

Dual EC

A Standardized Back Door?

Ruben Niederhagen

Joint work with Stephen Checkoway, Adam Everspaugh, Matthew Fredrikson, Matthew Green, Tanja Lange, Thomas Ristenpart, Daniel J. Bernstein, Jake Maskiewicz, and Hovav Shacham.

Department of Mathematics and Computer Science (IMADA)

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,
- ▶ ...

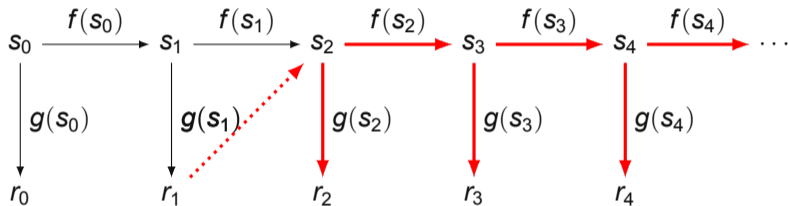
Must be impossible for an attacker to predict!

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,
⇒ update a secret internal state.



Broken if attacker learns internal state!

- June 2004 Dual EC appears in an ANSI draft.
- in 2004 RSA Security 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.

- 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.

Transport Layer Security (TLS)

- ▶ Used in the Internet for encryption of communication.

Examples:

- ▶ eMail transport,
- ▶ online banking,
- ▶ online shopping,
- ▶ ...
- ▶ Standard covers a vast 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

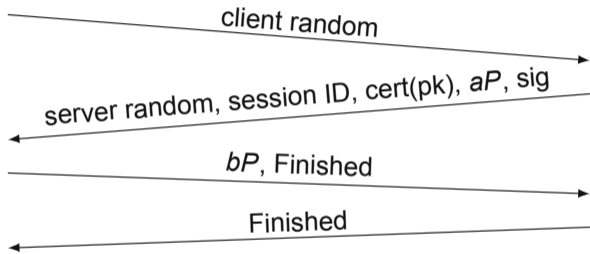
Client

generate
client random

generate b

Server

generate
session ID,
server random, a ,
signature nonce

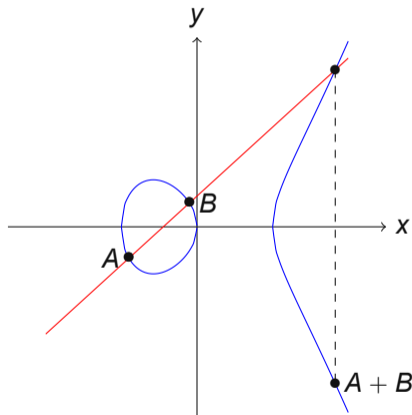


Common TLS implementations:

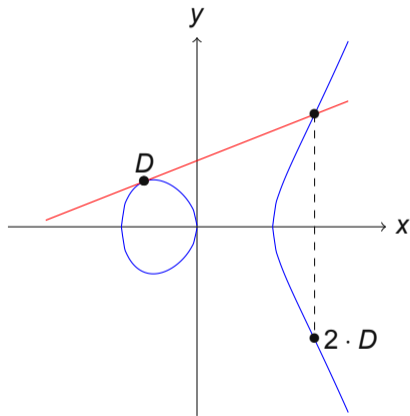
- ▶ RSA's BSAFE
 - ▶ RSA BSAFE Share for Java (BSAFE Java)
 - ▶ RSA BSAFE Share for C and C++ (BSAFE C)
- ▶ Microsoft's SChannel
- ▶ OpenSSL

All of these offer Dual EC.

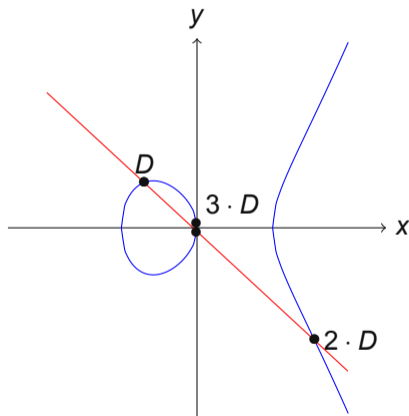
Example – Point Addition: $y = x^3 - x$ over \mathbb{R}



Example – Point Doubling: $y = x^3 - x$ over \mathbb{R}



Example – Scalar Mult.: $y = x^3 - x$ over \mathbb{R}



Arithmetic on Elliptic Curves:

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

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

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 $B = k \cdot A$!

