A triage framework for digital forensics


Muhammad Shamraiz Bashir


Naeem Ahmed Khan

Muhammad Shamraiz Bashir, Muhammad Naeem Ahmed Khan, Shaheed Zulfikar Ali Bhutto Institute of Science and Technology, Islamabad, Pakistan

A sharp increase in malware and cyber-attacks has been observed in recent years. Analysing cyber-attacks on the affected digital devices falls under the purview of digital forensics. The Internet is the main source of cyber and malware attacks, which sometimes result in serious damage to the digital assets. The motive behind digital crimes varies – such as online banking fraud, information stealing, denial of services, security breaches, deceptive output of running programs and data distortion.

Since computers are the primary targets for such kinds of malicious activities, evidence can be derived from different storage media, both volatile and non-volatile, attached to the computing devices. The role of information security officer in an organisation is to identify and implement appropriate procedures to prevent and subdue such cyber-attacks. In the event of a cybercrime incident, the incident response team needs to respond quickly so that the damage created by such attacks can be minimised. Then, a meticulous forensic analysis follows that involves gathering digital evidence to trace the origin and nature of the attack and identify the perpetrators. Digital evidence can be gathered from log files, file system activity, encrypted files, hidden pictures, deleted files, password-protected files, memory content, running processes and so on.

Generally, digital forensic analysis consists of three major steps: data acquisition, data analysis and evidence extraction/presentation. In the first step, specific digital information is acquired which may be in the form of logs, emails, deleted files, registry entries or file system activities available on the infected machine. Then deep analysis is performed by means of different tools in the second step; such analysis may be in the form of static or live analysis. In the final step, the analysed data is transformed into an analysis report to make it admissible in a law court. A number of open-source and commercial tools are available that can help perform digital examinations. The amount of digital data is constantly increasing, therefore a substantial amount of time is needed to sift the forensic-related data and perform analysis on it. In this regard, computational power and sophisticated tools are essential during the digital forensic analysis for the efficient extraction of evidence from large volumes of data.

Best practice

The Association of Chief Police Officers (ACPO) of England and Wales has outlined best practice guidelines for computer-based evidence.[1] The four principles articulated by ACPO need to be observed in true letter and spirit to ensure the authenticity of digital evidence. Computer forensic investigation needs to comply with these good practice guidelines so that digital evidence can be admitted in court.

Today, live forensics are becoming complementary to static analysis. Live digital forensics play a vital role during system examinations due to the potential availability of digital evidence in volatile memory, such as running processes, network connections, opened ports and encryption keys. Volatile data is categorised as Tier 1 and Tier 2 data. Tier 1 data corresponds to critical system information such as logged in users, active network connections and processes; whereas, Tier 2 data pertains to scheduled tasks and clipboard content.[2] Using live forensics entails recovering and analysing memory content, processes, opened network ports, opened connections and files linked to the running processes. Because the physical memory is a volatile storage media, it requires keeping the system alive until a proper memory dump is obtained.

Static analysis is also known as conventional or offline analysis. In static analysis, the infected system is usually powered off and analysis is performed by making duplicate copies of the storage devices found on the infected machine, while in live forensic analysis the victim machine is supposed to be kept alive in order to obtain a dump of the memory contents. Yet there is a chance of alteration in the memory content due to the execution of a forensic tool to acquire the memory dump. So a key challenge is maintaining the integrity of physical memory and data while performing the live analysis. In a live system, memory is constantly altered, especially when the first responder interferes, inevitably, with the evidence; this is remedied by suitable forensics principles – see, for example, ACPO Principle 2 which states: "In circumstances where a person finds it necessary to access original data held on a computer or on storage

media, that person must be competent to do so and be able to give evidence explaining the relevance and the implications of their actions."

Live forensics, sometimes referred to as live incident response, is a methodology to extract memory, system processes and network-related information – even terminated and cached processes. Live forensics are necessary today because modern-day trojans, such as Witty, SQL Slammer and the Code Red worm, do not leave their footprint on the storage media and hence their existence can only be detected by analysing the physical memory content.

