Recent Advances in Single Packet Authorization

Michael Rash
Security Architect
Enterasys Networks, Inc.

http://www.cipherdyne.org/

HOPE Number Nine
NYC, July 2012
Agenda

- Design tradeoffs in PK/SPA systems
- fwknop-2.0
- Security aspects of fwknop development
- SPA in the Amazon Cloud
- The future of Single Packet Authorization
- Demo
PK/SPA Common Goals

• Firewall default-drop stance for protected services
• Passive collection of authentication information
• Firewall policies dynamically reconfigured for temporary authenticated access

• Consequences:
  - Makes scanning for vulnerable services impractical
  - Fundamentally changes the server side exploit model
  - Reduces visible attack surface
Typical Work Flow

- User wants SSH access behind PK/SPA firewall
- User executes PK/SPA client
- Firewall is reconfigured to allow SSH connections from the specified IP
- PK/SPA packet(s) passively monitored
- PK/SPA packet(s) never acknowledged in any way
- SSHD cannot be scanned for
- *Think beyond SSHD*
This is where the similarities in PK/SPA systems end...
About 40 PK/SPA implementations:

http://www.portknocking.org/
fwknop Design Goals

- Firewall default drop stance for protected services
- Passive collection of authentication information
- Support for Symmetric and Asymmetric ciphers
- Encrypted and non-replayable SPA packets
  - Do not want anything that trusts an IP in the network layer header
- Server portable to embedded systems
  - Do not want a heavyweight interpreted language (this is a trade off)
- Server portable to different firewall architectures and router ACL languages
  - Make sophisticated use of NAT
- Client portable to everything from Cygwin to the iPhone
  - Do not want to require raw socket manipulation of packet headers or admin privileges
- Library implementation of SPA protocol for greater portability and integration possibilities
fwknop-2.0

- Completely re-written in C
- fwknopd supports iptables, ipfw, and pf
- SPA protocol library implementation 'libfko'
  - perl and python bindings
- FORCE_NAT mode transparently NAT's authenticated connections
- iPhone and Android clients

$ git clone http://www.cipherdyne.org/git/fwknop.git
fwknop-2.0 Dependencies

$ ldd server/.libs/fwknopd

linux-vdso.so.1 => (0x00007fffe8d98000)
libfko.so.0 => /usr/lib/libfko.so.0 (0x00007f82282850f000)
libpcap.so.0.8 => /usr/lib/libpcap.so.0.8 (0x00007f8228282d8000)
libc.so.6 => /lib/libc.so.6 (0x00007f8227f53000)
libgpgme.so.11 => /usr/lib/libgpgme.so.11 (0x00007f8227d1e000)  ← optional
libgpg-error.so.0 => /lib/libgpg-error.so.0 (0x00007f8227b1a000)  ← optional
/lib64/ld-linux-x86-64.so.2 (0x00007f8228947000)
Old Perl Dependencies

- Digest::SHA
- Net::Pcap
- Crypt::CBC
- GnuPG::Interface
- Unix::Syslog
- Net::IPv4Addr
- MIME::Base64
- IPTables::Parse
- IPTables::ChainMgr
Acquiring SPA Data?

- fwknop runs libpcap + lightweight crypto layer
- Allows design goals to be achieved
- *Every* PK/SPA system must acquire data in *some* way
- This is about attack surface reduction in server-side software – changes the exploit model
- What do exploit frameworks do about sniffers?
Metasploit: Exploitation of pcap-Based Software

- **Snort**
  - Back Orifice preprocessor buffer overflow
  - DCE/RPC preprocessor buffer overflow

- **Wireshark**
  - LWRES dissector stack-based buffer overflow
  - packet-dect.c stack overflow
  - A few others...

- Exploits generally rely on complex code that is layered above libpcap
  - National Vulnerability Database (NVD) searches confirm this

- Network exploitation of non-pcap userspace software requires
  access to talk up the remote networking stack – kernel drivers and
  other kernel code is a different story
Things Are Not Always As They Seem...

