Live Solaris Evidence Gathering Instructions

May 1, 2006

Document Name:	solaris_evidence_gathering_v1.2.doc
Version:	V 1.2
Author(s):	Mark Furner, Compass Security AG
	Ivan Buetler, Compass Security AG
Date of Delivery:	May 1, 2006
Classification:	PUBLIC

GLÄRNISCHSTR. 7
POSTFACH 1671
CH-8640 RAPPERSWIL

Tel. +41 55-214 41 60
Fax +41 55-214 41 61
team@csnc.ch www.csnc.ch

Table of Contents

GLÄRNISCHSTR. 7
POSTFACH 1671
CH-8640 RAPPERSWIL

Tel. +41 55-214 41 60
Fax +41 55-214 41 61
team@csnc.ch www.csnc.ch

1 Solaris Investigation

1.1 Introduction

Having been commissioned to investigate a Solaris system on a number of occasions, Compass Security AG has decided to write a generic guide to live evidence gathering and imaging specific to Sun Solaris systems. This does not cover the analysis phase of an investigation.

Three phases are involved in evidence gathering: preparation, gathering of live / volatile evidence and gathering of non-volatile evidence. Disk imaging is categorized as non-volatile evidence, although this is not the case if the disk is left on-line, as when imaging critical systems. These phases are then followed by the analysis of the evidence. This advisory paper does not cover storage area networks (SANs) or distributed files systems. Where the size of a SAN is an issue for evidence collection, Compass Security advises imaging individual partitions, filesystems or shares where feasible. This should allow the capture of digital evidence fragments left on a filesystem.

A general principle is to cause as little alteration as possible to the file systems that will be imaged prior to imaging. Some tools (such as “find” or “chkrootkit”) alter files’ access times and other metadata that may destroy evidence of an attacker’s activities. For this reason, the investigation first gathers information in RAM, followed by imaging of the host’s file system. File system investigations are then carried out on a copy of the image.

1.2 To the Reader

This document is geared towards security teams, and individuals concerned with investigating Solaris systems. The purpose of this document is to describe in detail the initial steps of evidence-gathering to forensic standards.

1.3 Document Structure

Chapter	Content
1	Introduction and Description of Document
2	Preparatory Steps
3	Evidence Gathering
App. 1	Compass Evidence Script

GLÄRNISCHSTR. 7
POSTFACH 1671
CH-8640 RAPPERSWIL

Tel. +41 55-214 41 60
Fax +41 55-214 41 61
team@csnc.ch www.csnc.ch

1.4 Version Control

Version	Date	Changes	Author
0.1	07.09.2005	First Version	Mark Furner
0.2	13.09.2005	Added script	Mark Furner
1.01	12.12.2005	Edits	Mark Furner
1.02	17.01.2006	Edits	Mark Furner
1.03	31.01.2006	Review	Ivan Bütler
1.04	01.02.2006	Edits, added network bridge subsection	Mark Furner
1.05	27.02.2006	Review, comments	Ivan Bütler
1.06	27.02.2006	Edits, restructure and add nmap	Mark Furner
1.07	24.04.2006	Edits, some visual corrections, script update	Mark Furner
1.08	16.05.2006	Added feedback remarks on encryption etc	Mark Furner
1.1	24.05.2006	New script version (test form with truss)	Mark Furner
1.11	07.06.2006	Minor edits	Mark Furner

GLÄRNISCHSTR. 7
POSTFACH 1671
CH-8640 RAPPERSWIL

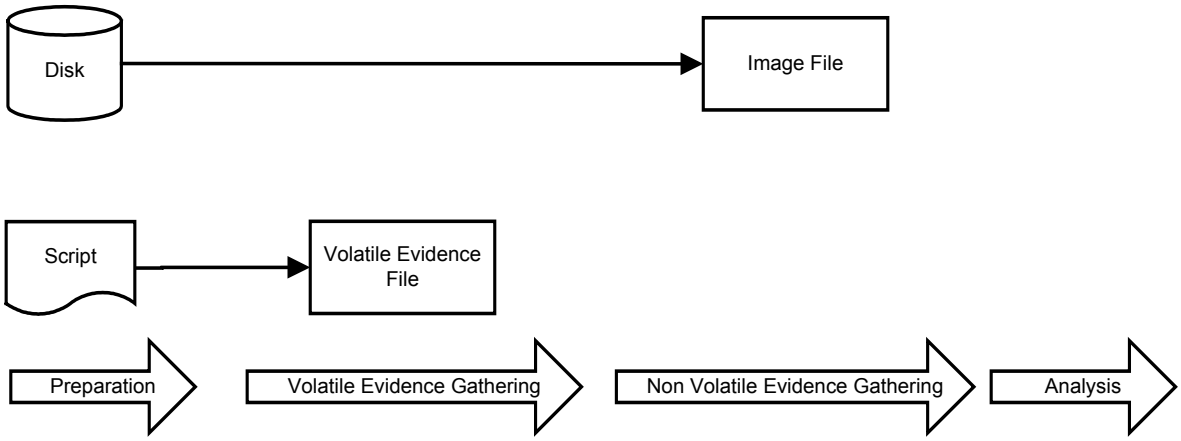
Tel. +41 55-214 41 60
Fax +41 55-214 41 61
team@csnc.ch www.csnc.ch

1.5 Suggested Procedure and Guiding Principles

Digital evidence is extremely vulnerable to alteration or tampering, whether deliberate or unintentional; it is volatile to varying degrees. It therefore requires special procedures to protect evidence from changes during and after gathering. Checksums are made of evidence files, so that the integrity of the evidence data may later be confirmed. According to RFC 3227,[1] evidence should be gathered in the order of volatility, which requires prioritizing evidence so that, for example, evidence in RAM is gathered at the earliest opportunity. This guide will offer a practical implementation of RFC 3227 for gathering evidence on Solaris systems.

Activities during each stage should be meticulously documented to ensure that each action can be accounted for at a later date. Forms for digital evidence handling (chain of evidence) are useful here, as are log books.

Evidence gathering should take place in three stages: preparation, gathering of volatile evidence, followed by gathering of non-volatile or less volatile evidence. These are the procedural steps we discuss in this document. An overview of the process is shown in the diagram below:



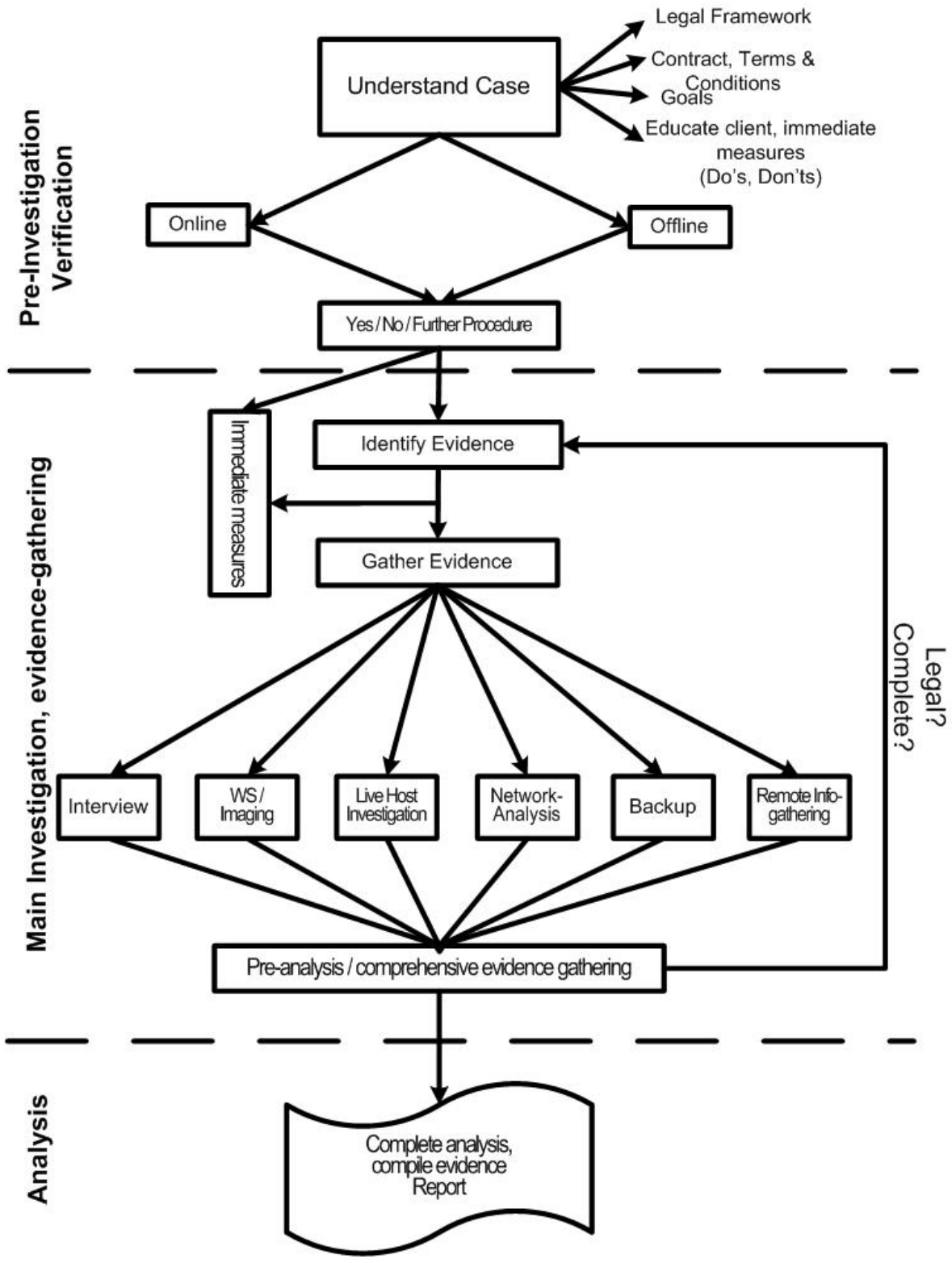
After the evidence has been gathered, a full analysis can proceed. However during the evidence-gathering stage it is often possible to gain preliminary results that can give hints to further evidence or give provisional results. This is a heuristic (learning) cycle that typically continues until the investigating team considers that enough evidence for the task at hand has been gathered and is restricted by a time window given for investigative activities.

A general overview of the investigations cycle is offered in the diagram on the following page. The heuristic loop is represented in the stages of identifying evidence, gathering evidence (with various possible methods) and pre-analysis.

[1] Guidelines for Evidence Collection and Archiving, D. Brezinski and T. Killalea; http://www.ietf.org/rfc/rfc3227.txt.

GLÄRNISCHSTR. 7
POSTFACH 1671
CH-8640 RAPPERSWIL

Tel. +41 55-214 41 60
Fax +41 55-214 41 61
team@csnc.ch www.csnc.ch



GLÄRNISCHSTR. 7
POSTFACH 1671
CH-8640 RAPPERSWIL

Tel. +41 55-214 41 60
Fax +41 55-214 41 61
team@csnc.ch www.csnc.ch

2 Preparation

2.1 Prepare Evidence Storage Media

Compass Security uses Linux-formatted disks to store evidence since our evidence collection procedures and analysis prefers Linux tools. For investigations of UFS or proprietary file systems it may be necessary to use other file systems. Evidence disks are first entirely overwritten with zeroes, and then partitioned and formatted with a journalled file system with large file support, for greater stability. Every access and use of these disks is documented using chain-of-evidence forms that detail who used the disks when and for what.

First we prepare the disk.

Goal	Prepare storage disk for evidence
Preparation	Wipe and format a hard drive

Tasks

No	Description of Test	Expected Result	Actual Result	PASS FAIL
1	Overwrite entire disk with zeroes	Disk filled with '0'		
2	Format disk: create 1 partition	1 partition		
3	Create file system with large file support	File system ready to receive image		

Details Concerning No. 1 and 2

Format the disk with one partition (premise: first IDE cable, master). As a general experience, the zeroing process takes approximately 1 hour for 60GB using an IDE cable and "DMA" access mode. Under Linux steps 1 and 2 could look like this:

```
[Zero the disk]
dd if=/dev/zero of=/dev/hda bs=1M

[Format the disk]
fdisk /dev/hda

[New partition] n
[Primary partition] p
[Partition number] 1
[First sector: default] [enter]
[Last sector: default] [enter]
[Partition type] t
[Partition type number] 83
[Verify] v
[Write to disk] w
```

GLÄRNISCHSTR. 7
POSTFACH 1671
CH-8640 RAPPERSWIL

Tel. +41 55-214 41 60
Fax +41 55-214 41 61
team@csnc.ch www.csnc.ch

Under Solaris the dd command is similar but there is no "/dev/zero" in Solaris and the device name is different. For example /dev/rdsk/c0t0d0s2 refers to the third partition of the first disk on the first controller, which spans the whole disk and is used for backup, "t" refers to "target", and is specific to SCSI or fibre-channel disks.

```
[Zero the disk]
dd if=00 of=/dev/rdsk/c0t0d1s2 bs=1M

[Format the disk]
format

[Select the disk from the selection]
1
[Selection:
we want a large data partition for data, e.g. partition 0, usually reserved for the root
file system]
partition
format
label

[quit the format application]
```

Details Concerning No. 3

For Linux we install an Ext3 file-system with large file support and journal onto the newly created partition. This should not take longer than a minute for a large drive (e.g. 200GB).

```
mkfs.ext3 -b 4096 -j -T largefile /dev/hda1
```

Solaris: we install a UFS2 file system onto the root partition.

```
newfs /dev/rdsk/c0t0d1s0
```

GLÄRNISCHSTR. 7
POSTFACH 1671
CH-8640 RAPPERSWIL

Tel. +41 55-214 41 60
Fax +41 55-214 41 61
team@csnc.ch www.csnc.ch

2.2 Gather Trusted and Tested Binaries

In general incident response scenarios, it is advisable to use trusted programs to gather evidence: programs that have been gathered from a “clean” system with same OS and patch level as the system to be investigated. If a root-kit or backdoor has been installed, software already installed on the server may have been compromised, or altered to hide processes. Using these programs cannot guarantee “clean” results; “trusted binaries” can mitigate the problem and are therefore considered good practice. Compass Security recommends the use of statically compiled software where possible (ie. programs that do not refer to other files that might also be compromised), gathered from a trusted system. These “trusted binaries” should be run from an external CDROM or USB stick/USB disk.

Solaris programs can be tested using “truss” to see which libraries the programs rely on.

```
truss -topen /cdrom/forensics/usr/bin/netstat -an
open("/var/ld/ld.config", O_RDONLY)             = 3
open("/usr/lib/libdhcpagent.so.1", O_RDONLY)    = 3
open("/usr/lib/libcmd.so.1", O_RDONLY)          = 3
open("/usr/lib/libsocket.so.1", O_RDONLY)       = 3
open("/usr/lib/libnsl.so.1", O_RDONLY)          = 3
open("/usr/lib/libkstat.so.1", O_RDONLY)        = 3
open("/usr/lib/libc.so.1", O_RDONLY)            = 3
open("/usr/lib/libdl.so.1", O_RDONLY)           = 3
open("/usr/lib/libmp.so.2", O_RDONLY)           = 3
open("/usr/platform/SUNW,UltraAX-i2/lib/libc_psr.so.1", O_RDONLY) = 3
open("/etc/default/inet_type", O_RDONLY)        Err#2 ENOENT
open("/dev/kstat", O_RDONLY)                    = 3
open("/dev/ip", O_RDWR)                         Err#13 EACCES
ip open: Permission denied
can't open mib stream: Bad file number
```

Solaris is a difficult platform for static compilation, and it is not always feasible to create static binaries from source. Where this is the case, library files should be documented and included on the CDROM or storage media used. Testing with truss will establish which files still require access to the host’s file system. This remaining file system access should be clearly documented for each program run so that this “external” influence on the host’s file system can be taken into account during analysis. In the example below (details concerning task no. 1), most library files were successfully bound into the bash shell / CDROM but those that were not should be carefully documented.