Performing triage

The majority of the literature agrees that triage should be performed on a live system. The term 'triage' is mainly used in a live forensics context, but we argue that there can be triage in 'dead' forensics – see for example what Bulk_extractor does. Bulk_extractor is a feature extractor tool and is capable of processing different parts of the disk in parallel. It scans disk images and extracts useful information of forensic interest without parsing the file system – hence, it is well-suited for rapid triage and cross-drive analysis.

This paper describes a forensic analysis procedure covering several phases from identification of the 'victim' machine to report generation. We propose a framework for performing digital forensics by setting out the comprehensive steps involved in its triage. We also describe step-by-step procedures to perform forensic analysis on the compromised machines and store all these activities in a database for later use. Triage in the proposed framework is performed after the data acquisition and before the detailed analysis phase.

Related work

Forensic analysis cannot be properly performed if the system has been powered off because the memory content containing useful information about the running processes and opened ports are lost. In addition, it is nearly impossible for the investigators to perform investigation and analysis on encrypted data. Decryption keys can be recovered from the memory dump in both Windows and Linux by using the TrueCrypt tool.[3] It is recommended that drives encrypted using BitLocker should be unsealed first.[4] A 48-digit password file is used to decrypt such data, or this can be done by the brute force method on the physical memory dump file. The Volatility framework comes with a number of plugins that support extracting artefacts from memory such as running processes, open network sockets and network connections along with DLLs and open files for each process. The beauty of the Volatility tool is that it supports extracting artefacts from Windows hibernation files and Windows crash dump files.

Cafegrid can also be used to extract digital evidence from data structures of running programs available in memory on Windows and Linux.[5] It locates the memory structure and provides details about the allocation, de-allocation and swapping of memory during program execution. Such information is quite helpful for forensic analysts while performing live analysis on evidentiary data.

Volatile data

Volatile and non-volatile data acquisition is an important step for digital forensics. Data can be acquired by using different tools – for example, the Linux 'grep' command can be used for searching and analysing the data.[6] But the caveat is that grep cannot work on data unless they are in a suitable text form. Digital evidence can be collected automatically by using the XML-based language XOVAL, which can be used to acquire data, set the primitives for the forensic analyst and then present the data in a declarative way.[7]

Forensic analysis of mobile devices has also been addressed in the contemporary research and to this end, Li et al state that by using certain procedures systematically, malicious code executing on a mobile operating system can be identified and its functionality can be deduced by using Logcat and PCAP.[8] On the contrary, pcap files or memory dumps are not guaranteed to be compatible with grep searches particularly if someone searches for process memory allocation. The pattern of program code, encryption of strings and code obfuscation is analysed through deobfuscation and decompilation tools, which help locate malicious events.

Triage and self-knowledge adaptation can also be used in mobile forensics for data gathering, evidence examination, presentation of documents for evidence and reporting the investigative results about that specific device.[9] Triage on mobile devices is performed in three phases. Evidentiary data is collected in the first phase, followed by removing noise from the forensically relevant data and finally classification of the data is performed. In this phase, the ordering and prediction of relevant data is carried out by using knowledge management and classification algorithms which reduce the size of evidentiary data required for analysis.

Forensic analysis can be performed by running scripts under different forensic tools, such as FDumper and Fundl, stored on a DVD or USB flash storage device.[10] These tools acquire the memory information and perform analysis of open ports, network connections, previously executed commands, user logon history etc. Evidence collection can be done by storing the LECT (Linux Evidence Collection Tool) on a USB flash storage device.[11] After identification of the target machine, it collects the forensic-relevant evidence, such as hashes and timestamps, in the console mode. LECT does not change the memory content to maintain integrity of the evidence. The other true triage tools are kludge (a remote information gathering script), tr3secure (script to grab

volatile information) and TriageIR (tool to collect information from a system). The scripts for all these tools have been posted on Google Code by their authors.

Lempereur *et al* discussed a framework to monitor the behavior of heterogeneous computers at runtime.[12] The status of the network connections in a system and among the host machines, hidden programs and files can be monitored by using the proposed framework. Security policy violations on local and network machines are also monitored and the forensically relevant data can also be stored on local, remote or network machines. During the forensic analysis, the nature of the applications running on the compromised machine and user interaction/activities can be monitored by using Niglant32.[13] It captures the memory image used by the application during its execution and then categorises the instances of user activities with memory allocation. This tool can also be used to perform pattern matching by segregating the text strings and provides a brief history of the application and activities performed by the users.

