Cryptography and Network Security
Cryptographic Hash Functions

Km Simran Jaiswal(20CS60R58)

13-Nov-2020

1 Introduction

Cryptographic hash functions is used to ensure integrity of data. We will discuss some algorithm in Random Oracle Model. These algorithm uses class of randomized algorithm called as Las Vegas Algorithm. Las Vegas algorithm may or may not terminate but if it terminates then it always give correct results.
ϵ → Average case success probability: If the probability that algorithm returns a correct answer averaged over all instances of problem is at least ϵ.
Q: Number of hash queries in algorithm.

2 Algorithm Find-Second Preimage

Algorithm 1: FIND-SECOND-PREIMAGE(h, x, Q)

```
$y \longleftarrow h(x)$
choose $X_0 \subseteq X \setminus \{x\}$, $|X_0|$=$Q-1$
for each $x_0 \in X_0$ do
    if ($h(x_0) = y$) then
        return ($x_0$)
return (failure)
```

This algorithm is definitely correct because $x_0 \neq x$ [x has been removed from X_0] and satisfy the condition that $h(x) = h(x_0)$ and $x \neq x_0$.The algorithm may fail because its success depend upon random selection of X_0

For any $X_0 \subseteq X \setminus \{x\}$ with $|X_0|$=$Q-1$, the success probability of this algorithm is ϵ=1-(1-1/M)$^{Q-1}$.

3 Algorithm Find-Collision

Algorithm 2: FIND-COLLISION(h, Q)

```
choose $X_0 \subseteq X \setminus \{x\}$, $|X_0|=Q$
for each $x \in X_0$ do
    $y \longleftarrow h(x)$
if ($y_x = y_{x'}$ for some $x \neq x'$ then
    return $(x, x')$)
else
    return (failure)
```

First we check pair x and x' in X_0 such that $h(x) = h(x')$ and $x \neq x'$. If pair is found then there is success and algorithm return pair else return failure.

Theorem : For any $X_0 \subseteq X \setminus \{x\}$ with $|X_0|=Q$ the success probability of this algorithm is

$$\epsilon = 1 - \left(\frac{M-1}{M}\right)\left(\frac{M-2}{M}\right)\dots\left(\frac{M-Q+1}{M}\right)$$

If there exist input x_1 and x_2 such that $h(x_1) = y_1$ and $h(x_2) = y_2$ then probability that $y_1 \neq y_2$ is $\left(1 - \frac{1}{M}\right)$.
Similarly for another input x_3 such that $h(x_3) = y_3$ then probability that $y_3 \neq y_1$ or $y_3 \neq y_1$ is$\left(1 - \frac{1}{M}\right)\ \left(1 - \frac{2}{M}\right)$.
On continuing this for all the Q inputs we get $\left(1 - \frac{1}{M}\right)\ \left(1 - \frac{2}{M}\right)\ \dots \left(1 - \frac{Q-1}{M}\right)$.
Hence success probability becomes

$$\epsilon = 1 - \left(1 - \frac{1}{M}\right)\left(1 - \frac{2}{M}\right)\dots\left(1 - \frac{Q-1}{M}\right) \tag{1}$$

Now,

$$\begin{aligned}\left(1 - \frac{1}{M}\right)\left(1 - \frac{2}{M}\right)\dots\left(1 - \frac{Q-1}{M}\right) &\approx \prod_{k=1}^{Q-1} e^{\frac{k}{M}} \qquad \text{(By birthday paradox)}\\ &\approx e^{\sum_{k=1}^{Q-1}\frac{k}{M}}\\ &\approx e^{\frac{Q(Q-1)}{2M}}\\ &\approx e^{\frac{Q^2}{2M}}\end{aligned}$$

Putting this in Eq(1) we get

$$\epsilon = 1 - e^{\frac{Q^2}{2M}} \tag{2}$$

If we consider that ϵ=0.5 then

$$1 - e^{\frac{Q^2}{2M}} \approx 0.5$$

Taking log on both sides

$$\begin{aligned}\frac{Q^2}{2M} &\approx -ln\frac{1}{2}\\ \frac{Q^2}{2M} &\approx ln(2)\\ Q &\approx \sqrt{2mln(2)}\\ Q &\approx 1.17\sqrt{M}\end{aligned}$$

Hence we can write

$$Q = \mathrm{O}(\sqrt{M}) \tag{3}$$

Thus our algorithm is $(\frac{1}{2}, \mathrm{O}\sqrt{M})$ because number of query $Q = \mathrm{O}(\sqrt{M})$ and $\epsilon = \frac{1}{2}$.
If we assume that $\mid Q \mid = 80$ bits i.e. we can do 2^{80} computation then to protect against collision then $\mid M \mid$ should be at least 160 bits.

