Security of Elliptic Curves: domain parameters & implementations

Centre for Research on Cryptography and Security

Ján Jančár*

Masaryk University

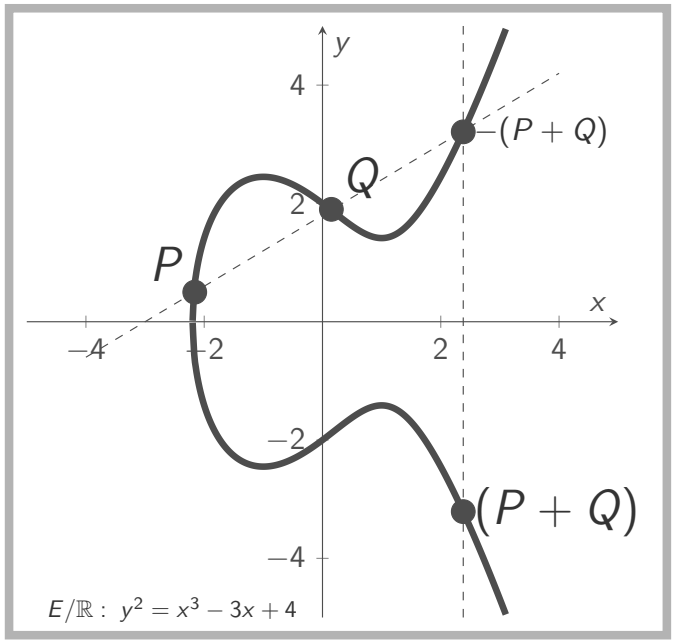
Brno, Czech Republic

SantaCrypt 2018

29.11.2018

*In collaboration with Petr Švenda, Marek Sýs and Vladimír Sedláček.

Table of Contents



- Cryptosystems based on the ECDLP on elliptic curves over $\mathbb{F}_p$ or $\mathbb{F}_{2^m}$
- ECDH(E) and ECDSA are most popular ECIES, EdDSA, ECMQV ...
- Security depends on parameters and implementations

! **Problem:** Classes of weak curves

! **Problem:** Challenging to implement correctly/securely

Shared domain parameters: (p, a, b, G, n, h)
$E/\mathbb{F}_p: \; y^2 \equiv x^3 + ax + b$
generator G
$|G| = n, \;\; |E(\mathbb{F}_p)| = nh$

ECDH

- private key $d_A \in [1, n-1]$, public key $W_B = [d_B]G$
- shared secret $H(([d_A]W_B)_x)$

ECDSA

- private key $d \in [1, n-1]$, public key $W = [d]G$, message m
- sign:
  - $k \leftarrow_R [1, n-1]$
  - $r = ([k]G)_x$
  - $s = k^{-1}(H(m) + rd) \mod n$
  - signature (r, s)
- verify:
  - $u_1 = H(m)s^{-1} \mod n$
  - $u_2 = rs^{-1} \mod n$
  - $r \stackrel{?}{=} ([u_1]G + [u_2]W)_x$

- Classes of weak curves, w.r.t. ECDLP:
  - Smooth generator order $\Longrightarrow$ Pohlig-Hellman decomposition
  - Anomalous curves $\Longrightarrow$ Additive transfer
  - Low embedding degree $\Longrightarrow$ Multiplicative transfer
  - Extension field curves $\Longrightarrow$ Index-calculus method
  - Small complex multiplication discriminant $\Longrightarrow$ Speedups to discrete logarithm algorithms
- **Current solution:** Use standard curves
- ? What if there is a publicly unknown vulnerability in some standard named curves?*

*Bernstein et. al, How to manipulate curve standards: a white paper for the black hat

- Has to implement finite field arithmetic for $\mathbb{F}_p$ or $\mathbb{F}_{2^m}$
- Has to choose a curve model, or use what some standard requires
- Has to choose point representation, affine/projective/Jacobian/mixed/...?
- Has to choose & implement addition-formulas
- Has to choose & implement scalar-multiplication algorithm
- Has to implement the cryptosystem, with key validation

- No point validation in ECDH $\Longrightarrow$ Invalid / Twist / Degenerate curve attacks
- Some point validation in ECDH $\Longrightarrow$ Small-subgroup attacks
- Side-channels $\Longrightarrow$ Timing attacks / Power analysis
- Faults
- Incorrect computation $\Longrightarrow$? nothing / private key leak ?

