An update on the backdoor in Juniper's ScreenOS

Us: Stephen Checkoway, Shaanan Cohney, Matthew Green, Nadia Heninger, Eric Rescorla, and **Hovav Shacham**.
Important contributions by: H.D. Moore, Samuel Neves, Willem Pinckaers, and Ralf-Philipp Weinmann.

Juniper security advisory, 17 Dec 2015

Administrative Access (CVE-2015-7755) allows unauthorized remote administrative access to the device. Exploitation of this vulnerability can lead to complete compromise of the affected device.

This issue only affects ScreenOS 6.3.0r17 through 6.3.0r20. **No other Juniper products or versions of ScreenOS are affected by this issue.**

Upon exploitation of this vulnerability, the log file would contain an entry that '**system**' had logged on followed by password authentication for a username.

Example:

Normal login by user **username1**:

```
2015-12-17 09:00:00 system warn 00515 Admin user username1 has logged on via SSH from …..
2015-12-17 09:00:00 system warn 00528 SSH: Password authentication successful for admin
user 'username1' at host …
```

Compromised login by user **username2**:

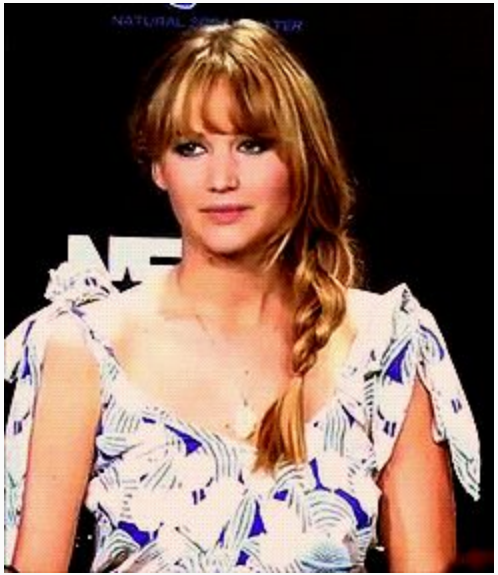
```
2015-12-17 09:00:00 system warn 00515 Admin user system has logged on via SSH from …..
```

The login backdoor

Extra check in `auth_admin_internal` allows admin login using password “`<<< %s(un='%s') = %u`”

```
ADD             R0, R5, #0x44
LDR             R1, =aSUnSU ; "<<< %s(un='%s') = %u"
BL              strcmp
CMP             R0, #0
BNE             loc_13DC78
MOV             R0, #0xFFFFFFFD
LDMDB           R11, {R4-R8,R11,SP,PC}
```

(from ARM disassembly by H.D. Moore)



Idea already worked out in *Phrack*, 2009

```
                    ==Phrack Inc.==

              Volume 0x0d, Issue 0x42, Phile #0x05 of 0x11

|=-----------------------------------------------------------------------=|
|=---------------=[     Netscreen of the Dead:       ]=------------------=|
|=-=[ Developing a Trojaned Firmware for Juniper ScreenOS Platforms  ]=--=|
|=-----------------------------------------------------------------------=|
|=-----------------------------------------------------------------------=|
|=--------------------=[     By graeme@lolux.net      ]=------------------=|
|=-----------------------------------------------------------------------=|
```

Changed constants in an H.D. Moore diff

P-256 Weierstraß *b*

P-256 *P* x coord

P-256 field order

```
      5AC635D8AA3A93E7B3EBBD5576986BC651D06B                C3E27D2604B
      6B17D1F2E12C4247F8BCE6E563A440F277037D812DEB33A0F4A139
      FFFFFFFF00000000FFFFFFFFFFFFFFFFBCE6FAADA7179E84F3B9CAC2FC632551
bad:  9585320EEAF81044F20D55030A035B11BECE81C785E6C933E4A8A131F6578107
good: 2c55e5e45edf713dc43475effe8813a60326a64d9ba3d2e39cb639b0f3b0ad10
nist: c97445f45cdef9f0d3e05e1e585fc297235b82b5be8ff3efca67c59852018192
```

Reverse engineering shows changed values are x coords for Dual EC point *Q*

ScreenOS was FIPS certified, but not with Dual EC

ScreenOS on NIST's RNG validation list: "ANSI X9.31 [TDES-3Key]".