Parameters:

Here: Elliptic curve over finite field 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 :

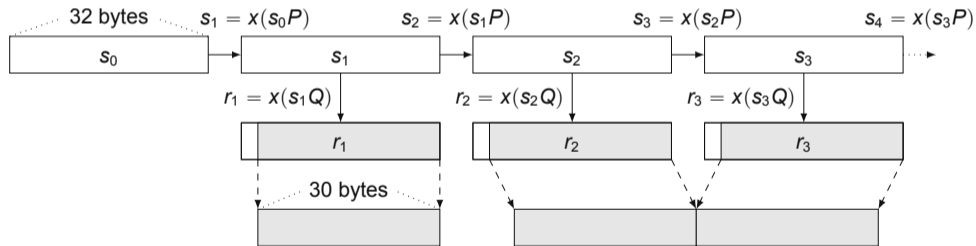
$$P_x = 0x6b17d1f2e12c4247f8bce6e563a440f277037d812deb33a0f4a13945d898c296,$$

$$P_y = 0x4fe342e2fe1a7f9b8ee7eb4a7c0f9e162bce33576b315ececbb6406837bf51f5;$$

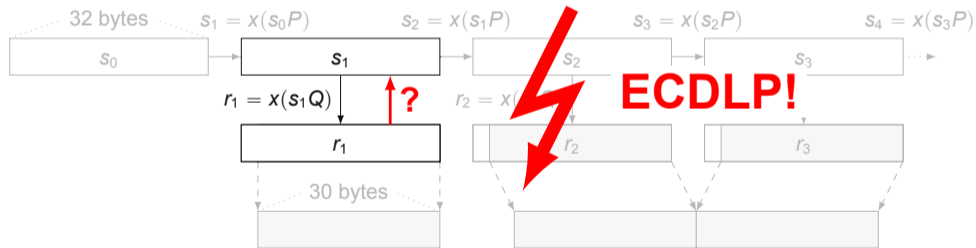
$$Q_x = 0xc97445f45cdef9f0d3e05e1e585fc297235b82b5be8ff3efca67c59852018192,$$

$$Q_y = 0xb28ef557ba31dfcbdd21ac46e2a91e3c304f44cb87058ada2cb815151e610046.$$

Points Q and P on an elliptic curve.

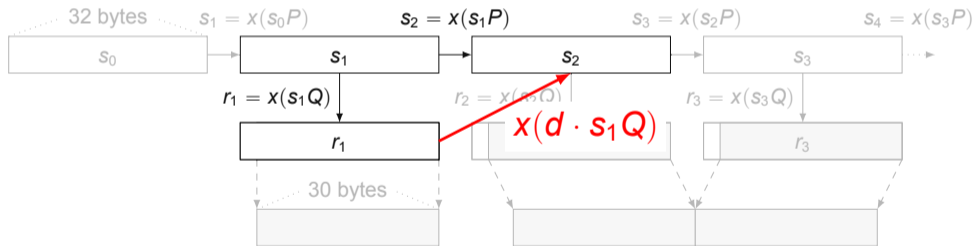


Points Q and P on an elliptic curve.

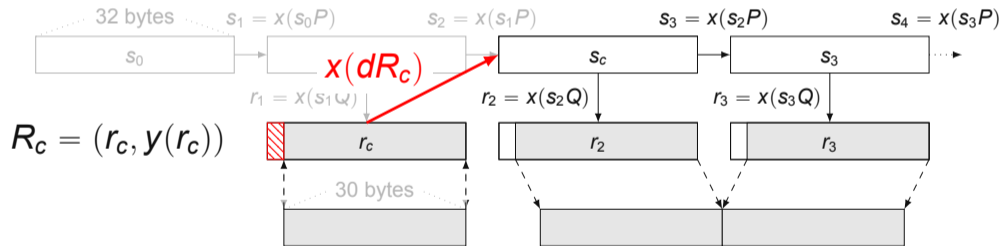


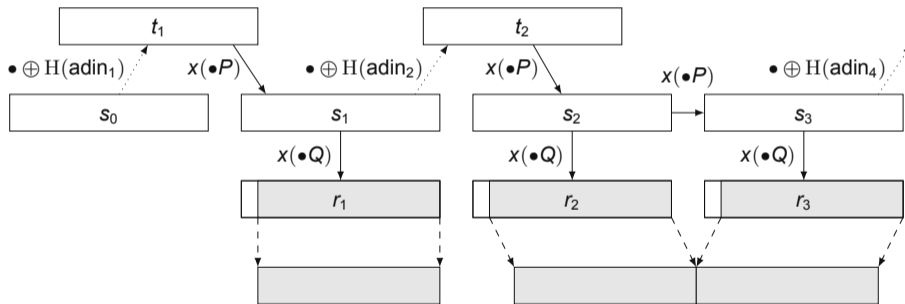
Points Q and $P = dQ$ on an elliptic curve.