GLÄRNISCHSTR. 7
POSTFACH 1671
CH-8640 RAPPERSWIL

Tel. +41 55-214 41 60
Fax +41 55-214 41 61
team@csnc.ch www.csnc.ch

Goal	Gather the programs needed later to capture process information
Preparation	Access to a trusted system with same OS version

Tasks

No	Description of Test	Expected Result	Actual Result	PASS FAIL
1	Gather trusted versions of OS tools (where possible as static binaries)	Programs gathered. File system access documented		
2	Compile programs that are not installed on this system (where possible as static binaries)	Extra binaries compiled. File system access documented		
3	Prepare a script to gather evidence	Script tested and ready		
4	Make checksums of programs and script	Checksum file		
5	Burn the files to CDROM/DVD/USB stick with checksums	CDROM for use on Solaris system		

Details Concerning No. 1

Below is a list of binaries that may typically be required for a first incident response. This list is sorted in the order of volatility of the information the binaries are designed to gather.

The “bash” command shell program offers a trusted command line shell from which to run the other programs, and offers the chance to bind in most library files a program might call using the “/etc/profile” file (in our example /cdrom/forensics/etc/profile).

The live evidence gathering script should be started within a trusted version of bash from CD, and the bash shell should refer to an edited “/etc/profile” file that is also supplied from CD. This profile file will bind in the library files for the bash shell instance where possible.[2]

Commands run from the CDROM or source of gathered binaries are prepended either with the full path, a variable ($SECURE_SOLARIS_BINARIES/bash), or by changing to the relevant directory and adding “./” to the front of the command (./bash). This forces the shell to search in that directory for the binary before using a standard system program file. Relative path commands, such as “usr/bin/bash” or “./usr/bin/bash”, should be careful not to leave an initial slash since “/usr/bin/bash” will call the host’s version of bash and ignore the relative path.

[2] Example CDs can be downloaded from www.ufsdump.org, but it is advisable to make ones own.

GLÄRNISCHSTR. 7
POSTFACH 1671
CH-8640 RAPPERSWIL

Tel. +41 55-214 41 60
Fax +41 55-214 41 61
team@csnc.ch www.csnc.ch

This is an example of how a program might be tested for file system access. Truss shows a list of files opened by the program in various modes (for example RDONLY is read-only, not written) followed by the normal program output.

```
# 1) navigate to the CDROM to simplify paths
cd /cdrom/forensics/

# 2) The profile file can look like this:
cat etc/profile
LD_LIBRARY_PATH='/cdrom/forensics/usr/lib: /cdrom/forensics/usr/local/lib'
PATH='/cdrom/forensics/usr/bin: /cdrom/forensics/usr/sbin/: /cdrom/forensics/usr/ucb/:
/cdrom/forensics/sbin'
export PATH LD_LIBRARY_PATH

# 3) The trusted bash shell is called from the CDROM with the profile file:
./usr/bin/bash –rcfile ./etc/profile

# 4) Test with truss
./cdrom/usr/bin/truss -topen ./cdrom/usr/bin/netstat –an

# Alternatively, run the program directly using the trusted bash shell...

./cdrom/usr/bin/bash --rcfile ./cdrom/etc/profile ' ./cdrom/usr/bin/truss -topen
./cdrom/usr/bin/netstat -an  '
open("/var/ld/ld.config", O_RDONLY)             = 3
open("/cdrom/forensics/usr/lib/libdhcpagent.so.1", O_RDONLY) = 3
open("/cdrom/forensics/usr/lib/libcmd.so.1", O_RDONLY) = 3
open("/cdrom/forensics/usr/lib/libsocket.so.1", O_RDONLY) = 3
open("/cdrom/forensics/usr/lib/libnsl.so.1", O_RDONLY) = 3
open("/cdrom/forensics/usr/lib/libkstat.so.1", O_RDONLY) = 3
open("/cdrom/forensics/usr/lib/libc.so.1", O_RDONLY) = 3
open("/cdrom/forensics/usr/lib/libdl.so.1", O_RDONLY) = 3
open("/cdrom/forensics/usr/lib/libmp.so.2", O_RDONLY) = 3
open("/usr/platform/SUNW,UltraAX-i2/lib/libc_psr.so.1", O_RDONLY) = 3
open("/etc/default/inet_type", O_RDONLY)        Err#2 ENOENT
open("/dev/kstat", O_RDONLY)                    = 3
open("/dev/ip", O_RDWR)                         Err#13 EACCES
ip open: Permission denied
can't open mib stream: Bad file number
```

The lists of suitable programs in this and the following section are not intended to be exhaustive but should fulfill the tasks we describe in this paper. These programs are available for Solaris versions 8 to 10. Solaris 9 has the tool Ptrace and Solaris 10 adds Dtrace for interrogating systems. Since the approach chosen by Compass involves use of Bash with Bash environment variables this should be cross compatible across different operating system versions, although it leaves out the improved functionality of Dtrace in Solaris 10, for example. Readers are invited to adapt their own versions of the script.

Order of use	Program Name	Command Line	Description
1	arp	arp -an	Show current ARP connection table
2	netstat	netstat –ran netstat -an	Show routing table and cache Show open network connections
3	ndd	ndd /dev/ip ipv4_ire_status	Show forwarding table
4	ifconfig	ifconfig -a	Show network card configuration

Order of use	Program Name	Command Line	Description
5	/usr/ucb/ps	ps -auxwww	Show open processes NB there are two versions of the ps program in Solaris systems, this version will show the whole command line, the standard version will truncate the line.
6	ls, awk, sort	ls -a1l /proc/ | awk '{print $1 " " $2 " " $3 " " $4 " " $5 " " $6 " " $7 " "$8 " " $9 }' | sort -t " " -n -k9n	Get long listing of /proc directory to establish ownership of processes, numerical sort by PID
7	pldd	pldd [PID]	List dynamic libraries linked into the process [PID], including shared objects
8	pcred	pcred [PID]	Print credentials (effective, real, saved UIDs and GIDs) of process [PID]
9	pmap	pmap -x [PID]	Print the address map of process [PID] (with read/write/exec to file)
10	pfiles	pfiles [PID]	Gives fstat and fcntl information on all open files for process [PID]
11	ptree	ptree [PID]	List process trees within the PID, which child processes indented from parents
12	pwdx	pwdx [PID]	working directory of process
13	lsof	lsof –i	List open files with network connections

Details Concerning No. 2

The following non-system programs are useful and may have to be compiled from scratch. Since the goal of the first phase of the evidence gathering is to avoid touching the file system as much as possible, these could be left out. They will access the file system and alter the access times of the files they touch, which could overwrite valuable metadata evidence in the form of inode or files' timestamps.

Program Name	Command Line	Description
2hash	2hash [filename]	Creates simultaneous MD5 and SHA1 checksums, which together are currently impossible to spoof http://crossrealm.com/2hash/
chkrootkit	chkrootkit -p /cdrom/bin	Checks for known rootkits. NB, the tools used by this program may alter the file system. This tool is therefore best used on a copy of the image. The command parameter forces the program to look to /cdrom/bin for the ancillary programs it requires.

