HMAC’s concluded

• \( \text{HMAC}(K, \text{text}) = H((K \oplus \text{opad}) || H((K \oplus \text{ipad}) || \text{text}))) \)
• \( H \) is a cryptographic hash like SHA-256
• \text{ipad}, the inner pad: the byte 0x36 repeated \( B \) times where \( B \) is key size
• \text{opad}, the outer pad: the byte 0x5c repeated \( B \) times
• Verification requires knowledge of \( K \).
Discrete log based public key systems
Discrete Log

• If \( b = a^x \), then \( \log_a(b) = x \). \( \log_a(y) \) is the discrete log function.

• If \( g = b^x \), then \( \log_a(g) = x \log_a(b) \).

• **Discrete Log Problem (DLP):** Given \( p, \) prime, \( a : \langle a \rangle = F_p^* \). \( b \equiv a^x \pmod{p} \), \( a \), unknown, find \( \log_a(b) \).

• **Computational Diffie Hellman Problem (CDHP):** Given \( p, \) prime, \( \langle a \rangle = F_p^* \). \( a^a \equiv b \pmod{p}, a^b \equiv c \pmod{p} \), find \( a^{ab} \equiv d \pmod{p} \).

• **Theorem:** \( \text{CDHP} \leq_P \text{DLP} \). If the factorization of \( p-1 \) is known and \( \phi(p-1) \) is \( O((\ln(p))^c) \) smooth then DLP and CDHP are equivalent.

• **Conclusion:** Exponentiation is a one way trap-door function.
El Gamal cryptosystem

- Alice, the private keyholder, picks a large prime, p, where \( p-1 \) also has large prime divisors (say, \( p = 2rq+1 \)) and a generator, g, for \( F^*_p \). \(<g> = F^*_p \). Alice also picks a random number, a (secret), and computes \( A = g^a \pmod{p} \). Alice’s public key is \(<A, g, p>\).

- To send a message, m, Bob picks a random b (his secret) and computes \( B = g^b \pmod{p} \). Bob transmits \((B, mA^b) = (B, C)\).

- Alice decodes the message by computing \( CB^{-a} = m \).

- Without knowing a, an adversary has to solve the Computational Diffie Hellman Problem to get m.

- Note: b must be random and never reused!
El Gamal Example

- Alice chooses
  - p=919, g=7.
  - a=111, A= $7^{111} = 461$ (mod 919).
  - Alice’s Public key is <919, 7, 461>

- Bob wants to send m=45, picks b= 29.
  - B=$7^{29} = 788$ (mod 919), $461^{29} = 902$ (mod 919),
  - C= (45)(902) = 154 (mod 919).
  - Bob transmits (788, 154).

- Alice computes $(788)^{-111} = 902^{-1}$ (mod 919).
  - (54)(902)+(-53)(919)=1. 54 = 902^{-1} (mod 919)
  - Calculates m= $(154)(54)=45$ (mod 919).
El Gamal Signature

• \(<g>= F_q^\ast\) A picks a random as in encryption.
• Signing: Signer picks \(k\): \(1 \leq k \leq p-2\) with \((k, p-1) = 1\) and publishes \(g^k\). \(k\) is secret.
• \(\text{Sig}_K(M, k) = (t, d)\)
  - \(t = g^k \pmod{p}\)
  - \(d = (M - gt)k^{-1} \pmod{p-1}\)
• \(\text{Ver}_K(M, t, d) \iff g^{kt} t^d = g^M \pmod{p}\)

• Notes: It’s important that \(M\) is a hash otherwise there is an existential forgery attack. It’s important that \(k\) be different for every message otherwise adversary can solve for key.
Timing

- Finding $g$ takes about $O(\lg(p)^3)$ operations, so does primality testing and raising $g$ to the $a$ power mod $p$.
- Encryption is also $O(\lg(p)^3)$ and so is decryption.
- Note that key generation is cheap but for safety, $p > w^2$, where $w$ is the “computational power” of the adversary.
Finding generators (Gauss)

