Constant-time BIGNUM is bollocks

B. B. Brumley, D.Sc. (Tech.)

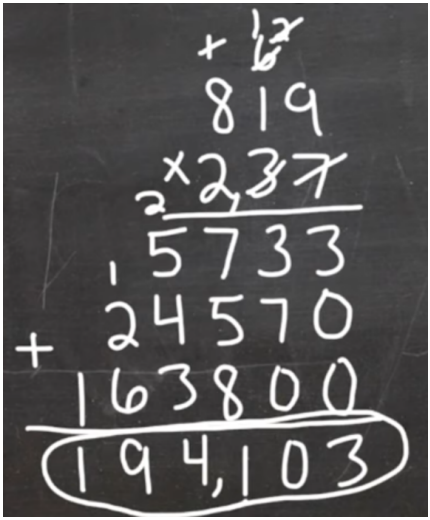
Director of Research, ESL Global Cybersecurity Institute (GCI)
Kevin O'Sullivan Endowed Professor, Department of Cybersecurity (CSEC)
Director, Platform Security Laboratory (PLATSEC)
Rochester Institute of Technology
Rochester, New York, USA
bbbics AT rit DOT edu

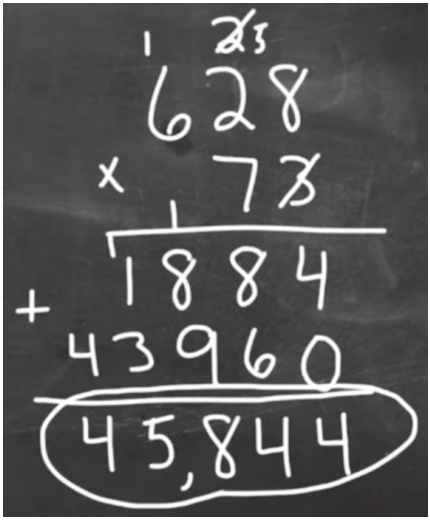
2011: My first source code contribution to OpenSSL

```
--- openssl-0.9.8g.orig/crypto/ecdsa/ecs_ossl.c
+++ openssl-0.9.8g/crypto/ecdsa/ecs_ossl.c
@@ -144,6 +144,14 @@
             }
         while (BN_is_zero(k));

+        /* We do not want timing information to leak the length of k,
+         * so we compute G*k using an equivalent scalar of fixed
+         * bit-length. */
+
+        if (!BN_add(k, k, order)) goto err;
+        if (BN_num_bits(k) <= BN_num_bits(order))
+            if (!BN_add(k, k, order)) goto err;
+
         /* compute r the x-coordinate of generator * k */
         if (!EC_POINT_mul(group, tmp_point, k, NULL, NULL, ctx))
         {
```

BIGNUM Side Channels 101





ECDSA signing: potential leakage points

$$r = x(kG) \bmod n$$
$$s = k^{-1}(H(m) + d \cdot r) \bmod n$$

Red: scalar multiplication (kG)

Orange: nonce inversion (k^{-1})

Blue: private-key multiplication ($d \cdot r$)

