Dual EC

a standardized back door

Ruben Niederhagen

Joint work with Stephen Checkoway,[1] Matthew Fredrikson,[2]
Matthew Green,[1] Tanja Lange,[3] Thomas Ristenpart,[2]
Daniel J. Bernstein,[3,5] Jake Maskiewicz,[4] and Hovav Shacham.[4]
Related work: network scan by Adam Everspaugh.[2]

[1]Johns Hopkins University, [2]University of Wisconsin,
[3]Technische Universiteit Eindhoven, [4]UC San Diego,
[5]University of Illinois at Chicago

ECC 2014, Oct. 8 2014

TU/e Technische Universiteit **Eindhoven** University of Technology

Random Numbers in Cryptography

Random numbers are crucial for cryptography:

- generation of private keys for authentication,
- generation of secret keys for encryption,
- generation of secret nonces for digital signatures,
- generation of ephemeral keys for perfect-forward secrecy,
- . . .

Random Numbers in Cryptography

Random numbers are crucial for cryptography:

- generation of private keys for authentication,
- generation of secret keys for encryption,
- generation of secret nonces for digital signatures,
- generation of ephemeral keys for perfect-forward secrecy,
- ...

Must be impossible for an attacker to predict!

Random Numbers in Cryptography

Challenges of random number generation:

- computers are built to be deterministic,
- “real” randomness is rare.

Random Numbers in Cryptography

Challenges of random number generation:

- computers are built to be deterministic,
- "real" randomness is rare.

Common approach:

- use *pseudo* random numbers,
- start with a random *seed*,
- compute subsequent values deterministically,
  $\Rightarrow$ update a secret internal state.

Random Numbers in Cryptography

s_0

Random Numbers in Cryptography

$$\begin{tikzcd}
s_0 \arrow[d, "g(s_0)"] \\
r_0
\end{tikzcd}$$

Random Numbers in Cryptography

$$\begin{tikzcd}
s_0 \arrow[r, "f(s_0)"] \arrow[d, "g(s_0)"] & s_1 \\
r_0 &
\end{tikzcd}$$

Random Numbers in Cryptography

$$\begin{tikzcd}
s_0 \arrow[r, "f(s_0)"] \arrow[d, "g(s_0)"] & s_1 \arrow[d, "g(s_1)"] \\
r_0 & r_1
\end{tikzcd}$$

Random Numbers in Cryptography

$$\begin{tikzcd}
s_0 \arrow[r, "f(s_0)"] \arrow[d, "g(s_0)"] & s_1 \arrow[r, "f(s_1)"] \arrow[d, "g(s_1)"] & s_2 \\
r_0 & r_1 &
\end{tikzcd}$$

Random Numbers in Cryptography

$$\begin{tikzcd}
s_0 \arrow[r, "f(s_0)"] \arrow[d, "g(s_0)"] & s_1 \arrow[r, "f(s_1)"] \arrow[d, "g(s_1)"] & s_2 \arrow[r, "f(s_2)"] \arrow[d, "g(s_2)"] & s_3 \arrow[r, "f(s_3)"] \arrow[d, "g(s_3)"] & s_4 \arrow[r, "f(s_4)"] \arrow[d, "g(s_4)"] & \cdots \\
r_0 & r_1 & r_2 & r_3 & r_4 &
\end{tikzcd}$$

Random Numbers in Cryptography

$$\begin{tikzcd}
s_0 \arrow[r, "f(s_0)"] \arrow[d, "g(s_0)"] & s_1 \arrow[r, "f(s_1)"] \arrow[d, "g(s_1)"] & s_2 \arrow[r, "f(s_2)"] \arrow[d, "g(s_2)"] & s_3 \arrow[r, "f(s_3)"] \arrow[d, "g(s_3)"] & s_4 \arrow[r, "f(s_4)"] \arrow[d, "g(s_4)"] & \cdots \\
r_0 & r_1 & r_2 & r_3 & r_4 &
\end{tikzcd}$$

Broken if attacker learns internal state!

Random Numbers in Cryptography

$$\begin{tikzcd}
s_0 \arrow[r, "f(s_0)"] \arrow[d, "g(s_0)"] & s_1 \arrow[r, "f(s_1)"] \arrow[d, "g(s_1)"] & s_2 \arrow[r, "f(s_2)", red] \arrow[d, "g(s_2)", red] & s_3 \arrow[r, "f(s_3)", red] \arrow[d, "g(s_3)", red] & s_4 \arrow[r, "f(s_4)", red] \arrow[d, "g(s_4)", red] & \cdots \\
r_0 & r_1 \arrow[ur, dotted, red] & r_2 & r_3 & r_4 &
\end{tikzcd}$$

Broken if attacker learns internal state!

Dual EC – a Standardized Back Door

Topic of this talk:

The "potential" back door in the *Dual Elliptic Curve Deterministic Random Bit Generator* (Dual EC) standardized by ANSI, ISO, and NIST.

History of Dual EC

June 2004 Dual EC appears in an ANSI draft.

History of Dual EC

June 2004	Dual EC appears in an ANSI draft.
in 2004	RSA makes Dual EC the default RNG in BSAFE.

History of Dual EC

June 2004	Dual EC appears in an ANSI draft.
in 2004	RSA makes Dual EC the default RNG in BSAFE.
in 2005	An ISO standard is published including Dual EC.

History of Dual EC

June 2004	Dual EC appears in an ANSI draft.
in 2004	RSA makes Dual EC the default RNG in BSAFE.
in 2005	An ISO standard is published including Dual EC.
Dec. 2005	A draft is released by NIST including Dual EC.

History of Dual EC

June 2004	Dual EC appears in an ANSI draft.
in 2004	RSA makes Dual EC the default RNG in BSAFE.
in 2005	An ISO standard is published including Dual EC.
Dec. 2005	A draft is released by NIST including Dual EC.
early 2006	Several researchers, e.g., Schoenmakers and Sidorenko, point out cryptographic weaknesses in Dual EC.

History of Dual EC

June 2004	Dual EC appears in an ANSI draft.
in 2004	RSA makes Dual EC the default RNG in BSAFE.
in 2005	An ISO standard is published including Dual EC.
Dec. 2005	A draft is released by NIST including Dual EC.
early 2006	Several researchers, e.g., Schoenmakers and Sidorenko, point out cryptographic weaknesses in Dual EC.
June 2006	NIST SP 800/90A is published including Dual EC, ignoring the warnings.

History of Dual EC

June 2004	Dual EC appears in an ANSI draft.
in 2004	RSA makes Dual EC the default RNG in BSAFE.
in 2005	An ISO standard is published including Dual EC.
Dec. 2005	A draft is released by NIST including Dual EC.
early 2006	Several researchers, e.g., Schoenmakers and Sidorenko, point out cryptographic weaknesses in Dual EC.
June 2006	NIST SP 800/90A is published including Dual EC, ignoring the warnings. This includes Dual EC in FIPS 140-2, the typical certification for RNGs.

History of Dual EC

June 2004	Dual EC appears in an ANSI draft.
in 2004	RSA makes Dual EC the default RNG in BSAFE.
in 2005	An ISO standard is published including Dual EC.
Dec. 2005	A draft is released by NIST including Dual EC.
early 2006	Several researchers, e.g., Schoenmakers and Sidorenko, point out cryptographic weaknesses in Dual EC.
June 2006	NIST SP 800/90A is published including Dual EC, ignoring the warnings. This includes Dual EC in FIPS 140-2, the typical certification for RNGs.
Aug. 2007	Shumow and Ferguson demonstrate the basic back door.

Recent History of Dual EC

5 Sept. 2013	NSA's "Project Bullrun" is revealed by documents from Edward Snowden with the purpose `"to covertly introduce weaknesses into the encryption standards followed by hardware and software developers around the world."`

Recent History of Dual EC

5 Sept. 2013	NSA's "Project Bullrun" is revealed by documents from Edward Snowden with the purpose ``to covertly introduce weaknesses into the encryption standards followed by hardware and software developers around the world.'' The New York Times writes that ``the NSA had inserted a back door into a 2006 standard adopted by NIST [...] called the Dual EC DRBG standard.''

Recent History of Dual EC

5 Sept. 2013	NSA's "Project Bullrun" is revealed by documents from Edward Snowden with the purpose "to covertly introduce weaknesses into the encryption standards followed by hardware and software developers around the world." The New York Times writes that "the NSA had inserted a back door into a 2006 standard adopted by NIST [...] called the Dual EC DRBG standard."
19 Sept. 2013	RSA advises not to use Dual EC.

Recent History of Dual EC

5 Sept. 2013	NSA's "Project Bullrun" is revealed by documents from Edward Snowden with the purpose "to covertly introduce weaknesses into the encryption standards followed by hardware and software developers around the world." The New York Times writes that "the NSA had inserted a back door into a 2006 standard adopted by NIST [...] called the Dual EC DRBG standard."
19 Sept. 2013	RSA advises not to use Dual EC.
20 Dec. 2013	Reuters reports that NSA paid RSA \$10 million to use Dual EC as their default RNG.

Recent History of Dual EC

5 Sept. 2013	NSA's "Project Bullrun" is revealed by documents from Edward Snowden with the purpose `"to covertly introduce weaknesses into the encryption standards followed by hardware and software developers around the world."` The New York Times writes that `"the NSA had inserted a back door into a 2006 standard adopted by NIST [...] called the Dual EC DRBG standard."`
19 Sept. 2013	RSA advises not to use Dual EC.
20 Dec. 2013	Reuters reports that NSA paid RSA $10 million to use Dual EC as their default RNG.
21 Apr. 2014	NIST removes Dual EC from the standard.

Authors of Dual EC

Kelsey, in December 2013 slides:

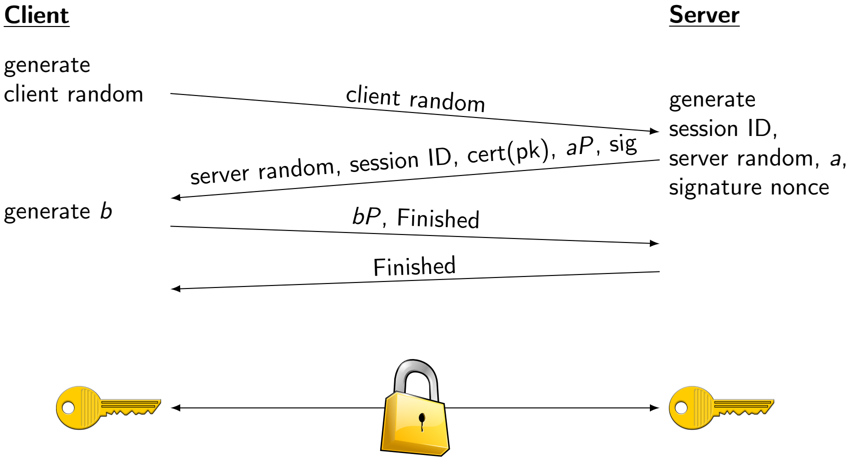
- Standardization effort by "NIST and NSA, with some participation from CSE".
- "Most of work on standards done by US federal employees (NIST and NSA, with some help from CSE)"
- The standard Dual EC parameters P and Q come "ultimately from designers of Dual EC DRBG at NSA".

Attack Target — TLS

Transport Layer Security (TLS)

- Used in the Internet for encryption of communication.
  Examples:
  - eMail transport,
  - online banking,
  - online shopping,
  - ...
- Standard covers a fast amount of protocols and optional features.
- Client and server agree on what parameters to use.
- Client and server agree on a random secret key.

TLS Handshake



Client
Server
generate client random
client random
generate session ID, server random, a, signature nonce
server random, session ID, cert(pk), aP, sig
generate b
bP, Finished
Finished

Attack Target — TLS

Common TLS implementations:

- RSA's BSAFE
  - RSA BSAFE Share for Java (BSAFE Java)
  - RSA BSAFE Share for C and C++ (BSAFE C)
- Microsoft's SChannel
- OpenSSL

All of these offer Dual EC.

Attack Target — TLS

Common TLS implementations:

- RSA's BSAFE
  - RSA BSAFE Share for Java (BSAFE Java)
  - RSA BSAFE Share for C and C++ (BSAFE C)
- Microsoft's SChannel
- OpenSSL

Remember: NSA paid RSA Security $10 million to use Dual EC as the default RNG!

All of these offer Dual EC.

Elliptic Curve Discrete Logarithm Problem

Arithmetic on Elliptic Curves

Operate on points $P = (x_P, y_P)$ on an elliptic curve:

- addition: $A + B = C$,
- scalar mul.: $k \cdot A = \underbrace{A + A + \cdots + A}_{k-\text{times}}$.

Elliptic Curve Discrete Logarithm Problem

Arithmetic on Elliptic Curves

Operate on points $P = (x_P, y_P)$ on an elliptic curve:

- addition: $A + B = C$,
- scalar mul.: $k \cdot A = \underbrace{A + A + \cdots + A}_{k-\text{times}}$.

Useful in Cryptography:

It is *easy* to compute $k \cdot A$, e.g.:

$$B = 243 \cdot A = A + 2A + 16A + 32A + 64A + 128A.$$

Cost: 5 additions and 7 doublings.

Elliptic Curve Discrete Logarithm Problem

Arithmetic on Elliptic Curves

Operate on points $P = (x_P, y_P)$ on an elliptic curve:

- addition: $A + B = C$,
- scalar mul.: $k \cdot A = \underbrace{A + A + \cdots + A}_{k-\text{times}}$.

Useful in Cryptography:

It is *easy* to compute $k \cdot A$, e.g.:

$$B = 243 \cdot A = A + 2A + 16A + 32A + 64A + 128A.$$

Cost: 5 additions and 7 doublings.

For given A and B, it is *hard* to find k such that $B = k \cdot A$!

Dual EC

Parameters

Here: elliptic curve over finite filed with NIST prime P-256.
(NIST SP800-90A also defines curves for P-384 and P-521.)

The elliptic curve is defined over $\mathbf{F}_p$ with $p = 2^{256} - 2^{224} + 2^{192} + 2^{96} - 1$.
The curve is given in short Weierstrass form

$$E : y^2 = x^3 - 3x + b, \text{ where}$$

$b = $ `0x5ac635d8aa3a93e7b3ebbd55769886bc651d06b0cc53b0f63bce3c3e27d2604b`.