GLÄRNISCHSTR. 7
POSTFACH 1671
CH-8640 RAPPERSWIL

Tel. +41 55-214 41 60
Fax +41 55-214 41 61
team@csnc.ch www.csnc.ch

Further commands used by chkrootkit should also be included on the CDROM, if this program is used, in order to prevent it from using programs installed on the investigated host. Where possible, they should be compiled statically, so that they do not refer to programs or libraries on the investigated host.

awk, cut, echo, egrep, find, head, id, ls, netstat, ps, strings, sed, uname

Further useful programs include:

Program Name	Command Example	Description
netcat	./netcat –n –w3 [IP receiver] [Port]	Send data across a network (e.g. from investigated host to forensic workstation)
cryptcat	./cryptcat –n –w3 –k [password] [IP of receiver] [Port]	Alternative to netcat, with encryption
dd	./dd if=[raw device] of=[file] bs=1M	dd is the tool used for imaging and copying the RAM.
bash	./bash	Instigates a trusted shell environment
memdump	./memdump ¦ netcat …	Program designed to capture memory
dcfldd	./dcfldd dd if=[raw device] of=[file] bs=1M hash=sha1 hashlog=[logfile]	Variation of dd with built-in checksum capability (the of command could be exchanged for “vf” to verify the output).

Depending upon the exact tasks or extent of the investigation further tools may be considered, such as the Sleuthkit / Autopsy forensic toolkits, or specialist disk analysis tools such as hdparm, SCSI disk tools (scsiinfo, scsi_info), encryption tools and disk / hex editors and network gathering and analysis tools. However, this paper supposes an investigation from a dedicated workstation and not on the host itself. Network dumps, for example, need not be gathered on the investigated host itself.

GLÄRNISCHSTR. 7
POSTFACH 1671
CH-8640 RAPPERSWIL

Tel. +41 55-214 41 60
Fax +41 55-214 41 61
team@csnc.ch www.csnc.ch

3 Evidence Gathering

3.1 Live System Evidence Gathering

The goal of this aspect of the investigation is to gather information on the processes running in the Solaris server with minimal alterations to evidence in the file system. Some live response or normal systems administration tools will alter last access times of files, which might remove important clues. For this reason it is important to restrict investigative activities to RAM or to the host's network activities before imaging, and to save all information gathered to external media such as floppies or an evidence disk.

Extensive interference with RAM, such as creating a RAM dump, can be dangerous on a heavily used system and hence this step may not be feasible. However, if an attack is still being carried out or was implemented in the very recent past, this may be the only source of evidence for the attack method. File system analysis may only provide clues of attacker activities after access was gained to the system.

Goal	Gather volatile system Information
Preparation	Access to Solaris machine, Compass script and approval

Tasks
Details for the respective tests are found below the table

No	Description of Investigation	Expected Result	Actual Result	PASS FAIL
1	Gather RAM process evidence in order of volatility using trusted programs	Evidence file(s)		
2	Optional: dump system memory	Evidence file(s)		
3	Optional: network traffic capture and scan	Network dump file		
4	Checksum evidence files gathered on workstation	MD5 or SHA1 in chain of evidence form		

Details Concerning No. 1

The Compass Security evidence-gathering script, which is printed in Appendix 1: Compass Evidence Gathering Script, used the programs listed on page 9, ordered according to approximate volatility of the evidence (RFC 3227).

The script output is sent to a forensic workstation listening on a given port (see the section on online imaging over a network, 3.2.1 Online Imaging over a Network).

GLÄRNISCHSTR. 7
POSTFACH 1671
CH-8640 RAPPERSWIL

Tel. +41 55-214 41 60
Fax +41 55-214 41 61
team@csnc.ch www.csnc.ch

```
[More recent Solaris systems will automount a CDROM]
cd /mnt/cdrom-compass/bin
./bash

./evidence-script.sh ¦ netcat –n –w3 [IP Workstation] [Port]
```

Details Concerning No. 2

There are three methods of dumping RAM in a Unix system. Either a dump can be provoked of the system memory using savecore and dumpadm to set up the correct parameters, dd or cat can be run against the memory device (for example /dev/mem or /dev/kmem), or there is a specialized Unix tool created by the developers of the TCT tools, memdump. The memdump program was specifically designed to capture memory using as small a footprint as possible, but should be tested before use. [3]

```
./memdump ¦ netcat –n –w3 [IP Workstation] [Port]
```

Some Solaris systems are able to dump memory while the system is running using savecore, as long as the correct parameters have been set up with dumpadm.[4] This latter command can also be used to specify a device to receive the core dump (such as an evidence disk), the dump may overwrite evidence if the core dump is written to a storage device that is part of the system. For this reason, it is better to send the dump over a network to another system (forensic workstation) or use a dedicated partition rather than a swap partition. Various tools are available to assist with the analysis of dump files.[5] Sun also offers this analysis as part of its service contracts and has courses on the topic. A detailed discussion is beyond the scope of this paper.

Details Concerning No. 3

Depending upon circumstance it may be useful to gather evidence outside the host: namely the network traffic leaving and entering it, and to scan the host to establish which services it is running. This can be advisable if the host is still under attack, in order to capture the attacker's actions and IP, or if a suspected attack needs to be verified. Installing sniffing software and acquiring a network traffic dump on the host itself will interfere with the running system and its file system. It is therefore advisable to insert a transparent bridge between the host and its network to sniff the traffic.

Two methods are readily available to gather this traffic evidence from the network: either port-mirroring on a switch (also known as port spanning or port monitoring), or a bridge device is placed between the host and the network to gather the traffic. In large switched networks it may be more

[3] For more information on capturing memory on Unix systems and Solaris in particular: Dan Farmer and Wietse Venema, Forensic Discovery, Addison Wesley Professional: 2005, p. 165ff. "memdump" is available from the authors' homepage, who are also the developers of forensic tools The Coroner's Toolkit (TCT): http://www.porcupine.org/forensics/tct.html. The manpage warns of several possible system crashes that can be caused by this program. It should be tested on a test system first.

[4] Venema and Farmer, p. 167. More information is available from the University of Princeton Solaris troubleshooting pages: http://www.princeton.edu/~psg/unix/Solaris/troubleshoot/coreanal.html and http://www.princeton.edu/~psg/unix/solaris/troubleshoot/savecore.html

[5] A short list of dump file analysis tools includes: Solaris Crash Analysis Tool 4.0, http://www.sun.com/download/products.xml?id=3e3af5aa, MemTool (quote from the website: "MemTool is not supported in anyway by Sun; use at your own risk. MemTool loads a kernel module into the target system, and may sometimes be incompatible with new Solaris patches. For this reason, do not use MemTool on a production system."), http://www.solarisinternals.com/si/tools/memtool/index.php; MDB (modular debugger), a default system utility since Solaris 8.

