Crippling Crypto: The Debian OpenSSL Debacle

The Last HOPE
Jacob Appelbaum, Dino Dai Zovi, Karsten Nohl
DILBERT By Scott Adams

TOUR OF ACCOUNTING
OVER HERE WE HAVE OUR RANDOM NUMBER GENERATOR.

NINE NINE NINE NINE NINE

ARE YOU SURE THAT'S RANDOM?

THAT'S THE PROBLEM WITH RANDOMNESS: YOU CAN NEVER BE SURE.

DEBIAN
You can never be sure.
Cryptographic Keys

- Foundation of Internet security
  - Encryption, Authentication, Digital Signatures
- Keys are at most as secure as they are hard to guess
  - High entropy needed (measured in bits)
    - Each bit of entropy doubles attack time
    - Alphanumerical Windows Lanman passwords: ≤ 36 bits
    - 80 bits and above considered “very secure”
- Keys often generated with OpenSSL package
  - Browsing: TLS/SSL, Administration: SSH, Anonymity: Tor, OpenVPN, …
Entropy is collected from several sources:
- /dev/urandom
- process ID, architecture, size(long)
- non-initialized memory (low entropy)
- Automated memory tool Valgrind flags “bug”
  - Reading from memory before writing to it
  - Debian packet maintainer removes alleged bug
    - Also removes most other entropy sources
    - Does not escalate the bug to the OpenSSL team
Debian OpenSSL Bug

- Entropy for key generation limited to 15 bits
- Affected various Linux distributions
  - Debian “Etch” 4.0
  - Ubuntu 7.04 – 8.04, Kubuntu, …
- Patched after one year on May 7, announced May 13
Weaknesses: SSH

Different types of keys are affected:
- Host keys
  - “known hosts” list, prevents MITM
- User keys for password-less login
  - Often locally generated, affects non-Debian boxes
  - High attack activity after patch
    (before announcement!)
- Session keys
  - Weak encryption if *either* host uses weak OpenSSL
Weaknesses: SSH (II)

- SSH is built to provide forward secrecy
  - Past sessions can’t be decrypted even with your key
  - Achieved through Diffie-Hellman key exchange
- Debian bug breaks forward secrecy
  - Even if all user/host keys are strong
  - Even if your machine’s RNG is strong
- Breaking RNG makes for stealthy backdoor
Fun with SSH:

http://www.cr[zero].org/progs/sshfun

- ssh_kex_keygen – generates weak keys
- bfssh – brute-forces weak keys (ssh -i key1 -i key2 ...
- ssh_decoder - whoops!
Weaknesses: Tor

- Bug affects anonymity of non-Debian users:
  - 300/1500 Tor relays were affected
    - Very small chance of picking 3 weak hosts: <0.2%
    - Weak keys were blacklisted almost immediately
  - Affected hidden services can be spoofed
  - 3 of 6 directory servers vulnerable
    - 4 needed to change server preference, build new Tor
- Tor users on Debian: The above and more
  - New Tor packet replaces weak keys
Weaknesses: TLS/SSL

- Spoofing web sites
  - Powerful in with DNS Poisoning / ARP Spoofing
- Decrypting traffic
  - Password, account numbers, TAN numbers, ...
- SSL also used for non http-traffic
  - Example: Updates for German tax-paying software are signed with weak key
Finding Weak Keys

- Generate list of all keys
  - Equally needed for exploiting and defending
- As exhaustive as possible:
  - 15 bit = 32,768 keys ...
  - … for each key size (1024, 2048, 4096, ...) ...
  - … and each platform (x86,x64,PPC,...)
- Generating single RSA 2048 key takes 1.5 s
  - We need hundreds-of-thousands, and quickly
  - Would take 5 days on single machine
On-Demand Computing

Amazon Web Services:

- Simple Scalable Storage (S3)
  - Web accessible infinite storage
- Elastic Compute Cloud (EC2)
  - Rent Xen 32-bit and 64-bit virtual machines
- Simple Queue Service (SQS)
  - Push and pop messages
Amazon provides VMs, including vulnerable Ubuntu

1. Populate a distributed queue with strings describing which keys to generate
   - “rsa/1024/65537/le32/0-FF”

2. Launch 20 VMs (the default limit)

3. Fetch key descriptors from queue, generate batches of keys, and store in S3
   - “rsa/1024/65537/le32/nornd/2a/2a/985d1c8f20b0d13d25bac1a5673340e5”
Generating 524,288 RSA keys
- 262,144 RSA keys on 20 32-bit instances
  - $0.10 \times 20 \text{ machines} \times 4 \text{ hours} = $8
- 262,144 RSA keys on 20 64-bit instances
  - $0.20 \times 20 \text{ machines} \times 2 \text{ hours} = $8

Decrypting your SSL traffic:
- Priceless
Finding Weak Web Certs

- Started from collections of sites
  - List of all US bank websites
  - Alexa 500, Open Directory
- Small crawl found dozens of weak certificates
- Larger crawl 3 weeks after bug was patched
  - Many newly assigned certificates
  - Thousands of weak certificates
- Weak certs stay valid until expiration date
- Didn’t find weak signing certificate
Attack Demo

DEBIAN RANDOM CAT

IZ NOT RANDOM
Defenses

- Certificate revocation lists
  - Broken/unsupported in almost all browsers
  - Never intended for large numbers of weak certs
- Blacklists
  - Weak 1024/2048-bit keys occupy 460 MB !!
  - Storing hashes often suffices, still 30+ MB
  - Implemented in patched SSH on Debian
    - only checks user/host keys, but not session key!
- Firefox Plug-In: SSL Blacklist
Automated software review is dangerous
- Especially of crypto, which is hard to understand

Security relies on more than strong ciphers

Generating your own randomness is pointless and dangerous

Key management is still major issue
- SSL: revocation big open problem
- SSH: keys often user generated
Questions?

```c
int getRandomNumber()
{
    return 4;  // chosen by fair dice roll.
    // guaranteed to be random.
}
```

Jacob Appelbaum  <jacob@appelbaum.net>
Dino Dai Zovi    <ddz@theta44.org>
Karsten Nohl     <nohl@virginia.edu>