- Find a generator, \( g \), for \( F_p^* \), \( n = (p-1) = p_1^{e_1} p_2^{e_2} \ldots p_k^{e_k} \).

\[
\text{while} \quad () \quad \{
\begin{align*}
\text{choose a random } \ g \in \mathbb{G} \\
\text{for} (i=1; \ i<e; \ k++) \quad \{
\begin{align*}
\quad b & = g^{n/p_i} \\
\quad \text{if} \ (b==1) \quad \break
\end{align*}
\}
\quad \text{if}(i>k) \\
\quad \text{return } \ g
\end{align*}
\}
\]

- \( \mathbb{G} \) has \( \phi(n) \) generators. Using the lower bound for \( \phi(n) \), the probability that \( g \) in line 2 is a generator is at least \( 1/(6 \ln \ln n) \).
Attack on reused nonce

- Suppose Bob reuses b for two different messages $m_1$ and $m_2$.
- An adversary, Eve, can see $<B, C_1>$ and $<B, C_2>$ where $C_i = Bm_i \pmod{p}$.
- Suppose Eve discovers $m_1$.
- She can compute $m_2 = m_1C_2C_1^{-1} \pmod{p}$.
- Don’t reuse b’s!
**DSA**

- **Alice**
  - \[2^{159} < q < 2^{160}, \ 2^{511+64t} < p < 2^{512+64t}, \ 1 \text{ c t c 8}, \ q|p-1\]
  - Select primitive root \(x\) (mod \(p\)); compute: \(g = x^{(p-1)/q}\) (mod \(p\))
  - Picks a random, 1cacq-1. \(A = g^a\) (mod \(p\))
  - Public Key: \((p, q, g, A)\). Private Key: \(a\).

- **Signature Generation**
  - Pick random \(k\), \(r = (g^k\) (mod \(p\))\) (mod \(q\)). Note: \(k\) must be different for each signature.
  - \(s = k^{-1}(h(M)+ar)\) (mod \(q\)). Signature is \((r,s)\)

- **Verification**
  - \(u = s^{-1}h(x)(\text{mod } q), \ v = (rs^{-1})(\text{mod } q)\)
  - Is \(g^u A^v = r\) (mod \(p\))?

- **Advantages over straight El Gamal**
  - Verification is more efficient (2 exponentiations rather than 3)
  - Exponent is 160 bits not 768
Baby Step Giant Step --- Shanks

- $g^x = y \pmod{p}$.
- $m \sim \sqrt{p}$.
- Compute $g^{mj}$, $0 \leq j < m$.
- Sort $(j, g^{mj})$ by second coordinate.
- Pick $i$ at random, compute $y^{g^{-i}} \pmod{p}$.
- If there is a match in the tables $y^{g^{-i}} = g^{mj} \pmod{p}$.
- $x = mj + i$ is the discrete log.
Baby Step Giant Step Example

- $p=193. \lfloor \sqrt{p} \rfloor = 13. \ m = 14. \ a = 5. \ b = 41.$
- $2 \times 193 + (-77) \times 5 = 1, \ a^{-1} = 116. \ a^{-14} = 189 \pmod{193}.$

<table>
<thead>
<tr>
<th>j</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
<th>11</th>
<th>12</th>
<th>13</th>
<th>14</th>
</tr>
</thead>
<tbody>
<tr>
<td>$a^j$</td>
<td>5</td>
<td>25</td>
<td>125</td>
<td>46</td>
<td>37</td>
<td>185</td>
<td>153</td>
<td>186</td>
<td>158</td>
<td>18</td>
<td>90</td>
<td>64</td>
<td>127</td>
<td>56</td>
</tr>
<tr>
<td>$ba^{-mj}$</td>
<td>26</td>
<td>77</td>
<td>78</td>
<td>74</td>
<td>90</td>
<td>26</td>
<td>89</td>
<td>30</td>
<td>73</td>
<td>94</td>
<td>10</td>
<td>153</td>
<td>160</td>
<td>132</td>
</tr>
</tbody>
</table>

- So $ba^{-(14 \times 5)} = 90 = a^{11} \pmod{193}$.
- Thus $b = a^{14 \times 5 + 11} = a^{81} \pmod{193}$.
- $L_5(41) = 193$. 
Discrete log Pollard $\rho$

- $x_{i+1} = f(x_i)$
  - $f(x_i) = b x_i$, if $x_i \in S_1$.
  - $f(x_i) = x_i^2$, if $x_i \in S_2$.
  - $f(x_i) = a x_i$, if $x_i \in S_3$.