Remote Timing Attacks are Still Practical*

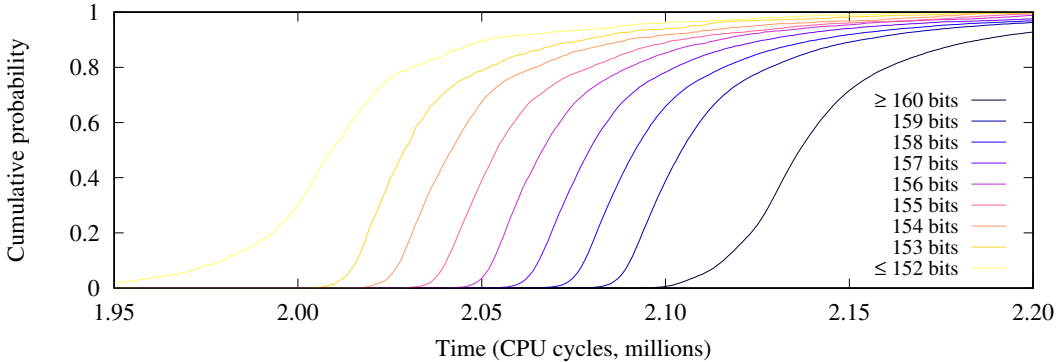
Billy Bob Brumley and Nicola Tuveri

Aalto University School of Science, Finland
{bbrumley,ntuveri}@tcs.hut.fi

Abstract. For over two decades, timing attacks have been an active area of research within applied cryptography. These attacks exploit cryptosystem or protocol implementations that do not run in constant time. When implementing an elliptic curve cryptosystem with a goal to provide side-channel resistance, the scalar multiplication routine is a critical component. In such instances, one attractive method often suggested in the literature is Montgomery's ladder that performs a fixed sequence of curve and field operations. This paper describes a timing attack vulnerability in OpenSSL's ladder implementation for curves over binary fields.

```
/* find top most bit and go one past it */
i = scalar -> top - 1; j = BN_BITS2 - 1;
mask = BN_TBIT ;
while (!( scalar -> d[i] & mask )) { mask >>= 1; j --; }
mask >>= 1; j - -;
/* if top most bit was at word break , go to next word */
if (! mask )
{
  i - -; j = BN_BITS2 - 1;
  mask = BN_TBIT ;
}
for (; i >= 0; i - -)
{
  for (; j >= 0; j - -)
  {
    if ( scalar ->d[ i] & mask )
...
```

OpenSSL 2010



Vulnerability Note VU#536044

OpenSSL leaks ECDSA private key through a remote timing attack

Original Release date: 17 May 2011 | Last revised: 01 Jun 2011

Overview

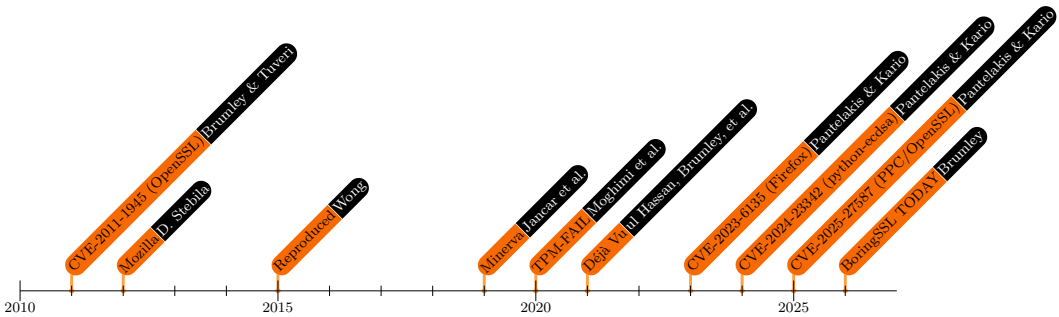
The OpenSSL ladder implementation for scalar multiplication of points on elliptic curves over binary fields is susceptible to a timing attack vulnerability. This vulnerability can be used to steal the private key of a TLS server that authenticates with ECDSA signatures and binary curves.

Description

Billy Bob Brumley's and Nicola Tuveri's paper "Remote Timing Attacks are Still Practical" states:

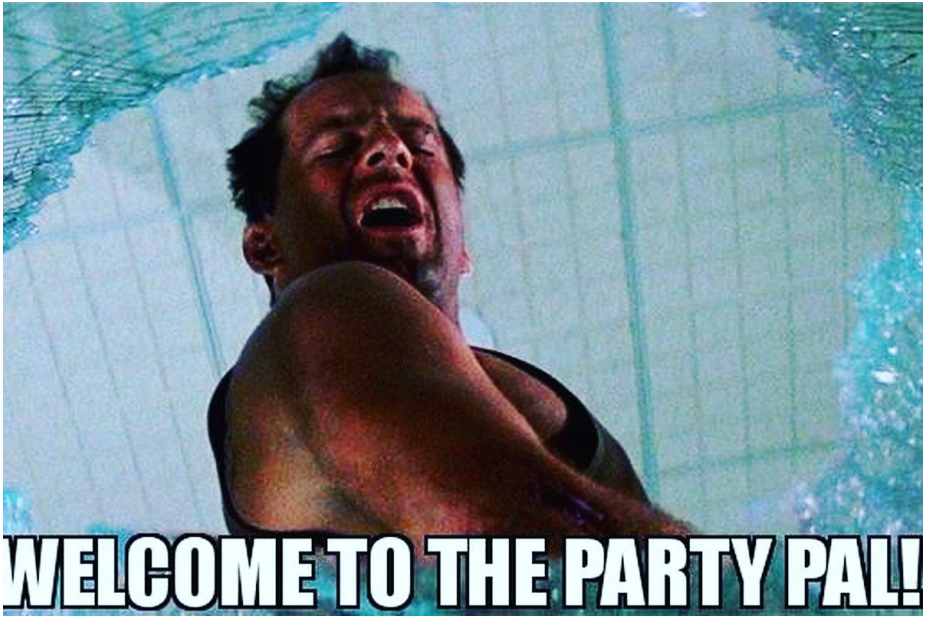

WHAC
-A-
MOLE
FREE PLAY

Timeline: Effective bit lengths can be secrets, too



(Yes, TODAY.)


