Cryptography and Network Security

Digital Signature Standard

Elgamal

Schnorr

Session Meta Data

Author	**Dr T Sree Sharmila**
Reviewer	
Version Number	1.0
Release Date	22 July 2018

Revision History

Revision Date	Details	Version no.
		1.0

Agenda

- Introduction
- Digital signature (DS) model
  - Attacks & Forgeries
- DS requirement
- Direct DS
- Elgamal DS
- Schnorr DS
- Digital Signature Standard
- Digital Signature Algorithm
- DSS overview
- Summary
- Test your understanding
- References

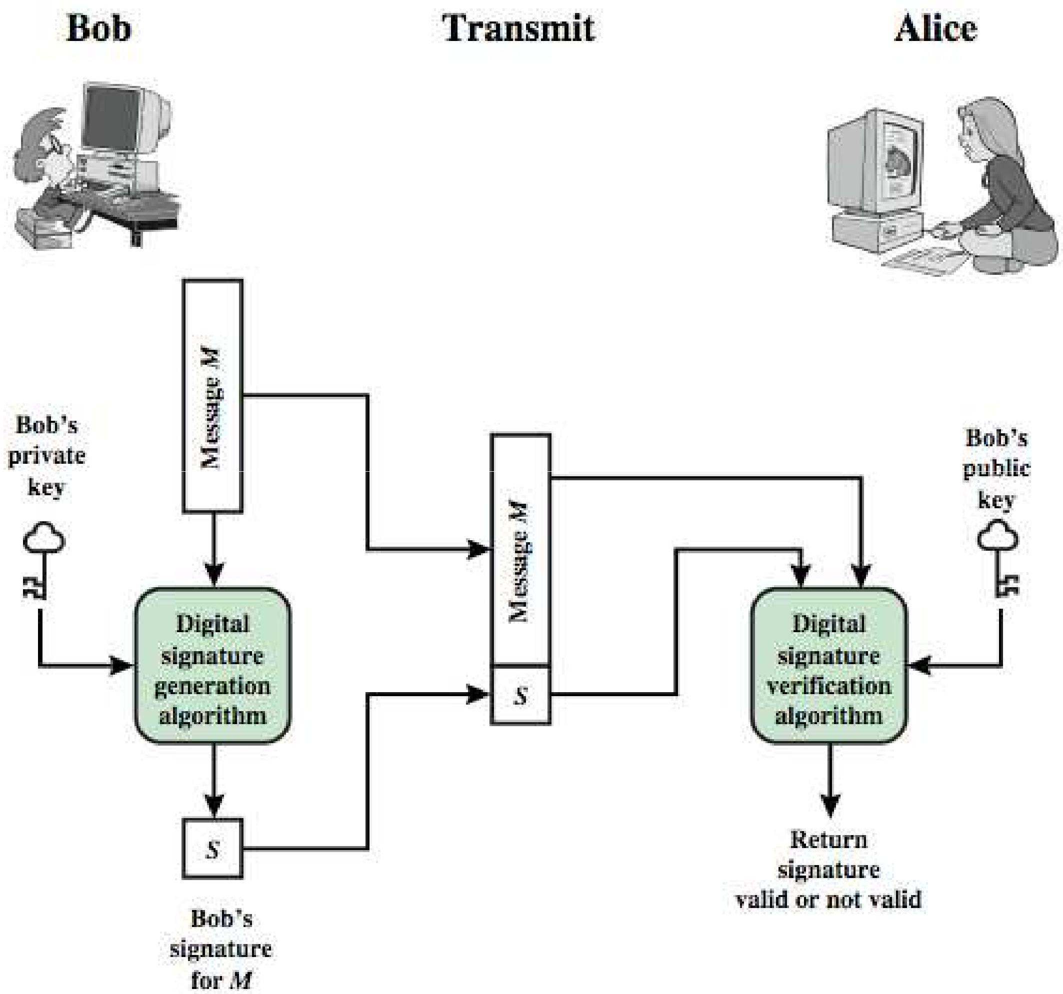
Introduction - Digital Signatures

- have looked at message authentication
  - but does not address issues of lack of trust
- digital signatures provide the ability to:
  - verify author, date & time of signature
  - authenticate message contents
  - be verified by third parties to resolve disputes
- hence include authentication function with additional capabilities

