Elliptic-curve cryptography XI
Identificaion schemes and signatures

Tanja Lange

Eindhoven University of Technology

2MMC10 – Cryptology

Schnorr identification protocol – How to prove you know a

P, n known. Alice has published $Q = aP$ as her public key.

Alice (prover)		**Bob (verifier)**
A commits to $r \leftarrow_R \{0, 1, \ldots, n-1\}$	$\xrightarrow{R \leftarrow rP}$	B picks challenge
	$\xleftarrow{h}$	$h \leftarrow_R \{0, 1, \ldots, n-1\}$
$s \leftarrow r + ha \bmod n$	$\xrightarrow{s}$	verifies $sP \stackrel{?}{=} R + hQ$

Schnorr identification protocol – How to prove you know a

P, n known. Alice has published $Q = aP$ as her public key.

Alice (prover)		**Bob (verifier)**
A commits to $r \leftarrow_R \{0, 1, \ldots, n-1\}$	$\xrightarrow{R \leftarrow rP}$	B picks challenge
	$\xleftarrow{h}$	$h \leftarrow_R \{0, 1, \ldots, n-1\}$
$s \leftarrow r + ha \bmod n$	$\xrightarrow{s}$	verifies $sP \stackrel{?}{=} R + hQ$

Valid choices always verify:

Schnorr identification protocol – How to prove you know a

P, n known. Alice has published $Q = aP$ as her public key.

Alice (prover)		**Bob (verifier)**
A commits to $r \leftarrow_R \{0, 1, \ldots, n-1\}$	$\xrightarrow{R \leftarrow rP}$	B picks challenge
	$\xleftarrow{h}$	$h \leftarrow_R \{0, 1, \ldots, n-1\}$
$s \leftarrow r + ha \bmod n$	$\xrightarrow{s}$	verifies $sP \stackrel{?}{=} R + hQ$

Valid choices always verify: $sP = (r + ha)P = R + hQ$.

Does this prove that Alice knows a?

Schnorr identification protocol – How to prove you know a

P, n known. Alice has published $Q = aP$ as her public key.

Alice (prover)		**Bob (verifier)**
A commits to $r \leftarrow_R \{0, 1, \ldots, n-1\}$	$\xrightarrow{R \leftarrow rP}$	B picks challenge
	$\xleftarrow{h}$	$h \leftarrow_R \{0, 1, \ldots, n-1\}$
$s \leftarrow r + ha \bmod n$	$\xrightarrow{s}$	verifies $sP \stackrel{?}{=} R + hQ$

Valid choices always verify: $sP = (r + ha)P = R + hQ$.

Does this prove that Alice knows a?

If she knew h before sending R

Schnorr identification protocol – How to prove you know a

P, n known. Alice has published $Q = aP$ as her public key.

Alice (prover)		**Bob (verifier)**
A commits to $r \leftarrow_R \{0, 1, \ldots, n-1\}$	$\xrightarrow{R \leftarrow rP}$	B picks challenge
	$\xleftarrow{h}$	$h \leftarrow_R \{0, 1, \ldots, n-1\}$
$s \leftarrow r + ha \bmod n$	$\xrightarrow{s}$	verifies $sP \stackrel{?}{=} R + hQ$

Valid choices always verify: $sP = (r + ha)P = R + hQ$.

Does this prove that Alice knows a?

If she knew h before sending R she could put $R = -hQ, s = 0$,

Schnorr identification protocol – How to prove you know a

P, n known. Alice has published $Q = aP$ as her public key.

Alice (prover)		**Bob (verifier)**
A commits to $r \leftarrow_R \{0, 1, \ldots, n-1\}$	$\xrightarrow{R \leftarrow rP}$	B picks challenge
	$\xleftarrow{h}$	$h \leftarrow_R \{0, 1, \ldots, n-1\}$
$s \leftarrow r + ha \bmod n$	$\xrightarrow{s}$	verifies $sP \stackrel{?}{=} R + hQ$

Valid choices always verify: $sP = (r + ha)P = R + hQ$.