Cryptographic model

Forensically relevant data integrity is very important for forensic analysis in order to generate accurate output. For this purpose, a cryptographic model is used to maintain the data secrecy while performing digital investigations.[14] The bit stream of the image is examined on the index files and are then encrypted again rather than examining the whole memory content. Static and live digital analysis can be performed simultaneously to enhance understanding of activities and events performed on the victim machine.[15] The memory dump containing a list of processes along with their start time is taken before the system is powered off. The system is subsequently booted using a virtual machine to perform live analysis from the memory dump. This static and live combination provides assistance for digital investigators to analyse memory content as well as the system image of the running machine.

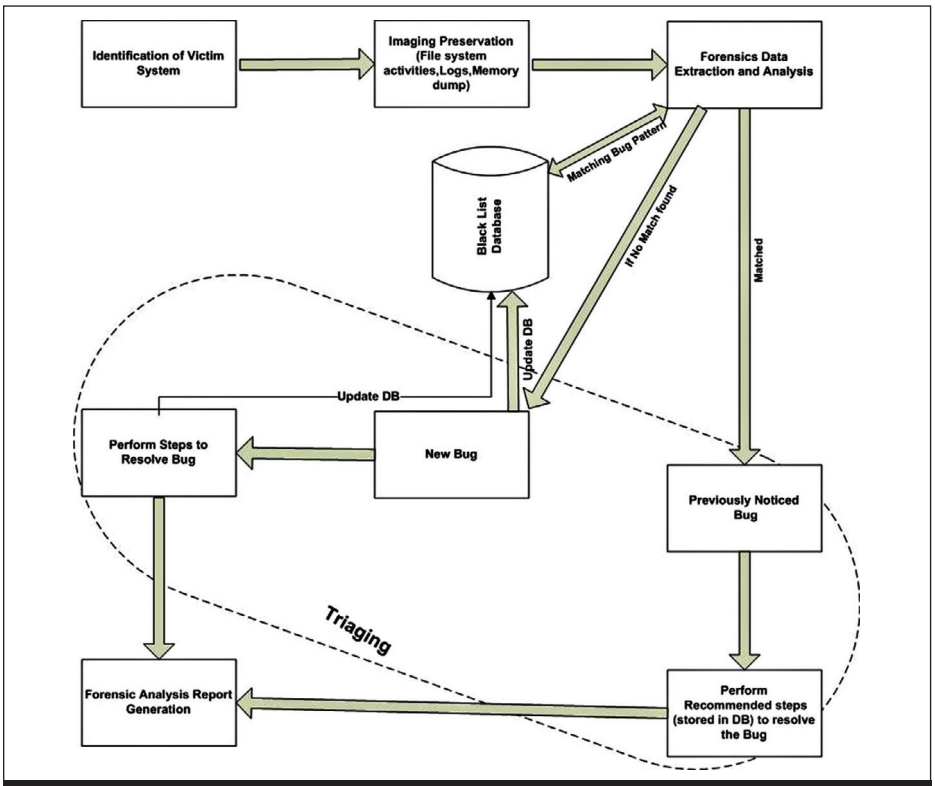


Figure 1: Proposed digital forensics triage framework.

To perform live forensics, different phases such as evidence extraction, analysis, examination and presentation of evidence should be clearly separated to maintain the credibility of the digital forensics.[16] The level of integrity of the evidentiary information pertains to verifying the extent to which the gathered data is affected by the data collection tool. It is essential to ascertain that the acquired data is consistent and fulfils the prerequisites to perform the forensic analysis; for instance, the evidentiary data is not tampered with and the procedure adopted for analysis is persistent enough to produce the accurate analysis. Khan *et al* proposed machine learning approaches for post-event timeline reconstruction.[17,18] The proposed techniques, however, are based on static analysis of the data enclosures. In the static forensic analysis paradigm, the Windows registry plays an important role and forensic scientists obtain valuable information from the registry and use it for extracting evidence.[19]

Proposed framework

Figure 1 depicts our proposed system framework for digital forensics triage that consists of the following five phases:

- Identification and isolation of the victim machine.
- Data preservation – imaging, memory dump, log files, file system activities etc.
- Extraction of data important from a forensic perspective.
- Comparison of forensically extracted data with a blacklist database (Triage).
- Evidentiary reports.

