The DSA digital signature scheme

GIANLUCA DINI
Dept. of Ingegneria dell'Informazione
University of Pisa
email: gianluca.dini@unipi.it

Version: 07/04/2025

1

Digital signatures

THE ELGAMAL SIGNATURE SCHEME

2

Elgamal in a nutshell

- Invented in 1985
- Based on difficulty of discrete logarithm
- Digital signature operations are different from the cipher operations

In schoolbook RSA operations are the same, but in general encryption scheme is different from the digital signature scheme. RSA is the only case in which everything is the same except key switching

Apr-25 Digital signatures 3

3

Key generation

- Choose a large prime *p*
- Choose a primitive element α of (a subgroup of) $\mathbb{Z}_p^*$
- Choose a random number *d* ∈ {2, 3,…,p – 2}
- Compute β = α^d mod p
- pubK = (p, α, β)
- privK = d

(α^{p-1} according to Fermat little theorem is 1)

α is a generator, usually in the set {2,…, p-2} generator could be p-1, but it is discarded usually: Suppose p-1=g.

Apr-25 Digital signatures 4

4

$(p-1)^2 = p^2 - 2p + 1 \equiv 1 \bmod p$

So p-1 is order 2 so you produce a subgroup of order 2.

Signature generation

- Input message x
- Choose an ephemeral key k_E in $\{0, 1, \underbrace{2, p-2}_{\text{EXCLUDE } 0,1,2}\}$ such that $\gcd(k_E, p-1) = 1$
- Compute the signature parameters
  - $r \equiv \alpha^{k_E} \bmod p$
  - $s \equiv (x - d \cdot r)k_E^{-1} \bmod p - 1$
  - (r, s) is the digital signature
- Output $\langle x, (r, s)\rangle$

Apr-25 Digital signatures 5

5

Signature verification

- Let
  - (p, α, β) be the public key;
  - x be the message and
  - (r, s) be the digital signature
- Compute $t \equiv \beta^r \cdot r^s \bmod p$
- **If** ($t \equiv \alpha^x \bmod p$)
  return True;
  else return False

Apr-25 Digital signatures 6

6

Proof

1. Let $t \equiv \beta^r \cdot r^s \equiv (\alpha^d)^r(\alpha^{kE})^s \equiv \alpha^{d \cdot r + kE \cdot s} \bmod p$
2. If $\beta^r \cdot r^s \equiv \alpha^x \bmod p$ then $\alpha^x \equiv \alpha^{d \cdot r + kE \cdot s} \bmod p$ [Eq. a]
3. According to Fermat's Little Theorem Eq.a holds if $x \equiv d \cdot r + k_E \cdot s \bmod p - 1$
4. from which the construction of parameter $s = (x - d \cdot r)k_E^{-1} \bmod p - 1$

Apr-25 Digital signatures 7

7

Computational aspects

- Key generation
  - Generation of a large prime (1024 bits)
  - True random generator for the private key
  - Exponentiation by square-and-multiply (compute public key)
- Signature generation r and s are in the size of the prime
  - $|s| = |r| = |p|$ thus $|x, (r, s)| = 3\ |x|$ (*dig sig expansion*) as large as message
  - One exponentiation by square-and-multiply
  - One inverse $k_E^{-1} \bmod p$ by EEA (pre-computation)
- Signature verification I assume multiplicat. neglegeable?
  - Two exponentiations by square-and-multiply
  - One multiplication

Apr-25 Digital signatures 8

8

Security aspects

must be authentic

- The verifier must have the correct public key
- The DLP must be intractable
- ➔ Ephemeral key K_E cannot be reused
  - If K_E is reused the adversary can compute the private key *d* and impersonate the signer
- ➔ Existential forgery for a random message x unless it is hashed

scheme is subject to ex. forgery

Reuse of ephemeral key

- If the ephemeral key k_E is reused, an attacker can easily compute the private key d
  - Proof
    - Message x_1 and x_2 and the reused ephemeral key k_E
    - $(x_1, (s_1, r))$ and $(x_2, (s_2, r))$ where $r \equiv \alpha^{kE} \bmod p$
      - $s_1 \equiv (x_1 - d \cdot r) \cdot k_E^{-1} \bmod p - 1$ [Eqn. a]
      - $s_2 \equiv (x_2 - d \cdot r) \cdot k_E^{-1} \bmod p - 1$ [Eqn. b]
      - *Eqn.a and Eqn.b is a system in two unknowns* (k_E and d) and two equations
        - » $s_1 - s_2 \equiv (x_1 - x_2) \cdot k_E^{-1} \bmod p - 1$
        - » $k_E \equiv (x_1 - x_2) \cdot (s_1 - s_2)^{-1} \bmod p - 1$
        - » $d \equiv (x_1 - s_1 \cdot k_E) \cdot r^{-1} \bmod p - 1$

This becomes an easily solveable linear system

Q.E.D.