GLÄRNISCHSTR. 7
POSTFACH 1671
CH-8640 RAPPERSWIL

Tel. +41 55-214 41 60
Fax +41 55-214 41 61
team@csnc.ch www.csnc.ch

convenient to configure a switch to mirror the port that consigns traffic to the host. The mirrored port is directed to a forensic workstation that collects the host's network traffic. This solution is less likely to result in a slow-down of traffic than the bridge solution, which, depending upon throughput, may act as a bottleneck. Port mirroring is commonly used by Network Intrusion Detection Systems (NIDS). Switch configuration is beyond the scope of this paper, since this dependent upon hardware and platform.[6]

A forensic workstation or laptop with sufficient storage capacity and two Ethernet ports can be configured as a bridge. A network bridge does not have an IP address of its own, will not show up on traceroute queries, and is largely invisible to the network and an attacker. The network cable can be unplugged from the evidence host and into the bridge with minimal interruption to traffic. Most recent Linux kernels of the 2.4 and 2.6 series have bridging support incorporated into the kernel, and the required administrative tools can be installed via the bridge-utils packages. Bridging can then be set up using the brctl command set. More information is available from The Linux Documentation Project.[7]

```
[Set the network interfaces to promiscuous mode for sniffing]
ifconfig eth0 promisc up
ifconfig eth1 promisc up

[Set up the bridge with name br0]
brctl addbr br0

[Add network interfaces eth0 and eth1 to bridge]
brctl addif br0 eth0
brctl addif br0 eth1

[Zero the MAC-addresses of the interfaces]
ifconfig eth0 0.0.0.0
ifconfig eth0 0.0.0.0

[Start the bridge]
ifconfig br0 up
```

Assuming that the investigations laptop is running under Linux and its network bridging interface is "br0", as in the example above, the network dump could be started as shown below. This command does not resolve addresses to names for reasons of speed (-n) and provides verbose output, for example decoding SMB packets (-vvv), captures the whole packets (-s 0) and writes the dump to a file (-w); each line is given a timestamp in the default format (date and time, -tttt).

```
tcpdump –tttt –i br0 –n –vvv –s0 –w [filename]
```

It is possible that some packets will be lost in a busy network connection; a second terminal can be used to follow the growth of the dump file, if required, using tail -f (for "follow"):

```
tail –f [filename]
```

[6] For more information on port mirroring in switched networks, see: http://www.networkdictionary.com/howto/NetworkAnalyzer.php

[7] The Linux Documentation Project: http://www.tldp.org/HOWTO/BRIDGE-STP-HOWTO/index.html. See also http://linux-net.osdl.org/index.php/Bridge.

GLÄRNISCHSTR. 7
POSTFACH 1671
CH-8640 RAPPERSWIL

Tel. +41 55-214 41 60
Fax +41 55-214 41 61
team@csnc.ch www.csnc.ch

Since the investigator has inserted a network device between the host and the rest of the network, this device can now be used to scan the host, if required, for illicit services and other signs of a successful attack. Nmap is useful here. The following command will assume that the host is live and not try to ping it, in case it blocks the request at a firewall (-P0), starts a TCP syn and UDP scan (the former causes less, if any, logging on the host –sS and -sU), does not attempt to resolve the IP address using DNS (-n), provides verbose output (-vv) in a grep-able form (all information on one line, better for automating searches later, -oG).

```
nmap –P0 –sS –sU –O –vv –n –oG [filename] [ip of host]
```

Details Concerning No. 4

All evidence files should be given cryptographic checksums as soon as they are gathered. This is especially important in live evidence gathering where the source evidence is being altered all the time. Current best practice is moving away from MD5 checksums to SHA1, since the first weaknesses in the MD5 algorithm are becoming apparent. Various tools are available for generating checksums on Solaris: md5 and sha1, the openssl digest modules, or 2hash.

The checksums are the start of the chain of evidence, and are used to prove that no subsequent alterations were made to the evidence files. Hence they should be recorded in log books or the chain of evidence forms.

```
2hash evidencefile

openssl dgst -sha1 evidencefile
#Alternative
openssl sha1 evidencefile
```

GLÄRNISCHSTR. 7
POSTFACH 1671
CH-8640 RAPPERSWIL

Tel. +41 55-214 41 60
Fax +41 55-214 41 61
team@csnc.ch www.csnc.ch

3.2 Imaging

Goal	Create a digital bit-stream copy of operating system disk
Preparation	Decision if “online” or “offline” imaging

Online versus Offline investigation

Online imaging involves copying the disk without interrupting the running system. Since the system is still running while the disk is being copied, the system will temporarily buffer write actions to the disk and implement these after the process has finished. This may have implications for certain data structures, such as online databases: this type of imaging can result in broken data consistency. A verification of the disk by checksums is also no longer possible: the disk has not been “frozen” by being taken offline and is still in use; will be immediately altered. Therefore from an evidentiary standpoint this is not a preferred method, but may be required when examining critical live systems.

Offline forensic imaging requires the system to be shut down and imaged in an inactive state. The disk is typically removed from the system and imaged on another computer. Since it is not in use, verification of the disk state is possible before and after the imaging, and it can be demonstrated that no changes were made to the disk by the evidence capture process. If the disk remains offline, then any subsequent change to the data on the disk can be ascertained using the checksums.

With large RAID 5-type disk sets or SAN systems imaging the disks independently may not be feasible. This is especially a problem with large-scale systems that have proprietary hardware RAID controllers: the investigator would need the same hardware controller to reconstruct the RAID, a software-based reconstruction may not be possible. In these circumstances live imaging of the host, or a controlled boot of the (offline) host is preferable. These would allow the extraction of the data using the proprietary hardware into a neutral format.

From an evidentiary aspect offline imaging makes it easier to prove that evidence presented later in court is the same as that found on the disk, but this can be unrealistic with critical infrastructure.

3.2.1 Online Imaging over a Network

Imaging over a network requires a valid account and connection, and an IP address from the evidence host that can be reached by the workstation. The hard drives of the evidence host should be un-mounted if possible so that the raw disk devices can be addressed. However this is the slowest method of imaging discussed in this paper.

Another aspect of network imaging speed is encryption. Within a trusted network it is not advisable to use encryption since it will considerably slow down the imaging process: netcat is the tool of choice. However, when the image is to be transferred out of a trusted network encryption should be considered to protect the data from third parties and may be a legal requirement. A comparison of algorithm speeds showed that MD5 could be one-third to over three times faster than SHA1 and twofish, depending on test environment.[8]

[8] A Windows and a 64-bit Linux environment were used for testing:
http://www.eskimo.com/~weidai/benchmarks.html

Goal	Log into and image evidence host; receive image and data at workstation
Preparation	Correct account details, ssh client or netcat, compiled sources for netcat or cryptcat IP address and adequate disk space for workstation

Tasks

No	Description of Activity (H: host W: workstation)	Expected Result	Actual Result	PASS FAIL
1	Is it possible to log into the evidence host from the workstation	Yes		
2	H: Are the evidence host hard drives un-mounted or is it possible to un-mount them? If not, is it possible to image the disks off the host without damaging the system?	Yes		
3	H: If the disks have been un-mounted: Make checksum of each storage media prior to imaging and send to listening workstation	Checksum(s) sent		
4	W: Listen for checksum(s)	Checksum received		
5	H: prtvtoc or fdisk listing of hard-drives on evidence host and sent to workstation	fdisk listing sent		
6	W: Listen for fdisk info	fdisk info received		
7	H: Image hard drive(s) with dd over the network using encryption (ssh or cryptcat)	Image sent		
8	W: Listen for image	Image file received		
9	W: Verify image integrity	Checksum matches image file		

Details Concerning No. 3

In a trusted environment netcat is the tool of choice since it is faster. The string $SECURE_SOLARIS_BINARIES denotes the path to the trusted binaries used (for example on the CDROM).

Make MD5 or SHA1 checksums of the disks before imaging. This step is only meaningful if the disks could be unmounted, but is important in the chain of evidence. It proves the investigator to prove at any later stage that the image used for analysis is the same as the state of the disk before imaging.

```
$SECURE_SOLARIS_BINARIES/sha1 [/devicename] ¦ $SECURE_SOLARIS_BINARIES/netcat -n -w3 [IP Workstation] [Port]
```

GLÄRNISCHSTR. 7
POSTFACH 1671
CH-8640 RAPPERSWIL

Tel. +41 55-214 41 60
Fax +41 55-214 41 61
team@csnc.ch www.csnc.ch

Details Concerning No. 4

Open a listening port and receive the checksum on the workstation:

```
nc -l -p [Port] > /mnt/evidence/[devicename].sha1

cat /mnt/evidence/[devicename].sha1
```

Details Concerning No. 5

(This is dealt with in section 3.2.3 below.) prtvtoc is typically not statically compiled and will refer to several library files on the disks in the /var/ld, /usr/lib or /usr/platform folders, altering the access times for these files. fdisk does not appear to refer to system files.