4 Comparison of security criteria

We will study two reductions:

1. Collision to second-Preimage
2. Collision to Preimage

Collision resistant of hash function is extremely critical. If hash function is collision resistant then it guarantees against Preimage and second-Preimage problem. We assume that Preimage can be solved using randomized algorithms.

4.1 Collision to Second Preimage reduction

We will assume that there is an algorithm that can solve Second Preimage problem. Using that we can solve Collision problem.

```
Algorithm 3: COLLISION-TO-SECOND-PREIMAGE($h$)
external ORACLE-SECOND-PREIMAGE
Choose $x \in X$ uniformly at random
if (ORACLE-SECOND-PREIMAGE $(h, x) = x'$) then
    return $(x, x')$
else
    return (failure)
```

For collision we know $h(x) = h(x')$ and $x \neq x'$. There is no additional for checking whether $x \neq x'$ because it is an oracle and it will assure that condition of collision is satisfied.
Since it is Las Vegas Algorithm if it terminates and give output then it will be correct. Thus $h(x) = h(x')$ and $x \neq x'$ and hence collision is found.
ORACLE-SECOND-PREIMAGE is (ϵ, q) algorithm therefore Collision to Second Preimage is also (ϵ, q) Las Vegas Algorithm.

4.2 Collision to Preimage reduction

Algorithm 4: COLLISION-TO-PREIMAGE(h)

```
external ORACLE-PREIMAGE
Choose $x \in X$ uniformly at random
$y \longleftarrow h(x)$
if (ORACLE PREIMAGE $(h, y) = x'$ and $x \neq x'$) then
    return $(x, x')$
else
    return (failure)
```

Although,ORACLE-PREIMAGE will give me corresponding preimage of y but there is a chance $x = x'$. To ensure that $x \neq x'$ an additional check is required.

Theorem: Suppose $h : X \to Y$ is a hash function where $\mid X \mid$ and $\mid Y \mid$ and $\mid X \mid \geq 2 \mid Y \mid$ .Suppose ORACLE-PREIMAGE is a (1,Q) algorithm for preimage for a hash function h then Collision to Preimage is (1/2,Q+1) algorithm for collision for a given hash function h.

Proof:
Let us suppose $\mid X \mid = M$ and $\mid Y \mid = N$. We induce partition on our input such that if x_1 and x_2 are two elements in same partition then $h(x_1) = h(x_2)$.
Partition are labelled as C_i then collection of partition is denoted by set $C = \{C_1, C_2, \ldots, C_M\}$.
Now Given x,we can define

$$Pr[success \mid x] = \frac{\mid [x] \mid -1}{\mid [x] \mid}$$

Average case probability of success is given by:

$$\begin{aligned}
Pr[success] &= \sum_{x} Pr[success \mid x] * Pr(x) \\
&= \frac{1}{\mid X \mid} \sum_{x} Pr[success \mid x] \\
&= \frac{1}{\mid X \mid} \sum_{x} \frac{\mid [x] \mid -1}{\mid [x] \mid} \\
&= \frac{1}{\mid X \mid} \sum_{c \in C} \sum_{x \in X} \frac{\mid [c] \mid -1}{\mid [c] \mid} \\
&= \frac{1}{\mid X \mid} \sum_{c \in C} (\mid [c] \mid -1) \\
&= \frac{1}{\mid X \mid} \sum_{c \in C} (\mid [c]) - \frac{1}{\mid X \mid} \sum_{c \in C} (1) \\
&= \frac{1}{\mid X \mid} \mid X \mid - \frac{1}{\mid X \mid} \mid Y \mid \\
&= 1 - \frac{\mid Y \mid}{\mid X \mid}
\end{aligned}$$

Since $\mid X \mid \geq 2 \mid Y \mid$

$$\begin{aligned}
Pr[success] &\geq 1 - \frac{\mid Y \mid}{2 \mid X \mid} \\
&\geq 1 - \frac{1}{2} \\
&\geq \frac{1}{2}
\end{aligned}$$

From this we can say that probability of success is at least 1/2. Hence we have proved that Collision to Preimage is (1/2,Q+1) algorithm for collision for a given hash function h.

5 References

Class Notes