Does this prove that Alice knows a?

If she knew h before sending R she could put $R = -hQ, s = 0$,
or, less suspicious, pick $s \leftarrow_R \{0, 1, \ldots, n-1\}$, put $R = sP - hQ$.

Consequence 1: Alice has chance $1/n$ of cheating by guessing h. ✓

Consequence 2: If for fixed R Alice can answer for challenges $h_1 \neq h_2$ she knows a;

Schnorr identification protocol – How to prove you know a

P, n known. Alice has published $Q = aP$ as her public key.

Alice (prover)		**Bob (verifier)**
A commits to $r \leftarrow_R \{0, 1, \dots, n-1\}$	$\xrightarrow{R \leftarrow rP}$	B picks challenge
	$\xleftarrow{h}$	$h \leftarrow_R \{0, 1, \dots, n-1\}$
$s \leftarrow r + ha \bmod n$	$\xrightarrow{s}$	verifies $sP \stackrel{?}{=} R + hQ$

Valid choices always verify: $sP = (r + ha)P = R + hQ$.

Does this prove that Alice knows a?
If she knew h before sending R she could put $R = -hQ, s = 0$,
or, less suspicious, pick $s \leftarrow_R \{0, 1, \dots, n-1\}$, put $R = sP - hQ$.

Consequence 1: Alice has chance $1/n$ of cheating by guessing h. ✓
Consequence 2: If for fixed R Alice can answer for challenges $h_1 \neq h_2$ she knows a; but doing so reveals a. (see exercises).
Consequence 3: Bob does not learn anything about a as he could have produced the "transcript" $[R, h, s]$ without Alice. ✓

Schnorr identification protocol – How to prove you know a

P, n known. Alice has published $Q = aP$ as her public key.

Alice (prover)		**Bob (verifier)**
A commits to		
$r \leftarrow_R \{0, 1, \ldots, n-1\}$	$\xrightarrow{R \leftarrow rP}$	B picks challenge
	$\xleftarrow{h}$	$h \leftarrow_R \{0, 1, \ldots, n-1\}$
$s \leftarrow r + ha \bmod n$	$\xrightarrow{s}$	verifies $sP \stackrel{?}{=} R + hQ$

Valid choices always verify: $sP = (r + ha)P = R + hQ$.

Does this prove that Alice knows a?
If she knew h before sending R she could put $R = -hQ, s = 0$,
or, less suspicious, pick $s \leftarrow_R \{0, 1, \ldots, n-1\}$, put $R = sP - hQ$.

Consequence 1: Alice has chance $1/n$ of cheating by guessing h. ✓
Consequence 2: If for fixed R Alice can answer for challenges $h_1 \neq h_2$ she knows a; but doing so reveals a. (see exercises).✓
Consequence 3: Bob does not learn anything about a as he could have produced the "transcript" $[R, h, s]$ without Alice. ✓

Schnorr signature

P, n known. Alice has published $Q = aP$ as her public key.

Alice (prover)		**Bob (verifier)**
A commits to		
$r \leftarrow_R \{0, 1, \ldots, n-1\}$	$\xrightarrow{R \leftarrow rP}$	B picks challenge
	$\xleftarrow{h}$	$h \leftarrow_R \{0, 1, \ldots, n-1\}$
$s \leftarrow r + ha \bmod n$	$\xrightarrow{s}$	verifies $sP \stackrel{?}{=} R + hQ$

Signatures are non-interactive. Alice signs message m.
Bob later verifies signature, obtains proof that a was used.

Schnorr signature

P, n known. Alice has published $Q = aP$ as her public key.

Alice (prover)		**Bob (verifier)**
A commits to		
$r \leftarrow_R \{0, 1, \ldots, n-1\}$	$\xrightarrow{R \leftarrow rP}$	B picks challenge
	$\xleftarrow{h}$	$h \leftarrow_R \{0, 1, \ldots, n-1\}$
$s \leftarrow r + ha \bmod n$	$\xrightarrow{s}$	verifies $sP \stackrel{?}{=} R + hQ$

Signatures are non-interactive. Alice signs message m.
Bob later verifies signature, obtains proof that a was used.