Details Concerning No. 6

Open a listening port and receive the fdisk information:

```
nc -l -p [Port] > /mnt/evidence/fdisk_[devicename].log

cat /mnt/evidence/fdisk_[devicename].log
```

Details Concerning No. 7

Using SSH:

```
$SECURE_SOLARIS_BINARIES/dd if=/dev/rdsk/c1t1d0s2 bs=32k ¦ $SECURE_SOLARIS_BINARIES/ssh
192.168.100.1 "dd bs=32k of=/mnt/hdc1/solaris_[systemname].dd"
```

Using the encrypted version of netcat, cryptcat:

```
$SECURE_SOLARIS_BINARIES/dd if=/dev/rdsk/c1t1d0s2 bs=32k ¦ $SECURE_SOLARIS_BINARIES/gzip -c
¦  $SECURE_SOLARIS_BINARIES/cryptcat -n -w3 -k [password] [IP Workstation] [Port]

[Variation with netcat]
$SECURE_SOLARIS_BINARIES/dd if=/dev/rdsk/c1t1d0s2 bs=32k ¦ $SECURE_SOLARIS_BINARIES/gzip -c
¦ $SECURE_SOLARIS_BINARIES/netcat -n -w3 [IP Workstation] [Port]
```

Details Concerning No. 8

Open a listening port and receive the compressed image.

```
nc -l -p [Port] > /mnt/evidence/image_[devicename].dd.gz

gunzip /mnt/evidence/image_[devicename].dd.gz

[Variation with cryptcat]
cryptcat -k [password] -l -p [Port] > /mnt/evidence/image_[devicename].dd.gz
```

GLÄRNISCHSTR. 7
POSTFACH 1671
CH-8640 RAPPERSWIL

Tel. +41 55-214 41 60
Fax +41 55-214 41 61
team@csnc.ch www.csnc.ch

Details Concerning No. 9

Verify the image against the checksum of the original source device.

```
sha1sum /mnt/evidence/image_[devicename].dd

cat /mnt/evidence/[devicename].sha1
```

GLÄRNISCHSTR. 7
POSTFACH 1671
CH-8640 RAPPERSWIL

Tel. +41 55-214 41 60
Fax +41 55-214 41 61
team@csnc.ch www.csnc.ch

3.2.2 Offline Imaging

Tasks

No	Description of Investigation	Expected Result	Actual Result	PASS FAIL
1	System shutdown or removal of disk from running system	System down		
2	Remove disk(s)	System disk safely removed and unharmed		
3	Install disks to imaging host with write blocker if possible	Disk ready for imaging		
4	Boot imaging host with a secure operating system (e.g. Linux)	Secure operating system running		
5	Mount the clean, wiped destination disk (preparation: section 2.1)	Destination disk ready to receive image		
6	Make checksum of source disk	Checksum file on destination disk		
7	(If relevant) gather controller information on the source disk	Information file on destination disk		
8	Image source disk to file on destination disk	Byte stream copy of source disk in a file		
9	Checksum image file on destination disk	1. Checksum file of image files.		
10	Compare checksums on the destination disk (of source disk and image file)	1. Both checksums of source disk and destination file are identical.		

Details Concerning No. 1

From an evidentiary system it is preferable to avoid an ordered shutdown of the system using normal system commands. In a hacked system, the shutdown processes may have been Trojaned by leaving “logic bombs” in the form of evidence cleanup scripts. The disk to be imaged is simply pulled from the system or the system is shut down suddenly by removing the power supply. The possibility of data inconsistency must be balanced against this risk. Some Sun servers with certain

RAID configurations, such as RAID 1, mirroring, support hot-plugging: it is possible to remove a disk for imaging while the system is live without breaking system consistency.

Details Concerning No. 3

If the source disk is an IDE disk, it should be jumpered as slave to prevent accidental access on boot. Write blockers can be used to prevent accidental writes to the evidence disks. These are hardware devices that are inserted between the evidence disk and the motherboard controller. Their function is to trap and cache write orders sent to the evidence disks. The write actions are instead maintained in cache. The operating system is 'tricked' into believing that the write was successfully implemented (necessary, for example, for the safe functioning of some operating systems.) If no write blocker is available, then on no account should the disk be placed in a live Windows operating system, since this will automatically write to it and alter the evidence.

Another common precaution at this stage is to jumper the evidence disks as slave and install them on the secondary IDE channel (if IDE disks). In the case of an accidentally uncontrolled boot, many operating systems search first for a boot disk on the primary master channel. Under Linux disks jumpered as secondary slave will typically be recognized as "/dev/hdd" depending on the operating system version (SATA devices may be addressed differently).

Details Concerning No. 4

Given the tendency of Windows operating systems to write to attached disks Compass Security recommends using a Unix variant operating system for imaging, or a Knoppix or variant boot CDROM. This latter version of Linux allows precise control over the boot parameters. The option "noswap" should be used at all costs, since this prevents the operating system from writing a swap file to the disk. (Forensic versions of Knoppix, such as Helix, should have this option built in.) Booting to the command prompt without a GUI will save time. The example below shows the Knoppix boot parameters used by Compass for our imaging device, which force DMA access mode for IDE disks and limit BIOS energy saving modes:

```
knoppix 2 failsafe noswap noapm noapic dma
```

If the disk is not IDE or SCSI but from a fibre-channel system it may be necessary to use specialized Sun hardware with the correct controller to image the device. Since Sun, as most Unix systems, will not write to an un-mounted non-system disk, the source disk can be built into the system, a checksum made and imaged un-mounted and unaltered (see section 3.2.3 Online Imaging on Host).

Details Concerning No. 5, 6

A useful way to check whether the attacked disks have been recognized correctly by the operating system at boot is to check the boot messages using "dmesg".

With imaging it is vital that the evidence disk be properly labeled and differentiated from the disk that is used to host the image. Compass distinguishes between the "source" disk, which is the Solaris system disk with evidence, and the "destination" disk, which is the disk prepared to hold the image file. Since Knoppix automatically mounts drives in read-only mode, the following commands will create a directory and mount the destination disk in write mode (we assume that the destination

disk is jumpered to master and connected to the first IDE channel). The last mount command will confirm whether the disk is mounted correctly. The source disk should not be mentioned in the mount output. Use of the Knoppix automounter in the graphical user interface (by default KDE) is not advised since it can clash with a command-line based management of devices.

Since Compass labels the destination disks as DEST[number] creating a temporary directory of this name is a convenient way to avoid confusion with the source drive.

```
dmesg
mkdir /mnt/dest
mount -w -t auto /dev/hda1 /mnt/dest
mount
```

The checksum programs md5/sha1 (Solaris) or md5sum/sha1sum (Linux) are commonly part of most distributions. A checksum of an entire Solaris disk must use the second partition (slice), the backup slice, which covers the whole disk:

```
[Solaris]
sha1 /dev/rdsk/c1t1d0s2 > /mnt/dest/evidence_c1t1d0s2.sha1

[Linux]
sha1sum /dev/hdd > /mnt/dest/evidence_hdd.sha1
```

Details Concerning No. 7

For IDE disks the command

```
hdparm -giI /dev/hdd
```

will provide information from the IDE drive controller that will not be copied during the imaging process. This information provides statistics on the drive but also alerts to the possibility of hidden sectors on the disk. The Sleuthkit tools disk_stat and disk_sreset can confirm the existence of hidden “host protected areas” on IDE disks, which the latter tool can temporarily remove.

