FIPS 186-4: Summary of the Digital Signature Standard (DSS)

The basic gist: analogous to the hand-written signature

- Sender (Bob) digitally signs document, establishing that he is the document **owner**.
- **Verifiable:** recipient (Alice) can verify that the owner is who he says he is.
- **Non-forgeable & non-refutable**: Alice can prove that Bob is the one who signed the document, and as such Bob cannot deny if he signed a document.
- Similarly, a digital signature (DS) needs to satisfy these three constraints: verifiable, non-forgeable and non-refutable.

Eg: Think of Bob wanting to withdraw money from a bank using a check.

For a message, m:

- Bob will sign (i.e. encrypt) the message m using his **private key** (k_B^-), such that he'll be sending: $k_B^-(m)$, along with his public key (k_B^+) – which is public (i.e. known to all).

NB: Despite using the Public Key encryption system, where a message m would usually be encrypted by using the public key, in this case, Bob will use his private key.

- To verify that Bob is the one sending the signed message $k_B^-(m)$:
    - k_B^- is secret and only known to Bob.
    - The pair (m, $k_B^-(m)$) will be sent over the network.
    - By taking Bob's public key, Alice can check if $k_B^+\left(k_B^-(m)\right) = m$
    - Alice can confirm that Bob was indeed the person who signed it, as she has access to the original message m from the pair, and by applying k_B^+, she gets the same m, from the encrypted $k_B^-(m)$.

- To check if the encrypted message is forged or not:
    - Say, Alice has received $k_T^-(m)$, which Alice doesn't know that m is signed by Trudy – but is being considered as Bob's encrypted message.
    - When Alice applies k_B^+, to check if $k_B^+\left(k_T^-(m)\right) = m$, is considered very improbable!
    - This reasoning can also be used to provide irrefutable proof that Bob did sign the message, should Bob decide to challenge the fact that he didn't sign a document.

Using message digests:

- By sending the pair (m, $k_B^-(m)$), essentially we're sending two copies of the original message, so (m, $k_B^-(m)$) = $2m$.
- It is computationally expensive to use public-key encryption for long messages, as seen in RSA. Thus, the goal: want a fixed length, easy-to-compute "digital fingerprint".

- In FIPS 186-4, an approved hash function (MD5: 128 bit; SHA-1: 160 bit) is used to make the message m, smaller.
- Why hash function?
  - produces fixed-size irrespective how large m is.
  - If we're given a message digest x, it is computationally infeasible to find the original message m, such that: x = $H(m)$.

Digital Signature using signed message digest

- Bob sends digitally signed message:
  - m: some large message
  - H: hash function is applied to m, s.t. $x = H(m)$
  - Bob then digitally signs (encrypts) the message digest $k_B^-(x)$
  - Bob then sends Alice: $(m, x, k_B^-(x))$

- Alice can then verify the signature integrity:
  - H is previously agreed upon, so she can apply it on m, to get x_A.
  - Then by doing: $k_B^+\left(k_B^-(x)\right) = x_A'$
  - If $x_A' = x_A$, then she can definitively state that Bob signed m.

- This approach should prevent a "man in the middle attack".
  Say, if Trudy constructs m', it would be infeasible for her to construct a $H(m') = H(m)$.

A brief overview on another variation on the Digital Signature Algorithm (DSA):

RSADSA

- RSADSA key = (RSA private key, RSA public key)
  - private key used to compute the Digital Signature
  - public key used to verify the Digital Signature.

- Just like in general RSA:
  - The public key is computed modulus N [where N = pq; p,q are both (large) primes]; NB: standard length for N: 1024, 2048 and 3062 bits.
  - Public key exponent: e
  - Thus, the RSA public key = (N, e)
  - The private key follows a similar format, except with a private key exponent: d
  - Thus, the RSA private key = (N, d)

- Security measures: p, q, d are all secret and is generated via approved random bit generator; N, e are known to the public.