Public-key cryptography in Tor and pluggable transports

Tanja Lange

Technische Universiteit Eindhoven

09 June 2016
Attend Roger’s talk on Friday.
Motivation

Motivation #1 Channels are spying on our (meta-)data.
Motivation #2 Channels are modifying our (meta-)data.
Motivation #3 Channels interrupt and block suspicious communication.
DH key exchange

“Jefferson” \( (x, y) \) → Censor → “Madison”

- Censor wants to block Tor (or whatever) traffic.
- Censor knows that Tor uses curve \( E : y^2 = x^3 + ax + b \) over finite field \( \mathbb{F}_p \).
- Jefferson sends \( (x, y) \) on \( E \).
- Censor intercepts message, parses it as two field elements, checks whether \( (x, y) \) is a point on \( E \). If so, break connection.
- Hasse’s theorem says there are around \( p \) points on \( E \) over \( \mathbb{F}_p \); that’s very small compared to \( p^2 \) pairs. Random chance \( 1/p \).
DH key exchange

“Jefferson” $x_1, x_2, x_3 \cdots \rightarrow \text{Censor} \rightarrow \text{“Madison”}$

- Jefferson sends $x$, belonging to $(x, y)$ on $E$.
- Each connection starts with a DH handshake, so there are several $x_i$.
- Censor intercepts message, parses it as one field element, checks whether $x_i$ belongs to a point $(x_i, y_i)$ on $E$. If so sufficiently often, break connection.
- Hasse’s theorem says there are around $p$ points on $E$ over $\mathbb{F}_p$. Most come in pairs $(x, \pm y)$.
- About half of all values in $\mathbb{F}_p$ appear as $x$-coordinates.
- Random chance $1/2^n$ after $n$ messages.
- This ignores $p$ not being a power of 2, e.g. worse for $p = 2^{256} - 2^{224} + 2^{192} + 2^{96} - 1$. 
Wanted!

- Make transmission of points indistinguishable from random strings.
- Have significant fraction of all points covered.

This still leaves a lot of problems

Censor can cut all communication.

But once traffic looks uniformly random (symmetric crypto has a much easier time on this) it can be steganographically layered on top of "accepted" communication.

Needed for Telex (Wustrow, Wolchok, Goldberg, and Halderman; USENIX 2011) and StegoTorus (Weinberg, Wang, Yegneswaran, Briesemeister, Cheung, Wang, and Boneh; ACM CCS 2012).

Needed also for kleptography (exfiltrating keys to the adversary), e.g. Young and Yung SCN 2010.
Wanted!

- Make transmission of points indistinguishable from random strings.
- Have significant fraction of all points covered.
- This still leaves a lot of problems
  - Censor can cut all communication.
  - Censor can cut all https traffic.
Wanted!

- Make transmission of points indistinguishable from random strings.
- Have significant fraction of all points covered.
- This still leaves a lot of problems
  - Censor can cut all communication.
  - Censor can cut all HTTPS traffic.
- But once traffic looks uniformly random (symmetric crypto has a much easier time on this) it can be steganographically layered on top of “accepted” communication.
- Needed for Telex (Wustrow, Wolchok, Goldberg, and Halderman; USENIX 2011) and StegoTorus (Weinberg, Wang, Yegneswaran, Briesemeister, Cheung, Wang, and Boneh; ACM CCS 2012).
Wanted!

- Make transmission of points indistinguishable from random strings.
- Have significant fraction of all points covered.
- This still leaves a lot of problems
  - Censor can cut *all* communication.
  - Censor can cut all HTTPS traffic.
- But once traffic looks uniformly random (symmetric crypto has a much easier time on this) it can be steganographically layered on top of “accepted” communication.
- Needed for Telex (Wustrow, Wolchok, Goldberg, and Halderman; USENIX 2011) and StegoTorus (Weinberg, Wang, Yegneswaran, Briesemeister, Cheung, Wang, and Boneh; ACM CCS 2012).
- Needed also for kleptography (exfiltrating keys to the adversary), e.g. Young and Yung SCN 2010.
How to use the idea

