β for some $m \in [0, q^2 - 1]$. $m = x_1 + qx_0$ (writing m in base q)

Define
$$h(m) = \alpha^{x_0} \beta^{x_1} \pmod{p}$$

Input can't be arbitrarily large $m \le q^2 - 1$ But h(m) so <math>h(m) is smaller than m. Output is "about" square root size of input.

This hash seems to have strong collision resistance. Prove this by showing that if we can find a collision, we can solve the discrete log problem. So since α, β are both primitive roots $(\mod p)$, there exists a such that $\alpha^a \equiv \beta \pmod{p}$.

Proposition: If we can find m_1 and m_2 with $h(m_1) = h(m_2)$ then we can find a (solve the discrete log problem $loga = L_{\alpha}(\beta)$)

Proof: write $m_1 = x_0 + x_1q$ and $m_1 = y_0 + y_1q$ then $h(m_1) = \alpha^{x_0}\beta^{x_1} \pmod{p}$ and $h(m_2) = \alpha^{y_0}\beta^{y_1} \pmod{p}$ if $h(m_1) = h(m_2)$ then $\alpha^{x_0}\beta^{x_1} \equiv \alpha^{y_0}\beta^{y_1} \pmod{p}$ $\alpha^{x_0-y_0}\beta^{x_1-y_1} \equiv 1 \pmod{p}$ since $\beta = \alpha^a$ $\alpha^{x_0-y_0}(\alpha^a)^{x_1-y_1} \equiv (\mod{p})$ $\alpha^{x_0-y_0+a(x_1-y_1)} \equiv 1 \pmod{p}$ Since α is a primitive root this means the exponent is a multiple of p-1so $(x_0 - y_0) + a(x_1 - y_1) \equiv 0 \pmod{p}$ $-a(x_1 - y_1) \equiv x_0 - y_0 \pmod{p-1}$ Solve this equation $(\mod{p-1})$ to find a. There can't be more than two possibilities for a (try them both).

So we solved DLP for $a = L_{\alpha}(\beta)$. Since we think this is hard, finding collisions must be hard.

Discrete Log hash is too slow to be used in practice. In practice, the hashes used are MD5, SHA-0, SHA-a, and RIPEMD-60.

Merkle-Damgård Construction

Used by most modern hashes. Suppose we have a function f takes in strings of length n and produces strings of length s. $\ell = n - s$ To hash m. pad it with enough zeroes so its length is a multiple of ℓ . Break the input into t string of length ℓ Initialize H to some fixed string of length S. For i in [1...t] $\mathbf{H} = \mathbf{f}(\mathbf{H} \parallel m_i)$ } output H Hard part is choosing a good f. This appears to be fairly secure (strongly collision resistant)

MD5 has outputs with 128 bits and SHA-1 has outputs with 160 bits.

In 2005 mathematicians found collisions for MD5 and SHA-0 (similar to SHA-1). So NSA and NIST are encouraging people to use SHA-2 which has 256, 384, or 512 bits instead. How many inputs do we need to try before we expect to find a collision?

Birthday Paradox

How many people need to be in a room before the probability that 2 of them have the same birthday is greater than 50%? Answer is 23.

Answer is 25. Probability that 2 people share a birthday is $\frac{1}{365}$. If we have 3 people, the probability that none share a birthday is $(1 - \frac{1}{365})(1 - \frac{2}{365})$ In general for n people, this probability is $(1 - \frac{1}{365})(1 - \frac{2}{365})...(1 - \frac{n}{365})$ If n = 23, $(1 - \frac{1}{365})(1 - \frac{2}{365})...(1 - \frac{22}{365}) = 0.493$ In general N things randomly choose r of them. Probability of there being a match is $1 - (1 - \frac{1}{N})(1 - \frac{2}{N})...(1 - \frac{r-1}{N}) \approx 1 - e^{\frac{-r^2}{2N}}$ (this is a good approximation for large N) for large N)