SCSI disks can use the scsiinfo or scsi_info commands to get controller information.

Details Concerning No. 8

Using dd to image the following command will copy the entire disk. Using a blocksize (bs) of 32k will speed up the write process for disks with fast write speeds and large buffer memories built in (default is a block size 512 bytes). Other possible parameters commonly used for imaging are “conv=noerror,sync” should the source disk have bad sectors. The default mode below will fail if there are bad sectors on the disk. One suggestion for easier management of evidence images is to use a uniform naming convention, using a project or system name. (This example assumes that the source disk is jumpered as slave on the second IDE controller).

```
dd if=/dev/hdd of=/mnt/dest/solaris_[system-name].dd bs=32k
```

3.2.3 Online Imaging on Host

GLÄRNISCHSTR. 7
POSTFACH 1671
CH-8640 RAPPERSWIL

Tel. +41 55-214 41 60
Fax +41 55-214 41 61
team@csnc.ch www.csnc.ch

Should offline imaging not be feasible, imaging can still be done while the system is live. The image should be piped off the system if possible so to minimize interference with the evidence being imaged. However, where this is not possible, an external disk should be formatted with UFS and mounted to receive the image. Since the imaging process is competing for system resources and not being copied offline, but disk-to-disk, this method is much slower than offline imaging. Online imaging on the host itself, the current scenario, is rarely used since it requires the live incorporation of a new hard drive into the system (hot plugging). This is feasible on later Solaris systems (for example, 9 and 10), but should be regarded as the last possible option, after offline imaging and online network imaging.

Tasks

No	Description of Investigation	Expected Result	Actual Result	PASS FAIL
1	Attach and mount destination disk on the evidence host	Destination disk ready to receive image		
2	Get device information	File on destination disk		
3	Byte stream copy (image) of source disk to file	File on destination disk		

Details Concerning No. 1

Online imaging techniques are discussed in greater detail in the sections to follow.

It may be necessary to rewrite the Solaris system device file for the newly attached disk using the command devfsadm so that the controller and device can be addressed by the operating system. In our example, we mount the third slice of our destination disk to hold the image data.

```
devfsadm
mkdir /mnt/dest
mount /dev/dsk/c1t1d0s3 /mnt/dest
```

Details Concerning No. 2

Before imaging, it can be useful to confirm which disks are attached to the system, to avoid unpleasant surprises later on. prtvtoc is one Solaris command to print the Virtual Table of Contents (and is hence of interest for SPARC systems). It gives useful information on the attached and mounted disks, and their allocated space and partitioning. Since the command refers to several library files on the host, it will alter some of the timestamps of the filesystem to be investigated; these library files may also have been trojaned. One example of the prtvtoc output is given below:

```
$SECURE_SOLARIS_BINARIES/prtvtoc /dev/rdsk/c1t1d0s0
* /dev/rdsk/c1t1d0s0 partition map
*
* Dimensions:
*     512 bytes/sector
*     424 sectors/track
*      24 tracks/cylinder
*   10176 sectors/cylinder
*   14089 cylinders
*   14087 accessible cylinders
*
* Flags:
*   1: unmountable
*  10: read-only
*
* Unallocated space:
*       First     Sector    Last
*       Sector     Count    Sector
*     6309120 18446744073703252672     10175
*   143349312 18446744073572511424   6309119
*    27271680 111854592 139126271
*   143328960     20352 143349311
*
*                          First     Sector    Last
* Partition  Tag  Flags    Sector     Count    Sector  Mount Directory
       0      2    00      10176   6298944   6309119
       1      3    01    6309120  20962560  27271679
       2      5    00          0 143349312 143349311
       3     15    01          0     10176     10175
       4     14    01      10176 143339136 143349311
       6      7    00  139126272   4202688 143328959
```

fdisk can also be used to write the disk geometry to screen (-g for label geometry, -G for physical geometry).

```
$SECURE_SOLARIS_BINARIES/fdisk -gG [/raw/device/name]
```

Details Concerning No. 3

Solaris operating systems offer a raw disk device (/dev/rdsk) to access devices and partitions directly, and standard partitioning schemes offer a second “slice” (partition) for backup purposes that cover the entire disk. This is the target for imaging. In the example below, the backup slice (s2) of the first device or disk (d0) of the second target (t1, more important for SCSI disks) of the second IDE controller (c1, counting begins at 0) is being imaged, hence /dev/rdsk/c1t1d0s2.

The operating system buffers any write actions to the disk during imaging, so normal operations can continue, but the disk will be altered immediately after the imaging is complete since it is still in use.

```
$SECURE_SOLARIS_BINARIES/dd if=/dev/rdsk/c1t1d0s2 of=/mnt/dest/solaris_[system-name].dd
bs=32k
```

GLÄRNISCHSTR. 7
POSTFACH 1671
CH-8640 RAPPERSWIL

Tel. +41 55-214 41 60
Fax +41 55-214 41 61
team@csnc.ch www.csnc.ch

Appendix 1: Compass Evidence Gathering Script

A1.1 Volatile Information Gathering Script

This script should be run from the mount point of the CDROM (for example /cdrom/forensic) and the resulting evidence sent to an external device as described in the script code.

Command line:

```
# Move to the mount point
cd /cdrom/forensic
# Start a trusted version of Bash
usr/bin/bash --rcfile etc/profile
# Optional: start a script
usr/bin/script
# Start evidcence-gathering script
usr/bin/bash --rcfile etc/profile ./solaris_volatile_evidence.sh
```

The etc/profile file:

```
PERL='/cdrom/forensic/usr/local/bin/perl'
LD_LIBRARY_PATH='/cdrom/forensic/usr/lib:/cdrom/forensic/usr/local/lib:/cdrom/forensic/usr/l
ib/64:/cdrom/forensic/usr/local/lib/64'
LD_CONFIG=etc/ld.conf
PATH='/cdrom/forensic/usr/bin:/cdrom/forensic/usr/sbin:/cdrom/forensic/usr/ucb:/cdrom/forens
ic/sbin: /cdrom/forensic/usr/local/bin:/cdrom/forensic/usr/local/sbin'
export PATH PERL LD_LIBRARY_PATH LD_CONFIG
```