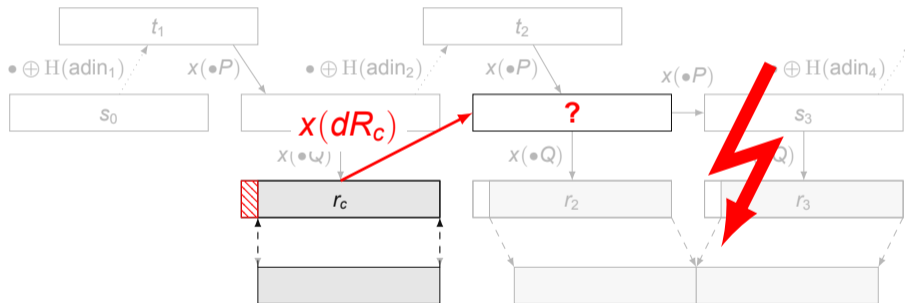
$$s_2 = x(s_1 P) = x(s_1 \cdot dQ)$$

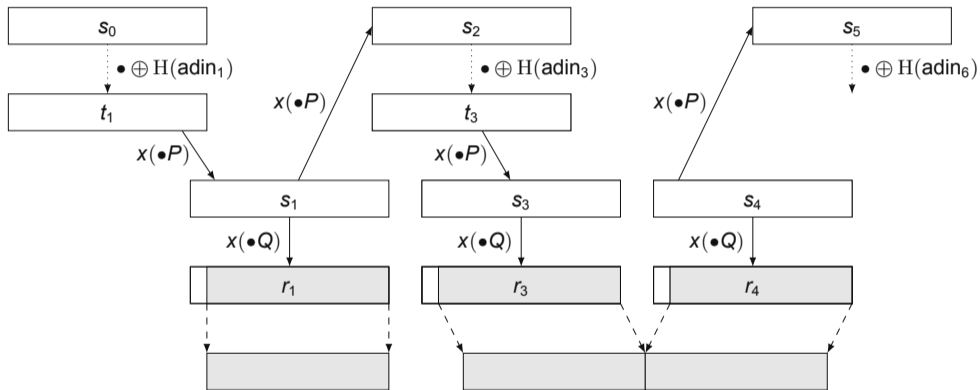


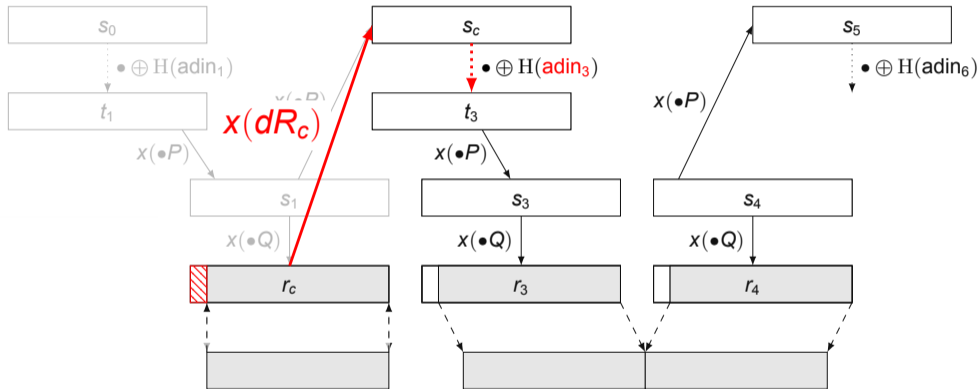
Points Q and $P = dQ$ on an elliptic curve.











Questionnaire

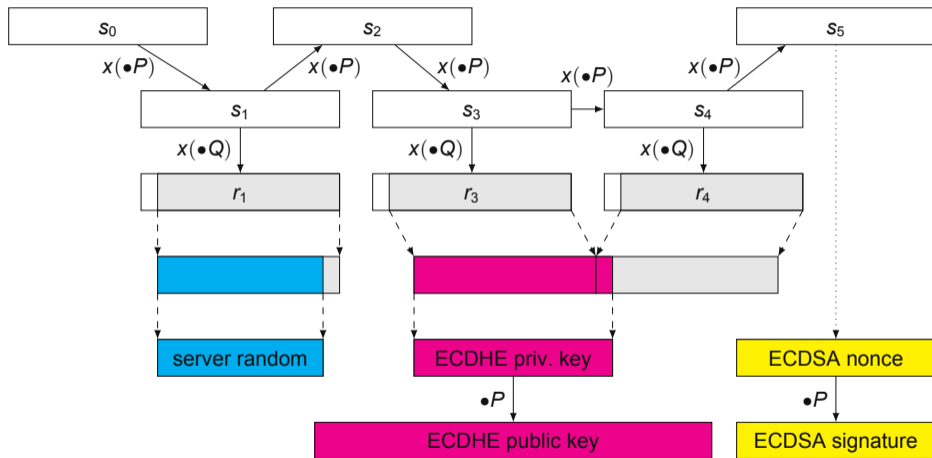
Attack targets in our analysis:

