Hash Functions & MSG Authentication

Birthday Paradox

MSG Authentication - verifying msg hasn't been altered.

A simple method to achieve

Alice → Bob

1. M
2. \( E_{Bob-Public}(M) \) - encrypt msg w/ Bob's Public key
3. \( E_{Alice-Private}(E_{Bob-Public}(M)) \) → send to Bob

When Bob receives ...
4. \( D_{Alice-Public}(C) \)
5. \( D_{Bob-Private}(D_{Alice-Public}(C)) \) → M

If this all works, you get a meaningful msg, then it must have been Alice who sent the MSG.

* (THIS IS SLOW!!!)
### Two Different Goals

1) **Message Authentication**: verifying that a message has not been altered since it was sent.

2) **Digital Signature**: proof of who authored a message.

### Hash Functions

1. **Input**: any size bitstring
2. **Output**: fixed-size output typically either $n$ bits or value 0, MAX-1.
3. It's possible that $x \neq y$ but $H(x) = H(y)$. This is called a collision.

### Properties of a Good Hash Func

1) Given an output value $y$, it should computationally infeasible to find a value $x$, s.t. $H(x) = y$.

2) Given an input $x$, it should be computationally infeasible to find some $x'$, with $x' \neq x$ but $H(x') = H(x)$.

3) Hard to find ANY pair $x, x'$, with $x \neq x'$ such that $H(x) = H(x')$.

<table>
<thead>
<tr>
<th>Inputs</th>
<th>Outputs</th>
</tr>
</thead>
<tbody>
<tr>
<td>261 abin</td>
<td>257 iron</td>
</tr>
<tr>
<td>223 yean</td>
<td>252 lucen</td>
</tr>
<tr>
<td>374 deem</td>
<td>232 rash</td>
</tr>
<tr>
<td>245 churn</td>
<td>232 rash</td>
</tr>
<tr>
<td>258 10080</td>
<td>261 tesley</td>
</tr>
</tbody>
</table>
If I pick a number in range \([1, n]\), then have 12 people also pick a rand number \([1, n]\) with \(k < n\) probability one of their numbers matches mine < \(\frac{k}{n}\).

Let people all pick a rand number \([1, n]\) what's the probability that some 2 people pick the same number?

What's prob all #s are different?
\[
\frac{n}{n} \times \frac{n-1}{n} \times \frac{n-2}{n} \times \frac{n-3}{n} \times \cdots \frac{n-k}{n} \leq 1
\]

1st person 2nd person

Birthday paradox \(n = 365\), \(k = 23\) for the formula, the frist probability is \(\leq \frac{1}{2}\) all are different!
4 Modes of Use

a) \( E(K, [m \parallel H(m)]) \)

Encrypt everything. Receiver gets msg, they decrypt to reveal \( m' \) and \( H(m') \). Now to verify the message wasn't altered calculate \( H(m') \) and make sure it's equal to \( H(m) \).

b) \( m \parallel E(K, H(m)) \)

Everyone can read but receiver can verify that it's not altered.

c) \( m \parallel H(m \parallel S), S = \text{secret value} \)

Only receiver knows \( S \), so they can calculate \( m' \parallel S \) received msg.
Calc \( H(m' \parallel S) \) to see if it matches \( H(m \parallel S) \).

\( \) Alice private

\( \) Bob shared private key

d) \( E(K, [m \parallel E(PR_{Alice}, H(m))]) \)

Proves Alice sent it but w/o using slow public key on the whole msg.