- User gains access to NetB from NetA
- Attacker: Which system to attack?
- SPA server can be anywhere on the routing path of an SPA packet – not just the SPA destination IP
- SPA packet source IP can be spoofed too
- Neither the SPA source nor destination IP matters
fwknop: Security-Focused Development
Security Aspects of fwknop Development

- Usage of run time memory checkers (valgrind)
- Usage of static analyzers (splint, wishlist: Coverity – expensive!)
- Usage of compile time security options
- Automated testing
  - Automated function coverage support
  - Automated valgrind usage and flagged function comparisons
- SPA protocol review
- Fuzzing (TODO)
Test Suite

- All major SPA functionality is tested/validated
- Compilation warning checks
- Security aspects of compiled binaries are verified (hardening-check from Kees Cook)
- `--enable-valgrind` mode
- `--diff` mode across test runs
- `fwknop-2.0/test/test-fwknop.pl`
Test Suite:

# ./test-fwknop.pl

[build security] [client] Position Independent Executable (PIE)......pass (3)
[build security] [client] stack protected binary....................pass (4)
[build security] [client] fortify source functions..................pass (5)
[build security] [client] read-only relocations.....................pass (6)
[build security] [client] immediate binding.........................pass (7)
[build security] [server] Position Independent Executable (PIE).....pass (8)
[build security] [server] stack protected binary....................pass (9)
[build security] [server] fortify source functions..................pass (10)
[build security] [server] read-only relocations.....................pass (11)
[build security] [server] immediate binding.........................pass (12)

• This is enabled via:
  - gcc ... -fstack-protector-all -fstack-protector -fPIE -pie -D_FORTIFY_SOURCE=2
    -Wl,-z,relro -Wl,-z,now
Test Suite: Rijndael SPA Cycle

# ./test-fwknop.pl

[Rijndael SPA] [client+server] complete cycle (tcp/22 ssh).........pass (43)

# head output/43_fwknopd.test

Fri May 10 19:01:34 2012 CMD: LD_LIBRARY_PATH=../lib/.libs
../server/.libs/fwknopd -c conf/default_fwknopd.conf -a
conf/default_access.conf -d run/digest.cache -p run/fwknopd.pid -i lo
--foreground --verbose --verbose

process_spa_request() CMD: '/sbin/iptables -t filter -A FWKNOP_INPUT
-p 6 -s 127.0.0.2 --dport 22 -m comment --comment _exp_1328904099 -j
ACCEPT 2>&1' (res: 0, err: )

Added Rule to FWKNOP_INPUT for 127.0.0.2, tcp/22 expires at 1328914099
...
Test Suite: Bug Hunting with Valgrind

• Development cycle becomes:
  
  # ./test-fwknop.pl --enable-valgrind
  
  - Code code code...
  
  # ./test-fwknop.pl --enable-valgrind
  # ./test-fwknop.pl --diff
  
  - Look for new errors reported by valgrind and fix
  
  $ git add ... , git commit
Example: crypto_update Branch

# ./test-fwknop.pl --include "appended" --enable-valgrind

[+] Starting the fwknop test suite...

args: --include appended --enable-valgrind

Saved results from previous run to: output.last/

[Rijndael SPA] [client+server] appended data to SPA pkt............pass (1)
[GnuPG (GPG) SPA] [client+server] appended data to SPA pkt............pass (2)
What Does Valgrind Say?

# ./test-fwknop.pl --diff
+Conditional jump or move depends on uninitialised value(s)
  + at 0x48384D6: rij_decrypt (cipher_funcs.c:263)
  + by 0x483A34A: fko_decrypt_sp_data (fko_encryption.c:158)
  + by 0x483AE9B: fko_new_with_data (fko_funcs.c:210)
  + by 0x10CC29: incoming_sp (incoming_sp.c:245)
  + by 0x10DB40: process_packet (process_packet.c:200)
  + by 0x4861E63: ??? (in /usr/lib/i386-linux-gnu/libpcap.so.1.1.1)
  + by 0x4864667: pcap_dispatch (in /usr/lib/i386-linux-gnu/libpcap.so.1.1.1)
  + by 0x10D607: pcap_capture (pcap_capture.c:223)
  + by 0x10A668: main (fwknopd.c:299)
+ Uninitialised value was created by a heap allocation
  + at 0x482BE68: malloc (in /usr/lib/valgrind/vgpreload_memcheck-x86-linux.so)
  + by 0x483A317: fko_decrypt_sp_data (fko_encryption.c:154)
  + by 0x483AE9B: fko_new_with_data (fko_funcs.c:210)
  + by 0x10CC29: incoming_sp (incoming_sp.c:245)
  + by 0x10DB40: process_packet (process_packet.c:200)
  + by 0x4861E63: ??? (in /usr/lib/i386-linux-gnu/libpcap.so.1.1.1)
  + by 0x4864667: pcap_dispatch (in /usr/lib/i386-linux-gnu/libpcap.so.1.1.1)
  + by 0x10D607: pcap_capture (pcap_capture.c:223)
  + by 0x10A668: main (fwknopd.c:299)
The Fix