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

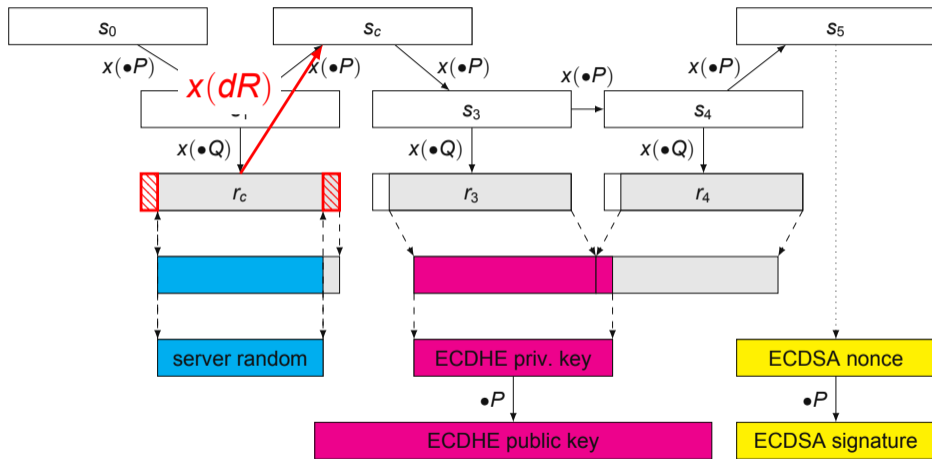
- ▶ RSA's BSAFE
 - ▶ RSA BSAFE Share for Java (BSAFE Java)
 - ▶ RSA BSAFE Share for C and C++ (BSAFE C)
- ▶ Microsoft's SChannel
- ▶ OpenSSL-fixed