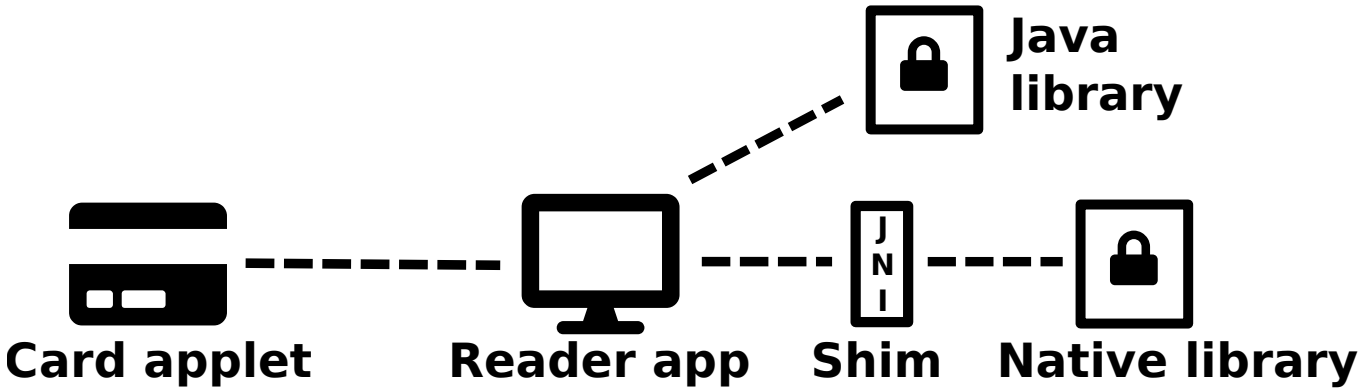
- OpenSSL bug in modular multiplication on NIST P-256.[1]
- OpenSSL timing private key leak in ECDSA.[2]
- Invalid curve attacks against TLS servers.[3]
- Private key leak in Go standard library on NIST P-256.[4]
- Java/NSS point arithmetic bug.[5, 6]
- Bluetooth Low Energy invalid curve attacks against many implementations.[7]
- Missing point validation step in CryptoNote based cryptocurrencies.[8]

- Very popular secure elements
- Payment cards, SIM cards, digital signatures, ePassports, eID ...
- **JavaCard**, .NET, MULTOS
- Upload applet(s), then communicate with them using ISO7816-4 APDUs
- Offers ECDH and ECDSA over $\mathbb{F}_p$ and $\mathbb{F}_{2^m}$ curves, also key generation
- Works with the short Weierstrass curve model
- Proprietary APIs sometimes available, but to customers only
- ! **Problem:** Black-box implementation. Has to trust manufacturer or certifications.

Black-box testing

Do black-box testing.

- Use past errors in implementations to test current ones
- Use additional data from knowledge of open-source implementations to test edge-cases
- Use data that is 'interesting' in some way, special classes of curves, weak curves, malformed data ...
- Do cross-verification of large number of inputs-outputs
- ☑ Usable with black-box access
- ☑ May catch a mis-behaving/insecure implementation, or learn about its workings
- ☑ Independent of manufacturer and certifications



- Tool for black-box testing elliptic curve cryptography implementations
- JavaCard applet, reader app, tool for software libraries
- Offers utility functions: key generation/ECDH/ECDSA
- Offers running of test suites: hundreds of curves, points, individual test cases, 12 test suites, 5 main categories
- Testing (currently) only on cards

crocs-muni.github.io/ECTester

Support & Performance

- Does the card have ECDH, if so which?
- Does the card have ECDSA, if so which?
- Does the card compute ECDH correctly for the test-vectors (from SECG, NIST CVP, Brainpool)?
- Are curves present by default, if so which?
- Do the card results of ECDH/ECDSA verify using another implementation?
- Measure duration of the operations

Public key validation

- Public key during ECDH should be verified to be on the correct subgroup of the correct curve
- If check is missing:
  - Short Weierstrass $\Longrightarrow$ invalid curve attacks
  - Edwards, Montgomery ... $\Longrightarrow$ twist and degenerate curve attacks
  - Any model $\Longrightarrow$ small-subgroup attacks
- Check might be evaded using a compressed point

$$y^2 \equiv x^3 + ax + b$$

- Active attack on static ECDH [3]
- Some short Weierstrass addition formulas do not use b parameter of curve equation
- Scalar multiplication assumes the point is on curve
- b value might be 'sneaked' in by the attackers public point
- Scalar multiplication is then performed on invalid curve

1. Pre-compute set of curves, with points of order p_i
2. Send points of low prime order on invalid curves during ECDH
3. Each ECDH result leaks private key mod p_i
4. Recover private key using CRT

- Montgomery and Brier-Joye x-only scalar multiplication
- Computes correctly with point on the quadratic twist [9]
- Quadratic twist might not be as secure as original curve

Public key validation

Degenerate curve attack

- Points of form $(0, y)$ for $y \in \mathbb{F}_p$
- Certain addition formulas on Edwards, Montgomery . . . curves
- Scalar multiplication is equivalent to exponentiation in $\mathbb{F}_p$ [10]
- Sub-exponential discrete logarithm via index-calculus method
- $\mathbb{F}_p$ is much smaller than what is used in DH usually

- ANSI X9.62 defines format for compressed points
- Only need x-coordinate and one bit for y
- y recovered as square root of rhs: $y = sqrt(x^3 + ax + b)$
- If rhs is not a quadratic-residue, modular square root might compute false result
- If output of decompression is assumed on curve
- Can sneak in point not on curve

Signature verification

- JavaCard uses ASN.1 DER encoding of (r, s) signature pair
- Is the parsing of this correct? What happens if we give it malformed data?
- It should hold that $r, s \in [1, n-1]$, what if not? Try specific values such as 0, 1, $n-1$, n, ...

