Single Packet Authorization

Michael Rash
Senior Security Architect
G2, Inc.

http://www.g2-inc.com/
http://www.cipherdyne.org/

DojoCon
2009.11.06
Agenda

- Single Packet Authorization
  - The basics
  - Community status
- Introducing the SPA C implementation
  - fwknop-c client, upcoming fwknop-c server, and the libfko library
  - Development strategy
  - Supported operating systems
- Advanced topics
  - SPA through HTTP proxies
  - Port randomization for both the SPA packet and NAT’d services
  - Creating “ghost” services with SPA
- Live demo
The Basics...

- Service protection behind a default-drop packet filter. Anyone scanning for such a service cannot even see that it is listening – let alone exploit a vulnerability or brute-force a password in the protected service.*

- Access granted only after passively collected information is verified.

- SPA is next-generation port knocking, with strong encryption and non-replayability.

* This is not to say that the firewall itself or the packet collection mechanism has no vulnerabilities.
SPA Network Architecture

- SPA packet
- SSH/HTTP

11.1.1.1

Internet

SPA packet

22.2.2.2

SSH server

HTTP server
Why Not Just Look for Brute Force Password Guessing Attempts?

- DenyHosts, fail2ban, custom log parsers, etc...
- “Relay Server Tactic Dupes Auto-Reporting”
  - http://www.theregister.co.uk/2008/07/14/brute_force_ssh_attack/
- Exploits commonly have nothing to do with guessing a weak password (Debian OpenSSL vuln, overflow vulnerabilities from time to time)... and this is only SSH.
From PK to SPA

• Gain access to sshd after:
  • Single packet to port 12345 (nmap or a web browser can function as a PK client).
  • Or... multiple packets to a sequence of pre-defined ports (generally need a custom client unless the sequence is < 3 ports long).
  • Or... multiple packets form an encrypted sequence with a shared key (really need a custom client).
  • Or... a single packet with appropriately built application layer data (this is SPA).
PK vs. SPA – Complexity vs. Protocol Limitations
SPA and the Security Community

- fwknop downloads (all versions):
  - 2006: 2,768
  - 2007: 6,976
  - 2008: 18,292 (9 software releases)
  - 2009: 10,503 (so far this year with 3 software releases – significant development time devoted to libfko)
User Contributions

- The big one: libfko + C client/server + FKO perl bindings (Damien Stuart)
- morpheus-fwknop UI (Daniel Lopez)
- HTTP proxy support (Jonathan Bennett)
- ipfw 'sets' support (Julien Picalaus)
- iptables cross-connection persistence (Martin Tan)
- ssh-fwknop (Richard Lundeen – Google Code project)
The fwknop-1.9.12 release

- Uses the FKO perl module by default.
- Has the ability to recover from interface outage and admin down/up cycles – useful when fwknopd is deployed in conjunction with DHCP or ppp end points.
- HTTP proxy support.
- Can acquire SPA packets via UDP or TCP sockets directly – no libpcap required in either of these modes.

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

- Over 30 total port knocking and/or SPA implementations ([http://www.portknocking.org/](http://www.portknocking.org/)).
- Each with a slight variation on PK or SPA, though few are regularly updated except for fwknop (which has 36 releases since 2004).
- The most interesting competing implementation is *knockknock* by Moxie Marlinspike.
Trends?

- SPA usage is up, but widespread deployment has a long way to go.
- A modifier will be efforts to package SPA software for various platforms, and efforts to support different firewalls and/or router ACL's.
- People still concentrate on detection of SPA vs. exploitation of SPA.
- Open question: To what extent are PK/SPA techniques used by the blackhat community or in botnets? ...This would make a great topic for a research paper.
Old SPA Man-In-The-Middle Attack

- Given that people concentrate on detection, it's only fair to present an attack as well.
- fwknop has *not* been vulnerable since 2006.
- It would be interesting to determine which other SPA implementations are also vulnerable to this.
fwknop-c + libfko

- libfko is a C library that third party applications can link against in order to implement the SPA protocol.
- Simplifies the implementation of both SPA client and server applications.
- Small footprint brings SPA to embedded systems that have limited resources (e.g. OpenWRT on a small router), and to systems where there is no perl interpreter and no compiler installed.
- FKO perl bindings already exist, with other language support planned.
libfko

- Supports Linux, FreeBSD, Mac OS X, Solaris, and Windows.
- The SPA packet format is built by libfko functions via an SPA-context data structure.
- Depends on gpgme for GunPG SPA operations.
- SPA packet format:

  random_data:user:timestamp:version:mode:access_str:internal_digest

  4070524269054661:root:1257137439:1.9.12:1:127.0.0.2,tcp/22:-
  1:0ey4FayNQIUSnS0qL5q4EMYaOWIXGSVODbtXQ2EQUas
libfko API

SPA packet data is built from a series of get_* and set_* functions:

- DLL_API int fko_set_rand_value(fko_ctx_t ctx, const char *val);
- DLL_API int fko_set_username(fko_ctx_t ctx, const char *spoof_user);
- DLL_API int fko_set_timestamp(fko_ctx_t ctx, int offset);
- DLL_API int fko_set_spa_message_type(fko_ctx_t ctx, short msg_type);
- DLL_API int fko_set_spa_message(fko_ctx_t ctx, const char *msg_string);
- DLL_API int fko_set_spa_nat_access(fko_ctx_t ctx, const char *nat_access);

- Once the SPA packet data is built, the client sends it out on the wire based on transmission needs (UDP vs. other socket type, auto-resolution of external NAT address of local network, etc.).
FKO perl module

#!/usr/bin/perl -w

use FKO;

my $fko = FKO->new();

my $err = $fko->spa_message('0.0.0.0,tcp/22');

### error checking...

$err = $fko->spa_data_final();

my $spa_data = $fko->spa_data();

### send over UDP socket...

exit $err;
fwknop-c client/server

- The fwknop-c client is finished, and passes the fwknop test suite.
- The server is currently in development – will depend on libpcap, and the tricky part is handling the underlying firewall interface. The perl version depends on IPTables::Parse and IPTables::ChainMgr.
- Will be highly portable considering where libfko already runs.
morpheus-fwknop UI
Advanced Topics + Live Demos...
Example 1: SPA over an HTTP proxy

- Requires a relaxation of the “single” part of SPA.
- Need to be able to set HTTP headers such that a proxy (such as Squid) recognizes where the SPA/HTTP request goes.
- The fwknop client builds an HTTP request with a leading '/', and the remainder is normal base64 encoded SPA data.
- Follow-on connections are made as usual.
Example 2: Port Randomization

• SPA destination port is randomized AND the service port itself is randomized (with NAT rules building the appropriate access).

• Essentially asking to access a service via a non-standard port.

• To an observer, difficult to identify what is going on without looking at every packet – no correspondence between connections and “expected” port numbers.

• Live demo...
Example 3: Creating a “Ghost” Service with SPA

• On the server side, and service can be offered over a port which fwknopd co-opts for other access for your source IP.

• Example: the server can be running a webserver on port 80, but NAT'd access to sshd can be requested through port 80 for the SPA client IP. Everyone else always just sees the HTTP server.

• Live demo...
Conclusions

• The security community is gradually embracing SPA in some cases, but there is a long way to go.

• Full fwknop-c server support is on the way, and client support exists today. OpenWRT server support will not be far behind.

• Effective NAT integration implies advantages in the face of attackers armed with packet sniffers.
Questions?

http://www.g2-inc.com/
http://www.cipherdyne.org/

michael.rash@g2-inc.com
mbr@cipherdyne.org