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

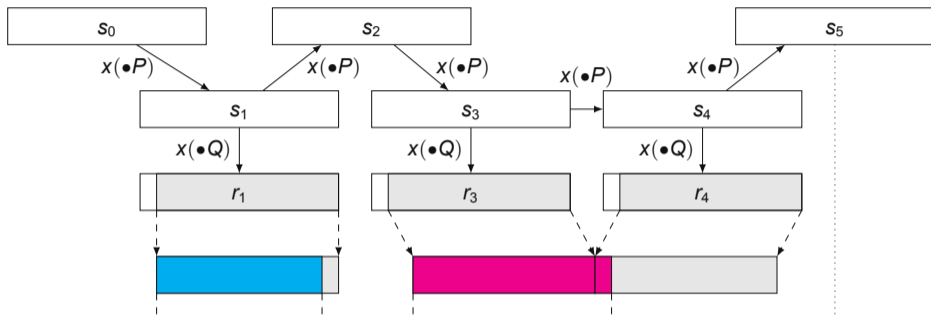
Example: BSAFE-Java



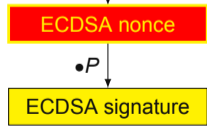
Example: BSAFE-Java

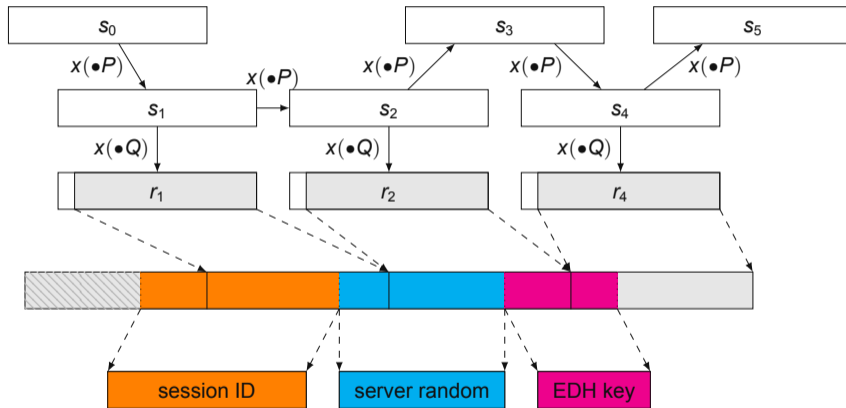


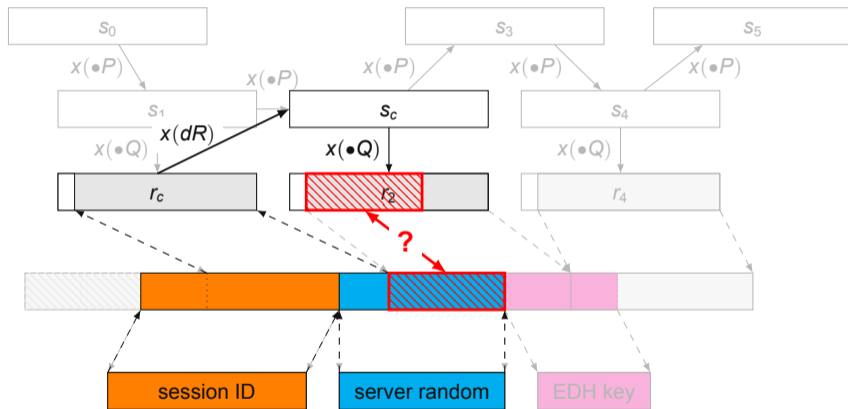