Agenda

- Introduction
- Digital signature (DS) model
  - Attacks & Forgeries
- DS requirement
- Direct DS
- Elgamal DS
- Schnorr DS
- Digital Signature Standard
- Digital Signature Algorithm
- DSS overview
- Summary
- Test your understanding
- References

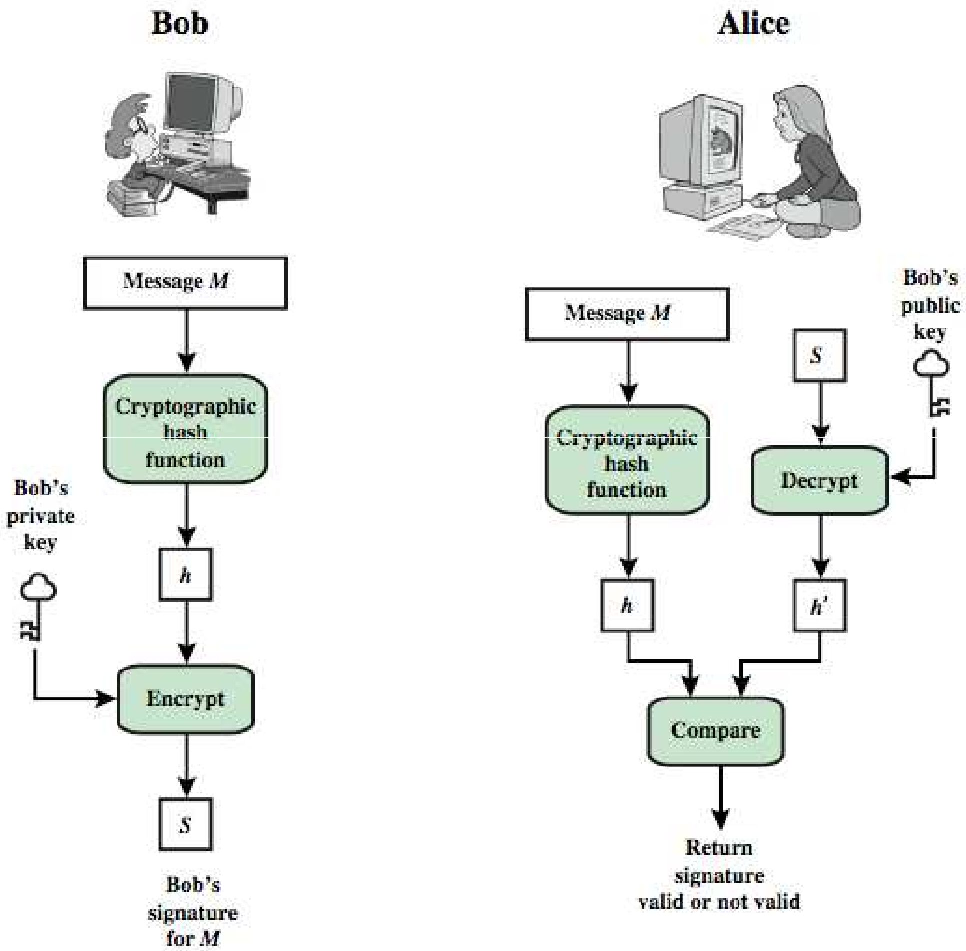
Digital Signature Model



Digital Signatures Model

- Figure shows generic model of the process of making and using digital signatures.
- Bob can sign a message using a digital signature generation algorithm.
- The inputs to the algorithm are the message and Bob's private key.
- Any other user, say Alice, can verify the signature using a verification algorithm, whose inputs are the message, the signature, and Bob's public key.