Existential Forgery Attack [→]

- The attack

Man in the middle

Alice	Adversary	Bob
		privK = d, pubK = (p, α, β)

< ---------------(p, α, β)-------------------

1. select i, j, s.t. gcd(j, p – 1) = 1
2. compute the signature
   $r \equiv \alpha^{i} \cdot \beta^{j} \bmod p$
   $s \equiv -r \cdot j^{-1} \bmod p - 1$
3. compute the message
   $x \equiv s \cdot i \bmod p - 1$

verification < ------------------(x, (r, s))---------------

$t \equiv \beta^{r} \cdot r^{s} \bmod p$ since

$t \equiv \alpha^{x} \bmod p$ ➔ valid signature!

We can prove that if we don't hash message adversary can forge

You can prove that choosing in a specific way r, s, you get signature

Apr-25 Digital signatures 11

11

Attack is existential: x is a side product of a computation

Existential forgery attack [→]

- Proof
  - **Step 1.** $t \equiv \beta^{r} \cdot r^{s} \equiv (\alpha^{d})^{r} \cdot (\alpha^{i} \cdot \beta^{j})^{s} \equiv (\alpha^{d})^{r} \cdot (\alpha^{i} \cdot \alpha^{d \cdot j})^{s} \equiv \alpha^{d \cdot r} \cdot (\alpha^{i+d \cdot j})^{s}$

    $\equiv \alpha^{d \cdot r} \cdot (\alpha^{i+d \cdot j})^{s} \equiv \alpha^{d \cdot r} \cdot \alpha^{(i+d \cdot j) \cdot (-r \cdot j^{-1})} \equiv$

    $\equiv \alpha^{d \cdot r} \cdot \alpha^{-d \cdot r} \cdot \alpha^{-r \cdot i \cdot j^{-1}} \equiv \alpha^{s \cdot i} \bmod p$ [Eqn. A]
  - **Step 2.** As the message was constructed as $x \equiv s \cdot i \bmod p$ then Eqn. a $t \equiv \alpha^{s \cdot i} \equiv \alpha^{x} \bmod p$ which is the condition to accept the signature as valid

Apr-25 Digital signatures 12

12

Existential forgery attack

- **Existential forgery**. In Step 3, the adversay computes message x whose semantics (s)he cannot control
- **How to avoid the forgery**. The attack is not feasible if the message is hashed:

$$s \equiv (H(x) - d \cdot r)k_E^{-1} \bmod p - 1$$

Apr-25 Digital signatures 13

13

Digital Signatures

DIGITAL SIGNATURE ALGORITHM (DSA)

Apr-25 Digital signatures 14

14

Introduction

→ DSA is a variation to make it more efficient, obtained by exploiting subgroups

- The Elgamal scheme is rarely used in practice
- DSA is a more popular variant
  - It's a federal US government standard for digital signatures (DSS)
  - It was proposed by NIST
- Advantages of DSA w.r.t. Elgamal
  - Signature is only 320 bits
  - Some attacks against Elgamal are not applicable to DSA

Key Generation

very large

1. Generate a prime p with $2^{1023} < p < 2^{1024}$.
2. Find a prime divisor q of p−1 with $2^{159} < q < 2^{160}$.
3. Find an element α with ord(α) = q, i.e., α generates the *subgroup with q elements*.
4. Choose a random integer d with 0 < d < q. → Private key selected in a subgroup
5. Compute $\beta \equiv \alpha^d \bmod p$.
6. The keys are now: pubK = (p,q,α,β); privK = (d)

Makes Diffie Hellman hard

Central idea

- DSA uses two cyclic groups
  - $\mathbb{Z}_p^*$, the order of which has bit lenght 2014 bit
  - H_q, a 160-bit subgroup of $\mathbb{Z}_p^*$
  - This setup yields shorter signatures
- Other combinations are possible

p	q	signature
1024	160	320
2048	224	448
3072	256	512

Apr-25 Digital signatures 17

8

17

Calculations are made mod q to make things more efficient.