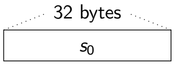
Dual EC defines two points, a base point P and a second point Q:

$P_x = $ `0x6b17d1f2e12c4247f8bce6e563a440f277037d812deb33a0f4a13945d898c296`,
$P_y = $ `0x4fe342e2fe1a7f9b8ee7eb4a7c0f9e162bce33576b315ececbb6406837bf51f5`;

$Q_x = $ `0xc97445f45cdef9f0d3e05e1e585fc297235b82b5be8ff3efca67c59852018192`,
$Q_y = $ `0xb28ef557ba31dfcbdd21ac46e2a91e3c304f44cb87058ada2cb815151e610046`.

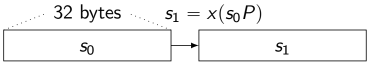
Dual EC — Basic Procedure

Points Q and P on an elliptic curve.



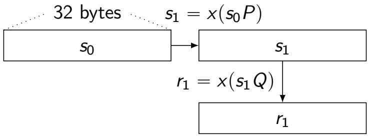
Dual EC — Basic Procedure

Points Q and P on an elliptic curve.



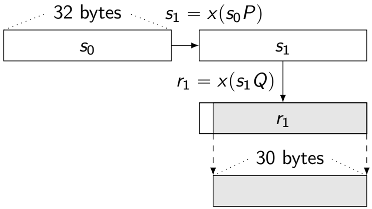
Dual EC — Basic Procedure

Points Q and P on an elliptic curve.



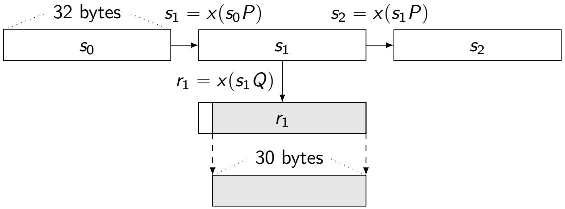
Dual EC — Basic Procedure

Points Q and P on an elliptic curve.



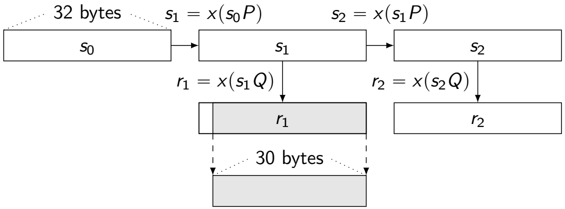
Dual EC — Basic Procedure

Points Q and P on an elliptic curve.



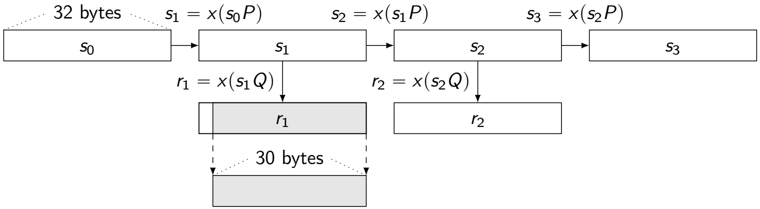
Dual EC — Basic Procedure

Points Q and P on an elliptic curve.



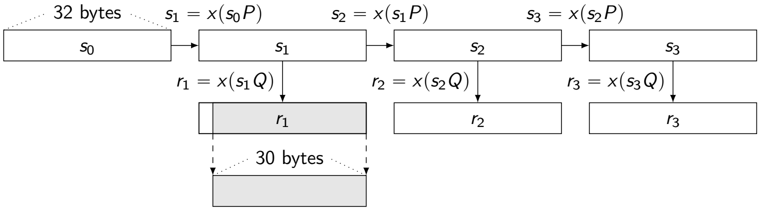
Dual EC — Basic Procedure

Points Q and P on an elliptic curve.



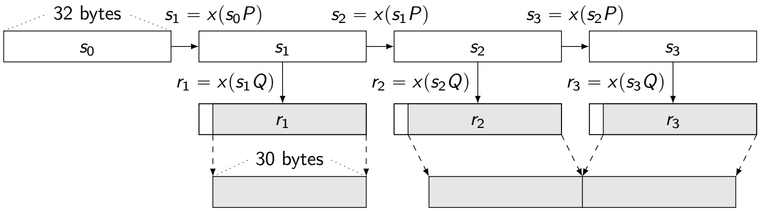
Dual EC — Basic Procedure

Points Q and P on an elliptic curve.



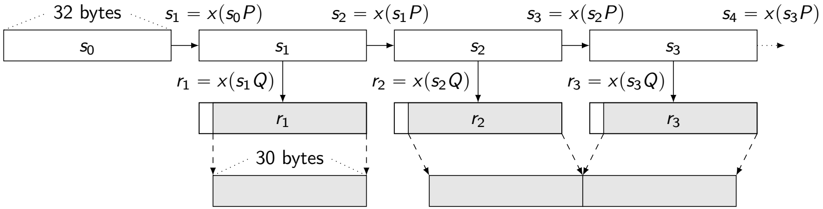
Dual EC — Basic Procedure

Points Q and P on an elliptic curve.



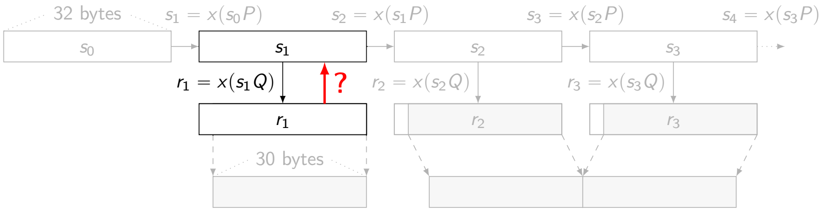
Dual EC — Basic Procedure

Points Q and P on an elliptic curve.



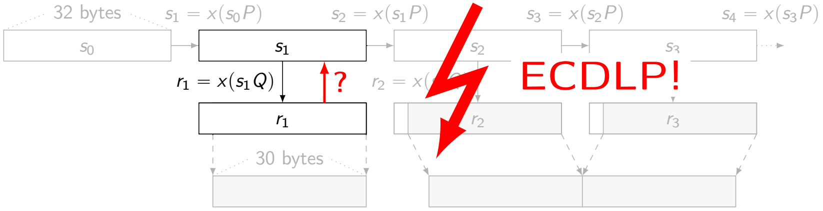
Dual EC — Basic Procedure

Points Q and P on an elliptic curve.



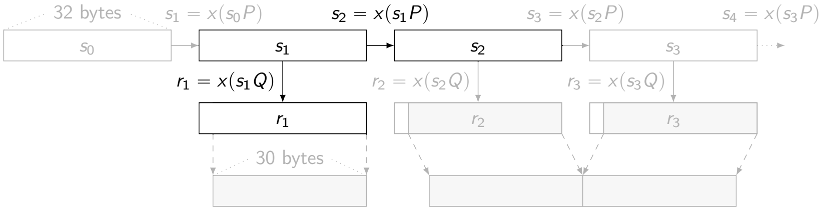
Dual EC — Basic Procedure

Points Q and P on an elliptic curve.



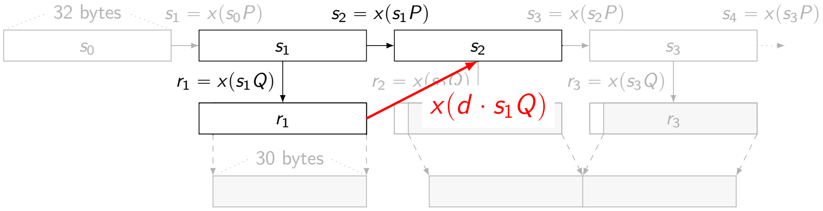
Shumow and Ferguson – the Basic Attack

Points Q and $P = dQ$ on an elliptic curve.



Shumow and Ferguson – the Basic Attack

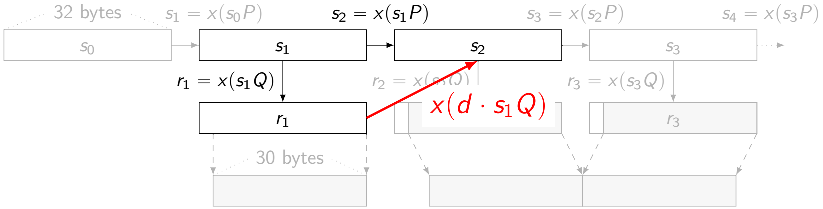
Points Q and $P = dQ$ on an elliptic curve.



Shumow and Ferguson – the Basic Attack

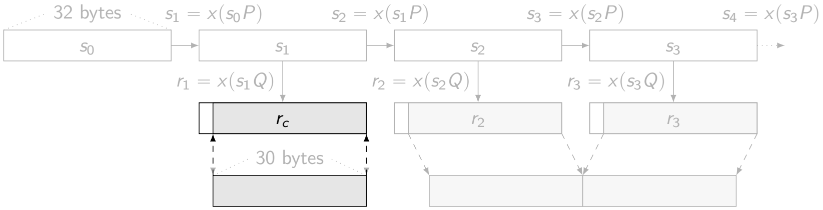
Points Q and $P = dQ$ on an elliptic curve.

$$s_2 = x(s_1 P) = x(s_1 \cdot dQ)$$



Shumow and Ferguson – the Basic Attack

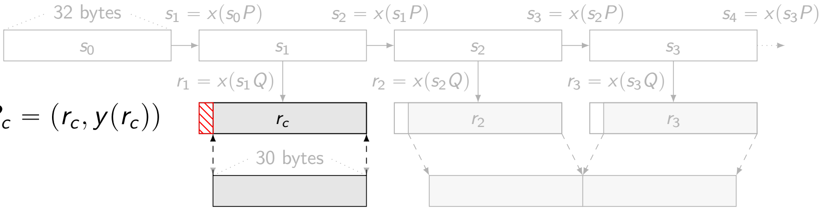
Points Q and $P = dQ$ on an elliptic curve.



Shumow and Ferguson – the Basic Attack

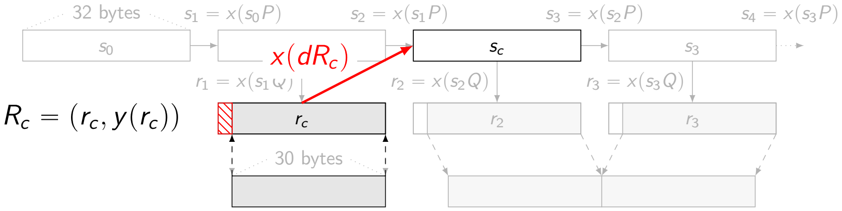
Points Q and $P = dQ$ on an elliptic curve.

$R_c = (r_c, y(r_c))$



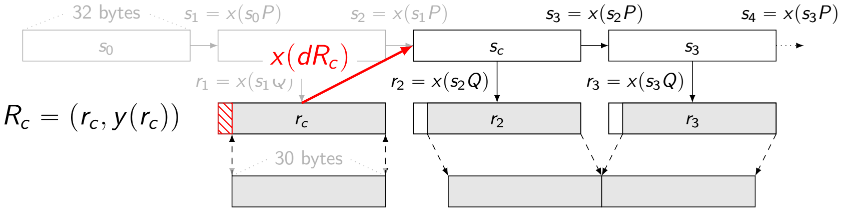
Shumow and Ferguson – the Basic Attack

Points Q and $P = dQ$ on an elliptic curve.

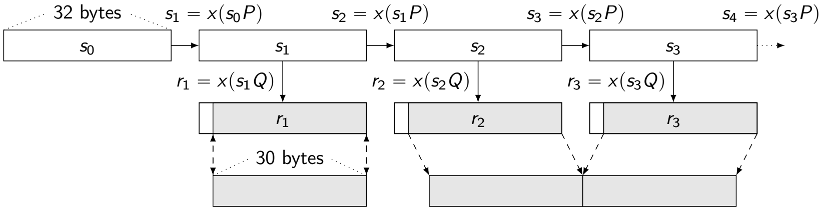


Shumow and Ferguson – the Basic Attack

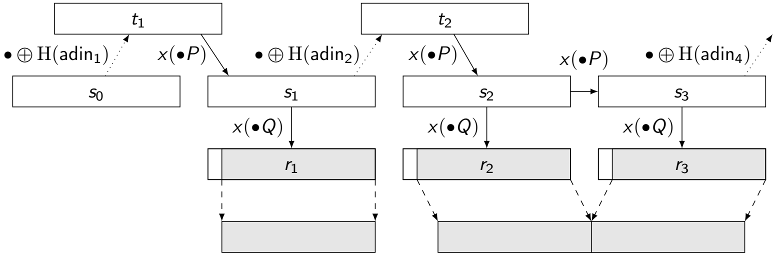
Points Q and $P = dQ$ on an elliptic curve.