Digital Signature Model



Agenda

- Introduction
- Digital signature (DS) model
  - Attacks & Forgeries
- DS requirement
- Direct DS
- Elgamal DS
- Schnorr DS
- Digital Signature Standard
- Digital Signature Algorithm
- DSS overview
- Summary
- Test your understanding
- References

Attacks and Forgeries

- attacks
  - key-only attack
  - known message attack
  - generic chosen message attack
  - directed chosen message attack
  - adaptive chosen message attack
- break success levels
  - total break
  - selective forgery
  - existential forgery

Attacks and Forgeries

- **Key-only attack**: C only knows A's public key.
- **Known message attack**: C is given access to a set of messages and signatures.
- **Generic chosen message attack**: C chooses a list of messages before attempting to breaks A's signature scheme, independent of A's public key. C then obtains from A valid signatures for the chosen messages. The attack is generic because it does not depend on A's public key; the same attack is used against everyone.
- **Directed chosen message attack**: Similar to the generic attack, except that the list of messages is chosen after C knows A's public key but before signatures are seen.
- **Adaptive chosen message attack**: C is allowed to use A as an "oracle." This means the A may request signatures of messages that depend on previously obtained message-signature pairs.

Attacks and Forgeries

- [GOLD88] then defines success as breaking a signature scheme as an outcome in which C can do any of the following with a non-negligible probability:
- **Total break**: C determines A's private key. • Universal forgery: C finds an efficient signing algorithm that provides an equivalent way of constructing signatures on arbitrary messages.
- **Selective forgery**: C forges a signature for a particular message chosen by C.
- **Existential forgery**: C forges a signature for at least one message. C has no control over the message. Consequently this forgery may only be a minor nuisance to A.

Agenda

- Introduction
- Digital signature (DS) model
  - Attacks & Forgeries
- DS requirement
- Direct DS
- Elgamal DS
- Schnorr DS
- Digital Signature Standard
- Digital Signature Algorithm
- DSS overview
- Summary
- Test your understanding
- References

Digital Signature Requirements

- must depend on the message signed
- must use information unique to sender
  - to prevent both forgery and denial
- must be relatively easy to produce
- must be relatively easy to recognize & verify
- be computationally infeasible to forge
  - with new message for existing digital signature
  - with fraudulent digital signature for given message
- be practical save digital signature in storage

Agenda

- Introduction
- Digital signature (DS) model
  - Attacks & Forgeries
- DS requirement
- Direct DS
- Elgamal DS
- Schnorr DS
- Digital Signature Standard
- Digital Signature Algorithm
- DSS overview
- Summary
- Test your understanding
- References

Direct Digital Signatures

- involve only sender & receiver
- assumed receiver has sender's public-key
- digital signature made by sender signing entire message or hash with private-key
- can encrypt using receivers public-key
- important that sign first then encrypt message & signature
- security depends on sender's private-key

Agenda

- Introduction
- Digital signature (DS) model
  - Attacks & Forgeries
- DS requirement
- Direct DS
- Elgamal DS
- Schnorr DS
- Digital Signature Standard
- Digital Signature Algorithm
- DSS overview
- Summary
- Test your understanding
- References

ElGamal Digital Signatures

- signature variant of ElGamal, related to D-H
  - so uses exponentiation in a finite (Galois)
  - with security based difficulty of computing discrete logarithms, as in D-H
- use private key for encryption (signing)
- uses public key for decryption (verification)
- each user (eg. A) generates their key
  - chooses a secret key (number): $1 < x_A < q-1$
  - compute their **public key**: $y_A = a^{x_A} \bmod q$