The script (solaris_volatile_evidence.sh):

```
#! ./usr/bin/bash --rcfile ./etc/profile

# Compass Security Volatile Evidence
# Gathering Script for Solaris
#
# System: 64-bit SPARC with SunOS 5.8
#
# Purpose: gather volatile evidence in RAM prior to imaging.
# The programs used should on no account alter the file system.
#
# Usage:
# 1) Start a bash from the CDROM forcing use of the etc/profile file:
# /cdrom/forensic/usr/bin/bash --rcfile ./etc/profile
#
# 2) cd /cdrom/forensic
# 3) Run the script (optional with script command first)
#
# usr/bin/bash --rcfile ./etc/profile ./solaris_volatile_evidence.sh
#                     2>&1 [MOUNTPOINT Destination disk]/evidence-file.txt
#       OR using netcat:
# usr/bin/bash --rcfile ./etc/profile ./solaris_volatile_evidence.sh 2>&1 \
#       ¦ ./usr/local/bin/nc -n -w3 [IP] [Port]
# If you get "Broken Pipe" errors, then add this onto the front of this line:
#     trap exit PIPE ;
#
# Author: Mark Furner, 08 Sept. 2005
```

```
#
# Alterations:
# Simple version without procedures, MF 08.09.2005
# Altered checking procedures for programs on CDROM, MF 27.09.2005
# Forced use of CDROM binaries, MF 18.01.2006
# Tweaks and path variables, MF 05.04.2006
# Added bit about Broken Pipe, MF 02.05.2006
# Further tweak to date call, MF 16.05.2006
# Added to and altered memory sections, removed disk tools (prtvtoc etc), MF 24.05.2006

# Set variables for binaries
CURDIR=`./usr/bin/pwd`
BINDIR="$CURDIR/usr/bin"
SBINDIR="$CURDIR/usr/sbin"
UCBBINDIR="$CURDIR/usr/ucb"

# Start with warning to user
$BINDIR/echo "All output can be redirected to your choice of storage or netcat pipe."
$BINDIR/echo "Invoke this script only in a trusted shell using the correct environment
variables."
$BINDIR/echo

# Get local time and date
$BINDIR/echo "Local system time and date at start is:"
$BINDIR/date
$BINDIR/echo

# Gather evidence
$BINDIR/echo "************************************"
$BINDIR/echo "Starting Volatile Evidence Gathering"
$BINDIR/echo "************************************"
$BINDIR/echo

$BINDIR/echo "****************"
$BINDIR/echo "Running arp -a"
$BINDIR/echo "****************"
$SBINDIR/arp -a
$BINDIR/echo
$BINDIR/echo "***********************************************"
$BINDIR/echo "Output of netstat -arn routing table and cache"
$BINDIR/echo "***********************************************"
$BINDIR/netstat -arn
$BINDIR/echo
$BINDIR/echo "*******************"
$BINDIR/echo "Output of netstat -an open connections"
$BINDIR/echo "*******************"
$BINDIR/netstat -an
$BINDIR/echo
$BINDIR/echo "*******************************************************"
$BINDIR/echo "Output of ndd /dev/ip ip_forwarding forwarding status"
$BINDIR/echo "for IPv4 first then IPv6: 0 off 1 is ON"
$BINDIR/echo "*******************************************************"
$SBINDIR/ndd -get /dev/ip ip_forwarding
$BINDIR/echo
$SBINDIR/ndd -get /dev/ip ip6_forwarding
$BINDIR/echo
$BINDIR/echo "*****************************************"
$BINDIR/echo "Output of ifconfig -a network device status"
$BINDIR/echo "*****************************************"
$SBINDIR/ifconfig -a
$BINDIR/echo
$BINDIR/echo
"*********************************************************************************************"
$BINDIR/echo "Output of /usr/ucb/ps -auxwww (from cdrom) verbose process information with
start time"
$BINDIR/echo
"*********************************************************************************************"
```

GLÄRNISCHSTR. 7
POSTFACH 1671
CH-8640 RAPPERSWIL

Tel. +41 55-214 41 60
Fax +41 55-214 41 61
team@csnc.ch www.csnc.ch

```
$UCBBINDIR/ps -auxww
$BINDIR/echo
$BINDIR/echo "******************************************************"
$BINDIR/echo "Use ls and awk to find who started/owns which processes "
$BINDIR/echo "******************************************************"
$BINDIR/ls -a1l /proc/ | $BINDIR/awk '{print $1 " " $2 " " $3 " " $4 " " $5 " " $6 " " $7 "
"$8 " " $9  }' |\
  $BINDIR/sort -t " " -n -k9n
$BINDIR/echo
$BINDIR/echo "*********************************************************************************"
$BINDIR/echo "Loop through each process in /proc to get environment information and stats"
$BINDIR/echo "*********************************************************************************"
$BINDIR/echo
# Solaris 9 has pargs, but proc tools work for Solaris >=8

# pldd [PID]
# List dynamic libraries linked into each process, including shared objects
# explicitly attached using dlopen.
# prcred [PID ¦ user]
# Print credentials (effective, real, saved UIDs and GIDs) of each process
# pmap -x [PID]
# print the address map of each process (with read/write/exec to file)
# pfiles [PID]
# Gives fstat and fcntl info on all open files in each process
# ptree [PID]
# process trees within the PID, which child processes indented from parents
# pwdx [PID]
# working directory of process

PIDarray=`$BINDIR/ls -la /proc | $BINDIR/awk '{print $9}'`
for PIDno in $PIDarray ; do
        $BINDIR/echo "****************************************"
        $BINDIR/echo "*** Getting information on PID $PIDno ***"
        $BINDIR/echo "****************************************"
       $BINDIR/echo " *** pldd for PID $PIDno *** "
       $BINDIR/pldd $PIDno
       $BINDIR/echo " *** pcred for PID $PIDno *** "
       $BINDIR/pcred $PIDno
       $BINDIR/echo " *** pmap -x for PID $PIDno *** "
       $BINDIR/pmap -x $PIDno
        $BINDIR/echo " *** pfiles for PID $PIDno *** "
        $BINDIR/pfiles $PIDno
        $BINDIR/echo " *** ptree for PID $PIDno *** "
        $BINDIR/ptree $PIDno
        $BINDIR/echo " *** pwdx for PID $PIDno *** "
        $BINDIR/pwdx $PIDno
       $BINDIR/echo "*** Completed PID $PIDno ***"
       $BINDIR/echo
done

$BINDIR/echo "************************************************"
$BINDIR/echo " Dump of process environment information complete"
$BINDIR/echo "************************************************"
$BINDIR/echo


$BINDIR/echo "*********************************************"
$BINDIR/echo "Output of lsof -i find open files to processes"
$BINDIR/echo "*********************************************"
$BINDIR/lsof -i
$BINDIR/echo

# Disk section: these tools now removed
#$BINDIR/echo "********************"
#$BINDIR/echo "Output of isainfo -vk"
#$BINDIR/echo "********************"
#$BINDIR/truss -topen $BINDIR/isainfo -vk
```

GLÄRNISCHSTR. 7
POSTFACH 1671
CH-8640 RAPPERSWIL

Tel. +41 55-214 41 60
Fax +41 55-214 41 61
team@csnc.ch www.csnc.ch

```
#$BINDIR/echo "******************"
#$BINDIR/echo "Output of showrev -p"
#$BINDIR/echo "******************"
#$BINDIR/truss -topen $BINDIR/showrev -p
#$BINDIR/echo "*************"
#$BINDIR/echo "Output of crle"
#$BINDIR/echo "*************"
#$BINDIR/truss -topen $BINDIR/crle
#$BINDIR/echo "*****************"
#$BINDIR/echo "Output of prtconf"
#$BINDIR/echo "*****************"
#$BINDIR/truss -topen $SBINDIR/prtconf
#$BINDIR/echo "*****************"
#$BINDIR/echo "Output of swap -l"
#$BINDIR/echo "****************"
#$BINDIR/truss -topen $SBINDIR/swap -l
#$BINDIR/echo "*****************"
#$BINDIR/echo "Output of prtvtoc"
#$BINDIR/echo "****************"
#$BINDIR/truss -topen $SBINDIR/prtvtoc /dev/rdsk/c0t0d0s0

$BINDIR/echo "************************************"
$BINDIR/echo "First stage of evidence"
$BINDIR/echo "gathering completed. "
$BINDIR/echo "Consider making a copy of /proc and "
$BINDIR/echo "a RAM dump using memdum, and "
$BINDIR/echo "proceeding to imaging. "
$BINDIR/echo "Make a checksum of the evidence file. "
$BINDIR/echo "************************************"
$BINDIR/echo

# Date and time at end of investigation
$BINDIR/echo "Local system date and time at end of script run: "
$BINDIR/date
# End
```

GLÄRNISCHSTR. 7
POSTFACH 1671
CH-8640 RAPPERSWIL

Tel. +41 55-214 41 60
Fax +41 55-214 41 61
team@csnc.ch www.csnc.ch