c) Verification of step (i') of ElGamal signatures requires checking if \( 1 \leq r' \leq p-1 \)

If this check is omitted, then Oscar can re-sign messages of his choice provided he has one valid signature and \( h(m)^{r'} \mod p-1 \) should exist.

Suppose \((r_1, s_1)\) is a signature for message \( m \).
O selects a message \( m' \) of his choice and computes

\[ h(m') \] and \( u = h(m')(h(m)^{r'}) \mod p-1 \)

\[ r' = u \mod p-1 \]

\( r' \) such that \( r' \equiv ru \mod (p-1) \) and \( r' \equiv r \mod p \)

Solve this by CRT, because \( p, p-1 \) are relatively prime.

The pair \((r'_1, s')\) is a signature for \( m' \) which would be accepted if \( 1 \leq r' \leq p-1 \) is ignored.
11.2 The Digital Signature Algorithm (DSA)

- Proposed by the NIST in Aug 91
- Standardized as FIPS 186, renamed DSS (Digital Signature Standard)
- Developed by the NSA (not publicly)
- DSA is a variant of the ElGamal signature scheme
- Needs a hash function \( h: \{0,1\}^* \rightarrow \mathbb{Z}_q \) as a building block

The standard prescribes \( q = 160 \) bits

**System parameters**

Each user generates a public and private key as follows:

1. Choose a prime \( q \) with \( 2^{159} < q < 2^{160} \) (160 bits)
2. Choose \( t, 0 \leq t \leq 8 \) further a prime \( p \) such that
   \[ 2^{512 + 64t} < p < 2^{512 + 64t} \quad \text{and} \quad q \mid p - 1 \]
   (Recommended by NIST from Oct 2001: \( t = 8 \), 1024 bits)
3. (i) Select \( g \in \mathbb{Z}_p^* \), compute \( \alpha = g \left( p - 1 \right) / q \) mod \( p \)
   (ii) If \( \alpha = 1 \), repeat step (i)
   (\( g \) is a generator of a cyclic subgroup of order \( q \) in \( \mathbb{Z}_p^* \))
4. Choose some random \( x \in \{1, \ldots, p - 2\} \) \( \text{mod} \ p \)
5. Compute \( \gamma = \alpha^x \) mod \( p \)
6. Public key: \((p, q, g, \gamma)\), private key \( x \)

**Signing a message \( m \in \{0,1\}^* \)**

1. Choose a random \( k \in \{1, \ldots, q - 1\} \)
2. \( r = (\alpha^k \text{ mod } p) \text{ mod } q \)
3. Compute \( k^{-1} \text{ mod } q \)
4. \( \tau = k^{-1} (h(m) + x \cdot r) \text{ mod } q \)
5. Signature \((r, \tau)\) \( (320 \text{ bits in total}) \)
Verifikasi of signature \( (\gamma, \zeta) \) on message \( m \):
1. Check if \( 0 < \gamma < q \) and \( 0 < \zeta < q \), otherwise decline
2. \( \omega = \gamma^{-1} \mod q \)
3. \( \mu_1 = (\omega h(m)) \mod q \), \( \mu_2 = (\gamma \cdot \zeta) \mod q \)
4. \( \nu = (\mu_1 \cdot \chi \cdot \mu_2 \mod p) \mod q \)
5. Accept the signature if \( \nu = \gamma \)

Proof that the verification is correct:
For a valid signature \( (\gamma, \zeta) \) it holds that
\[
h(m) \equiv k \cdot \omega - \gamma \cdot \zeta \mod q
\]
Hence:
\[
a^{\mu_1} \cdot \gamma \cdot \mu_2 \equiv a^{\mu_1 + \mu_2} \mod p
\]
\[
\mu_1 + \mu_2 \equiv \omega h(m) + \gamma \cdot \zeta \equiv \omega \cdot k \cdot \omega - \gamma \cdot \zeta + \gamma \cdot \zeta \equiv k \mod q
\]
\[
\nu \equiv \left( a^{k \cdot \omega + k} \mod p \right) \mod q = \left( a^k \mod p \right) \mod q = \gamma
\]
\[
\text{as desired if } 1 \cdot a^q \equiv 1 \mod p
\]

Security

- Security relies on two DL problems:
  a) \( \mu \not\equiv 0 \mod p \)
  b) \( \mu < q \leq \frac{p}{2} \) (\( < q \) denotes the subgroup \( \gamma \cdot \omega \), \( \gamma \cdot \omega \))

- Security principles of the ElGamal scheme carry over:
  - always choose \( \alpha \) a new \( k \)
  - use of hash functions is mandatory
  - always verify 1. in the verification procedure. Otherwise
  signatures for arbitrary messages can be generated provided one valid
  signature is known.
Remarks:

a) Modular exponentiation is in the range of (160 bits) (rather than 1024 El Gamal)

b) $k^{-1}$ may be generated, computed and shared in advance

c) Verification needs 2 instead of 3 modular exponentiation

d) Signature by DSA is about 320 bits, instead of 2048 bits for El Gamal.

e) In the verification step, also check, if $r \neq 0, 0 \neq 0$, otherwise the signature is rejected. But this happens with a very small probability.
12. Identification and Entity Authentication

This chapter considers techniques to allow the "verifier" to establish the identity of the "claimant," thereby preventing impersonation. Requirements on authentication protocols:

1. A is able to uniquely identify himself to B
2. B cannot reuse an identification exchange with A to impersonate A to a third party C. (non-reusability)
3. It is practically infeasible that a third party C can cause B to wrongly accept the identity of A. (impersonation)
4. Even if C observes the identification process between A and B very often he cannot impersonate A.

There are main categories of identification:

1. Something I know: password, PIN, private key
2. Something I possessed: key, magnetic-striped card, chipcard, PIN or password generator ...
3. Something I have: human physical characteristics, face recognition, biometric material pattern, hand-written signatures

12.1 Passwords

Fixed password schemes

Rather than storing a cleared user password pwd in a file, a hash value h(pwd) of each user password is stored. Verification is done by comparing the hash value of the entered password with the stored one for a given user.
Main attacks are:
- replay of fixed passwords
- exhaustive password search
- password-guessing and dictionary attacks

Defense strategies are:
- choose a random password or nearly random use of special characters (increase entropy)
- slowing down the password mapping
- salting passwords
  Extend the password by some random string, the salt, before hashing. Both the hashed password and the salt are stored as 
  \[ h(p\text{assword} \cdot salt) = salt \]

They does not complicate exhaustive search and simultaneous dictionary attacks against a large set of passwords

One-time passwords
- protects against eavesdropping and replay of passwords or "phishing"

TARPOT's protocol

Objective: A identifies himself to B
- use a one-way function \( H \)
- \( H_{k}(w) = \underbrace{H(H(\ldots H(H(w)))}}_{k\text{-times}} \)

Initial parameters: \( t \) : max number of identities, \( t = 100 \text{, } 000 \)
- A chooses an initial password \( w \)
- A transfers \( C_{0} = H^{t}(w) \) to B
- B initializes his counter for \( A \) to \( i_{A} = 1 \)

Protocol action for round \( i \):
- A computes \( C_{w} = H^{t-i}(w) \); transfers to B: \( (A, i, C_{w}) \)
- B checks that \( i = i_{A} \) and \( C_{w} = H(w_{i}) \). If both checks succeed, B accepts and sets \( i_{A} = i_{A} + 1 \) and stores \( C_{w} \)
- \( - \)