But, from an October, 2013 Juniper Knowledge Base article:

The following product families do utilize Dual_EC_DRBG, but do not use the pre-defined points cited by NIST:

1. ScreenOS*

* ScreenOS does make use of the Dual_EC_DRBG standard, but is designed to not use Dual_EC_DRBG as its primary random number generator. ScreenOS uses it in a way that should not be vulnerable to the possible issue that has been brought to light. Instead of using the NIST recommended curve points it uses self-generated basis points and then takes the output as an input to FIPS/ANSI X.9.31 PRNG, which is the random number generator used in ScreenOS cryptographic operations.

NetScreen RNG core (6.2, 6.3): produces 32 bytes into prng_output_buf

```
void prng_generate_block(void)
{
  ...
  prng_output_idx = 0;
  ++blocks_generated_since_reseed;
  if ( !prng_reseed_not_needed() )     // in default config, always returns 0
    prng_do_reseed();
  for ( ; (unsigned int)prng_output_idx <= 31; prng_output_idx += 8 )
  { /* obtain 8 bytes from X9.31, copy to offset in prng_output_buf */ }
}
```

NetScreen RNG reseed (6.2, 6.3): runs Dual EC, uses output to seed X9.31

```
void prng_do_reseed(void)
{
  ...
  if ( dual_ec_bytes(prng_output_buf, 32) != 32 )
    { /* log error */ }
  // set X9.31 seed and X9.31 DES subkeys using prng_output_buf:
  memcpy(&ansi_x9_31_seed,     prng_output_buf,   8 );
  memcpy(&ansi_x9_31_3des_key, prng_output_buf+8, 24);
  prng_output_idx = 32;
  ...
}
```

NetScreen RNG core (6.2, 6.3): Looking through the bug

```
void prng_generate_block(void)
{
  ...
  prng_output_idx = 0;
  ++blocks_generated_since_reseed;
  if ( !prng_reseed_not_needed() )     // in default config, always returns 0
    prng_do_reseed();   // sets prng_output_idx = 32
  for ( ; (unsigned int)prng_output_idx <= 31; prng_output_idx += 8 )
  { /* obtain 8 bytes from X9.31, copy to offset in prng_output_buf */ }
}
```

Willem Pinckaers first spotted the bug, from Ralf-Philipp Weinmann's disassembly

Never run. Does not overwrite `prng_output_buf`

RNG consumer: IKE nonce generation

IKE protocol uses nonces, equivalent to TLS client and server randoms

But: IKE doesn't specify nonce length

> Logjam authors' scan: >50% of responders use 20-byte nonces

ScreenOS (6.2, 6.3): Nonces are 32 bytes, directly from `prng_output_buf`.

> **This means they are unfiltered Dual EC outputs (30 bytes + 2 bytes)**
> With knowledge of dlog *Q*, recover RNG state, predict subsequent outputs

Does recovering Dual EC state from a nonce reveal the keys for *that* IPsec session?

ScreenOS constructs nonce **after** it constructs the key exchange message:

```
IKE<#.#.#.#> Construct ISAKMP header.
IKE<#.#.#.#> Msg header built (next payload #4)
IKE<#.#.#.#> Construct [KE] for ISAKMP
IKE<#.#.#.#> Construct [NONCE]
IKE<#.#.#.#> Construct [CERT-REQ]
IKE<#.#.#.# > Xmit : [KE] [NONCE] [CERT-REQ]
```

But nonces are **pregenerated** from RNG, pulled from a nonce FIFO as needed …

For comparison: ScreenOS 6.1.x series

X9.31 RNG (no Dual EC)

Reseeding after reasonable interval (10,000 blocks)

Seeding from (interrupt?) entropy gathering

Core `prng_generate_block` function produces **20** bytes

IKE nonces are 20 bytes, too

No nonce pregeneration

ScreenOS timeline

27 Oct 2008, 6.2.0r1:

- Introduced Dual EC
- Introduced bug in RNG code
- Made IKE nonces be 32 bytes
- Generated 2c55 point
- Added globals to RNG code
- Made RNG core produce 32 bytes
- Added nonce pregeneration table

12 Sep 2012, 6.2.0r15:

- Replaced 2c55 point with 9585 point

25 Apr 2014, 6.3.0r17:

- Added SSH backdoor

Juniper response

VPN Decryption (CVE-2015-7756) may allow a knowledgeable attacker who can monitor VPN traffic to decrypt that traffic. It is independent of the first issue.

This issue affects ScreenOS 6.2.0r15 through 6.2.0r18 and 6.3.0r12 through 6.3.0r20. **No other Juniper products or versions of ScreenOS are affected by this issue.**

There is no way to detect that this vulnerability was exploited.

This issue has been assigned CVE-2015-7756.

17 Dec 2015, disclosure and patch:

Removed SSH backdoor

Restored 2c55 point