average cost: $2^{31}(C_V + 5C_f)$

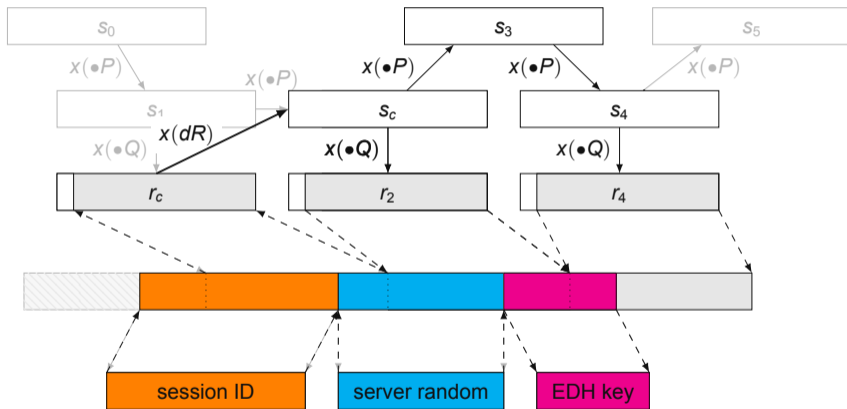


**Exposes longterm secret key!
Impersonation attack possible!**

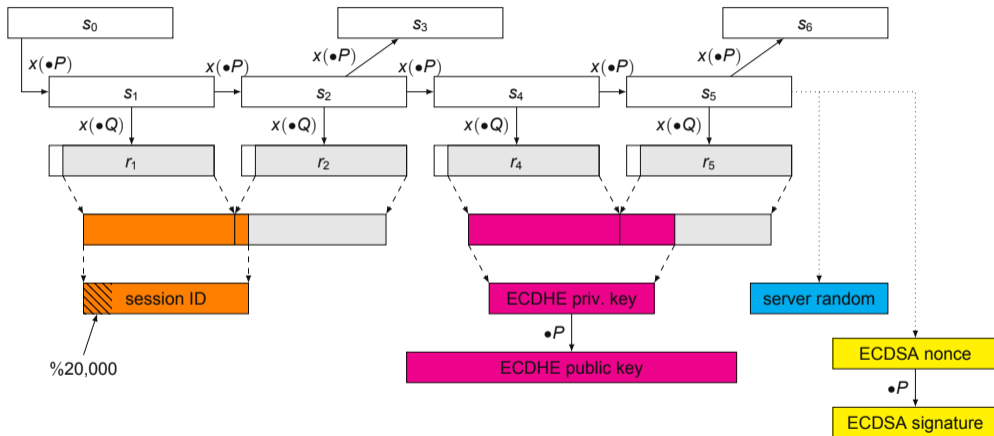


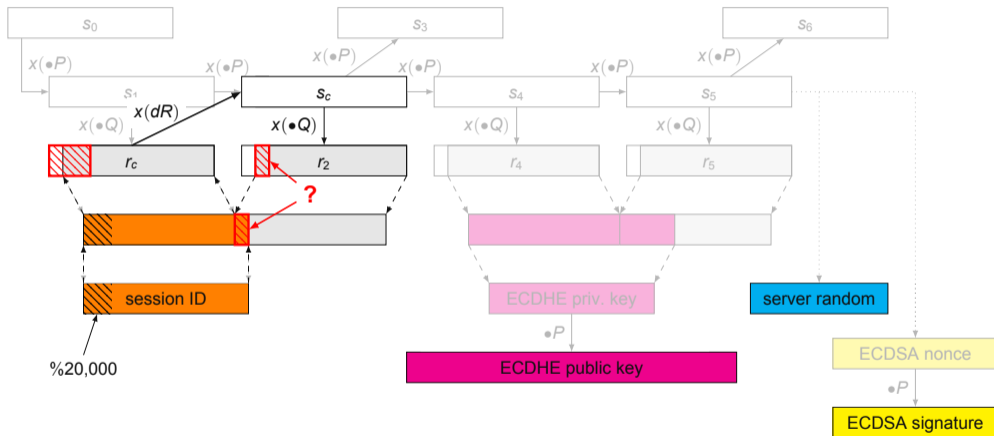


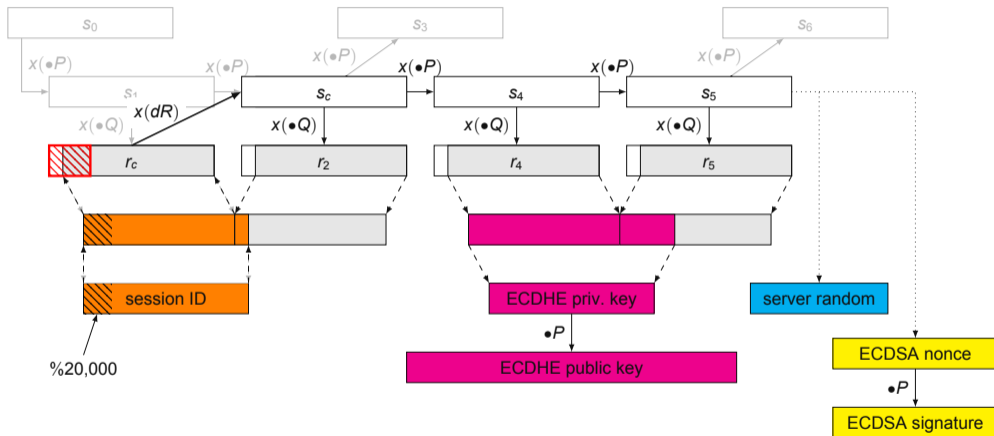