Let Alice choose h?!

Schnorr signature

P, n known. Alice has published $Q = aP$ as her public key.

Alice (prover)		**Bob (verifier)**
A commits to		
$r \leftarrow_R \{0, 1, \ldots, n-1\}$	$\xrightarrow{R \leftarrow rP}$	B picks challenge
	$\xleftarrow{h}$	$h \leftarrow_R \{0, 1, \ldots, n-1\}$
$s \leftarrow r + ha \bmod n$	$\xrightarrow{s}$	verifies $sP \stackrel{?}{=} R + hQ$

Signatures are non-interactive. Alice signs message m.
Bob later verifies signature, obtains proof that a was used.

~~Let Alice choose h?!~~ Enforce choice of $h = H(m)$ using hash function H.

Schnorr signature

P, n known. Alice has published $Q = aP$ as her public key.

Alice (prover) **Bob (verifier)**

A commits to

$r \leftarrow_R \{0, 1, \ldots, n-1\}$ $\xrightarrow{R \leftarrow rP}$ B picks challenge

$\xleftarrow{h}$ $h \leftarrow_R \{0, 1, \ldots, n-1\}$

$s \leftarrow r + ha \bmod n$ $\xrightarrow{s}$ verifies $sP \stackrel{?}{=} R + hQ$

Signatures are non-interactive. Alice signs message m.
Bob later verifies signature, obtains proof that a was used.

~~Let Alice choose h?!~~ Enforce choice of $h = H(m)$ using hash function H. Still a problem: she knows h before committing to R.

Schnorr signature

P, n known. Alice has published $Q = aP$ as her public key.

Alice (prover)		**Bob (verifier)**
A commits to		
$r \leftarrow_R \{0, 1, \ldots, n-1\}$	$\xrightarrow{R \leftarrow rP}$	B picks challenge
	$\xleftarrow{h}$	$h \leftarrow_R \{0, 1, \ldots, n-1\}$
$s \leftarrow r + ha \bmod n$	$\xrightarrow{s}$	verifies $sP \stackrel{?}{=} R + hQ$

Signatures are non-interactive. Alice signs message m.
Bob later verifies signature, obtains proof that a was used.

~~Let Alice choose h?!~~ Enforce choice of $h = H(m)$ using hash function H. Still a problem: she knows h before committing to R. Enforce order of choices by putting $h = H(R, m)$

Schnorr signature

P, n known. Alice has published $Q = aP$ as her public key.

Alice (prover)		**Bob (verifier)**
A commits to		
$r \leftarrow_R \{0, 1, \dots, n-1\}$	$\xrightarrow{R \leftarrow rP}$	B picks challenge
	$\xleftarrow{h}$	$h \leftarrow_R \{0, 1, \dots, n-1\}$
$s \leftarrow r + ha \bmod n$	$\xrightarrow{s}$	verifies $sP \stackrel{?}{=} R + hQ$

Signatures are non-interactive. Alice signs message m.
Bob later verifies signature, obtains proof that a was used.

~~Let Alice choose h?!~~ Enforce choice of $h = H(m)$ using hash function H. Still a problem: she knows h before committing to R. Enforce order of choices by putting $h = H(R, m)$

Sign: Signature is (R, s)
$r \leftarrow_R \{0, 1, \dots, n-1\}, R \leftarrow rP, h \leftarrow H(R, m), s \leftarrow r + ha \bmod n$.
Verify: $sP \stackrel{?}{=} R + hQ$.

EdDSA – Ed25519

Let $p = 2^{255} - 19, d = -121665/121666$ and

$$E : -x^2 + y^2 = 1 + dx^2y^2.$$

Base point P has prime order ℓ, $|E(\mathbf{F}_p)| = 8\ell$.