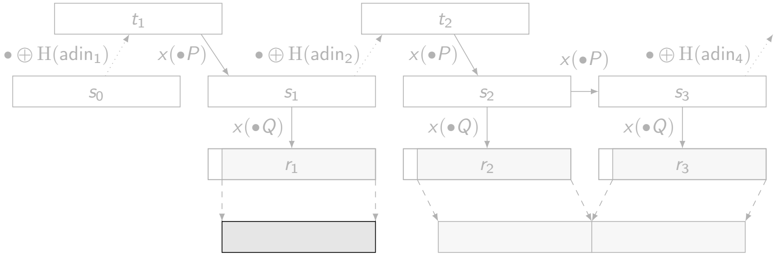
Shumow and Ferguson – the Basic Attack



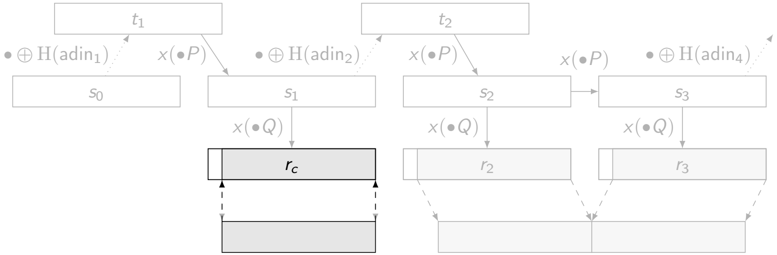
Dual EC — NISTSP800-90 in June 2006


t_1
t_2
$\bullet \oplus \mathrm{H}(\mathrm{adin}_1)$
$x(\bullet P)$
$\bullet \oplus \mathrm{H}(\mathrm{adin}_2)$
$x(\bullet P)$
$x(\bullet P)$
$\bullet \oplus \mathrm{H}(\mathrm{adin}_4)$
s_0
s_1
s_2
s_3
$x(\bullet Q)$
$x(\bullet Q)$
$x(\bullet Q)$
r_1
r_2
r_3

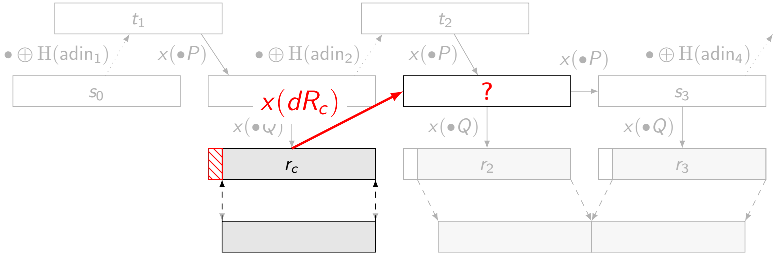
Dual EC — NIST SP800-90 in June 2006


t_1
t_2
$\bullet \oplus \mathrm{H}(\mathrm{adin}_1)$
$x(\bullet P)$
$\bullet \oplus \mathrm{H}(\mathrm{adin}_2)$
$x(\bullet P)$
$x(\bullet P)$
$\bullet \oplus \mathrm{H}(\mathrm{adin}_4)$
s_0
s_1
s_2
s_3
$x(\bullet Q)$
$x(\bullet Q)$
$x(\bullet Q)$
r_1
r_2
r_3

Dual EC — NIST SP800-90 in June 2006

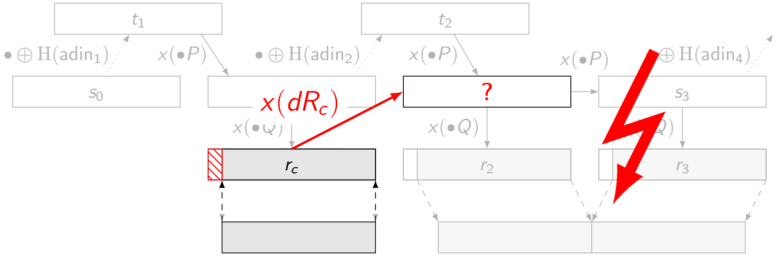

t_1
t_2
$\bullet \oplus \mathrm{H}(\mathrm{adin}_1)$
$x(\bullet P)$
$\bullet \oplus \mathrm{H}(\mathrm{adin}_2)$
$x(\bullet P)$
$x(\bullet P)$
$\bullet \oplus \mathrm{H}(\mathrm{adin}_4)$
s_0
s_1
s_2
s_3
$x(\bullet Q)$
$x(\bullet Q)$
$x(\bullet Q)$
r_c
r_2
r_3

Dual EC — NIST SP800-90 in June 2006



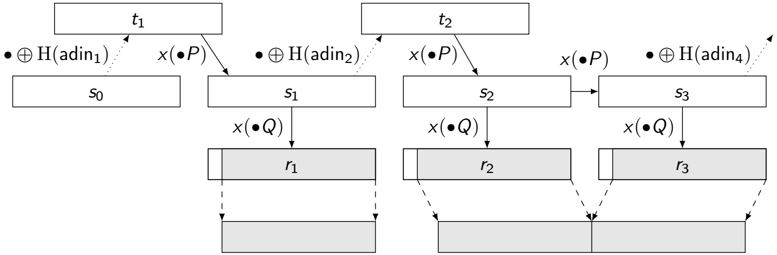
t_1
t_2
$\bullet \oplus \mathrm{H}(\mathrm{adin}_1)$
$x(\bullet P)$
$\bullet \oplus \mathrm{H}(\mathrm{adin}_2)$
$x(\bullet P)$
$x(\bullet P)$
$\bullet \oplus \mathrm{H}(\mathrm{adin}_4)$
s_0
$x(dR_c)$
?
s_3
$x(\bullet Q)$
$x(\bullet Q)$
$x(\bullet Q)$
r_c
r_2
r_3

Dual EC — NIST SP800-90 in June 2006

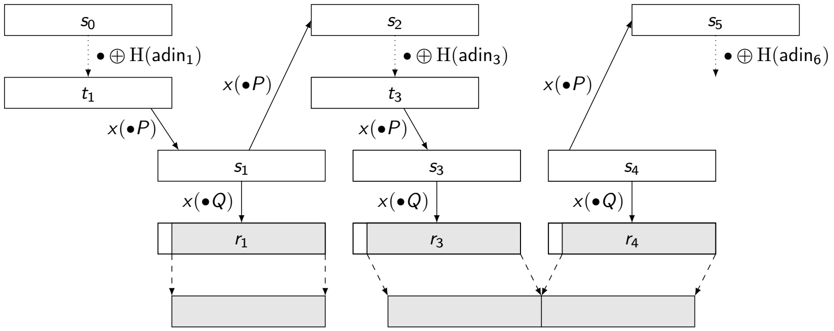


t_1
t_2
$\bullet \oplus \mathrm{H}(\mathrm{adin}_1)$
$x(\bullet P)$
$\bullet \oplus \mathrm{H}(\mathrm{adin}_2)$
$x(\bullet P)$
$x(\bullet P)$
$\oplus \mathrm{H}(\mathrm{adin}_4)$
s_0
$x(dR_c)$
?
s_3
$x(\bullet Q)$
$x(\bullet Q)$
$Q)$
r_c
r_2
r_3

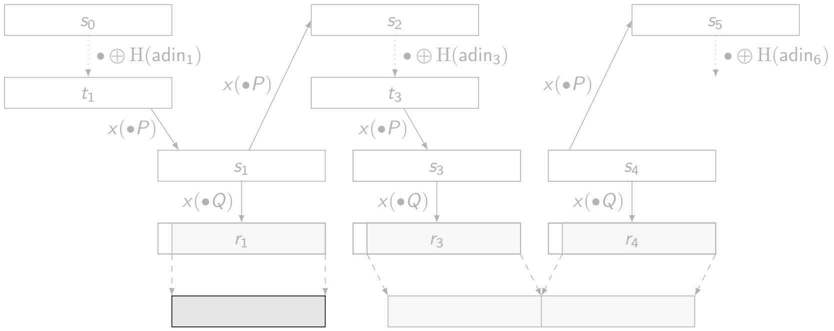
Dual EC — NIST SP800-90 in June 2006


t_1
t_2
$\bullet \oplus \mathrm{H}(\mathrm{adin}_1)$
$x(\bullet P)$
$\bullet \oplus \mathrm{H}(\mathrm{adin}_2)$
$x(\bullet P)$
$x(\bullet P)$
$\bullet \oplus \mathrm{H}(\mathrm{adin}_4)$
s_0
s_1
s_2
s_3
$x(\bullet Q)$
$x(\bullet Q)$
$x(\bullet Q)$
r_1
r_2
r_3

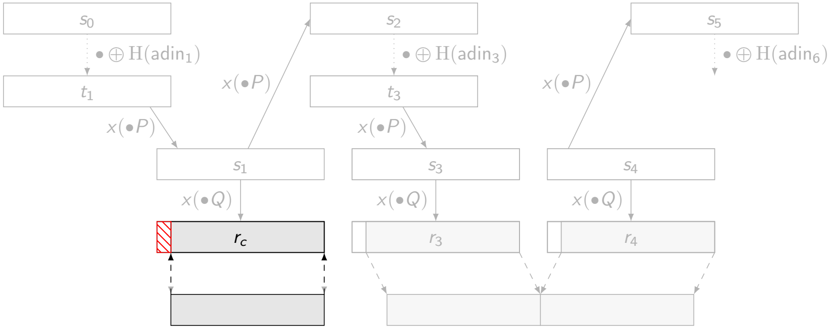
Dual EC — NIST SP800-90 in March 2007



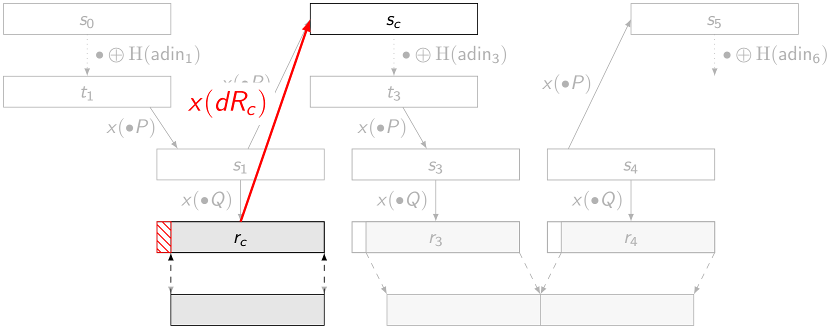
Dual EC — NIST SP800-90 in March 2007



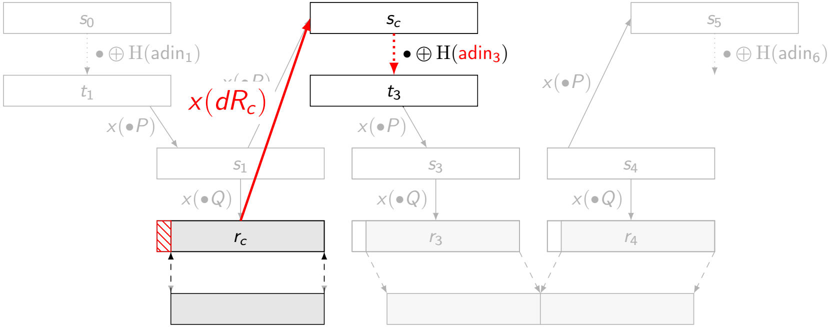
Dual EC — NIST SP800-90 in March 2007



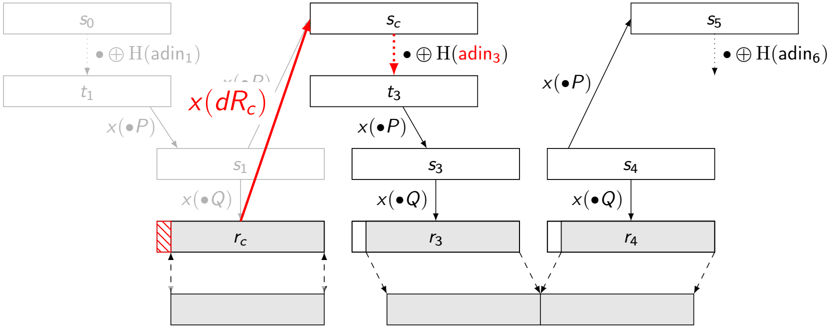
Dual EC — NIST SP800-90 in March 2007



Dual EC — NIST SP800-90 in March 2007



Dual EC — NIST SP800-90 in March 2007



Attack

Attack targets in our analysis:

In the real world, the attack is more complicated. We attacked:

- RSA's BSAFE
  - RSA BSAFE Share for Java (BSAFE Java)
  - RSA BSAFE Share for C and C++ (BSAFE C)
- Microsoft's SChannel
- OpenSSL

We replaced the points P and Q with known $P = dQ$; this required some reverse engineering of BSAFE and SChannel.

Attack

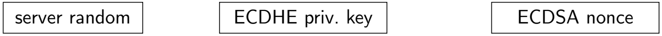
Attack targets in our analysis:

In the real world, the attack is more complicated. We attacked:

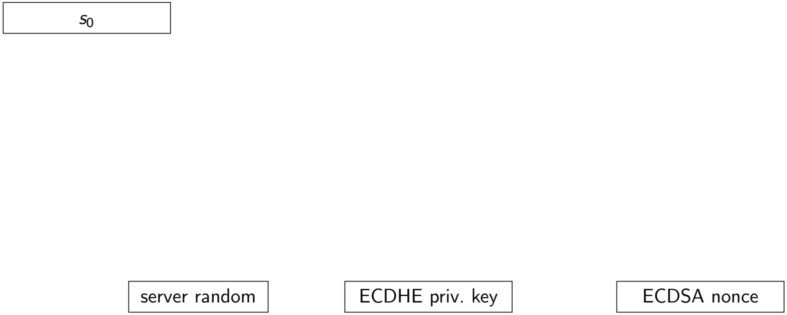
- RSA's BSAFE
  - RSA BSAFE Share for Java (BSAFE Java)
  - RSA BSAFE Share for C and C++ (BSAFE C)
- Microsoft's SChannel
- OpenSSL-fixed

We replaced the points P and Q with known $P = dQ$; this required some reverse engineering of BSAFE and SChannel.

