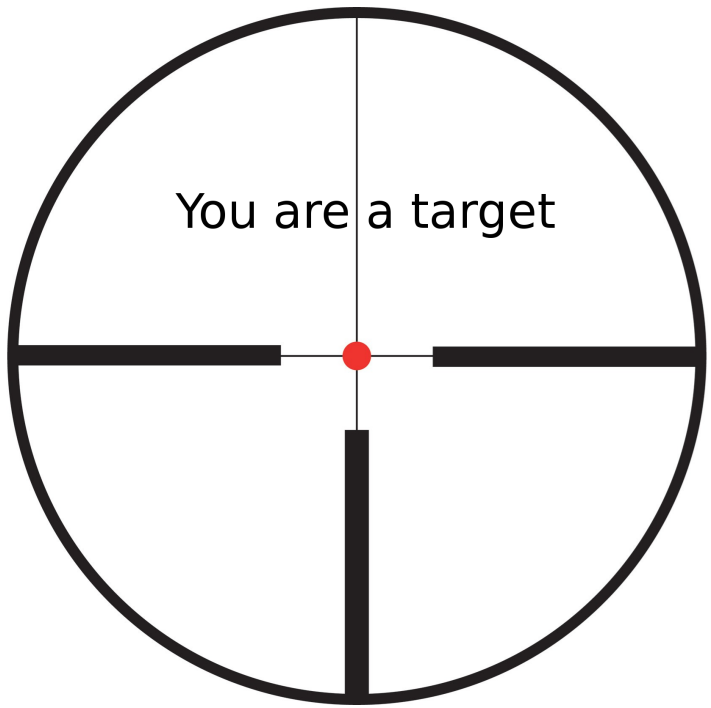
Our Current Threat Environment

Leif Nixon

October 7, 2013



The New York Times

Internet Attack Called Broad and Long Lasting by Investigators

SAN FRANCISCO, May 9 – The incident seemed alarming enough: a breach of a Cisco Systems network in which an intruder seized programming instructions for many of the computers that control the flow of the Internet.

Now federal officials and computer security investigators have acknowledged that the Cisco break-in last year was only part of a more extensive operation – involving a single intruder or a small band, apparently based in Europe – in which thousands of computer systems were similarly penetrated. […]

Attention is focused on a 16-year-old in Uppsala, Sweden. […]

In this wave of intrusions, many, many systems were compromised; supercomputers, military systems, private industry systems, universities…

Attacking sites the Stakkato way

1. Gain access to the site through stolen account.

Attacking sites the Stakkato way

1. Gain access to the site through stolen account.
2. Disable command line history by doing `unset HISTFILE`.

Attacking sites the Stakkato way

1. Gain access to the site through stolen account.
2. Disable command line history by doing `unset HISTFILE`.
3. Map the site:
   - Identify administrative users and groups. For example, is there a `/sw` filesystem with site-wide software, owned by a particular user or group?

Attacking sites the Stakkato way

1. Gain access to the site through stolen account.
2. Disable command line history by doing `unset HISTFILE`.
3. Map the site:
   - Identify administrative users and groups. For example, is there a `/sw` filesystem with site-wide software, owned by a particular user or group?
   - Check `/etc/hosts` and `.ssh/known_hosts`. Locate NFS servers.

Attacking sites the Stakkato way

1. Gain access to the site through stolen account.
2. Disable command line history by doing `unset HISTFILE`.
3. Map the site:
   - Identify administrative users and groups. For example, is there a `/sw` filesystem with site-wide software, owned by a particular user or group?
   - Check `/etc/hosts` and `.ssh/known_hosts`. Locate NFS servers.
   - Use `showmount` to find NFS clients. (Just cut'n'paste your standard `awk` command line to extract the hostnames into a temporary file.)

Attacking sites the Stakkato way

1. Gain access to the site through stolen account.
2. Disable command line history by doing `unset HISTFILE`.
3. Map the site:
   - Identify administrative users and groups. For example, is there a `/sw` filesystem with site-wide software, owned by a particular user or group?
   - Check `/etc/hosts` and `.ssh/known_hosts`. Locate NFS servers.
   - Use `showmount` to find NFS clients. (Just cut'n'paste your standard `awk` command line to extract the hostnames into a temporary file.)
   - Loop over hosts (cut'n'paste your standard bash `for` loop) and try to log in on them using your stolen account. Save output from `w` and `uname -a` in a temp file.