Our framework firstly identifies the affected machine and then we obtain a back-up of the machine by preserving the necessary logs and system files that can help understand the nature of the attack. In the next phase, forensically extracted data is analysed and compared with the blacklist database to identify whether we have to perform triaging or it is a new type of attack. This necessitates identification of the sequence of steps necessary to resolve the new attack.

Then finally our framework generates the analysis reports.

It is worth mentioning that most steps in our framework require human intervention. In addition, there are specific tools mentioned for each phase, but this does not mean that the framework cannot work with other tools. Also, there is a mix of open source and commercial tools used in the framework, some of which are suitable for Linux and some others for Windows OS. This is purposely done since most of the corporate networks are mix of Windows and Linux machines, and also on the pretext that the majority of the tools that offer support for either the platforms or their variants are available for cross-platform use. In general, a responder faces serious challenges on a crime scene and needs to be aware of the capabilities of his/her tools and have pertinent information of the suspect system before entering the scene. Moreover, a responder must have an admin account to get access to the suspect computers.

A brief description of each phase of the proposed framework is described below. The tools suggested in the different phases of our proposed framework are neither exhaustive not final and the reader may opt for other tools of his/her choice.

Identification and isolation

The first phase of our framework pertains to the identification and isolation of the affected system. There are different types of attacks, offences and digital crimes and an affected system may be part of a botnet. From the term 'affected system' we mean the affected machine reported by users or the incident response team. In this phase, the victim system can be identified through different means – for example, the user might report abnormal behaviour by application programs or services to the incident response team. From abnormal behaviour we mean that the application program is generating wrong output, services are not loading at all or the system becomes non-responsive etc. Another method of identification could be by monitoring the systems – eg, traffic monitoring, node monitoring, analysing the critical data where it resides or by using network administration tools.

During the identification phase, it is suggested that the victim machine should remain alive, but with the network cable unplugged to isolate the victim machine so that attack can be contained.

Data preservation

After identification and isolation of the victim machine, the next phase of our framework is to take a back-up or image of the memory dump, log files, history files and file system activities of the machine. We take a memory dump and copy the hard drive contents to an external hard drive or USB stick so that we can prevent tampering and alteration of the original data during the forensic analysis. These images are obtained by making a bit-by-bit copy of the hard drive of the suspected machine for a subsequent in-lab analysis. Another way that can be used is to take a memory dump and system drive back-up by connecting another system with the suspected machine on the same LAN. If the victim machine is not part of a critical infrastructure system, then the affected system can be switched off after taking the memory image and data back-up.

Generally, we can take a physical image of the hard drive through hardware and software tools. In our framework, we take a physical image by using software tools – ie, we take the memory dump by using the software PMDump. By using this tool, we can take the contents and running processes of memory in a dump file without stopping the processes. To take physical disk back-up, we use the Windows version of Disk Duplicate (DD) command. The main reason for using this tool is that it is open source and does not require rebooting the system during its execution. Alternatively, FTK imager Safeback, Encase or dd on Linux can be used to create a duplicate copy of the hard disk. The output file generated by the DD command is readable for many other tools and software such as the Windows kernel debugger. In this phase, we just preserve the memory dump and disk images of the victim machine. All the subsequent analyses are performed on the preserved images.

Data extraction

In this phase, we extract the relevant data to perform the forensic analysis. In this regard, we extract the list of the running processes, list of established network connections and the opened ports, a list of applications running on that machine and a list of the users logged-on at time when the system was affected. For this purpose, we use a toolkit Rootkit Revealer which contains different tools. Rootkit Revealer is capable of detecting rootkits in both user mode and kernel mode. To extract information about the running processes, we use the PSTools suite. The kind of information extracted and the tools used to extract that information is shown in Table 1.

The information extracted in this phase helps determine the nature of the attack during the investigative process. We use this information in the subsequent phases.