- $x_i = a^{a[i]} b^{b[i]}$.
  - $a[i] = a[i]$, if $x_i \in S_1$.
  - $a[i] = 2a[i]$, if $x_i \in S_2$.
  - $a[i] = a[i] + 1$, if $x_i \in S_3$.
  - $b[i] = b[i] + 1$, if $x_i \in S_1$.
  - $b[i] = 2b[i]$, if $x_i \in S_2$.
  - $b[i] = b[i]$, if $x_i \in S_3$.

- $x_{2i} = x_i \Rightarrow a_{2i} - a_i = L_a(b)(b_{2i} - b_i)$
Pollard $\rho$ example

- $p=229$, $n=191$, $b=228$, $a=2$. $L_2(228)=110$

<table>
<thead>
<tr>
<th>$i$</th>
<th>$x_i$</th>
<th>$a_i$</th>
<th>$b_i$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>228</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>279</td>
<td>0</td>
<td>2</td>
</tr>
<tr>
<td>3</td>
<td>92</td>
<td>0</td>
<td>4</td>
</tr>
<tr>
<td>4</td>
<td>184</td>
<td>1</td>
<td>4</td>
</tr>
<tr>
<td>5</td>
<td>205</td>
<td>1</td>
<td>5</td>
</tr>
<tr>
<td>6</td>
<td>14</td>
<td>1</td>
<td>6</td>
</tr>
<tr>
<td>7</td>
<td>28</td>
<td>2</td>
<td>6</td>
</tr>
<tr>
<td>8</td>
<td>256</td>
<td>2</td>
<td>7</td>
</tr>
<tr>
<td>9</td>
<td>152</td>
<td>2</td>
<td>8</td>
</tr>
<tr>
<td>10</td>
<td>304</td>
<td>3</td>
<td>8</td>
</tr>
<tr>
<td>11</td>
<td>372</td>
<td>3</td>
<td>9</td>
</tr>
<tr>
<td>12</td>
<td>121</td>
<td>6</td>
<td>18</td>
</tr>
<tr>
<td>13</td>
<td>12</td>
<td>6</td>
<td>19</td>
</tr>
<tr>
<td>14</td>
<td>144</td>
<td>12</td>
<td>38</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>$i$</th>
<th>$x_{2i}$</th>
<th>$a_{2i}$</th>
<th>$b_{2i}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>279</td>
<td>0</td>
<td>2</td>
</tr>
<tr>
<td>2</td>
<td>184</td>
<td>1</td>
<td>4</td>
</tr>
<tr>
<td>3</td>
<td>14</td>
<td>1</td>
<td>6</td>
</tr>
<tr>
<td>4</td>
<td>256</td>
<td>2</td>
<td>7</td>
</tr>
<tr>
<td>5</td>
<td>304</td>
<td>3</td>
<td>8</td>
</tr>
<tr>
<td>6</td>
<td>121</td>
<td>6</td>
<td>38</td>
</tr>
<tr>
<td>7</td>
<td>144</td>
<td>12</td>
<td>152</td>
</tr>
<tr>
<td>8</td>
<td>235</td>
<td>48</td>
<td>154</td>
</tr>
<tr>
<td>9</td>
<td>72</td>
<td>48</td>
<td>118</td>
</tr>
<tr>
<td>10</td>
<td>14</td>
<td>96</td>
<td>119</td>
</tr>
<tr>
<td>11</td>
<td>256</td>
<td>97</td>
<td>120</td>
</tr>
<tr>
<td>12</td>
<td>304</td>
<td>98</td>
<td>51</td>
</tr>
<tr>
<td>13</td>
<td>121</td>
<td>5</td>
<td>104</td>
</tr>
<tr>
<td>14</td>
<td>144</td>
<td>10</td>
<td>163</td>
</tr>
</tbody>
</table>

- $x_{14} = x_{28}$, $(b_{14} - b_{28}) = 125 \pmod{191}$, $L_2(228)=125^{-1} (a_{28} - a_{14}) = 110$.  

