Authentication

- Requirements include
  - validating the identity of the originator;
  - protecting the integrity of a message; and
  - non-repudiation by either the source or destination.
Symmetric encryption*

*Nobody else knows the key, nobody else could have sent the message: confidentiality, authentication, and integrity.

A qualification

Let $X$ be any input. If $X$ is a legitimate message $M$ produced by $E$ then $Y = D_X(M)$ will be plaintext of $M$. Otherwise,...

Suppose message $M$ could be any arbitrary bit pattern...
Internal error control

*Note that the order in which the FCS and encryption functions are performed is critical.

TCP segment format

<table>
<thead>
<tr>
<th>0</th>
<th>4</th>
<th>10</th>
<th>16</th>
<th>31</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Source port</td>
<td>Destination port</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Sequence number</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Acknowledgement number</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Data offset</td>
<td>Reserved</td>
<td>Flags</td>
<td>Window</td>
<td></td>
</tr>
<tr>
<td>Checksum</td>
<td></td>
<td>Urgent pointer</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Options + padding</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Application data</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Sometimes confidentiality isn’t needed

- In fact, sometimes confidentiality only gets in the way. We don’t encrypt our checks when we sign them.
- With message encryption, the protection is lost when the message is decrypted.
- In addition, there is an overhead associated with encryption and decryption.

Program authentication

- Authentication of a computer program in plaintext is a useful service.
- A “fingerprint” attached to computer code could be checked whenever assurance of its integrity was required.
**Hash functions**

**Definition.** A hash family is a four-tuple \((X, Y, K, H)\), where the following conditions are satisfied:
1. \(X\) is a set of possible messages
2. \(Y\) is a finite set of possible message digests or authentication tags.
3. \(K\), the keyspace, is a finite set of possible keys
4. For each \(K \in K\), there is a hash function \(h_K \in H\). Each \(h_K : X \rightarrow Y\).

**Message authentication codes**

- A keyed hash function is referred to as a message authentication code or **MAC**.
- MACs need not be reversible and are considered less vulnerable to being broken than encryption.
Message authentication & confidentiality

Unkeyed hash functions

- An unkeyed hash function (or simply hash function) is a hash family in which there is only one possible key.
- Like MACs unkeyed hash functions can be used for authentication.
What’s wrong with this picture?

Source A  

Destination B  

M  

H  

M  

H  

H(M)

Compare

We could . . .

Source A  

Destination B  

M  

H  

E  

K  

E_K(M || H(M))

M  

H  

H(M)

Compare

*Once we decrypted the message, we secure the message digest so that it cannot be altered.
Alternatively, we could . . .

* A rose by any other name . . .

Using a shared secret
If a hash is to be secure, three problems should be hard to solve

**Preimage**

*Given* $h : X \rightarrow Y$, and $y \in Y$.
*Find* $x \in X$ s.t. $h(x) = y$.

**Second Preimage**

*Given* $h : X \rightarrow Y$, and $x \in X$.
*Find* $x' \in X$ s.t. $h(x') = h(x)$.

**Collision.**

*Given* $h : X \rightarrow Y$.
*Find* $x, x' \in X$ s.t. $x \neq x'$ and $h(x') = h(x)$.

---

Random oracle model

- We model the "ideal" hash function $h : X \rightarrow Y$ chosen randomly from $F^{X \times Y}$.
- We are only permitted oracle access to the function $h$.
- Suppose $h(x)$ has been determined* if and only if $x \in X_0 \subseteq X$. Then $\Pr[h(x) = y] = 1/|Y|$ for all $x \in X \setminus X_0$.

*By querying an oracle for $h$. 
Las Vegas algorithms

- A Las Vegas algorithm is a random algorithm which may fail to give an answer, but if the algorithm does return an answer, then the answer must be correct.

- A randomized algorithm has worst-case success if, for every problem instance, the algorithm returns a correct answer with probability.

Solving Preimage

- Algorithm FindPreimage(h, y, q)
  
  choose any $X_0 \subseteq X$, $|X_0| = q$
  
  for each $x \in X_0$
    
    do if $h(x) = y$ then return $(x)$
  
  return (failure)

- For any $X_0 \subseteq X$, with $|X_0| = q$ the average-case success probability of FindPreimage is $\frac{1}{M} = 1 - (1 - 1/M)^q$.

*A Las Vegas algorithm with average-case success in which the number of oracle queries is at most $q$ is termed an $(1,q)$-algorithm.
Birthday paradox

Algorithm FindCollision($h, q$)
choose $X_0 \subseteq X$, $|X_0|=q$
for each $x \in X_0$
do $y_x \leftarrow h(x)$
if $y_x = y_{x'}$ for some $x' \neq x$
then return $(x, x')$
else return (failure)

For any $X_0 \subseteq X$, with $|X_0| = q$ the average-case success probability of FindCollision is

\[ \prod = 1 \prod_{i=1}^{M} \frac{1}{m} \prod_{i=1}^{M-2} \frac{1}{m} \cdots \frac{1}{m} \frac{1}{M} q + 1 \]

---

Estimating the odds

- We approximate $1-x$ for small $x$ by $e^{-x}$ using the first two terms of the series expansion 
  \[ e^{-x} = 1 - x + \frac{x^2}{2!} - \frac{x^3}{3!} + \cdots \]
- The probability of finding no collision is then

\[ e^{-\prod} = e^{-1} \prod_{i=1}^{M} \frac{1}{m} \prod_{i=1}^{M-2} \frac{1}{m} \cdots \frac{1}{m} \frac{1}{M} q + 1 \]

\[ = e^{-1} \prod_{i=1}^{M} \frac{i}{M} \prod_{i=1}^{M} \frac{1}{e^m} e^{-\prod} = e^{-\prod} e^{-\prod} e^{-\prod} = e^{-\prod} \frac{1}{2M} \]

---
Thus,

- We can estimate the probability of finding at least one collision to be \(1 - e^{-\frac{q(q-1)}{2M}}\).
- Denoting this probability by \(e\) we can solve for \(q\) as a function of \(M\) and \(e\):

\[
1 - e^{q(q-1)/2M}\]

\[
\frac{1}{2M}\ln(1/e)
\]

\[
q^2 - q - 2M\ln(1/e) = 0
\]

or

\[
q = \sqrt{2M\ln(1/e)}
\]

Birthday attacks

- The birthday attack imposes a lower bound on the sizes of secure message digests.
- But what harm could a birthday paradox cause?