- Let $S \subseteq \{0, 1\}^t$. Here: $S \subseteq \mathbb{F}_p$.
- Want map $\iota : S \rightarrow E(S)$ and inverse (limited to set $\iota(S)$).
- Want $\iota$ and $\iota^{-1}$ be efficiently computable and $\iota(S)$ be large in $E(\mathbb{F}_p)$, e.g. cover about half of all points.
- In DH, Jefferson picks $j$, computes $jP$. If $jP \notin \iota(S)$ he picks a new $j$. He sends $\iota^{-1}(jP)$. Same for Madison. On average 2 tries, only in local computation.
- In Schnorr signatures, signer Bob has public key $\tau_B = \iota^{-1}(bP)$ and private key $b$.
  To sign $m$, the sender picks random $r$ until $rP \in \iota(S)$, computes $\tau = \iota^{-1}(rP)$, $h = H(\tau || \tau_B || m)$, $s = r + hb \pmod{\ell}$. The signature is $(\tau, s)$.
- Signature verification:
  Compute $bP = \iota(\tau_B)$, $rP = \iota(\tau)$, $h = H(\tau || \tau_B || m)$.
  Compare $rP + h(bP)$ and $sP$.
  This works: $sP = (r + hb)P = rP + h(bP)$.
Two approaches . . . and their shortcomings

Assume that \( p \) is close to power of 2.

- Hash strings to curve points; increment till valid \( x \)-coordinate is found.
  - Points can have multiple preimages.
  - Points can have no preimages.
  - Really hard to get uniform distribution (reject with probability proportional to the number of preimages? How many are there? How to get deterministic map?).
  - Finding all the preimages means point counting.

- Use curve \( E \) and its quadratic twist \( E' \).
  - Each \( x \in \mathbb{F}_p \) belongs to two points: \((x, \pm y)\) on \( E \), \((x, \pm y)\) on \( E' \) or \((x, 0)\) on both curves.
  - Get uniformity by switching to right curve.
  - Requires two keys for everything (doubles key size).
  - Problems with parties choose non-matching curves in DH.
Elligator!

Joint work with Bernstein, Hamburg, and Krasnova (CCS 2013).

We use slightly different curve shape.

\[ y^2 = x^3 + Ax^2 + Bx \]

with \( AB(A^2 - 4B) \neq 0 \) (usually \( A = 0 \) included but not here).

- This curve has a point \((0, 0)\) of order 2.
- For \( B = 1 \) called *Montgomery curve* (can have \( C \) in \( Cy^2 \)).
- Tor uses Curve25519 in ntor for building circuits (see Friday?). Curve25519 is a Montgomery curve with \( A = 486662 \) and \( p = 2^{255} - 19 \).
Rewrite curve equation as $y^2 = x(x^2 + Ax + B)$.

Find two values $x_1, x_2$ such that

$$x_1^2 + Ax_1 + B = x_2^2 + Ax_2 + B \text{ and } x_1/x_2 \neq \Box.$$

In finite fields we have $\Box \cdot \Box = \Box$, so either $x_1$ or $x_2$ belongs to an $(x, y)$ on the curve (except for $y = 0$).

Transform equality into $x_1 + x_2 = -A$ (i.e. $x_1 = -A - x_2$).

Let $x_1/x_2 = ur^2$, where $u$ is a fixed non-square in $\mathbb{F}_p$.

Combine to $(-A - x_2)/x_2 = ur^2$, i.e. $x_2 = -A/(1 + ur^2)$ and $x_1 = -Aur^2/(1 + ur^2)$.

This defines map $\iota(r) = (x_1, \sqrt{x_1(x_1^2 + Ax_1 + B)})$ or $\iota(r) = (x_2, -\sqrt{x_2(x_1^2 + Ax_1 + B)})$ (pick the one defined).
Inverse map

- $\iota(S)$ is the set of $(x, y) \in E(\mathbb{F}_p)$ with
  - $x \neq -A$,
  - if $y = 0$ then $x = 0$, and
  - $-ux(x + A) = \Box$.

- If $(x, y) \in \iota(S)$ then $\bar{r} \in S$ is defined and $\iota(\bar{r}) = (x, y)$:

  $$\bar{r} = \begin{cases} 
  \sqrt{-x/((x + A)u)} & \text{if } y \in \sqrt{\mathbb{F}_p^2}; \\
  \sqrt{-(x + A)/(ux)} & \text{if } y \notin \sqrt{\mathbb{F}_p^2}.
  \end{cases}$$
Application to Curve25519

Here $q \equiv 1 \pmod{4}$ and $u = 2$ is a non-square.
Need to specify a square-root function for $\mathbb{F}_p$.