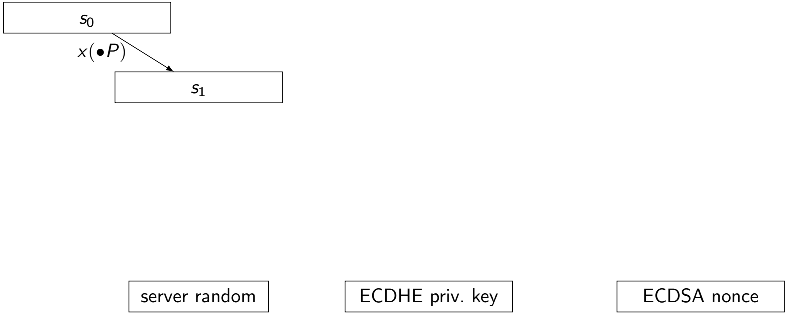
Attack — Example: BSAFE-Java



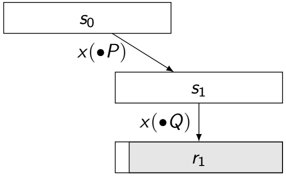
Attack — Example: BSAFE-Java

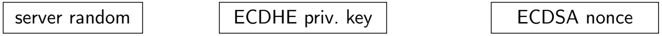


Attack — Example: BSAFE-Java



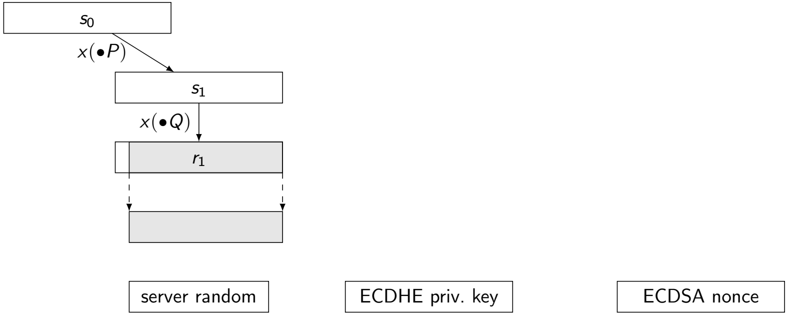
Attack — Example: BSAFE-Java



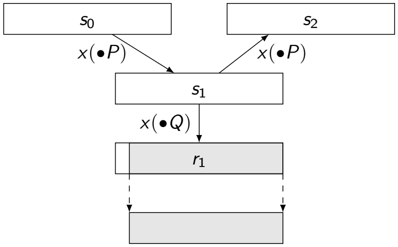


Technische Universiteit Eindhoven University of Technology

Attack — Example: BSAFE-Java



Attack — Example: BSAFE-Java



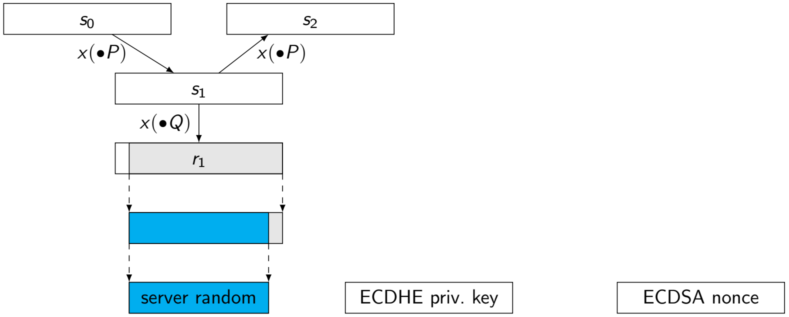
server random

ECDHE priv. key

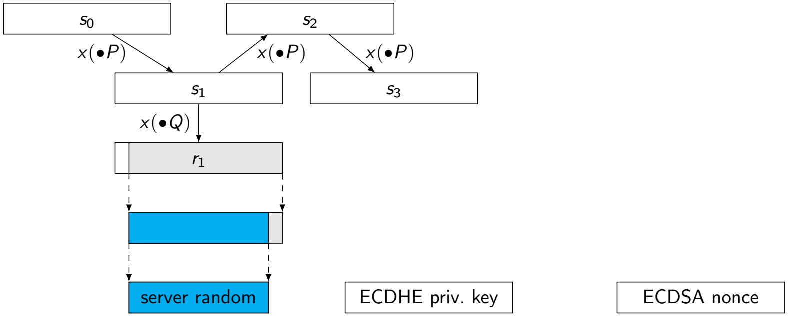
ECDSA nonce

Technische Universiteit
Eindhoven
University of Technology

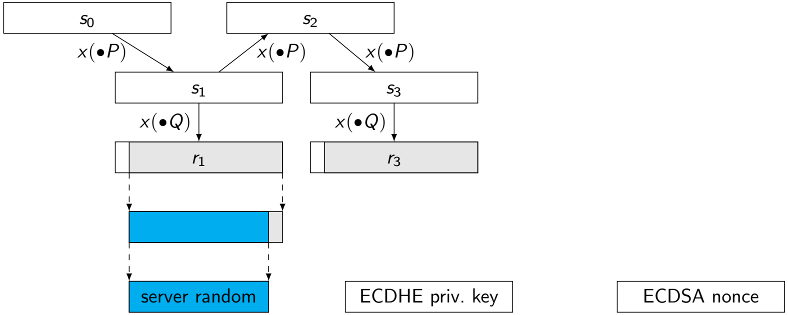
Attack — Example: BSAFE-Java



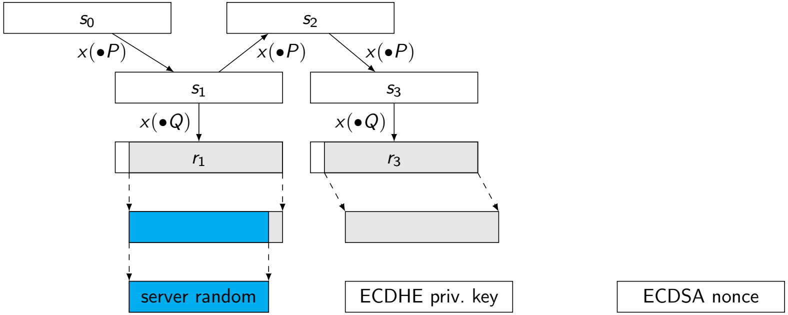
Attack — Example: BSAFE-Java



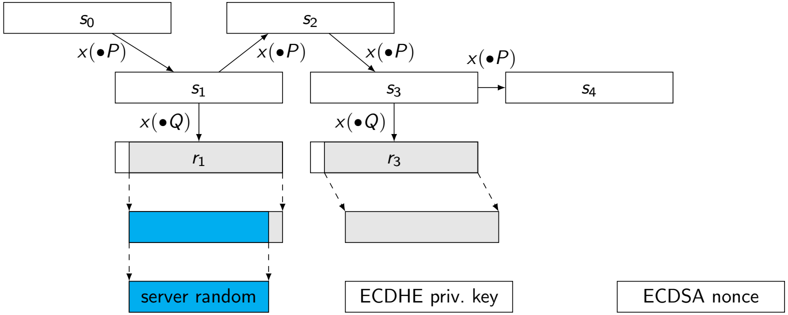
Attack — Example: BSAFE-Java



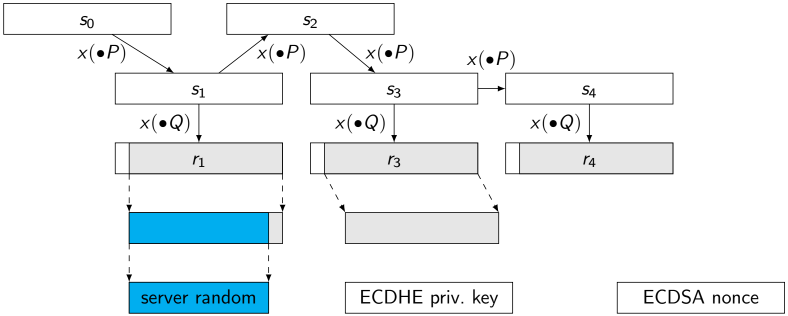
Attack — Example: BSAFE-Java



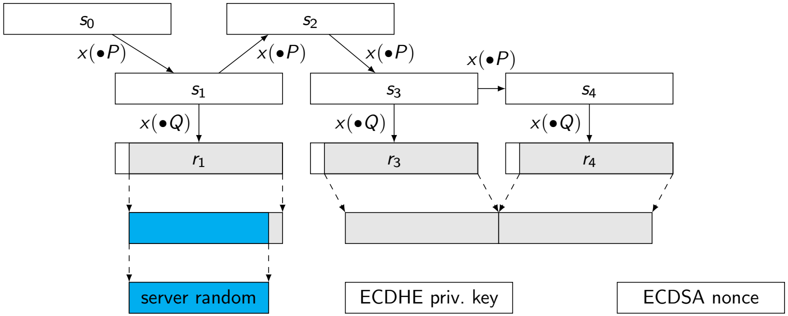
Attack — Example: BSAFE-Java



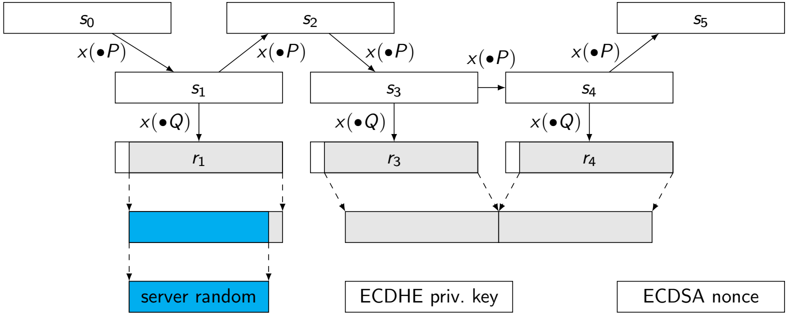
Attack — Example: BSAFE-Java