Attacking sites the Stakkato way

1. Gain access to the site through stolen account.
2. Disable command line history by doing `unset HISTFILE`.
3. Map the site:
   - Identify administrative users and groups. For example, is there a `/sw` filesystem with site-wide software, owned by a particular user or group?
   - Check `/etc/hosts` and `.ssh/known_hosts`. Locate NFS servers.
   - Use `showmount` to find NFS clients. (Just cut'n'paste your standard `awk` command line to extract the hostnames into a temporary file.)
   - Loop over hosts (cut'n'paste your standard bash `for` loop) and try to log in on them using your stolen account. Save output from `w` and `uname -a` in a temp file.
   - Use this data to find potential targets. Easiest target: Linux machine with unpatched kernel. Otherwise: use toolbox of standard exploits for Linux, Solaris, Irix or AIX machines. There is bound to be an old forgotten NFS client machine *somewhere*…

Attacking sites the Stakkato way

4. Acquire root on one of your target machines. Use `/tmp/...` as discreet working directory for compiling exploits, etc.

Attacking sites the Stakkato way

4. Acquire root on one of your target machines. Use `/tmp/.../` as discreet working directory for compiling exploits, etc.
5. At this point there are several ways to proceed, depending on the site configuration:
   - NFS filesystem mounted without root squashing, and without noexec/nosuid flags? Jackpot! Hide suid shell deep in directory hierarchy. Instant root access everywhere!

Attacking sites the Stakkato way

4. Acquire root on one of your target machines. Use `/tmp/.../` as discreet working directory for compiling exploits, etc.
5. At this point there are several ways to proceed, depending on the site configuration:
   - NFS filesystem mounted without root squashing, and without noexec/nosuid flags? Jackpot! Hide suid shell deep in directory hierarchy. Instant root access everywhere!
   - Otherwise, `su` to an administrative user and see if you can modify the site software, perhaps deploy your trusty ssh trojan.

Attacking sites the Stakkato way

4. Acquire root on one of your target machines. Use `/tmp/.../` as discreet working directory for compiling exploits, etc.
5. At this point there are several ways to proceed, depending on the site configuration:
   - NFS filesystem mounted without root squashing, and without noexec/nosuid flags? Jackpot! Hide suid shell deep in directory hierarchy. Instant root access everywhere!
   - Otherwise, `su` to an administrative user and see if you can modify the site software, perhaps deploy your trusty ssh trojan.
   - If that doesn't work, drop your ssh trojan into a user's home directory. (Change `.bash_profile` to put it first in `$PATH` if necessary.) Target an administrative user if possible – this may be a goldmine for root passwords.

Attacking sites the Stakkato way

4. Acquire root on one of your target machines. Use `/tmp/.../` as discreet working directory for compiling exploits, etc.
5. At this point there are several ways to proceed, depending on the site configuration:
   - NFS filesystem mounted without root squashing, and without noexec/nosuid flags? Jackpot! Hide suid shell deep in directory hierarchy. Instant root access everywhere!
   - Otherwise, `su` to an administrative user and see if you can modify the site software, perhaps deploy your trusty ssh trojan.
   - If that doesn't work, drop your ssh trojan into a user's home directory. (Change `.bash_profile` to put it first in `$PATH` if necessary.) Target an administrative user if possible – this may be a goldmine for root passwords.
6. Consider deploying the Suckit rootkit on Linux machines – snoops all entered passwords and provides a stealthy backdoor for remote root access.

Attacking sites the Stakkato way

…and then start all over again.

Attacking sites the Stakkato way

…and then start all over again.

Over 18 months, more than 1000 sites compromised, causing damage worth millions.

16-yo convicted for six cases of data intrusion. Suspended sentence because of age, plus a couple of EUR 10000 in damages.

The Stakkato intrusions

When did this happen?

The Stakkato intrusions

When did this happen?

2003–2005

The Titan incident – It starts with an IM...

Bellman (OoD): `NORDUNETTICKET-1253: "For security reasons, a subnet belonging to the NDGF facility in Norway has been closed down." Hmm...`

. . . and continues with an email

We may have found a modified sshd binary on one of Titan's login nodes. Not sure yet.

Determining the situation

- At least three hosts rooted; `login0`, `login1` and `master`
- Trojan ssh/sshd logging passwords to `/usr/share/kbd/keymaps/i386/azerty/c1`
- Grid attached systems apparently unharmed. (Phew.)

We can assume other sites serving the same user community are compromised

My goal: go after the intruders as far as possible.

Battlefield forensics

I received copies of the ssh binaries, and started looking for interesting strings.

Usually, the strings will be obfuscated by xor:ing with a single byte.

Not in this case; apparently something slightly more clever was used.

Battlefield forensics

Running the sshd under `strace` and `ltrace` in a sandbox showed what was going on, and revealed a potential backdoor root password:

`.ssh/authorized_keys2␣␣`

Battlefield forensics

```
$ ssh root@login0.titan.uio.no
root@login0.titan.uio.no's password:
Last login: Wed Aug 10 09:32:05 2011 from master.titan.uio.no
**************************************************************
*                                                            *
* Research Computing Services, University of Oslo, TITAN     *
*  ---------------------------------------------------       *
**************************************************************
login-0-0.local#
```

Battlefield forensics 2

I was going to mess with the systems remotely, so images were made of the system disks.