ElGamal Digital Signature

- Alice signs a message M to Bob by computing
  - the hash $m = H(M)$, $0 <= m <= (q-1)$
  - chose random integer K with $1 <= K <= (q-1)$ and $\gcd(K,q-1)=1$
  - compute temporary key: $S_1 = a^k \bmod q$
  - compute K^{-1} the inverse of $K \bmod (q-1)$
  - compute the value: $S_2 = K^{-1}(m-x_A S_1) \bmod (q-1)$
  - signature is: (S_1, S_2)
- any user B can verify the signature by computing
  - $V_1 = a^m \bmod q$
  - $V_2 = y_A^{S_1} S_1^{S_2} \bmod q$
  - signature is valid if $V_1 = V_2$

ElGamal Signature Example

- use field GF(19) $q=19$ and $a=10$
- Alice computes her key:
  - A chooses $x_A=16$ & computes $y_A=10^{16} \bmod 19 = 4$
- Alice signs message with hash $m=14$ as $(3,4)$:
  - choosing random $K=5$ which has $\gcd(18,5)=1$
  - computing $S_1 = 10^5 \bmod 19 = 3$
  - finding $K^{-1} \bmod (q-1) = 5^{-1} \bmod 18 = 11$
  - computing $S_2 = 11(14-16.3) \bmod 18 = 4$
- any user B can verify the signature by computing
  - $V_1 = 10^{14} \bmod 19 = 16$
  - $V_2 = 4^3.3^4 = 5184 = 16 \bmod 19$
  - since $16 = 16$ signature is valid

Agenda

- Introduction
- Digital signature (DS) model
  - Attacks & Forgeries
- DS requirement
- Direct DS
- Elgamal DS
- Schnorr DS
- Digital Signature Standard
- Digital Signature Algorithm
- DSS overview
- Summary
- Test your understanding
- References

Schnorr Digital Signatures

- also uses exponentiation in a finite (Galois)
  - security based on discrete logarithms, as in D-H
- minimizes message dependent computation
  - multiplying a 2*n-bit* integer with an *n-bit* integer
- main work can be done in idle time
- have using a prime modulus p
  - $p-1$ has a prime factor q of appropriate size
  - typically p 1024-bit and q 160-bit numbers

Schnorr Key Setup

- choose suitable primes p, q
- choose a such that $a^q = 1 \bmod p$
- (a,p,q) are global parameters for all
- each user (eg. A) generates a key
  - chooses a secret key (number): $0 < s_A < q$
  - compute their **public key**: $v_A = a^{-sA} \bmod q$

Schnorr Signature

- user signs message by
  - choosing random r with $0<r<q$ and computing $x = a^r \bmod p$
  - concatenate message with x and hash result to computing: $e = H(M \,||\, x)$
  - computing: $y = (r + se) \bmod q$
  - signature is pair (e, y)
- any other user can verify the signature as follows:
  - computing: $x' = a^y v^e \bmod p$
  - verifying that: $e = H(M \,||\, x')$