average cost: $30 \cdot 2^{15}(C_v + C_f)$

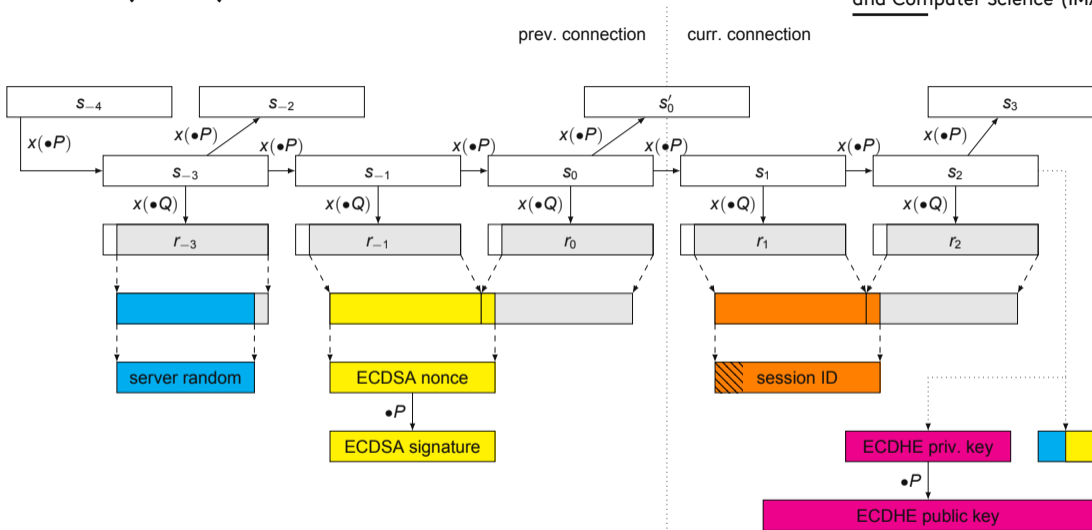




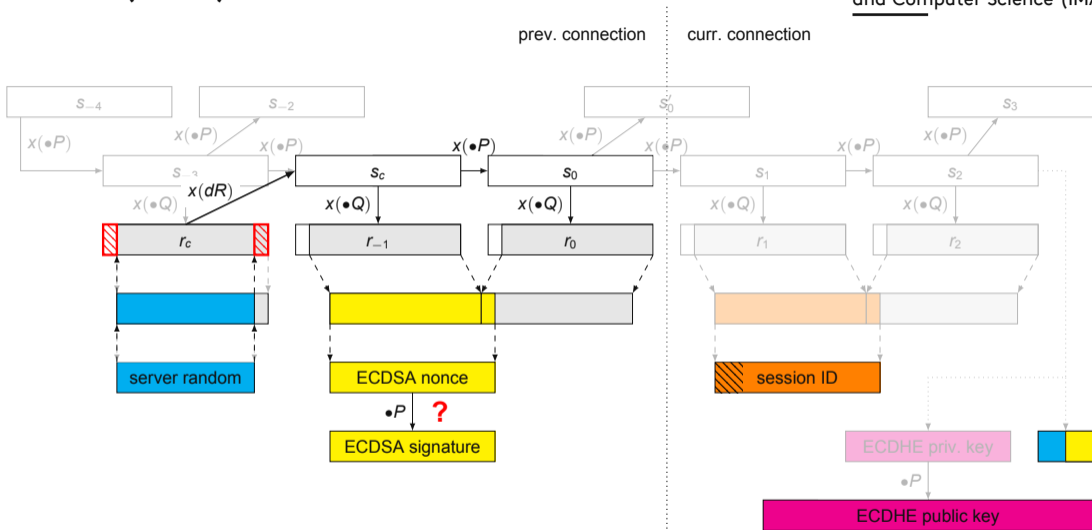


average cost: $2^{33}(C_v + C_f) + 2^{17}(5C_f)$

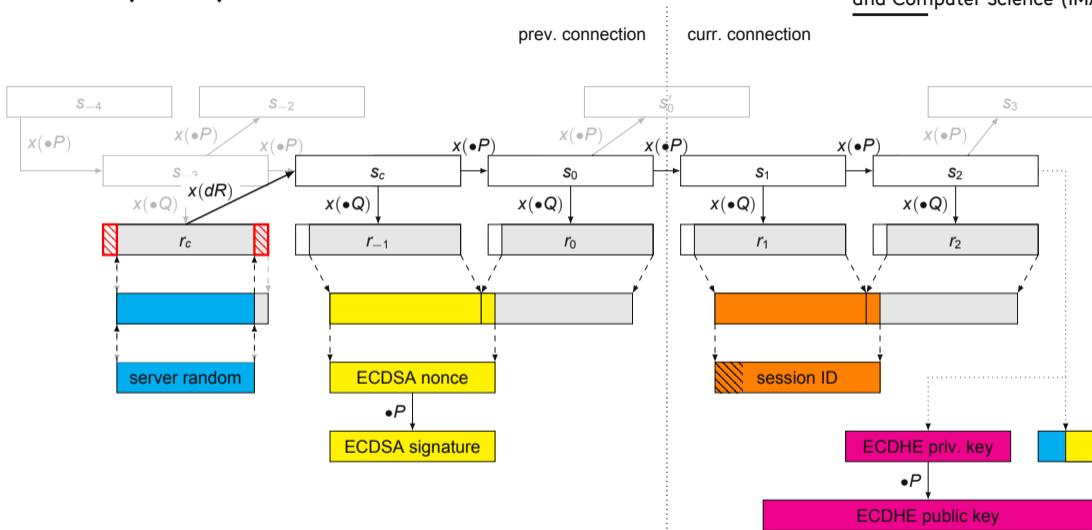
SChannel (cont.)



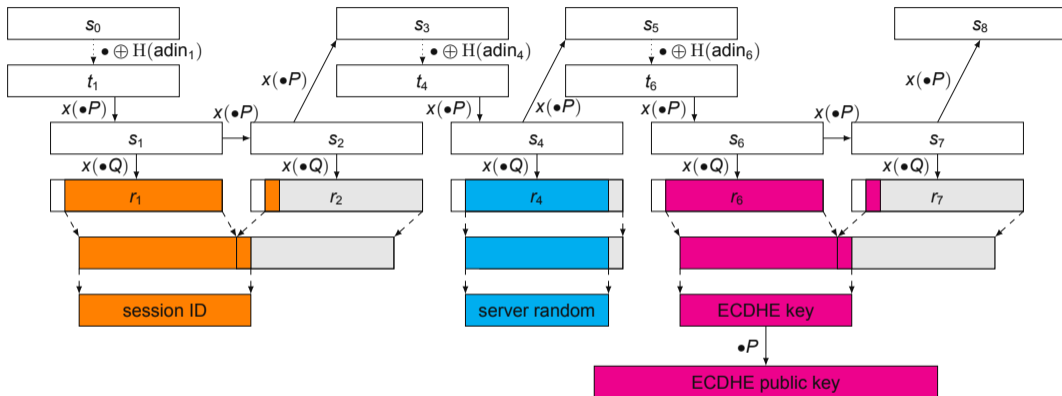
SChannel (cont.)