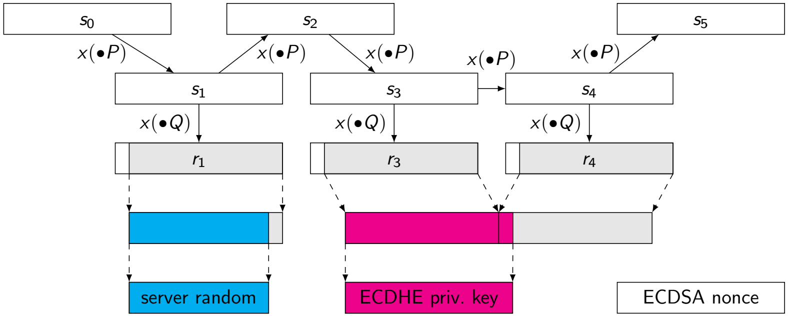
Attack — Example: BSAFE-Java



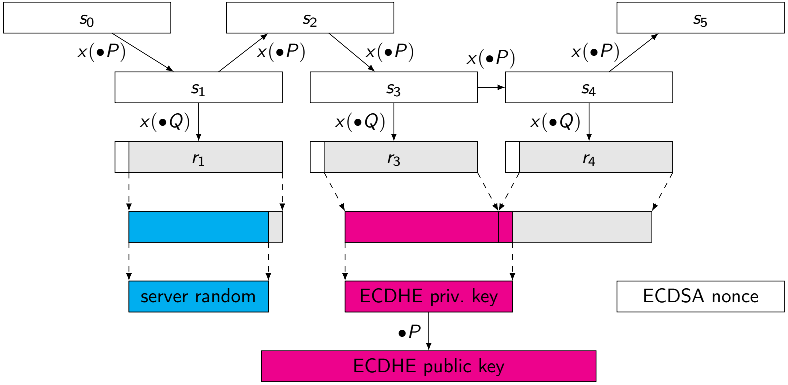
Attack — Example: BSAFE-Java



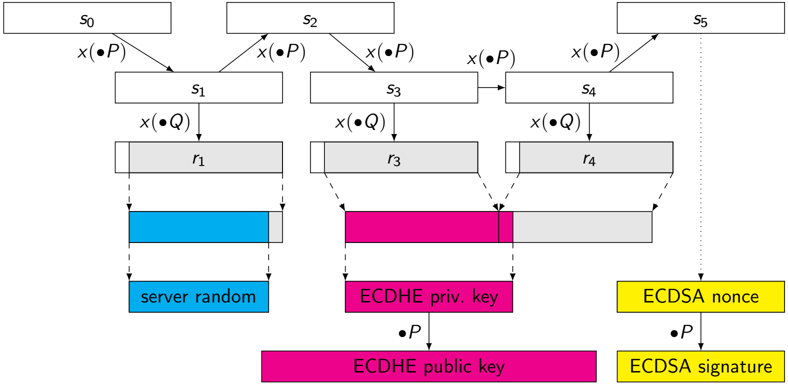
Attack — Example: BSAFE-Java



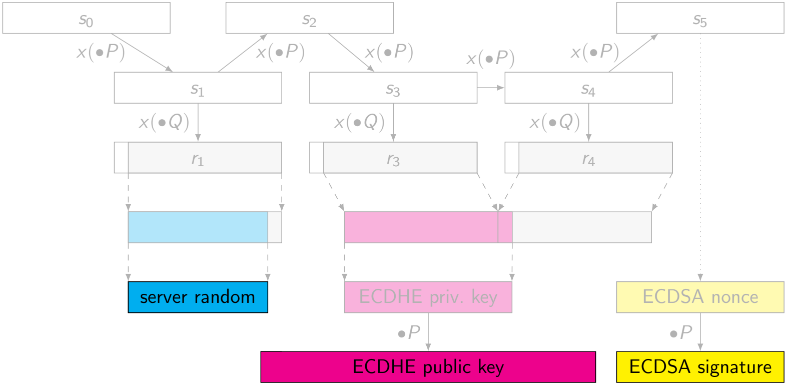
Attack — Example: BSAFE-Java



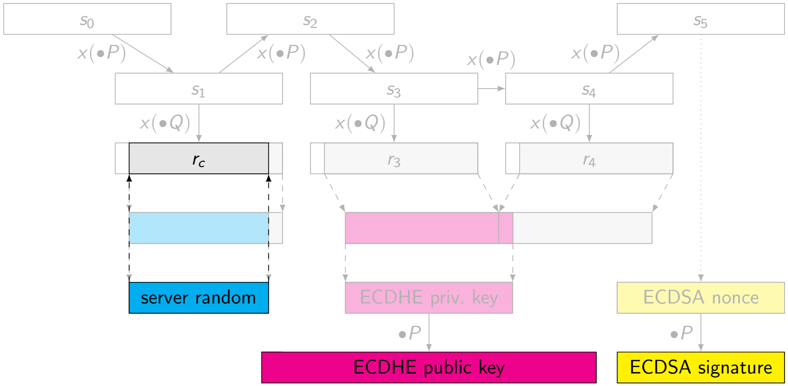
Attack — Example: BSAFE-Java



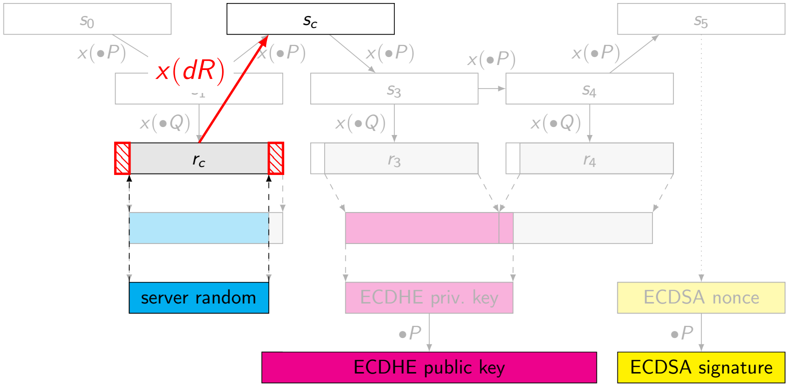
Attack — Example: BSAFE-Java



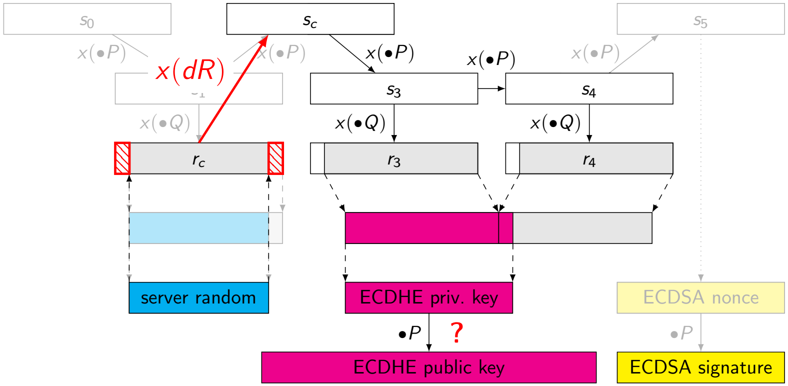
Attack — Example: BSAFE-Java



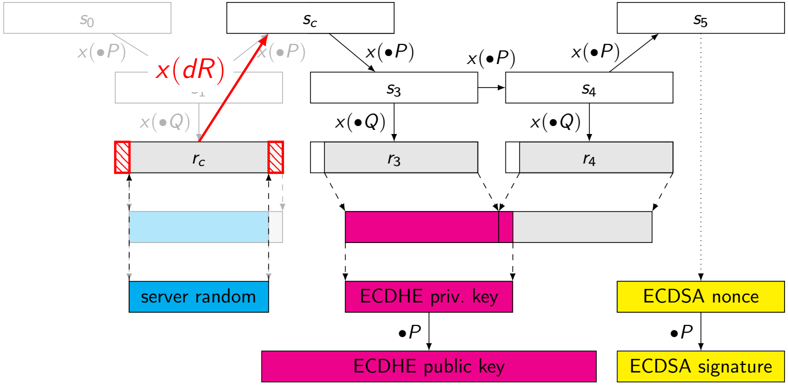
Attack — Example: BSAFE-Java



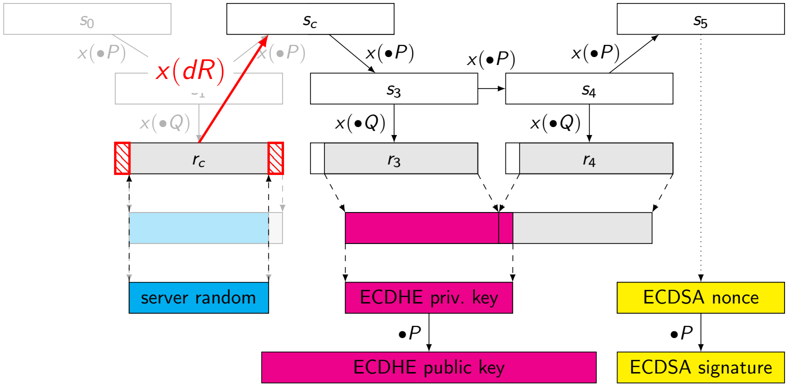
Attack — Example: BSAFE-Java



Attack — Example: BSAFE-Java

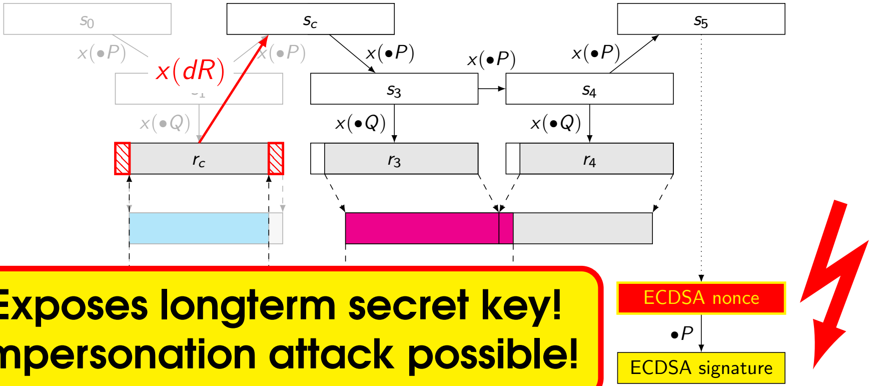


Attack — Example: BSAFE-Java



average cost: $2^{31}(C_v + 5C_f)$

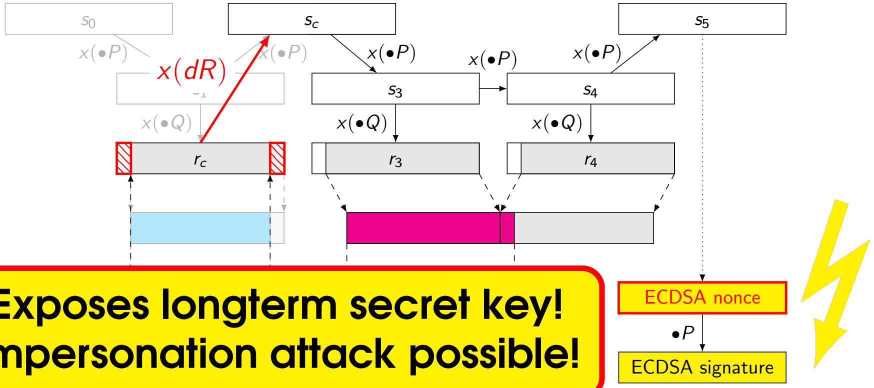
Attack — Example: BSAFE-Java



Exposes longterm secret key!
Impersonation attack possible!

average cost: $2^{31}(C_v + 5C_f)$

Attack — Example: BSAFE-Java



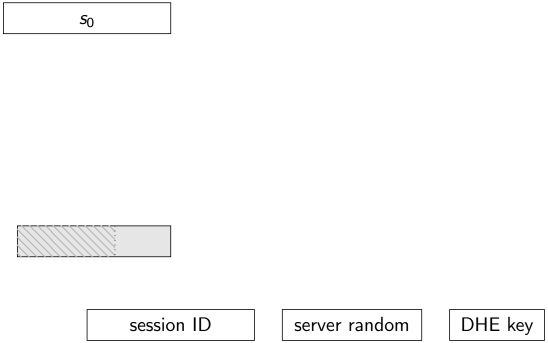
Exposes longterm secret key!
Impersonation attack possible!