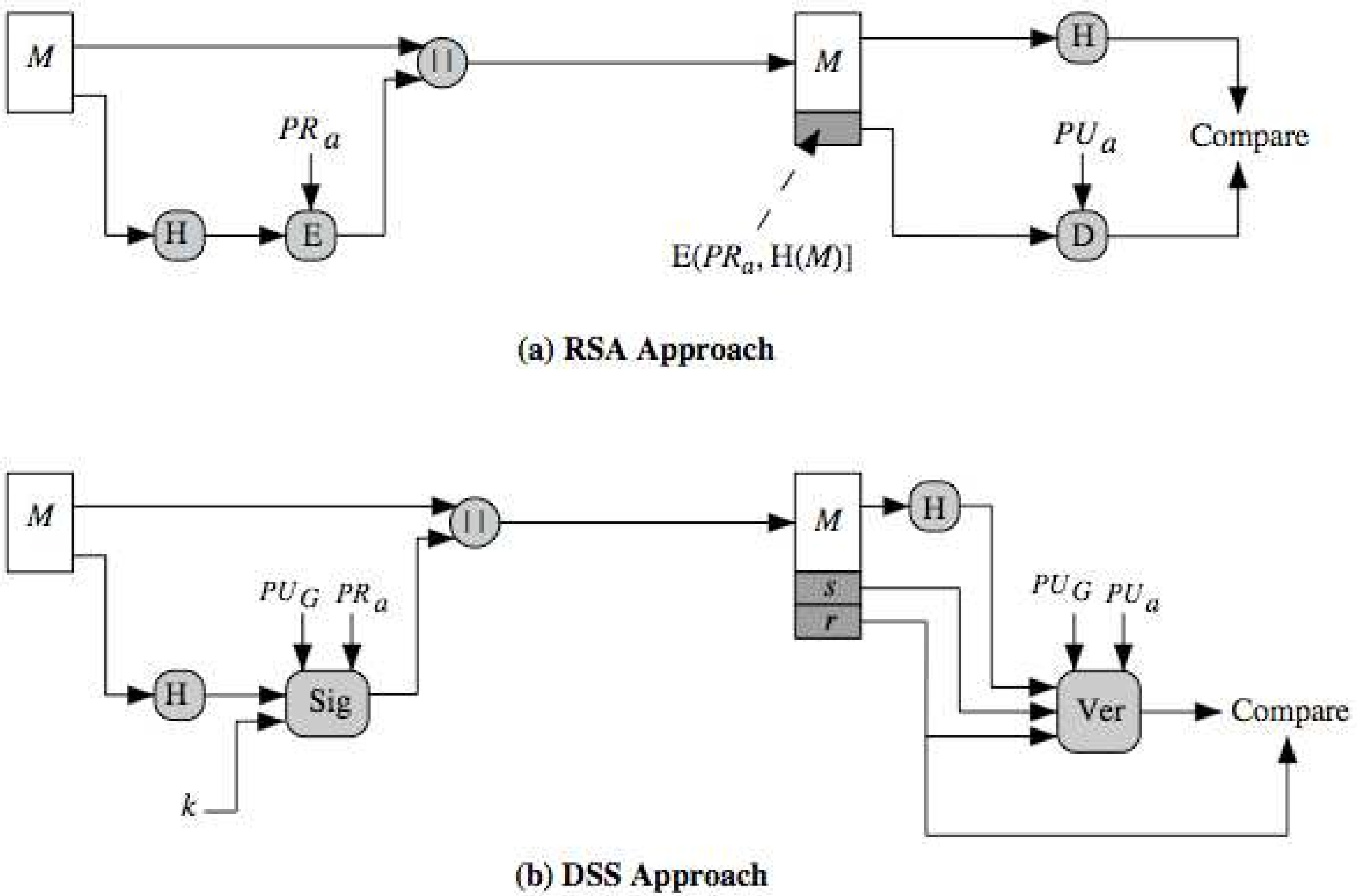
Agenda

- Introduction
- Digital signature (DS) model
  - Attacks & Forgeries
- DS requirement
- Direct DS
- Elgamal DS
- Schnorr DS
- Digital Signature Standard
- Digital Signature Algorithm
- DSS overview
- Summary
- Test your understanding
- References

Digital Signature Standard (DSS)

- US Govt approved signature scheme
- designed by NIST & NSA in early 90's
- published as FIPS-186 in 1991
- revised in 1993, 1996 & then 2000
- uses the SHA hash algorithm
- DSS is the standard, DSA is the algorithm
- FIPS 186-2 (2000) includes alternative RSA & elliptic curve signature variants
- DSA is digital signature only unlike RSA
- is a public-key technique

DSS vs RSA Signatures



DSS vs RSA Signatures

- In the RSA approach, the message to be signed is input to a hash function that produces a secure hash code of fixed length.
- This hash code is then encrypted using the sender's private key to form the signature.
- Both the message and the signature are then transmitted.
- The recipient takes the message and produces a hash code.
- The recipient also decrypts the signature using the sender's public key.
- If the calculated hash code matches the decrypted signature, the signature is accepted as valid.
- Because only the sender knows the private key, only the sender could have produced a valid signature.
- The DSS approach also makes use of a hash function.