YOU MERELY ADOPTED THE DARK
I WAS BORN IN IT


WELCOME TO THE PARTY PAL!

Déjà Vu: Side-Channel Analysis of Mozilla's NSS

ul Hassan, Brumley, et al., CCS 2020

multiple vendors (e.g., OpenSSL), which highlights a gap in the practice of CVE coordination among peer vendors.

Cui bono? NSS is certainly neither the first nor last security library to fall prey to SCA and failure to use constant-time implementations. Why is this a recurring event? Who should be held culpable? We note the break, fix, break cycle benefits several stakeholders due to perverse incentives—to mention a few: (i) it keeps software engineers in demand since these libraries are not "deploy and forget"; (ii) it keeps security engineers in demand since there is a steady stream of security issues to assess and address; (iii) it keeps security researchers busy with a perpetual flow of research topics to write papers about—including us. During judgment, the ancient Romans inquired *Cui bono?* or "Who benefits?" to identify suspects. Perhaps the incomplete list of key players above in this self-perpetuating meta-system is a good start.

Mitigations. During responsible disclosure to Mozilla, we made several FOSS contributions to assist in mitigating these issues and testing the fixes—all of which are now merged. (i) To solve the vul-

OpenSSL Foundation and Corportation BAC Meeting

Brno, May 2025

Peter Gutmann: “Timing attacks are bollocks”
Me: “Yes, and constant-time BIGNUM is a pipe dream anyway”
Pauli Dale: “BoringSSL did it”

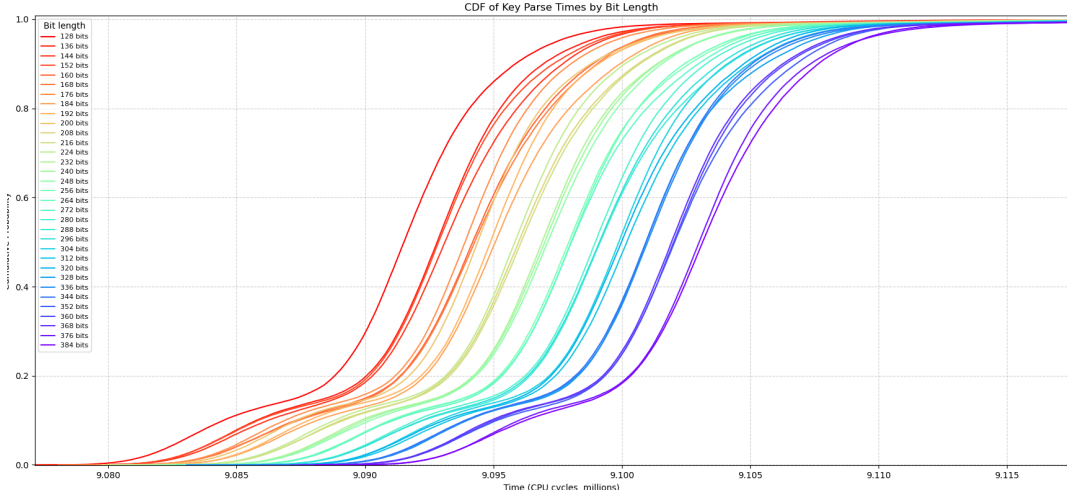
BORINGSSL 0DAY TIMING ATTACK

https://gitlab.com/platsec/boringssl-keyload-vuln

OpenSSL Corporation BAC / TAC Meeting

Brno, last week

BoringSSL 03 Oct 2025 (master)