average cost: $2^{31}(C_v + 5C_f)$

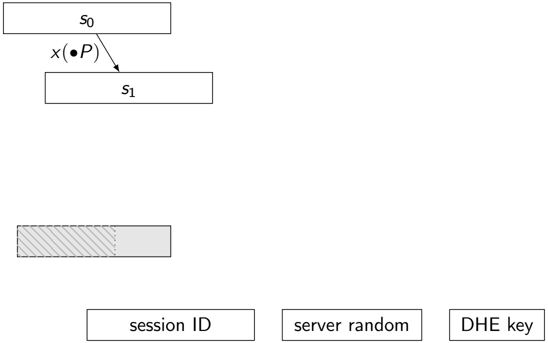
Attack — BSAFE-C

session ID | server random | DHE key

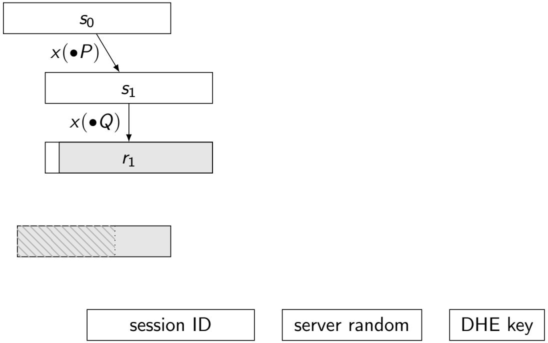
Attack — BSAFE-C



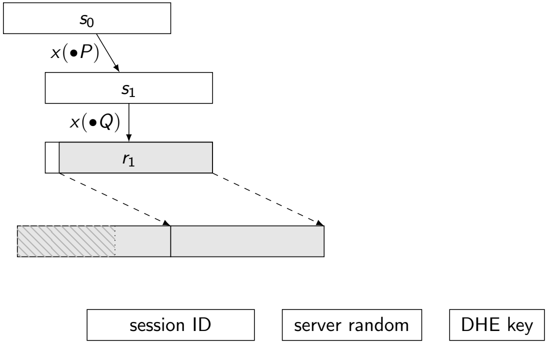
Attack — BSAFE-C



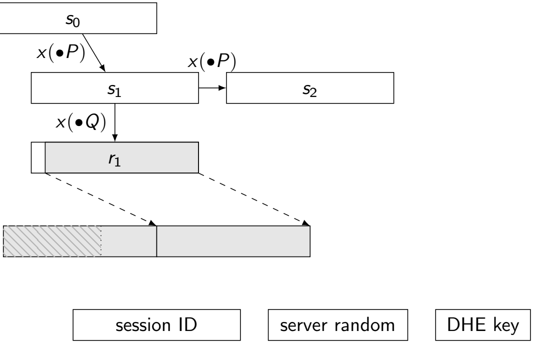
Attack — BSAFE-C



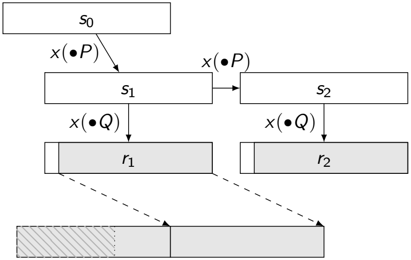
Attack — BSAFE-C



Attack — BSAFE-C

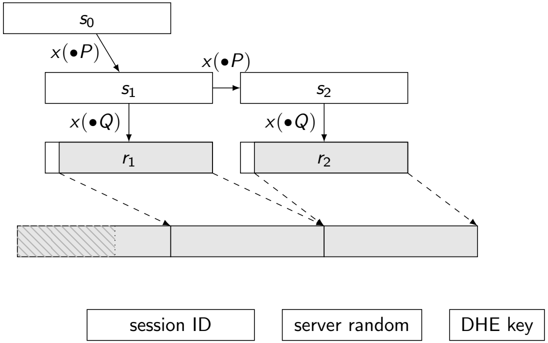


Attack — BSAFE-C

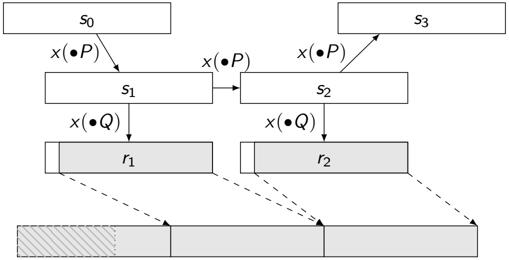


session ID | server random | DHE key

Attack — BSAFE-C



Attack — BSAFE-C

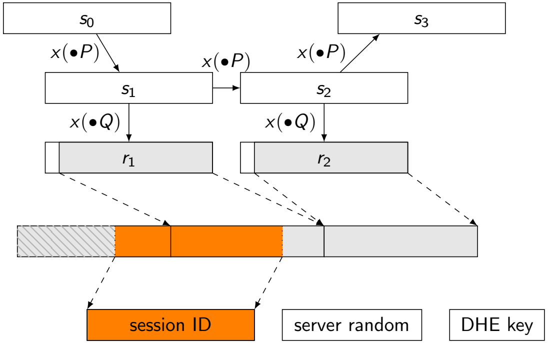


session ID

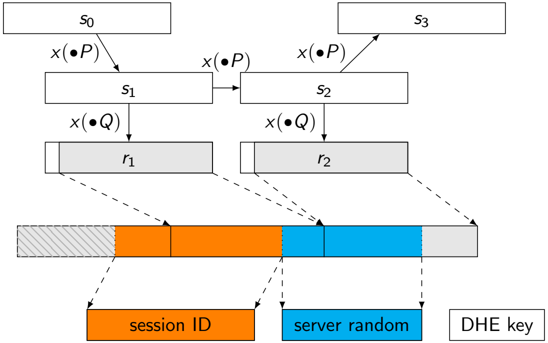
server random

DHE key

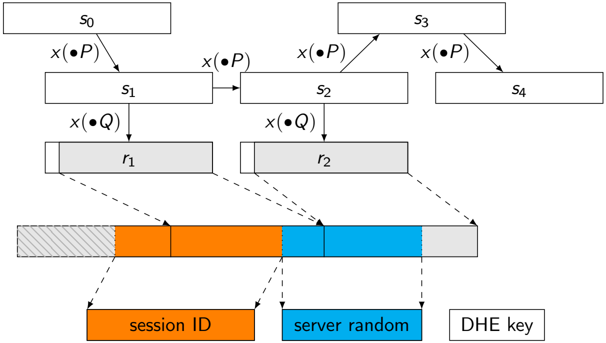
Attack — BSAFE-C



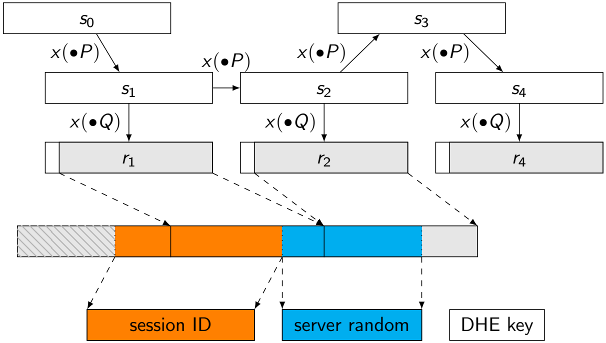
Attack — BSAFE-C



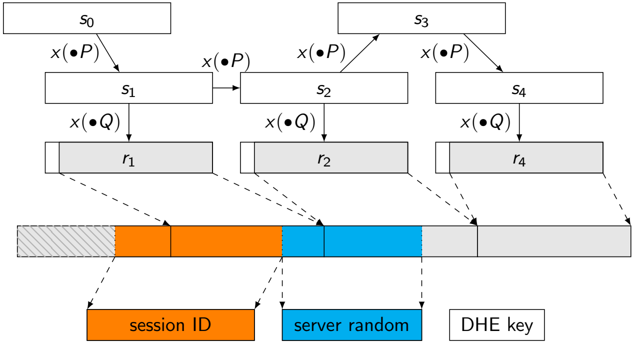
Attack — BSAFE-C



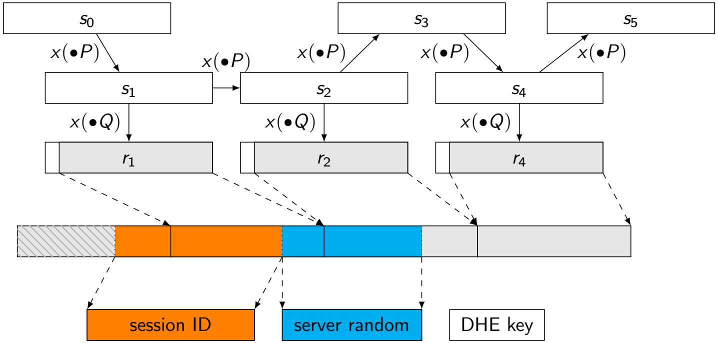
Attack — BSAFE-C



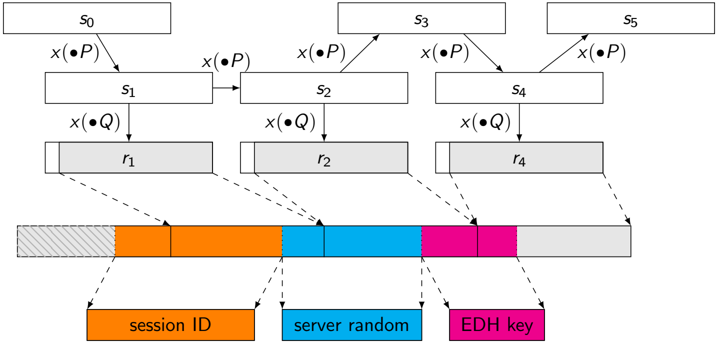
Attack — BSAFE-C



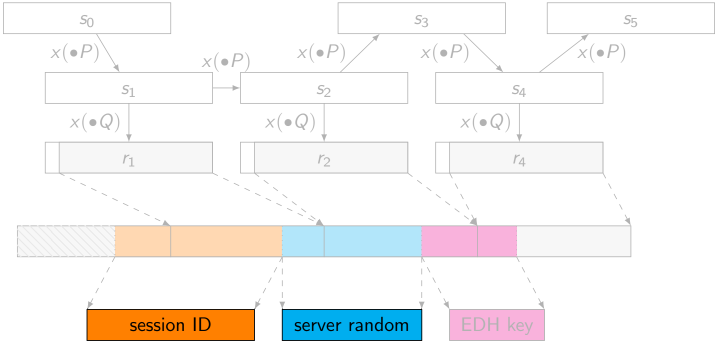
Attack — BSAFE-C



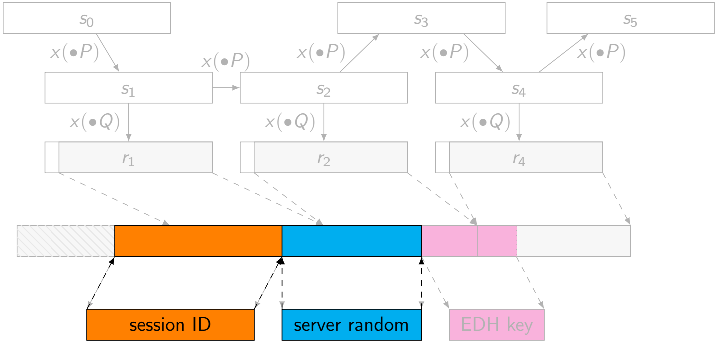
Attack — BSAFE-C



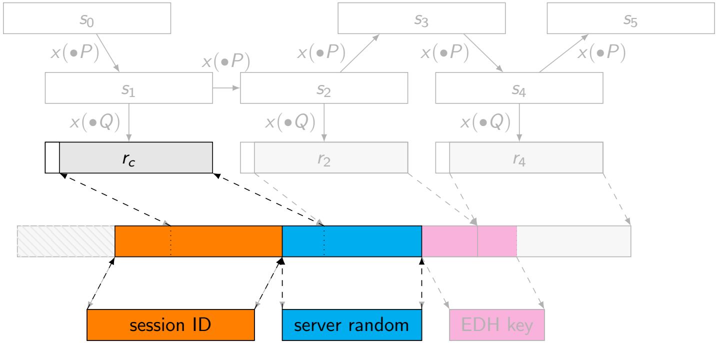
Attack — BSAFE-C



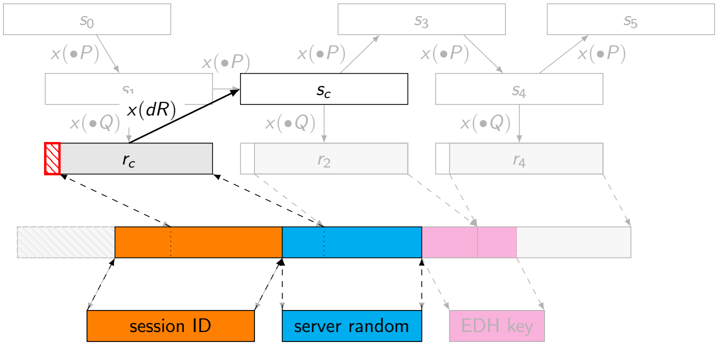
Attack — BSAFE-C



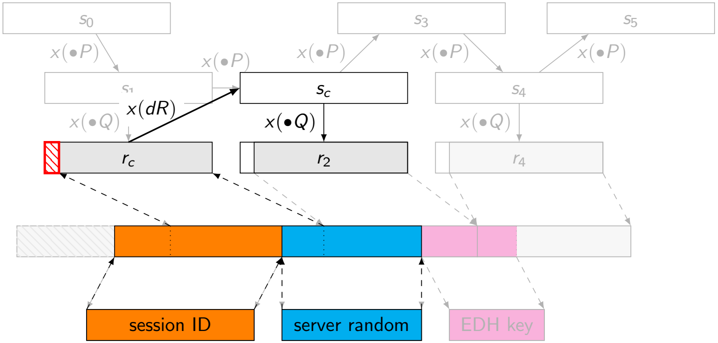
Attack — BSAFE-C



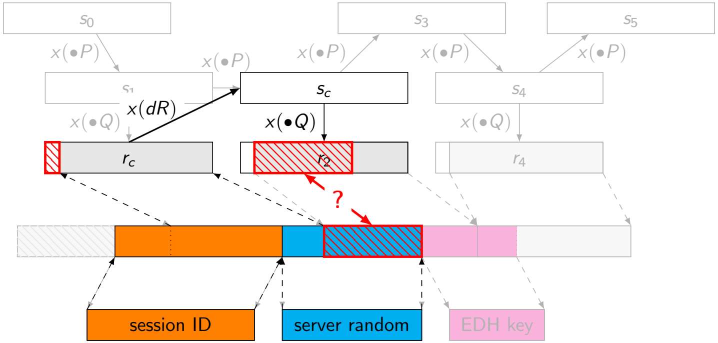
Attack — BSAFE-C



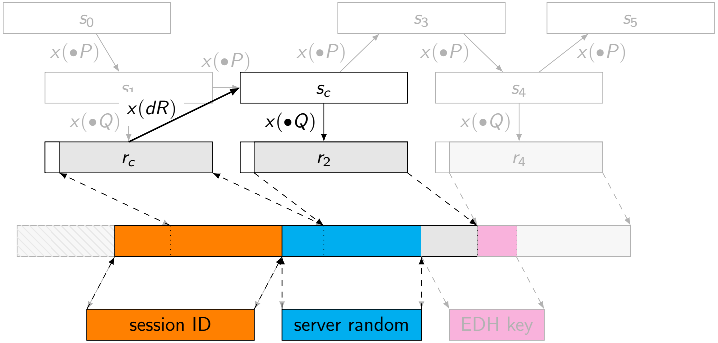
Attack — BSAFE-C



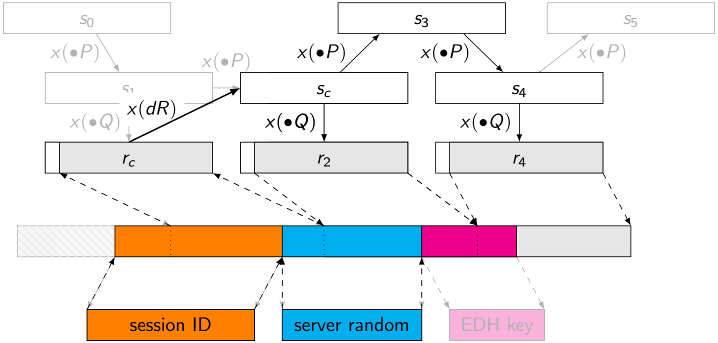
Attack — BSAFE-C



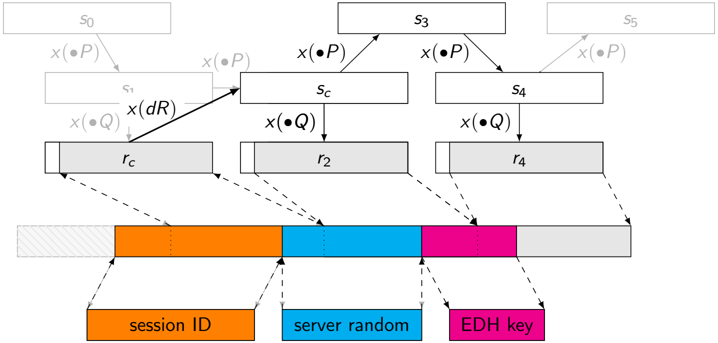
Attack — BSAFE-C



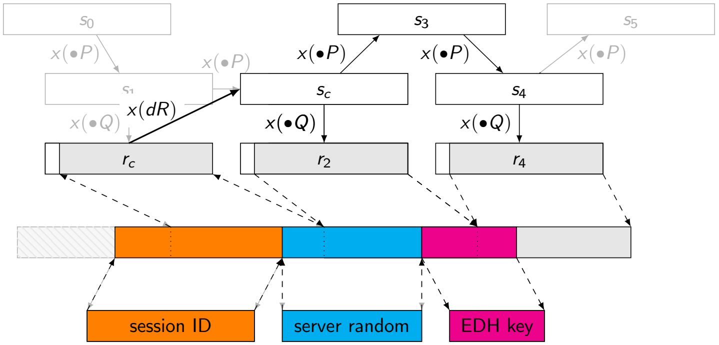
Attack — BSAFE-C



Attack — BSAFE-C

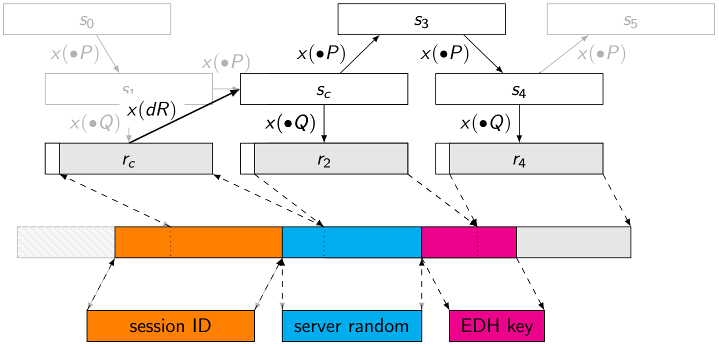


Attack — BSAFE-C



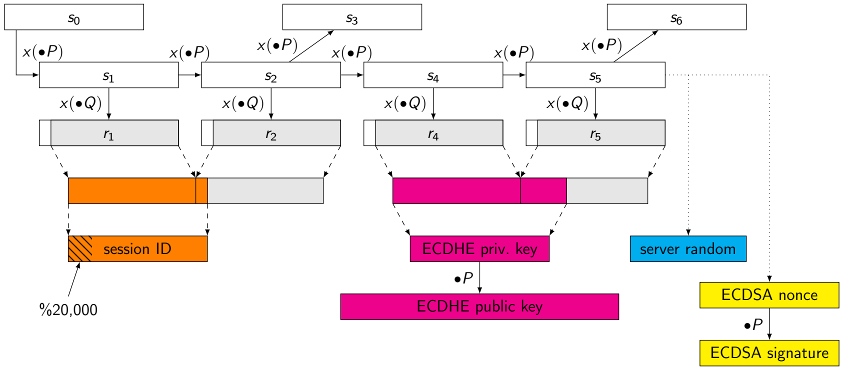
average cost: $2^{15}(C_v + C_f)$

Attack — BSAFE-C



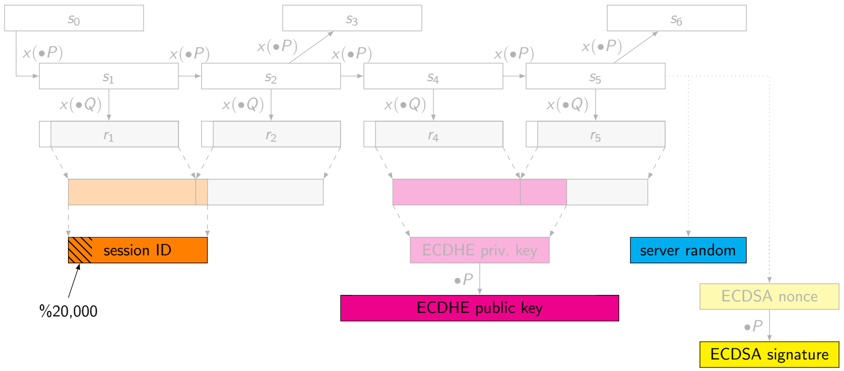
average cost: $30 \cdot 2^{15}(C_v + C_f)$