Scheme follows Schnorr, with some improvements:

- Put $h = H(R, Q, m)$ to reduce multi-target attacks.
- Verify $8sP = 8R + 8hQ$ to deal with cofactor (can also check without 8).
- Choose r pseudorandomly to avoid issues with bad randomness.

ECDSA – Elliptic curve digital signature algorithm

Similar setup, different equation for s.
More expensive due to inversions modulo n.
Mostly result of patent avoidance (Schnorr patent expired by now).

Sign: Signature is (R', s)
$r \leftarrow_R \{0, 1, \ldots, n-1\}, R \leftarrow rP, R' \leftarrow x(R) \bmod n$,
(R' is x-coordinate of R taken as integer, then reduced modulo n)
$s \leftarrow r^{-1}(H(m) + R'a) \bmod n$.

Verify: $w_1 \leftarrow s^{-1}H(m) \bmod n$ and $w_2 \leftarrow s^{-1} \cdot R' \bmod n$.
Check $x(w_1P + w_2Q) \equiv R' \bmod n$

ECDSA – Elliptic curve digital signature algorithm

Similar setup, different equation for s.
More expensive due to inversions modulo n.
Mostly result of patent avoidance (Schnorr patent expired by now).

Sign: Signature is (R', s)
$r \leftarrow_R \{0, 1, \ldots, n-1\}, R \leftarrow rP, R' \leftarrow x(R) \bmod n$,
(R' is x-coordinate of R taken as integer, then reduced modulo n)
$s \leftarrow r^{-1}(H(m) + R'a) \bmod n$.

Verify: $w_1 \leftarrow s^{-1}H(m) \bmod n$ and $w_2 \leftarrow s^{-1} \cdot R' \bmod n$.
Check $x(w_1P + w_2Q) \equiv R' \bmod n$

Alice's signature is valid:
$w_1P + w_2Q =$

ECDSA – Elliptic curve digital signature algorithm

Similar setup, different equation for s.
More expensive due to inversions modulo n.
Mostly result of patent avoidance (Schnorr patent expired by now).

Sign: Signature is (R', s)
$r \leftarrow_R \{0, 1, \ldots, n-1\}, R \leftarrow rP, R' \leftarrow x(R) \bmod n$,
(R' is x-coordinate of R taken as integer, then reduced modulo n)
$s \leftarrow r^{-1}(H(m) + R'a) \bmod n$.

Verify: $w_1 \leftarrow s^{-1}H(m) \bmod n$ and $w_2 \leftarrow s^{-1} \cdot R' \bmod n$.
Check $x(w_1P + w_2Q) \equiv R' \bmod n$

Alice's signature is valid:
$w_1P + w_2Q = (s^{-1}H(m))P + (s^{-1} \cdot R')Q =$
$(s^{-1}(H(m) + R'a))P = rP$, and so the x-coordinate of this expression equals $R' \equiv x(rP) \bmod n$.

ECDSA – Elliptic curve digital signature algorithm

Similar setup, different equation for s.
More expensive due to inversions modulo n.
Mostly result of patent avoidance (Schnorr patent expired by now).

Sign: Signature is (R', s)
$r \leftarrow_R \{0, 1, \ldots, n-1\}, R \leftarrow rP, R' \leftarrow x(R) \bmod n$,
(R' is x-coordinate of R taken as integer, then reduced modulo n)
$s \leftarrow r^{-1}(H(m) + R'a) \bmod n$.

Verify: $w_1 \leftarrow s^{-1}H(m) \bmod n$ and $w_2 \leftarrow s^{-1} \cdot R' \bmod n$.
Check $x(w_1P + w_2Q) \equiv R' \bmod n$

Alice's signature is valid:
$w_1P + w_2Q = (s^{-1}H(m))P + (s^{-1} \cdot R')Q =$
$(s^{-1}(H(m) + R'a))P = rP$, and so the x-coordinate of this expression equals $R' \equiv x(rP) \bmod n$.

Similar fragility about reuse of r, see PS3 Epic Fail (talk at 27C3).
r called a "nonce": number used only once.