JLM 20101208
Pohlig-Hellman

• \( p-1 = \prod_i q_i^{r[i]} \).
• Solve \( a^x = y \mod p \) for \( x \mod q_i^{r[i]} \) and use Chinese Remainder Theorem.
• \( x = x_0 + x_1 q + x_2 q^2 + \ldots + x_{r[i]-1} q^{r[i]-1} \).
• \( x \mod (p-1)/q = x_0 \mod (p-1)/q + (p-1)(\ldots) \).
• So \( b^{(p-1)/q} = a^{x_0 \mod (p-1)/q} \). Solve for \( x_0 \).
• The put \( g = ba^{-x_0} \) and solve \( g^{(p-1)/(q \times q)} = a^{x_1 \mod (p-1)/q} \).
• This costs \( O(\sum_{i=1}^r e_i (\lg(n)+\sqrt{q_i}) \).
Pohlig-Hellman example

- $p=251$. $a=71$, $b=210$, $\langle a \rangle = F_{251}^*$. $n=250 = 2 \times 5^3$.
- $L_{71}(210) = 1 \pmod{2}$.
- $x = x_0 + x_1 5 + x_2 5^2$.
- So $a^{n/5} = 71^{20}$. $b^{n/5} = 210^{20} = 149$.
  - $x_0 = L_{20}(149) = 2$.
  - $x_1 = 4$
  - $x_2 = 2$
- $x = 2 + 4 \times 5 + 2 \times 25 = 72 \pmod{125}$
- Applying CRT: $L_{71}(210) = 197$. 

JLM 20101208
Index Calculus

- \(g^x \equiv y \pmod{p}\). \(B = (p_1, p_2, \ldots, p_k)\).
- Precompute
  - \(g^{x_j} = p_1^{a_1} p_2^{a_2} \cdots p_k^{a_k}\)
  - \(x_j = a_1 \log_g(p_1) + a_2 \log_g(p_2) + \ldots + a_k \log_g(p_k)\)
  - If you get enough of these, you can solve for the \(\log_g(p_i)\)
- Solve
  - Pick \(s\) at random and compute \(y g^s = p_1^{c_1} p_2^{c_2} \cdots p_k^{c_k}\)
    then
  - \(\log_g(y) + s = c_1 \log_g(p_1) + c_2 \log_g(p_2) + \ldots + c_k \log_g(p_k)\)
- This takes \(O(e^{(1+\ln(p))\ln(\ln(p))})\) time.

- LaMacchia and Odlyzko used Gaussian integer index calculus variant to attack discrete log.
Index Calculus Example

- $p=229$. $a=6$. $<a> = F_{229}^*$. $n=228$. $b=13$. $S=\{2,3,5,7,11\}$.
- **Step 1**
  1. $6^{100} \pmod{229} = 180 = 2^2 \times 3^2 \times 5^1 \times 7^0 \times 11^0$.
  2. $6^{18} \pmod{229} = 176 = 2^4 \times 3^0 \times 5^0 \times 7^0 \times 11^1$.
  3. $6^{12} \pmod{229} = 165 = 2^0 \times 3^1 \times 5^1 \times 7^0 \times 11^0$.
  4. $6^{62} \pmod{229} = 154 = 2^1 \times 3^0 \times 5^0 \times 7^1 \times 11^1$.
  5. $6^{143} \pmod{229} = 198 = 2^1 \times 3^2 \times 5^0 \times 7^0 \times 11^1$.
  6. $6^{206} \pmod{229} = 210 = 2^1 \times 3^1 \times 5^1 \times 7^1 \times 11^0$.
- Taking $L_a()$ of both sides, we get:
  1. $100 = 2L_a(2) + 2L_a(3) + L_a(5) \pmod{228}$
  2. $18 = 4L_a(2) + L_a(11) \pmod{228}$
  3. $12 = L_a(3) + L_a(5) + L_a(11) \pmod{228}$
  4. $62 = L_a(2) + L_a(7) + L_a(11) \pmod{228}$
  5. $143 = L_a(2) + L_a(3) + L_a(11) \pmod{228}$
  6. $206 = L_a(2) + L_a(3) + L_a(5) + L_a(11) \pmod{228}$
Index Calculus example - continued