diff --git a/lib/fko_encryption.c b/lib/fko_encryption.c
index 5f1788a..af43a87 100644
--- a/lib/fko_encryption.c
+++ b/lib/fko_encryption.c
@@ -139,6 +139,15 @@ _rijndael_decrypt(fko_ctx_t ctx, const char *dec_key, int encryption_mode)

     cipher_len = b64_decode(ctx->encrypted_msg, cipher);

+     /* Since we're using AES, make sure the incoming data is a multiple of
+      * the blocksize
+     */
+     if((cipher_len % RIJNDAEL_BLOCKSIZE) != 0)
+     {
+         free(cipher);
+         return(FKO_ERROR_INVALID_DATA);
+     }

     /* Create a bucket for the plaintext data and decrypt the message
      * data into it.
      */
Coming Soon: HMAC Support

- HMAC-SHA256 coming in fwknop-2.2
  - \( \text{HMAC}(K,m) = H((K \oplus \text{opad}) \parallel H((K \oplus \text{ipad}) \parallel m)) \)
  - SPA encrypted message = \( m \parallel \text{HMAC} \)
  - \( K \neq \) encryption key

- fwknop uses the encrypt-then-authenticate paradigm
  - SSH uses encrypt-and-MAC
  - SSL uses MAC-then-encrypt
  - IPSEC uses encrypt-then-MAC \( \leftarrow \) provably INT-CTXT and IND-CCA2 secure
Why HMAC?

- In encrypt-then-authenticate mode:
  - Protection against things like the Vaudenay attack against SSL:
  - Can ignore bogus (inauthentic) data faster
  - Further reduction in potential attack surface
    - Can discard data *without running any decryption code*
    - libgpgme functions protected by more simplistic HMAC layer

[http://www.daemonology.net/blog/2009-06-24-encrypt-then-mac.html](http://www.daemonology.net/blog/2009-06-24-encrypt-then-mac.html)
Cross-Packet Ciphertext Entropy

- Strategy: generate lots of SPA packets, then measure total entropy at each byte position in slices
- We expect high levels of entropy if the usage of random data and encryption is done properly
- `extras/spa-entropy/spa-entropy.pl`

```
$ ./spa-entropy.pl -f spa_pkts.out -r -c 1000 --base64-decode
```
1,000 SPA Packets - Rijndael CBC Mode

SPA slice entropy (encryption mode: cbc)

min: 7.75 @ byte: 54, max: 7.86 @ byte: 115
1,000 SPA Packets - GnuPG ElGamal Cipher

SPA slice entropy (encryption mode: gpg)

min: 0.00 @ byte: 1, max: 7.86 @ byte: 368
How Good is /dev/urandom?

$ dd if=/dev/urandom count=1000 | ent
1000+0 records in
1000+0 records out
512000 bytes (512 kB) copied, 0.128497 s, 4.0 MB/s

**Entropy = 7.999625 bits per byte.**

Optimum compression would reduce the size of this 512000 byte file by 0 percent.

Chi square distribution for 512000 samples is 265.77, and randomly would exceed this value 50.00 percent of the times.