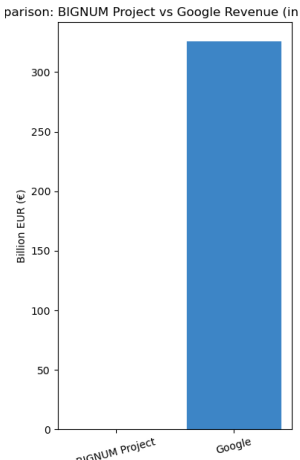

This not
BoringSSL's folt
CHANGE MY MIND


BUT WAIT
THERE'S MORE

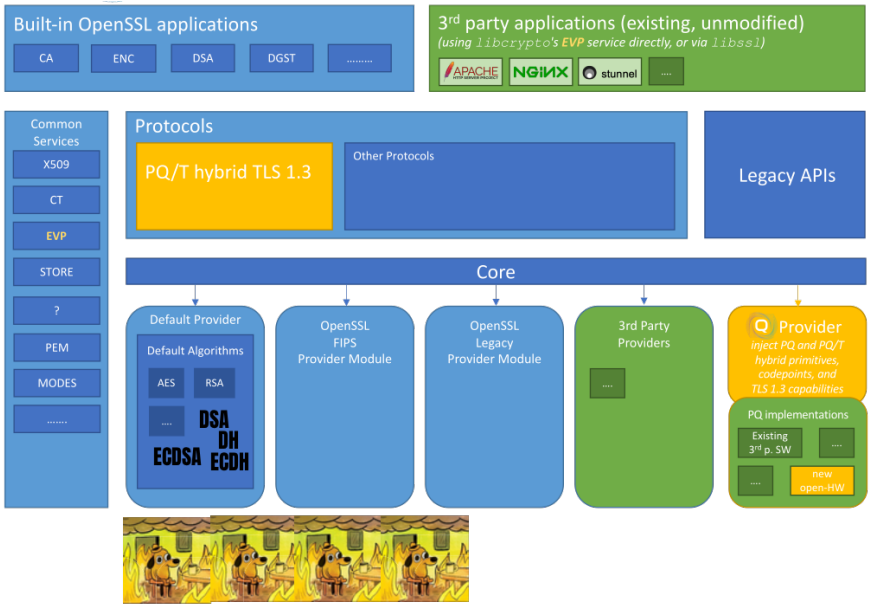
Sovereign Tech Fund Invests in OpenSSL

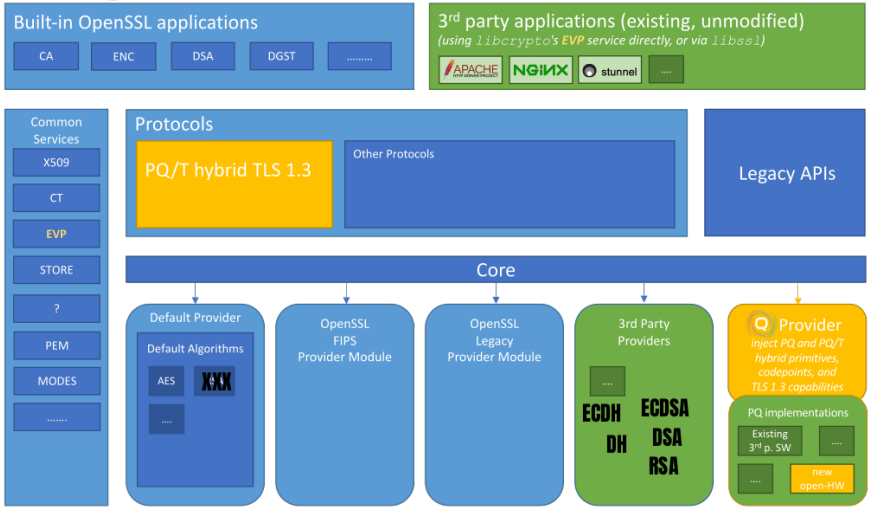
Aug 4, 2025

The OpenSSL Foundation is pleased to announce a €405,888 investment from the Sovereign Tech Fund to enhance timing side-channel resistance in the BIGNUM code and address a backlog of user-submitted GitHub issues.