• Review
  – $p=229. \ a=6. \ \langle a \rangle = F_{229}^*$. $n=228$. Solving, we got:
  – $L_a(2) = 21 \ (mod \ 228)$
  – $L_a(3) = 208 \ (mod \ 228)$
  – $L_a(5) = 98 \ (mod \ 228)$
  – $L_a(7) = 107 \ (mod \ 228)$
  – $L_a(11) = 162 \ (mod \ 228)$

• Step 2:
  – Recall $b=13$. Pick $k=77$
  – $13 \times 6^{77} = 147 = 3 \times 7^2 \ (mod \ 229)$
  – $L_6(13) = (L_6(3)+2L_6(7)-77) = 117 \ (mod \ 228)$
Diffie Hellman key exchange

Alice

A1: \( s = \min(p \text{ size}), N_a \in \{0, \ldots, 2^{256}-1\} \)

A2: Check \((p,q,g) X, \text{Auth}_B\), pick \(y \in \{0, \ldots, q-1\}\)

\( K = X^y \)

Bob

B1: Choose \((p,q,g), x \in \{0, \ldots, 2^{256}-1\}\)

B2: Check \(Y, \text{Auth}_A\)

\( K = Y^x \)
DH key exchange example

- \( p=3547, \ g=2. \)
- Alice: \( a=7. \)
- Bob: \( b=17. \)
- \( A \rightarrow B_1: A=128 \ (=2^7), \ Sign_A(SHA-2(128||r_1)) \)
- \( B \rightarrow A_1: B=3380 \ (=2^{17}), \ Sign_B(SHA-2(3380||r_2)) \)
- \( K=128^{17}=3380^7=362. \)
Access Control: authentication and authorization

- Authentication is process of identifying a security principal. Here are some ways:
  - Login/password or smart card/pin (user)
  - Cryptographic Hash (program)
  - Ability to decrypt (channel)
Authentication

• When logging on to a computer you enter
  – user name and
  – password
• The first step is called identification. You announce who you are.
• The second step is called authentication. You prove that you are who you claim to be.
• To distinguish this type of ‘authentication’ from other interpretations, we may refer specifically to entity authentication: The process of verifying a claimed identity.
Authentication

Login: jlm
Password: ********

Welcome John Manferdelli
>

Jan 18, 2007
Problems with Passwords

- Authentication by password is widely accepted and not too difficult to implement.
- Managing password security can be quite expensive; obtaining a valid password is a common way of gaining unauthorised access to a computer system.
- Typical issues
  - how to get the password to the user,
  - forgotten passwords,
  - password guessing,
  - password spoofing,
  - compromise of the password file.
Guessing Passwords

• Exhaustive search (brute force): Try all possible combinations of valid symbols up to a certain length.
• Intelligent search: search through a restricted name space, e.g. passwords that are somehow associated with a user like name, names of friends and relatives, car brand, car registration number, phone number,…, or try passwords that are generally popular.
• Typical example for the second approach: dictionary attack trying all passwords from an on-line dictionary.
• You cannot prevent an attacker from accidentally guessing a valid password, but you can try to reduce the probability of a password compromise.

Slide from Dieter Gollmann
Password Salting

- To slow down dictionary attacks, a **salt** can be appended to the password before encryption and stored with the encrypted password.
  - If two users have the same password, they will now have different entries in the file of encrypted passwords.
  - Example: Unix uses a 12 bit salt.
Access Control Matrix

• Capabilities:
  – access rights are stored with the subject
  – rows of the access control matrix

• Access Control Lists (ACLs)
  – access rights are stored with the object.
  – columns of the access control matrix.