I dumped in a copy of The Sleuth Kit and started looking at timelines.

Battlefield forensics 2

Jun 23 22:14:10 Prerequisite devel rpms installed through yum on login0, trojan openssh compiled

Jun 23 22:17:46 Trojan openssh installed

Jun 24 00:59:51 Same yum and compilation operations are performed on login1

Jul 15 23:33:47 Same yum and compilation operations are performed on master

Battlefield forensics 2

Jun 23 22:14:10 Prerequisite devel rpms installed through yum on login0, trojan openssh compiled

Jun 23 22:17:46 Trojan openssh installed

Jun 24 00:59:51 Same yum and compilation operations are performed on login1

Jul 15 23:33:47 Same yum and compilation operations are performed on master

But how did they get root?

Battlefield forensics 2

One weird thing stood out:

```
Thu Jun 23 22:08:14 37280 ..c.  r/rrwsr-xr-x root root 6684690
/bin/ping
```

The ctime on `/bin/ping` was updated just as the intruder started running things as root. Ping is setuid root – perhaps a backdoor was installed? But the binary seemed intact. Strange.

Why has ping been messed with?

Popular CVE-2010-3847 exploit:

```
$ mkdir /tmp/exploit
$ ln /bin/ping /tmp/exploit/target
$ exec 3< /tmp/exploit/target
$ rm -rf /tmp/exploit/
$ gcc -w -fPIC -shared -o /tmp/exploit payload.c
$ LD_AUDIT="\$ORIGIN" exec /proc/self/fd/3
sh-4.1# whoami
root
```

Elementary, my dear Watson

1. Making a hard link to the ping binary will update its ctime.
2. The system turned out to be vulnerable to CVE-2010-3847.

Conclusion: it's a good guess that this was how the system was rooted.

Tracing backwards

System logs had been tampered with, but by combining flow logs and the remaining system logs, we could identify an account belonging to a user from a European astrophysics facility as the likely source of the intrusion.

Tracing backwards

After establishing contact with the astrophysics facility, we found they were in a rather bad shape.

Their department network had lots of rooted machines with ssh trojans.

There were also rooted machines at their experiment site.

Tracing backwards

I worked with them to identify more victims, and we could find several more potentially compromised sites.

In the end, we found 3 or 4 big astrophysics sites across the world with compromised systems, before the incident disappeared over the horizon.

The Titan incident

When did this happen?

The Titan incident

When did this happen?

2011

EGI-20111231-01

Intrusions across Poland, Norway, Netherlands, Korea, Japan, Germany.

Replaced ssh binaries, password theft. Many, many compromised systems, including Dutch telecom giant KPN.

Dutch perpetrator finally caught.

EGI-20111231-01

Intrusions across Poland, Norway, Netherlands, Korea, Japan, Germany.

Replaced ssh binaries, password theft. Many, many compromised systems, including Dutch telecom giant KPN.

Dutch perpetrator finally caught. 16-year-old kid.

Nothing ever changes

90% of incidents in our community are because of stolen or weak ssh credentials.

Root escalations are almost always due to known security holes for which patches are available[1].

If we could improve these two factors, we would be in a much better shape.

[1] or because of stolen root passwords

Who?

The attackers generally fall into one of these categories:

- **Bounty hunters –** Are in it for the lulz and the bragging rights

Who?

The attackers generally fall into one of these categories:

- **Bounty hunters –** Are in it for the lulz and the bragging rights
- **Resource hoarders –** Want to use network, storage or CPU capacity for their own purposes; hosting pirated content, performing dDOS attacks, sending spam or mine bitcoins

Who?

The attackers generally fall into one of these categories:

- **Bounty hunters –** Are in it for the lulz and the bragging rights
- **Resource hoarders –** Want to use network, storage or CPU capacity for their own purposes; hosting pirated content, performing dDOS attacks, sending spam or mine bitcoins
- **Hacktivists –** Want to achieve political or ideological goals

Who?

The attackers generally fall into one of these categories:

- **Bounty hunters –** Are in it for the lulz and the bragging rights
- **Resource hoarders –** Want to use network, storage or CPU capacity for their own purposes; hosting pirated content, performing dDOS attacks, sending spam or mine bitcoins
- **Hacktivists –** Want to achieve political or ideological goals
- **Nation states –** ?

Conclusion

Our main threat is a sixteen-year-old kid using tactics from the previous decade.

We *should* be able to meet this.