Attack — SChannel

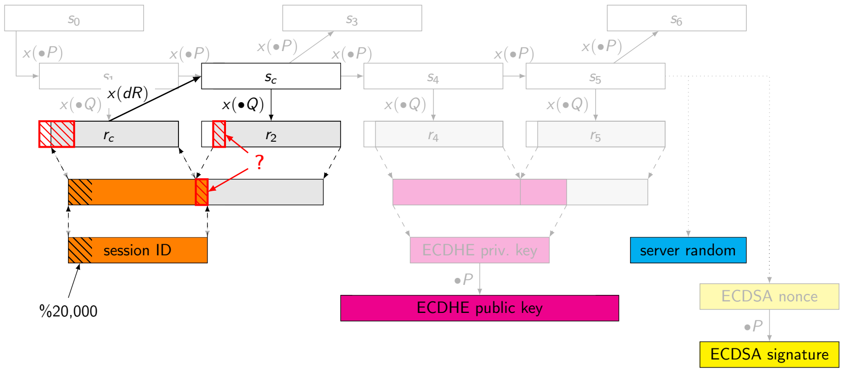


Technische Universiteit Eindhoven University of Technology

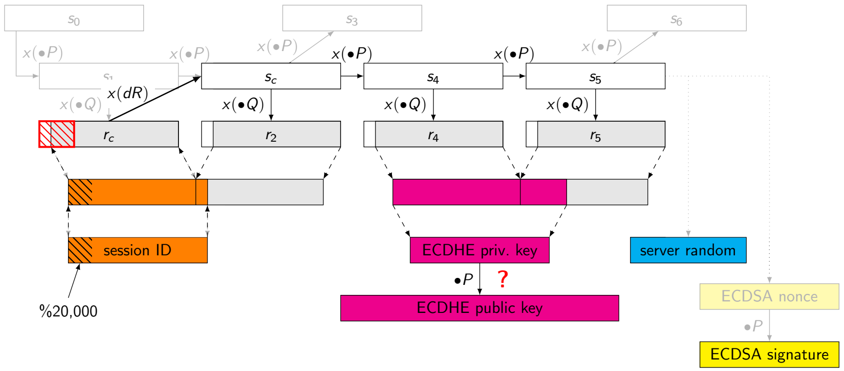
Attack — SChannel



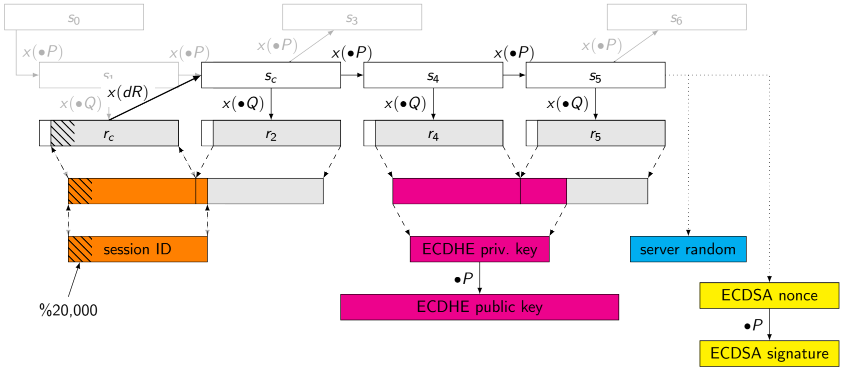
Attack — SChannel



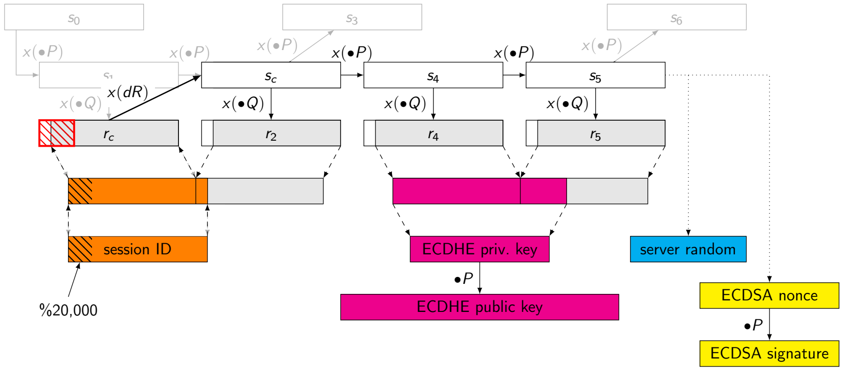
Attack — SChannel



Attack — SChannel

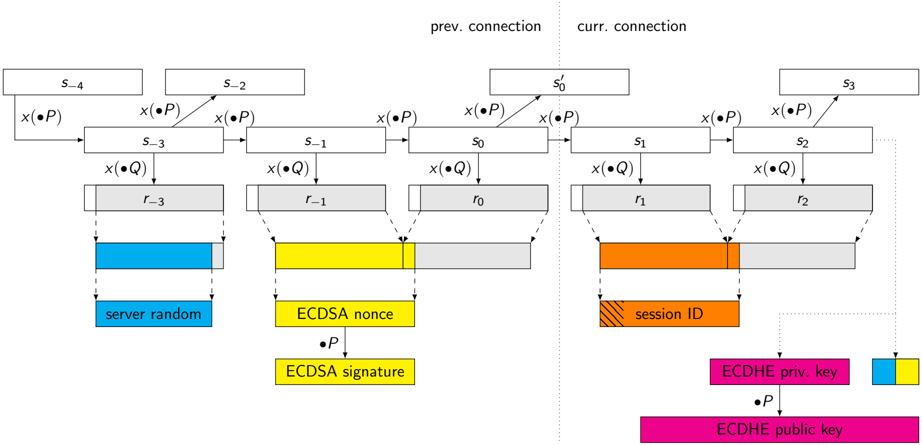


Attack — SChannel

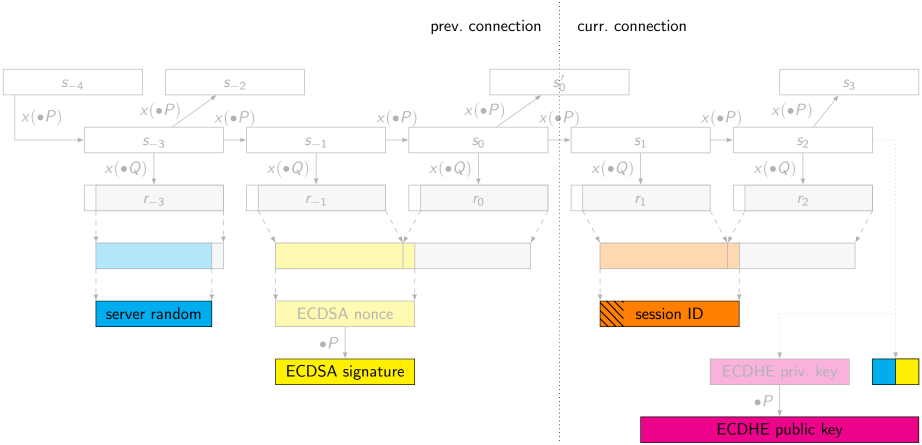


average cost: $2^{33}(C_v + C_f) + 2^{17}(5C_f)$

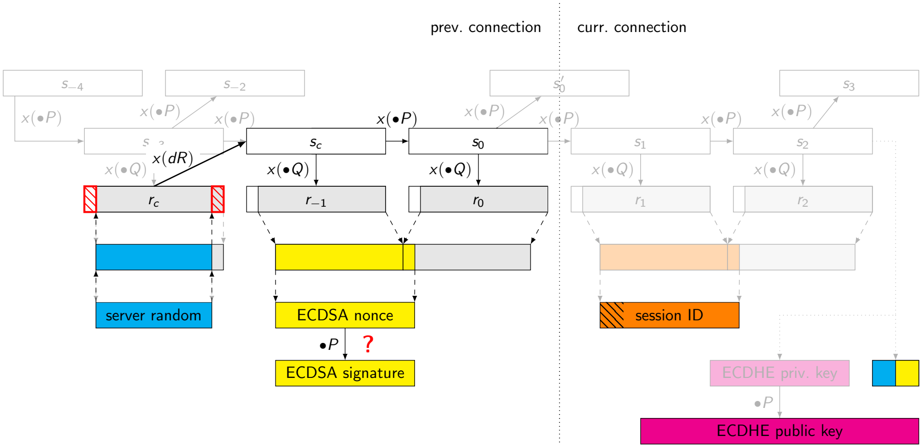
Attack — SChannel (cont.)



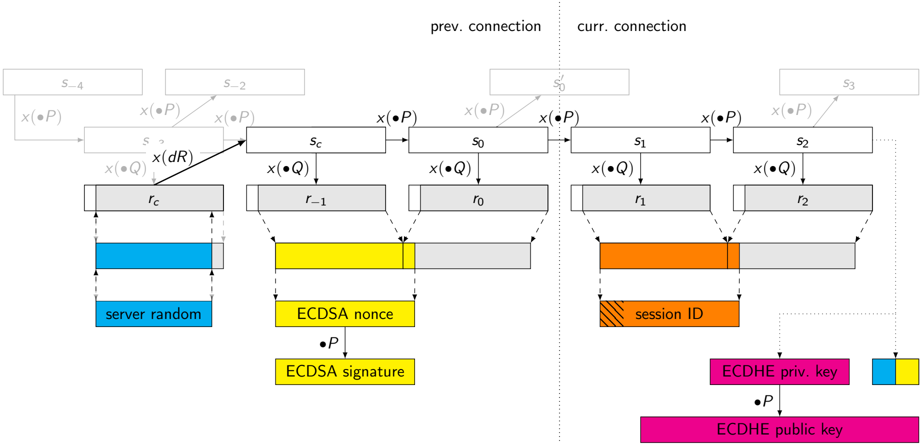
Attack — SChannel (cont.)



Attack — SChannel (cont.)

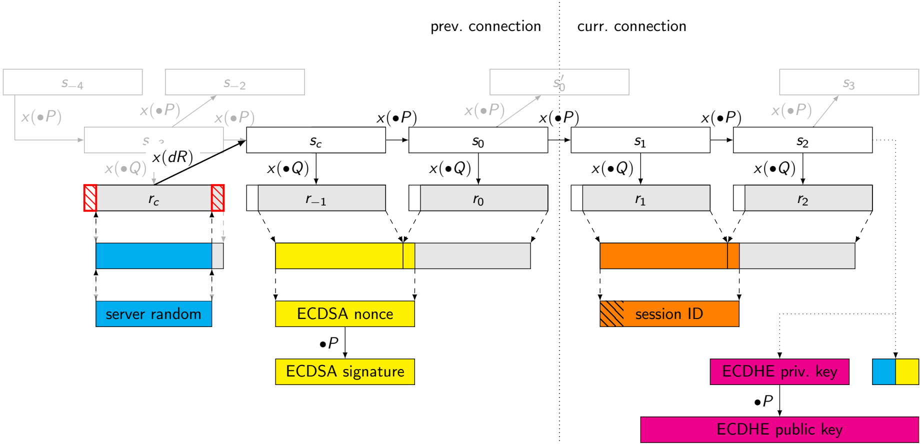


Attack — SChannel (cont.)



Technische Universiteit **Eindhoven** University of Technology

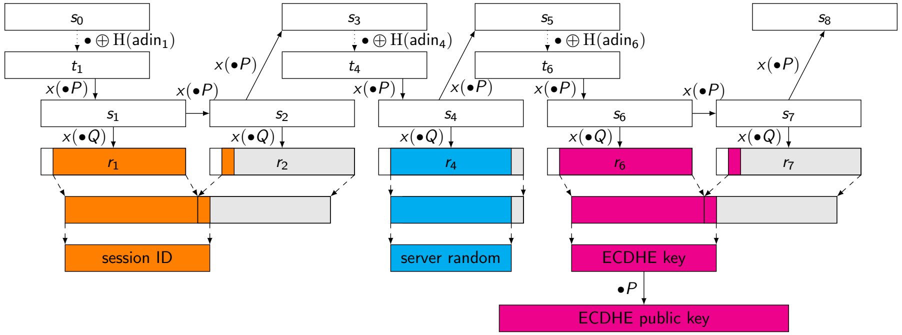
Attack — SChannel (cont.)