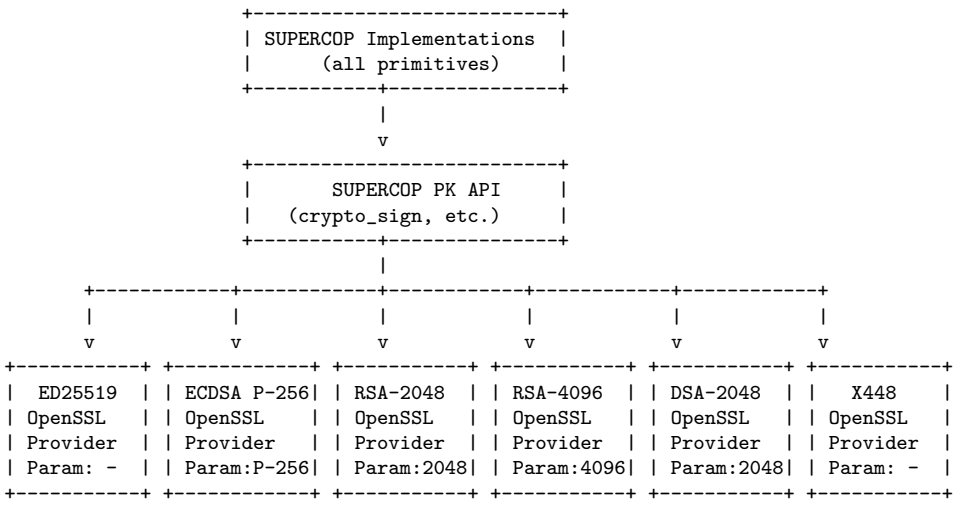

Built-in OpenSSL applications
CA
ENC
DSA
DGST
.....
3rd party applications (existing, unmodified)
(using libcrypto's EVP service directly, or via libssl)
APACHE
NGINX
stunnel
....
Common Services
X509
CT
EVP
STORE
?
PEM
MODES
.....
Protocols
PQ/T hybrid TLS 1.3
Other Protocols
Legacy APIs
Core
Default Provider
Default Algorithms
AES
RSA
....
DSA
DH
ECDSA
ECDH
OpenSSL FIPS Provider Module
OpenSSL Legacy Provider Module
3rd Party Providers
....
Provider
ct PQ and PQ/T
ybrid primitives,
codepoints, and
LS 1.3 capabilities
PQ implementations
Existing 3rd p. SW
....
....
new open-HW
BIGNUM

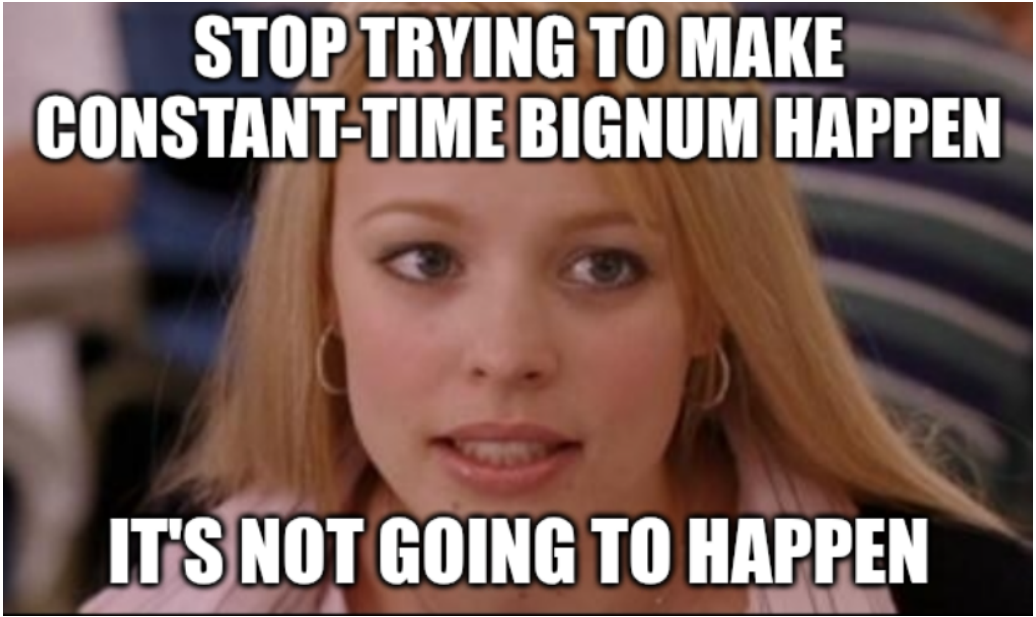

Built-in OpenSSL applications
CA
ENC
DSA
DGST
.....
3rd party applications (existing, unmodified)
(using libcrypto's EVP service directly, or via libssl)
APACHE
NGINX
stunnel
....
Common Services
X509
CT
EVP
STORE
?
PEM
MODES
.....
Protocols
PQ/T hybrid TLS 1.3
Other Protocols
Legacy APIs
Core
Default Provider
Default Algorithms
AES
RSA
....
DSA
DH
ECDSA
ECDH
OpenSSL FIPS Provider Module
OpenSSL Legacy Provider Module
3rd Party Providers
....
Provider
inject PQ and PQ/T hybrid primitives, codepoints, and TLS 1.3 capabilities
PQ implementations
Existing 3rd p. SW
....
....
new open-HW



SUPERCOP / S2N-BIGNUM?

OpenSSL Providers from SUPERCOP straightline implementations




STOP TRYING TO MAKE
CONSTANT-TIME BIGNUM HAPPEN
IT'S NOT GOING TO HAPPEN