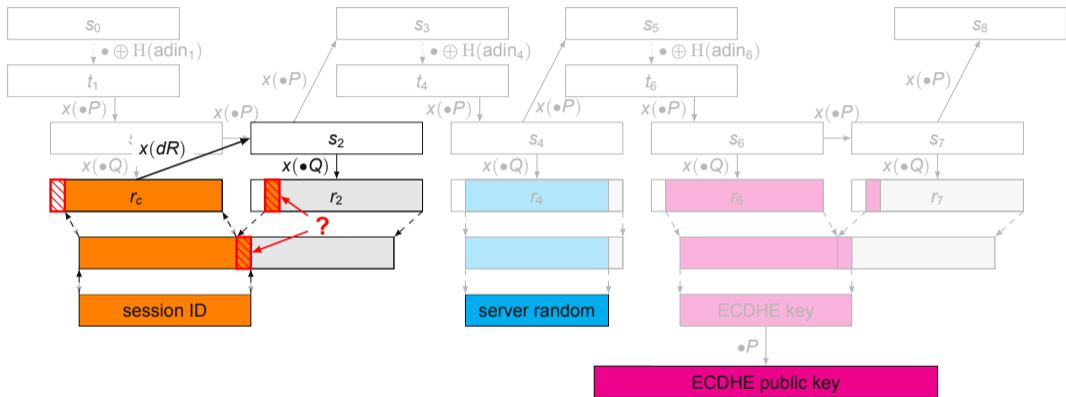


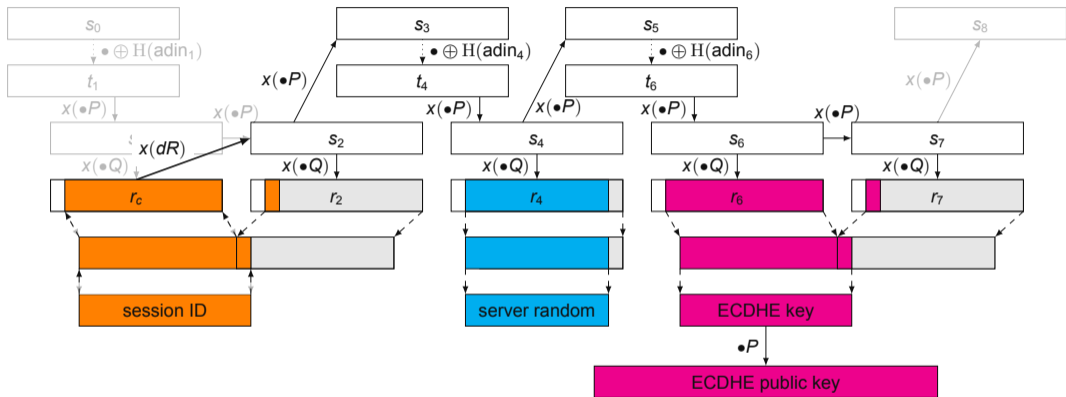
SChannel (cont.)



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



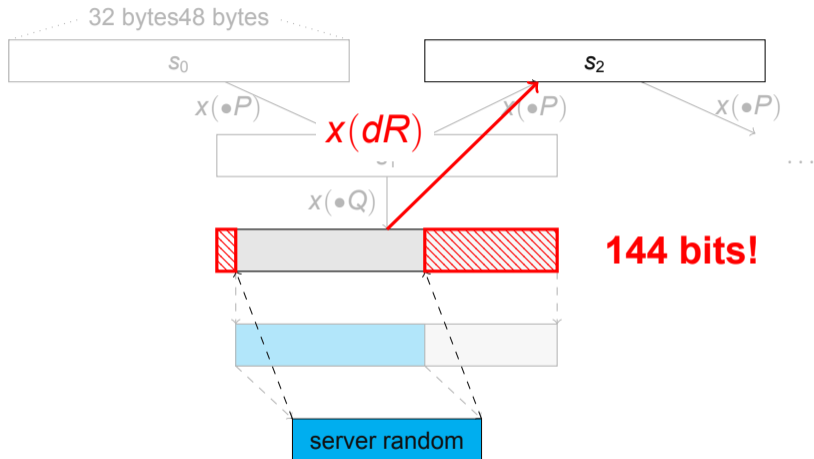




average cost: $2^{15}(C_v + C_f) + 2^{20+k+l}(2C_f) + 2^{13}(5C_f)$

Attack	Intel Xeon CPU		16 × AMD CPU Tot. Time (min)
	Avg. Time (min)	# for 1s	
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	$2^k \cdot 707$	$2^k \cdot 44,200$	$2^k \cdot 83.32$

Dual EC with P-384 and P-521



Draft for a proposed TLS extension named “Extended Random”:

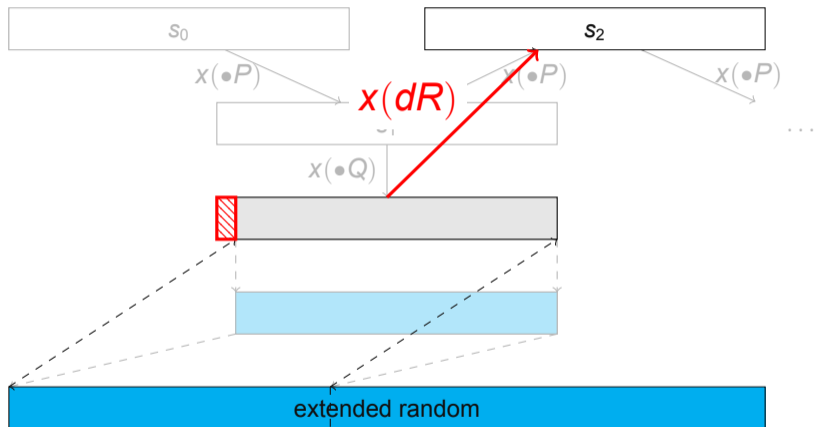
- ▶ allows client to request up to 2^{16} random bytes,

- ▶ 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!



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”.

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]]



US 20070189527A1

(19) **United States**

(12) **Patent Application Publication** (10) **Pub. No.: US 2007/0189527 A1**

Brown et al. (43) **Pub. Date: Aug. 16, 2007**

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

Publication Classification

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

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

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

(57) **ABSTRACT**

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

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.

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

Related U.S. Application Data

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

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

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.

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 DSA a back door that allows impersonation,
- ▶ proven to be practical in various TLS libraries,
- ▶ was default RNG in RSA Security'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!

Additional information:

- ▶ Checkoway, Fredrikson, Niederhagen, Everspaugh, Green, Lange, Ristenpart, Bernstein, Maskiewicz, and Shacham: “On the Practical Exploitability of Dual EC in TLS Implementations”. USENIX 2014.
- ▶ <https://projectbullrun.org/dual-ec/>