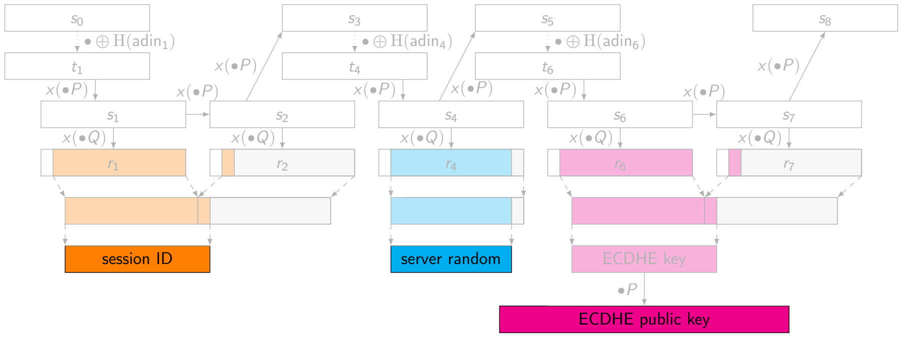
average cost: $2^{31}(C_v + 4C_f)$

Technische Universiteit Eindhoven University of Technology

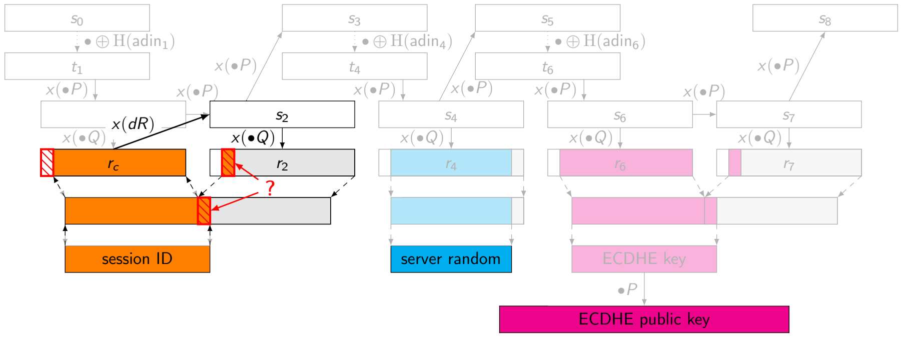
Attack — OpenSSL-fixed



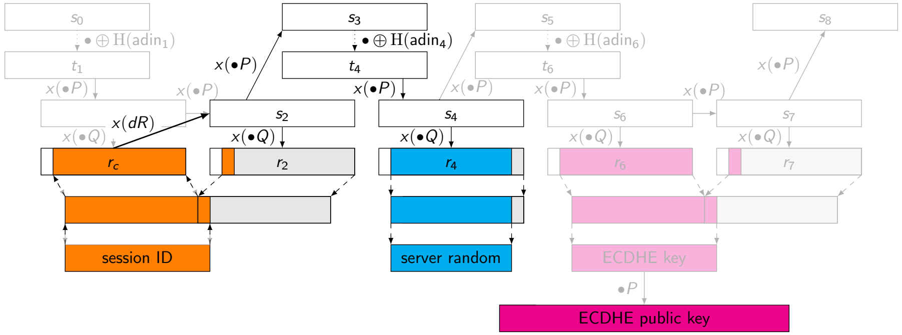
Attack — OpenSSL-fixed



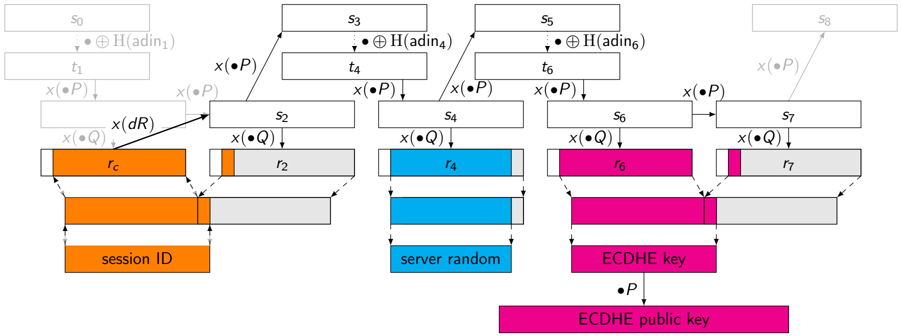
Attack — OpenSSL-fixed



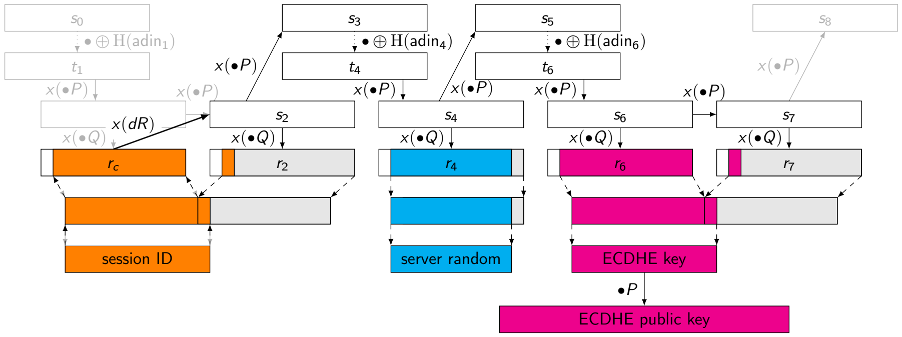
Attack — OpenSSL-fixed



Attack — OpenSSL-fixed



Attack — OpenSSL-fixed



$$\text{average cost: } 2^{15}(C_v + C_f) + 2^{20+k+l}(2C_f) + 2^{13}(5C_f)$$

TU/e Technische Universiteit Eindhoven University of Technology

Attack — Attack Timings

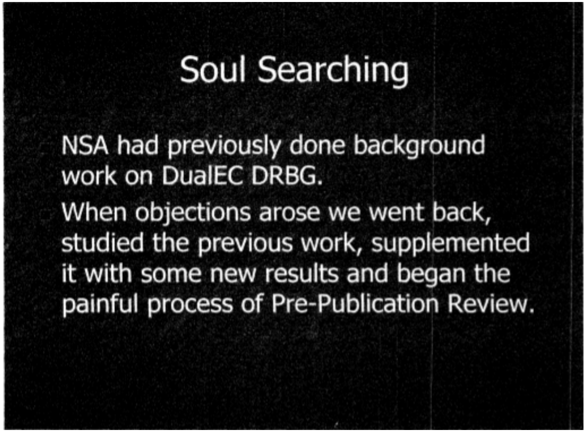
	Intel Xeon CPU		16 $\times$ AMD CPU
Attack	Avg. Time (min)	# for 1s	Tot. Time (min)
BSAFE-C v1.1	0.26	16	0.04
BSAFE-Java v1.1	641	38,500	63.96
SChannel I	619	37,100	62.97
SChannel II	1,760	106,000	182.64
OpenSSL-fixed I	0.04	3	0.02
OpenSSL-fixed II	707	44,200	83.32
OpenSSL-fixed III	$2^k \cdot 707$	$2^k \cdot 44{,}200$	$2^k \cdot 83.32$

Attack — Attack Timings

Attack	Intel Xeon CPU Avg. Time (min)	Intel Xeon CPU # for 1s	16 × AMD CPU Tot. Time (min)
BSAFE-C v1.1	0.26	16	0.04
BSAFE-Java v1.1	641	38,500	63.96
SChannel I	619	37,100	62.97
SChannel II	1,760	106,000	182.64
OpenSSL-fixed I	0.04	3	0.02
OpenSSL-fixed II	707	44,200	83.32
OpenSSL-fixed III	$2^k \cdot 707$	$2^k \cdot 44{,}200$	$2^k \cdot 83.32$

NIST FOIA

Two FOIA requests by Andrew Crocker and Nate Cardozo of EFF and Matthew Stoller and Rep. Alan Grayson. Files hosted by Matt Green at `https://github.com/matthewdgreen/nistfoia`.

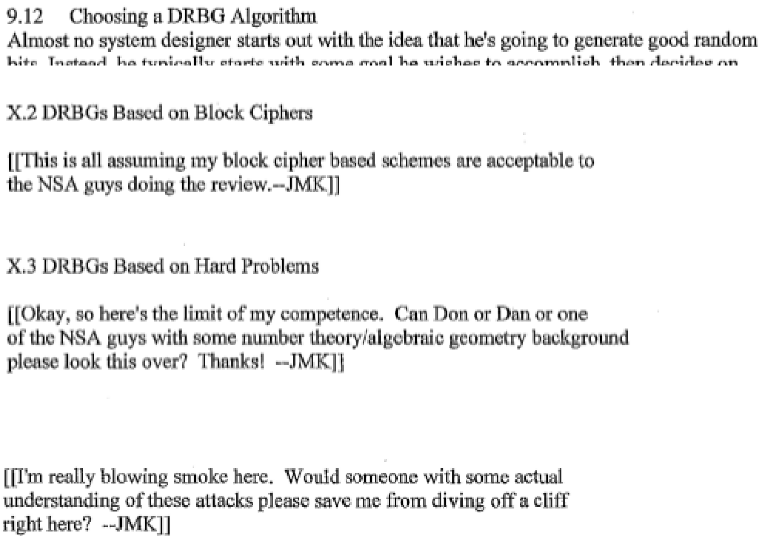


Soul Searching

NSA had previously done background work on DualEC DRBG.

When objections arose we went back, studied the previous work, supplemented it with some new results and began the painful process of Pre-Publication Review.

NIST FOIA



9.12 Choosing a DRBG Algorithm
Almost no system designer starts out with the idea that he's going to generate good random
bits. Instead, he typically starts with some goal he wishes to accomplish, then decides on

X.2 DRBGs Based on Block Ciphers

[[This is all assuming my block cipher based schemes are acceptable to
the NSA guys doing the review.--JMK]]

X.3 DRBGs Based on Hard Problems

[[Okay, so here's the limit of my competence. Can Don or Dan or one
of the NSA guys with some number theory/algebraic geometry background
please look this over? Thanks! --JMK]]

[[I'm really blowing smoke here. Would someone with some actual
understanding of these attacks please save me from diving off a cliff
right here? --JMK]]

Draft Proposal – Extended Random

Draft for a proposed TLS extension named "Extended Random":

- allows client to request up to 2^{16} random bytes,
- has a weak motivation:

  ```
  The rationale for this as stated by DoD is that the
  public randomness for each side should be at least
  twice as long as the security level for cryptographic
  parity, which makes the 224 bits of randomness provided
  by the current TLS random values insufficient.
  ```
- was co-authored by an employee of NSA.

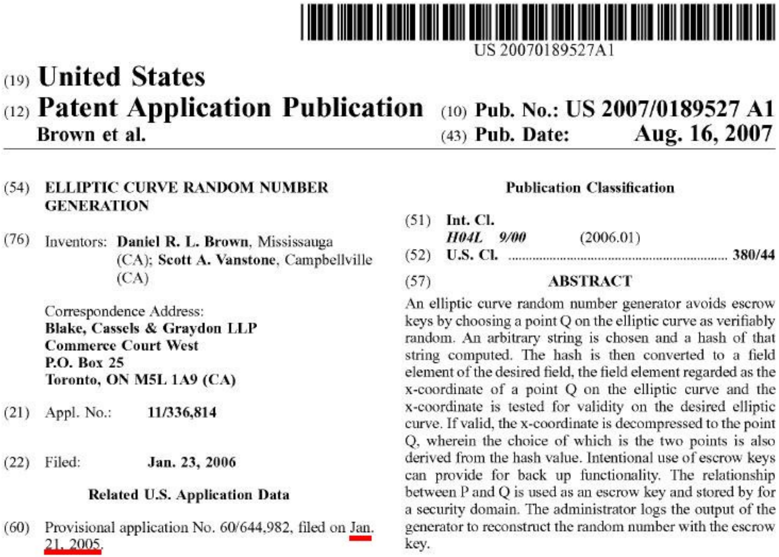
Draft Proposal – Extended Random

Draft for a proposed TLS extension named "Extended Random":

- allows client to request up to 2^{16} random bytes,
- has a weak motivation:
  `The rationale for this as stated by DoD is that the public randomness for each side should be at least twice as long as the security level for cryptographic parity, which makes the 224 bits of randomness provided by the current TLS random values insufficient.`
- was co-authored by an employee of NSA.

Makes Dual EC even more vulnerable!

Snippets from the patent application



US 20070189527A1

(19) **United States**
(12) **Patent Application Publication**
Brown et al.

(10) **Pub. No.: US 2007/0189527 A1**
(43) **Pub. Date: Aug. 16, 2007**

(54) **ELLIPTIC CURVE RANDOM NUMBER GENERATION**

(76) Inventors: **Daniel R. L. Brown**, Mississauga (CA); **Scott A. Vanstone**, Campbellville (CA)

Correspondence Address:
Blake, Cassels & Graydon LLP
Commerce Court West
P.O. Box 25
Toronto, ON M5L 1A9 (CA)

(21) Appl. No.: **11/336,814**

(22) Filed: **Jan. 23, 2006**

Related U.S. Application Data

(60) Provisional application No. 60/644,982, filed on Jan. 21, 2005.

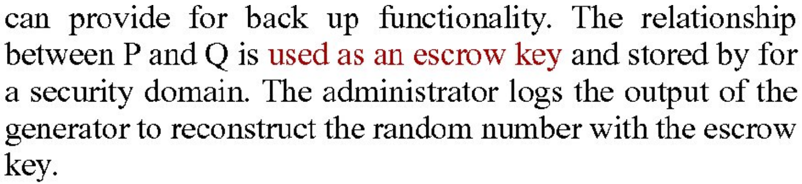
Publication Classification

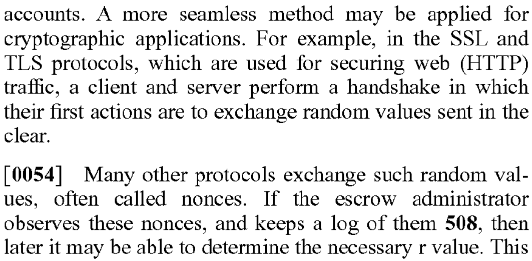
(51) **Int. Cl.**
H04L 9/00 (2006.01)
(52) **U.S. Cl.** **380/44**

(57) **ABSTRACT**

An elliptic curve random number generator avoids escrow keys by choosing a point Q on the elliptic curve as verifiably random. An arbitrary string is chosen and a hash of that string computed. The hash is then converted to a field element of the desired field, the field element regarded as the x-coordinate of a point Q on the elliptic curve and the x-coordinate is tested for validity on the desired elliptic curve. If valid, the x-coordinate is decompressed to the point Q, wherein the choice of which is the two points is also derived from the hash value. Intentional use of escrow keys can provide for back up functionality. The relationship between P and Q is used as an escrow key and stored by for a security domain. The administrator logs the output of the generator to reconstruct the random number with the escrow key.

Snippets from the patent application


can provide for back up functionality. The relationship between P and Q is used as an escrow key and stored by for a security domain. The administrator logs the output of the generator to reconstruct the random number with the escrow key.


accounts. A more seamless method may be applied for cryptographic applications. For example, in the SSL and TLS protocols, which are used for securing web (HTTP) traffic, a client and server perform a handshake in which their first actions are to exchange random values sent in the clear.

[0054] Many other protocols exchange such random values, often called nonces. If the escrow administrator observes these nonces, and keeps a log of them **508**, then later it may be able to determine the necessary r value. This

Additional Attack Cost – Patents

Dual EC patents:

Certicom (now part of Blackberry) has patents in multiple countries on:

- **Dual EC exploitation:** the use of Dual EC for key escrow and
- **Dual EC escrow avoidance:** modification to avoid key escrow.

Additional Attack Cost – Patents

Dual EC patents:

Certicom (now part of Blackberry) has patents in multiple countries on:

- **Dual EC exploitation:** the use of Dual EC for key escrow and
- **Dual EC escrow avoidance:** modification to avoid key escrow.

The patent filing history also shows that:

- Certicom knew the Dual EC back door by 2005,
- NSA was informed of the Dual EC back door by 2005, and
- the patent application, including examples of Dual EC exploitation, was publicly available in July 2006.

Summary

Dual EC — a standardized back door:

- (co-)authored by NSA,

Summary

Dual EC — a standardized back door:

- (co-)authored by NSA,
- may contain a back door (can neither be proven nor disproven),

Summary

Dual EC — a standardized back door:

- (co-)authored by NSA,
- may contain a back door (can neither be proven nor disproven),
- allows the back-door owner to compute all future random outputs,

Summary

Dual EC — a standardized back door:

- (co-)authored by NSA,
- may contain a back door (can neither be proven nor disproven),
- allows the back-door owner to compute all future random outputs,
- makes flaw in DSS a back door that allows impersonation,

Summary

Dual EC — a standardized back door:

- (co-)authored by NSA,
- may contain a back door (can neither be proven nor disproven),
- allows the back-door owner to compute all future random outputs,
- makes flaw in DSS a back door that allows impersonation,
- proven to be practical in various TLS libraries,

Summary

Dual EC — a standardized back door:

- (co-)authored by NSA,
- may contain a back door (can neither be proven nor disproven),
- allows the back-door owner to compute all future random outputs,
- makes flaw in DSS a back door that allows impersonation,
- proven to be practical in various TLS libraries,
- was default RNG in RSA's BSAFE library,

Summary

Dual EC — a standardized back door:

- (co-)authored by NSA,
- may contain a back door (can neither be proven nor disproven),
- allows the back-door owner to compute all future random outputs,
- makes flaw in DSS a back door that allows impersonation,
- proven to be practical in various TLS libraries,
- was default RNG in RSA's BSAFE library,
- back door becomes even stronger with proposed Extended Random,

Summary

Dual EC — a standardized back door:

- (co-)authored by NSA,
- may contain a back door (can neither be proven nor disproven),
- allows the back-door owner to compute all future random outputs,
- makes flaw in DSS a back door that allows impersonation,
- proven to be practical in various TLS libraries,
- was default RNG in RSA's BSAFE library,
- back door becomes even stronger with proposed Extended Random,
- it is not only standardized but even patented.

Summary

Dual EC — a standardized back door:

- (co-)authored by NSA,
- may contain a back door (can neither be proven nor disproven),
- allows the back-door owner to compute all future random outputs,
- makes flaw in DSS a back door that allows impersonation,
- proven to be practical in various TLS libraries,
- was default RNG in RSA's BSAFE library,
- back door becomes even stronger with proposed Extended Random,
- it is not only standardized but even patented.

How to fix it?

Summary

Dual EC — a standardized back door:

- (co-)authored by NSA,
- may contain a back door (can neither be proven nor disproven),
- allows the back-door owner to compute all future random outputs,
- makes flaw in DSS a back door that allows impersonation,
- proven to be practical in various TLS libraries,
- was default RNG in RSA's BSAFE library,
- back door becomes even stronger with proposed Extended Random,
- it is not only standardized but even patented.

Don't use Dual EC!

Summary

Additional information:

https://projectbullrun.org/dual-ec/

Summary

Additional information:

`https://projectbullrun.org/dual-ec/`

Questions?