<table>
<thead>
<tr>
<th></th>
<th>bill.doc</th>
<th>edit.exe</th>
<th>fun.com</th>
</tr>
</thead>
<tbody>
<tr>
<td>Alice</td>
<td>-</td>
<td>{exec}</td>
<td>{exec,read}</td>
</tr>
<tr>
<td>Bob</td>
<td>{read,write}</td>
<td>{exec}</td>
<td>{exec,read,write}</td>
</tr>
</tbody>
</table>
Digital signatures

Slide from Dieter Gollmann
Digital Signatures

- A has a public verification key and a private signature key (→ public key cryptography).
- A uses her private key to compute her signature on document $m$.
- B uses a public verification key to check the signature on a document $m$ he receives.
- This provides non-repudiation.
- Signature algorithm= hash+padding+private key operation

Slide from Dieter Gollmann
Bleichenbacher Attack on PKCS1

- Chosen-ciphertext attack.
- RSA PKCS #1 v1.5: \( c = (00 \ || \ 02 \ || \ r \ || \ 0 \ || \ m)^e \mod n \)
- Attacker can test if 16 MSBs of plaintext = ’02’.
- Attack: to decrypt a given ciphertext \( C \) do:
  - Pick \( r \in \mathbb{Z}_n \). Compute \( C' = r^e \cdot C = (r \cdot \text{PKCS1}(M))^e \).
  - Send \( C' \) to oracle and use response.
Side-Channel Attacks

- Some attack vectors …
  - Fault Attacks
  - Timing Attacks
  - Cache Attacks
  - Power Analysis
  - Electromagnetic Emissions
  - Acoustic Emissions
Berlekamp factorization

- \( f(x) = \prod_{i=1}^{t} f_i(x) \) over \( \mathbb{F}_p \), \( \deg(f(x))=n \). \( f_i(x) \) irreducible.
  
  \[ F = \{ f(x) \}; \]
  
  \[
  \text{for}(i=1; \ i<n; i++)
  \]
  
  \[
  x^{iq} = \sum_{j=0}^{n-1} q_{ij} x^j \mod f(x), \quad q_{ij} \in \mathbb{F}_p.
  \]

  Find basis \( \langle v_1, \ldots, v_t \rangle \) of null space of \( (Q-I_n) \);
  
  // \( w = w_0, \ldots, w_{n-1}. \) \( w(x) = w_0 + w_1 x + \ldots + w_{n-1} x^{n-1} \)
  
  \[
  \text{for}(i=1; \ i\leq t; i++) \{
  \]
  
  \[
  \text{for (h(x) \in F, deg (h)>1;)} \{
  \]
  
  Compute \( (h(x), v_i(x)-a), a \in \mathbb{F}_p; \)
  
  Replace \( h(x) \) in \( F \) with these;
  
  \[
  \}
  \]
  
  return \( (F) \);

- \( O(n^3+tpn^2) \), \( t = \# \) irreducible factors. Can be reduced to \( O(n^3+t \lg(p)n^2) \).
Berlekamp factorization example

• Factor $x^7 - 1$ over $F^2$.

\[
\begin{align*}
1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \\
0 & 0 & 1 & 0 & 0 & 0 & 0 & x & x^2 \\
0 & 0 & 0 & 0 & 1 & 0 & 0 & x^2 & x^4 \\
0 & 0 & 0 & 0 & 0 & 0 & 1 & x^3 & x^6 \\
0 & 1 & 0 & 0 & 0 & 0 & 0 & x^4 & x^1 \\
0 & 0 & 0 & 1 & 0 & 0 & 0 & x^5 & x^3 \\
0 & 0 & 0 & 0 & 0 & 1 & 0 & x^6 & x^5 
\end{align*}
\]

• Adding I and solving get:
  • 1
  • $x^4 + x^2 + x = x(x^3 + x + 1)$
  • $x^6 + x^5 + x^3 = x^3(x^3 + x^2 + 1)$
• Dividing into $x^7 - 1$, we get:
  • $(x+1)$