DSS vs RSA Signatures

- The hash code is provided as input to a signature function along with a random number *k* generated for this particular signature.
- The signature function also depends on the sender's private key (PR_a) and a set of parameters known to a group of communicating principals.
- We can consider this set to constitute a global public key (PU_G).
- The result is a signature consisting of two components, labeled s and r.
- At the receiving end, the hash code of the incoming message is generated.
- This plus the signature is input to a verification function.
- The verification function also depends on the global public key as well as the sender's public key (PU_a), which is paired with the sender's private key.
- The output of the verification function is a value that is equal to the signature component r if the signature is valid.
- The signature function is such that only the sender, with knowledge of the private key, could have produced the valid signature.

Agenda

- Introduction
- Digital signature (DS) model
  - Attacks & Forgeries
- DS requirement
- Direct DS
- Elgamal DS
- Schnorr DS
- Digital Signature Standard
- Digital Signature Algorithm
- DSS overview
- Summary
- Test your understanding
- References

Digital Signature Algorithm (DSA)

- creates a 320 bit signature
- with 512-1024 bit security
- smaller and faster than RSA
- a digital signature scheme only
- security depends on difficulty of computing discrete logarithms
- variant of ElGamal & Schnorr schemes

DSA Key Generation

- have shared global public key values (p,q,g):
  - choose 160-bit prime number q
  - choose a large prime p with $2^{L-1} < p < 2^{L}$
    - where L= 512 to 1024 bits and is a multiple of 64
    - such that q is a 160 bit prime divisor of (p-1)
  - choose $g = h^{(p-1)/q}$
    - where $1<h<p-1$ and $h^{(p-1)/q} \bmod p > 1$
- users choose private & compute public key:
  - choose random private key: $x<q$
  - compute public key: $y = g^{x} \bmod p$

DSA Signature Creation

- to **sign** a message M the sender:
  - generates a random signature key k, $k<q$
  - nb. k must be random, be destroyed after use, and never be reused
- then computes signature pair:

  $r = (g^k \bmod p) \bmod q$

  $s = [k^{-1}(H(M)+ xr)] \bmod q$
- sends signature (r,s) with message M

DSA Signature Verification

- having received M & signature (r,s)
- to **verify** a signature, recipient computes:

  $w = s^{-1} \bmod q$

  $u1 = [H(M)w] \bmod q$

  $u2 = (rw) \bmod q$

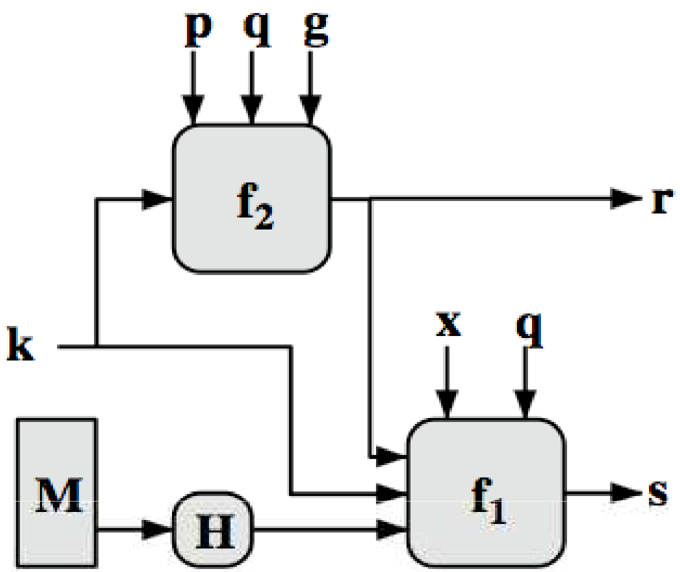
  $v = [(g^{u1} y^{u2}) \bmod p] \bmod q$