Arithmetic mean value of data bytes is 127.5076 (127.5 = random).
Monte Carlo value for Pi is 3.138715386 (error 0.09 percent).
Serial correlation coefficient is -0.001293 (totally uncorrelated = 0.0).
SPA in the Amazon Cloud

http://aws.amazon.com/
Amazon Web Services (AWS)

- AWS provides massive infrastructure for cheap on-demand costs

- Notable usages of AWS:
  - 42nd fastest supercomputer built in EC2:
    http://www.wired.com/wiredenterprise/2011/12/nonexistent-supercomputer/
  - Debian OpenSSL key debacle:
    http://trailofbits.files.wordpress.com/2008/07/hope-08-openssl.pdf

- We deploy SPA on Elastic Compute Cloud (EC2) and Virtual Private Cloud (VPC) networks
Amazon VPC Networks
The Perfect SPA Use Case

- Microsoft RDP vulnerability earlier this year (CVE-2012-0002)
- Full remote code execution potential, although Metasploit has a DoS module
- Problem: fwknop does not support a Windows firewall
SPA + NAT = Secure RDP Access

- Use an internal Ubuntu AWS image as a jump host
- fwknopd is deployed on the Ubuntu system

- **Any** VPC system is accessible through the Ubuntu system via SPA + NAT

- Only one Amazon Elastic IP is required
  - Changes the normal Amazon NAT+Elastic IP association model

- iptables+SPA extends Amazon's filtering capabilities – SPA not integrated into AWS border controls
### VPC Filtering Policy

The AWS Management Console shows a VPC Security Group with two security groups: default and ssh-access. The ssh-access group allows SSH access with specific port ranges.

**Security Group: ssh-access**

<table>
<thead>
<tr>
<th>Details</th>
<th>Inbound</th>
<th>Outbound</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Create a new rule:</strong></td>
<td><strong>Source</strong></td>
<td><strong>Action</strong></td>
</tr>
<tr>
<td>Custom TCP rule</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Port range:</strong></td>
<td><strong>Source</strong></td>
<td><strong>Action</strong></td>
</tr>
<tr>
<td>(e.g., 80 or 49152-65535)</td>
<td></td>
<td>Delete</td>
</tr>
<tr>
<td><strong>Source:</strong></td>
<td><strong>Port (Service)</strong></td>
<td><strong>Source</strong></td>
</tr>
<tr>
<td>0.0.0.0/0</td>
<td>ALL</td>
<td>10.0.0.0/16</td>
</tr>
<tr>
<td>(e.g., 192.168.2.0/24, sp-47a482e, or 12345/67890/default)</td>
<td>ALL</td>
<td>170/32</td>
</tr>
<tr>
<td><strong>TCP Port (Service)</strong></td>
<td><strong>Source</strong></td>
<td><strong>Action</strong></td>
</tr>
<tr>
<td>22 (SSH)</td>
<td></td>
<td>Delete</td>
</tr>
<tr>
<td>23 (TELNET)</td>
<td></td>
<td>Delete</td>
</tr>
<tr>
<td>80 (HTTP)</td>
<td></td>
<td>Delete</td>
</tr>
<tr>
<td>3389 (RDP)</td>
<td></td>
<td>Delete</td>
</tr>
<tr>
<td>12345</td>
<td></td>
<td>Delete</td>
</tr>
<tr>
<td><strong>UDP</strong></td>
<td></td>
<td>Delete</td>
</tr>
</tbody>
</table>
SPA + NAT = RDP Access
fwknop Client Command Line

- Ubuntu IP: 10.0.0.171
- Windows Server IP: 10.0.0.79
- External Elastic IP: 107.21.55.55

$ fwknop -A tcp/80 -N 10.0.0.79,3389 -R -D 107.21.55.55 --server-port 53

$ rdesktop -u Administrator 107.21.55.55:80
fwknopd Configuration

• Need NAT to work through the Ubuntu system, so in `fwknopd.conf`:

  ```
  ENABLE_IPT_FORWARDING    Y;
  ```

• The Windows host is not associated with an Elastic IP, so we want return traffic to go back through the Ubuntu host

• The Windows host only sees an RDP connection from the Ubuntu host – not from its true source over the Internet

  ```
  ENABLE_IPT_SNAT    Y;
  SNAT_TRANSLATE_IP  <ubuntu IP>
  ```
SPA NAT Access to RDP
The Future of Single Packet Authorization
The Future of SPA

- Mandatory Access Control support via SELinux and/or AppArmor
- Supported iPhone client (we are looking for a maintainer – please email me if interested)
- Further cloud computing extensions and integration points
- Packed binary protocol
- Tunneling mode extensions (DNS, HTTP, SMTP, Tor)
iPhone + Android fwknop Clients
Demo...
Linux Firewalls 2nd Edition
Input Please...
Questions?

mbr@cipherdyne.org

http://www.cipherdyne.org/fwknop/