- Given a square $a \in \mathbb{F}_p$, compute $b = a^{(q+3)/8}$.
  (Note that $q \equiv 5 \pmod{8}$, so $(q + 3)/8$ is an integer.)
  Then $b^4 = a^2$, i.e., $b^2 \in \{a, -a\}$.

- Define $\sqrt{a}$ as $|b|$ if $b^2 = a$ and as $|b\sqrt{-1}|$ otherwise.

- Here $|b|$ means $b$ if $b \in \{0, 1, \ldots, (q - 1)/2\}$, otherwise $-b$.

Cost of computing $\iota$:

- 1 square-root computation,
- 1 inversion,
- 1 computation of square-root selection
- a few multiplications.

Note that the inversion and the square-root computation can be combined into one exponentiation,
More motivation

Motivation #1  Channels are spying on our (meta-)data.
Motivation #2  Channels are modifying our (meta-)data.
Motivation #3  Channels interrupt and block suspicious communication.
Motivation #4  Network nodes want to know how many of them exist.
Hidden services/onion services

- For better protection against eavesdropping, users can reach Facebook at https://facebookcorewwwi.onion.
- This means their traffic never leaves the Tor network.
- Facebook advertises their .onion page, so their existence is public.
- Other public .onion pages are xmpp servers for chat.
- Reasons for private .onion sites
  - Use Tor to deal with stupid network configuration (e.g. at TU/e).
  - Local chat services using Ricochet.
  - Collaborative servers (small group, not public).
  - File sharing, online shops, ...
  - Secure drop sites.
- General idea is that nobody knows all the existing sites.
- See Roger’s talk for more details.
Related keys

- Alice has secret key $a$ and public key $A = aP$ on elliptic curve.
- These are known to people she wants to connect with.
- Alice’s server changes location every day and there are Directory Services (DS) providing locations based on keys.
Related keys

- Alice has secret key $a$ and public key $A = aP$ on elliptic curve.
- These are known to people she wants to connect with.
- Alice’s server changes location every day and there are Directory Services (DS) providing locations based on keys.
- DSs are used randomly, but all servers will likely come by in a month, so for fixed keys the directory knows all servers.
- Alice goes to a conference and doesn’t want to bring $a$, but throw-away keys $A'$ for each day, but
  - She doesn’t want to get a new certificate for $A'$.
  - She doesn’t want to distribute new public keys.
  - She wants to be able to decrypt after the trip, but not keep old $a'$.
- Idea (Zooko Wilcox-O’Hearn; Gregory Maxwell; Robert Ransom; Christian Grothoff):
  If $d = H(date)$ is public, anybody can compute $A + dP$ or $dA$ which are public keys for $a + d$ or $ad$.
- Put $d = H(date, A)$, for $d$ secret from those not knowing $A$.
- Also used in Bitcoin (BIP 32), Tahoe-LAFS, and GNUNet.
How to use this idea?

- Make .onion addresses harder to harvest by directory servers (Tor track # 8106).
- DSs store information on location of $A$ under the key $A$, along with a signature under $A$.
- Alice can produce signatures under $A'$ from having $da$.
- There is no authority limiting the number of keys and servers. Of course anybody can submit a fake entry $B$ with a signature for its alleged location under $B$.
- But: nobody other than Alice can produce signature under $A'$.
- Recall Schnorr signatures: Signature on $m$ is $(R, s)$ with $R = rP$, $h = H(R||A||m)$, $s = r + ha \pmod{\ell}$. 
  Verification:
  Compute $h = H(R||A||m)$ and compare $R + h(A)$ and $sP$. 

How to use this idea?

- Toss in some more: make \( d = H(date||P||A). \)
- DS receives location date for server \( A' \) with signature under \( A' \) using \( a' = da \). Checks signature and stores information.
- Authorized client computes \( A' \) from \( A \) and date; asks DS for information on \( A' \).
- Client verifies signature on information obtained from DS, using \( A' \).
- Verification can use precomputed \( A' \) or include extra \( d \) in equations.
- A bit more tricky in practice to deal with Ed25519, which has nontrivial cofactors.
- This involves lots of non-standard crypto assumptions and modeling.