Miscellaneous behavior tests

- To peak inside the black-box
- Barreto-Naehrig(pairing friendly), MNT curves, Curve25519 transformed into short Weierstrass
- Composite order curves
- Anomalous curves ($|E(\mathbb{F}_p)| = p$)
- Supersingular curves ($|E(\mathbb{F}_p)| = p + 1$)
- Low embedding degree d curves ($|E(\mathbb{F}_p)|$ divides $p^d - 1$)
- Parameters that are not a correct curve (composite p in $\mathbb{F}_p$)

Vulnerabilities & Edge-case values

- Values that trigger known bugs
  - OpenSSL modular multiplication bug
  - Java/NSS bug
  - Go stdlib bug
- Edge-case values of private key, around 0, around p when $p < n$, around n, bit patterns, windowing patterns ...

- Tested 8 card models (NXP, Infineon, G&D, Athena)
- Have support for OpenSSL, BoringSSL, wolfSSL, Bouncy Castle, SunEC, Crypto++, Botan, libtomcrypt, libgcrypt, Microsoft CNG
- Tested cards demonstrated behavior outside of JavaCard specification
- Support & Performance results are public

	Test result	Description	Result
−	✔	Tests of 192b ALG_EC_FP support: Some.	SUCCESS
2	✔	Allocate both keypairs 192b ALG_EC_FP	SUCCESS
2	✔	Generate both keypairs	SUCCESS
2	✔	Allocate both keypairs 192b ALG_EC_FP	SUCCESS
2	✔	Set custom curve parameters on both keypairs	SUCCESS
2	✔	Generate both keypairs	SUCCESS
−	✔	KeyAgreement tests.	SUCCESS
−	✔	Test of the ALG_EC_SVDP_DH KeyAgreement.	SUCCESS
4	✔	Allocate KeyAgreement(ALG_EC_SVDP_DH) object	SUCCESS
4	✔	ALG_EC_SVDP_DH of local pubkey and remote privkey(unchanged point)	SUCCESS
4	✖	ALG_EC_SVDP_DH of local pubkey and remote privkey(COMPRESSED point)	FAILURE
4	✔	Mean =137477828 ns, Median =137503097 ns, Mode =135560799 ns	SUCCESS
+	✔	Test of the ALG_EC_SVDP_DHC KeyAgreement.	SUCCESS
+	✔	Test of the ALG_EC_SVDP_DH_PLAIN KeyAgreement.	SUCCESS
+	✔	Test of the ALG_EC_SVDP_DHC_PLAIN KeyAgreement.	SUCCESS
3	✖	Allocate KeyAgreement(ALG_EC_PACE_GM) object	FAILURE
3	✖	Allocate KeyAgreement(ALG_EC_SVDP_DH_PLAIN_XY) object	FAILURE
+	✔	Signature tests.	SUCCESS

Legend

- ☰: Displays either a toggle icon for a test that is composed of several tests, or a depth of a simple test.
- **Test result:** The high-level result of a test of a group of tests.
- **Description:** A short description of what the test or a group of tests did.
- **Result:** The concrete result of a test or a group of tests. Can be one of:
  - `SUCCESS` - ✔ - The test was expected to pass and it did.
  - `FAILURE` - ✖ - The test was expected to pass but it failed.
  - `UXSUCCESS` - ✖ - The test was expected to fail but it succeeded.
  - `XFAILURE` - ✔ - The test was expected to fail and it did.
  - `ERROR` - ⚠ - There was an *unexpected* error while running the test.

Figure: Example result from ECTester website.

Conclusions & Future work

- Developed the tool for thorough testing of black-box ECC implementations
- Found issues in tested cards, more investigation necessary
- Test libraries
- Test more cards
- Side-channel analysis

Thanks for your time!

crocs-muni.github.io/ECTester

xjancar@fi.muni.cz

neuromancer.sk

MUNI FI

CROCS
Centre for Research on Cryptography and Security

References I

- Brumley, Barbosa, Page and Vercauteren; Practical realisation and elimination of an ECC-related software bug attack
- Brumley and Tuveri; Remote Timing Attacks are Still Practical
- Jager, Schwenk and Somorovsky; Practical Invalid Curve Attacks on TLS-ECDH
- CVE-2017-8932
- CVE-2017-10176
- CVE-2017-7781
- CVE-2018-5383
- Spagni; Disclosure of a Major Bug in CryptoNote Based Currencies
- Bernstein and Lange; `https://safecurves.cr.yp.to/`

- Neves and Tibouchi; Degenerate Curve Attacks
- Lim and Lee; A key recovery attack on discrete log-based schemes using a prime order subgroup