Signature Generation

1. Choose an integer as random ephemeral key k_E with $0 < k_E < q$.
2. Compute $r \equiv (\alpha^{k_E} \bmod p) \bmod q$.
3. Compute $s \equiv (SHA(x) + d \cdot r)k_E^{-1} \bmod q$. $q \neq 1$
   - SHA-1(·) produces a 160-bit value (size of q not size of p)
4. Digital signature is the pair (r, s)
   - 160 + 160 = 320 bit long

Apr-25 Digital signatures 18

18

Remember: DSA is an elgamal variation and they use $\mathbb{Z}_p^*$ and a subgroup on 160 bits to keep security of algorithm and make it more efficient.
You can also move on EC with point multiplication.

Signature Verification

1. Compute auxiliary value $w \equiv s^{-1} \bmod q$.
2. Compute auxiliary value $u_1 \equiv w \cdot SHA(x) \bmod q$.
3. Compute auxiliary value $u_2 \equiv w \cdot r \bmod q$.
4. Compute $v \equiv (\alpha^{u1} \cdot \beta^{u2} \bmod p) \bmod q$.
5. The verification follows from:
   - **If** ($v \equiv r \bmod q$)
     return TRUE
     else return FALSE

Apr-25 Digital signatures 19

19

Proof [→]

- We show that a signature (r, s) satisfies the verification condition $v \equiv r \bmod q$.
  - $s \equiv (SHA(x)+d\ r)k_E^{-1} \bmod q$ which is equivalent to $k_E \equiv s^{-1} SHA(x)+d\ s^{-1}\ r \bmod q$.
  - The right-hand side can be expressed in terms of the auxiliary values u1 and u2: $k_E \equiv u_1+d\ u_2 \bmod q$.
  - We can raise α to either side of the equation if we reduce modulo p: $\alpha^{kE} \bmod p \equiv \alpha^{u1 + d\ u2} \bmod p$

[→]

Apr-25 Digital signatures 20

20

Proof

- Since the public key value β was computed as $\beta \equiv \alpha^d \bmod p$, we can write: $\alpha^{kE} \equiv \alpha^{u1} \beta^{u2} \bmod p$.
- We now reduce both sides of the equation modulo q: $(\alpha^{kE} \bmod p) \bmod q \equiv (\alpha^{u1}\beta^{u2} \bmod p) \bmod q$.
- Since r was constructed as $r \equiv (\alpha^{kE} \bmod p) \bmod q$ and $v \equiv (\alpha^{u1}\beta^{u2} \bmod p) \bmod q$,
- this expression is identical to the condition for verifying a signature as valid: $r \equiv v \bmod q$.

Apr-25 Digital signatures 21

21

Computational aspects [→]

- Key Generation
  - The most challenging phase
    - Find a $\mathbb{Z}_p^*$ with 1024-bit prime p and a subgroup in the range of 2^{160}
    - This condition is fulfilled if $|\mathbb{Z}_p^*| = |p-1|$ has a prime factor q of 160 bit
  - General approch:
    - To find q first and then p

Apr-25 Digital signatures 22

22

Computational aspects [→]

- Signing
  - Computing **r** requires exponentiation
    - Operands are on 1024 bit
    - Exponent q is on 160 bit
      - On average 160 + 80 = 240 SQs and MULTs
    - Result is reduced mod q
    - Does not depend on message x so can be precomputed
  - Computing **s**
    - Involve 160-bit operands
    - The most costly operation is inverse

Apr-25 Digital signatures 23

23

Computational aspects

- Verification
  - Computing the auxiliary parameters w, u_1 and u_2 involves 160-bit operands
  - This is relatively fast

Apr-25 Digital signatures 24

24

Security

- We have to protect from two different DLPs
  1. $d = \log_\alpha \beta$ mod p.
     - Index calcolus attack
       - Prime p must be on 1024 bits for 80-bit security level
  2. α generates a subgroup of order q
     - Index calculus attack cannot be applied
     - Only generic DLP attacks can be used
       - Square-root attacks: Baby-step giant-step, Pollard's rho
       - Running time: $\sqrt{q} = \sqrt{2^{160}} = 80$
- Vulerable to k_E reuse
  - Analalogue to ElGamal

Apr-25 Digital signatures 25

25

Apr-25 Digital signatures 26

26