- if $v=r$ then signature is verified
- see Appendix A for details of proof why

Agenda

- Introduction
- Digital signature (DS) model
  - Attacks & Forgeries
- DS requirement
- Direct DS
- Elgamal DS
- Schnorr DS
- Digital Signature Standard
- Digital Signature Algorithm
- DSS overview
- Summary
- Test your understanding
- References

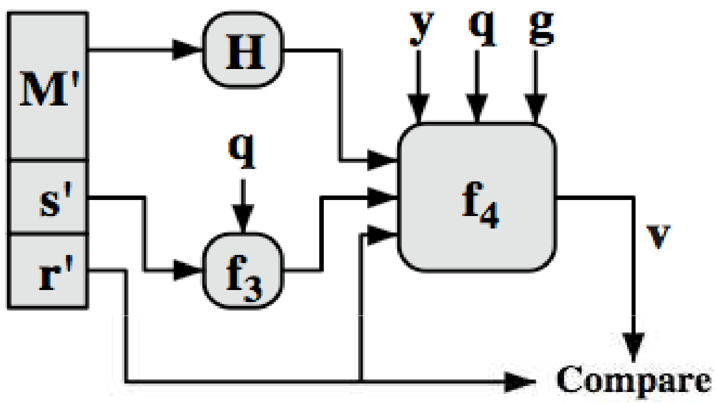
DSS Overview



$$s = f_1(H(M), k, x, r, q) = (k^{-1}(H(M) + xr)) \bmod q$$

$$r = f_2(k, p, q, g) = (g^k \bmod p) \bmod q$$

(a) Signing



$$w = f_3(s', q) = (s')^{-1} \bmod q$$

$$v = f_4(y, q, g, H(M'), w, r')$$

$$= ((g^{(H(M')w) \bmod q}\, y^{r'w \bmod q}) \bmod p) \bmod q$$

(b) Verifying

DSS Overview

- The structure of the algorithm, as revealed here is quite interesting.
- Note that the test at the end is on the value r, which does not depend on the message at all.
- Instead, r is a function of k and the three global public-key components.
- The multiplicative inverse of k (mod q) is passed to a function that also has as inputs the message hash code and the user's private key.
- The structure of this function is such that the receiver can recover r using the incoming message and signature, the public key of the user, and the global public key.

Agenda

- Introduction
- Digital signature (DS) model
  - Attacks & Forgeries
- DS requirement
- Direct DS
- Elgamal DS
- Schnorr DS
- Digital Signature Standard
- Digital Signature Algorithm
- DSS overview
- Summary
- Test your understanding
- References

Summary

- have discussed:
  - digital signatures
  - ElGamal & Schnorr signature schemes
  - digital signature algorithm and standard

Agenda

- Introduction
- Digital signature (DS) model
  - Attacks & Forgeries
- DS requirement
- Direct DS
- Elgamal DS
- Schnorr DS
- Digital Signature Standard
- Digital Signature Algorithm
- DSS overview
- Summary
- Test your understanding
- References

Test your understanding

1) Explain the following:
    1) Elgamal DS
    2) Schnorr DS
2) Explain digital signature model.

Agenda

- Introduction
- Digital signature (DS) model
  - Attacks & Forgeries
- DS requirement
- Direct DS
- Elgamal DS
- Schnorr DS
- Digital Signature Standard
- Digital Signature Algorithm
- DSS overview
- Summary
- Test your understanding
- References

References

1. William Stallings, Cryptography and Network Security, 6th Edition, Pearson Education, March 2013.
2. Charlie Kaufman, Radia Perlman and Mike Speciner, “Network Security”, Prentice Hall of India, 2002.