Triaging

In this phase, we describe the procedure to analyse and resolve an attack. In digital forensic investigations, sometime it takes a lot of time and effort to resolve novel attacks, particularly tracing the nature of a malware attack or locating the data files damaged/deleted by a hacker. Investigating such scenarios necessitates performing all the necessary steps from problem identification to problem resolution. If you have a database that contains the history of previously resolved malware or cyber-attacks (ie, the set of steps taken to resolve them), then it could help save a lot of

S#	Description of extraction activity	Tool used	Attributes extracted
1	List of remotely executed processes	PsExec	Process name, image name (ie, associated files), user name, memory occupied
2	List of files that were opened remotely	Psfile	Filename, path
3	SID of computer and user	PsGetSid	SID of computer, SID of logged on users
4	System information	PsInfo	OS, computer name, shared folders
5	List of running processes	PsList	Process name, image name (ie, associated files), user name, memory occupied
6	List of users which were logged on locally and through resource sharing	PsLoggedOn	Username, workgroup, status (local or remote user)
7	Details of log file events	PsLogList	Event ID, description
8	List of established network connections and all opened and closed ports along with the associated applications	Tcpview	Connection name, open ports plus closed ports for each connection
9	List of DDLs and EXEs loaded into the memory	ListModules	Filename, size in memory, type (DLL or EXE), process name to which it is linked
10	List of auto starting programs and their location. It also shows us the list of programs which are configured to run during the system boot-up along with login details and their entries in the Windows processes. These programs include the start-up folder, Run, runOnce and other registry keys	autorun tool	Application name, path
11	Signatures of the programs and processes	Forensic Toolkit (FTK)	MD5 hashes

Table 1: Attributes of the extracted information.

time during the investigative process. In this regard, we propose using the concept of a special blacklist database, which comprises the following attributes:

- ID (an auto-generated numeric value giving a unique ID for the investigative process).
- Name (name or kind of malware or cyber-attack).
- Description – about the malware and the damage done by it.
- Status (whether it has been resolved).
- Signature (placeholder to store unique signatures of malware/cyber-attack in the form of MD5 hashes).
- Counter-measures (step by step details of the tools and techniques used for resolution of the problem).

In this phase, we take the signature of the affected files that were identified in the previous phase and search for it in our blacklist database. If the signature of any of the affected files is found in the blacklist database, then it means that such an attack scenario (virus, malware, cyber-attack, unauthorised access etc) has already been observed or resolved. In this case, we check the status attribute corresponding to this attack scenario to determine whether it has been resolved in the past or its investigative process is still underway. If its status is found to be resolved, then we look for the counter-measures that were taken to resolve it and we can take the same counter-measures.

We also update the blacklist database periodically with the additional new signatures and information/knowledge that we learn while resolving the malware/cyber-attack. However, if we do not find the signature of the newly observed attack scenario in the blacklist database, then it means that it is a new sort of attack. In this case, we also update our database to reflect that a new attack scenario has been reported and we set its status to 'being resolved'. Afterwards, we follow the steps describe in our framework to resolve this newly observed attack by checking the hidden and unusual files/processes, open sockets and unusual application requests, auto-run applications and suspicious accounts including null accounts and deleted file list. Based on these findings, we perform the following specific steps:

- We perform a full system scan with Norton Anti-virus.
- Complete file system analysis is performed using SleuthKit and PsTool suite. We identify and recover the deleted files.
- We analyse the volatile memory by using the Memoryze and Volatility utilities.
- Log file analysis is carried out using the PsTool suite.
- We check network file ports and connection with the Tcpview software.
- If some files are password protected, then we use lophtcrack and lepton crack utilities for password cracking.
- By using Ollydbg and WinDbg, we analyse and perform the static and behavioural analysis of the new attack.

The use of the above tools against different steps of the analysis procedure has been described as a guide only and a forensic analyst can use tools of his/her own choice for analysing an attack scenario. After performing the aforesaid steps, we also update our database with the successful steps used to resolve the attack. Moreover, the term 'triaging' seems to refer to a certain part of the investigation process.

Evidentiary reports

The final steps of our proposed framework relate to data visualisation and report generation. It is an important fea-

ture of our framework that it generates summary and detailed analysis reports in a descriptive way encompassing every step of the investigative process. It also generates reports for the various events that occurred on the system and the steps (along with their description) taken to resolve the attack. The analysis reports are also generated in XML format so that these can be easily exported into other tools. We can also perform searching on the reports through several attributes such as event, filename and status of attack.

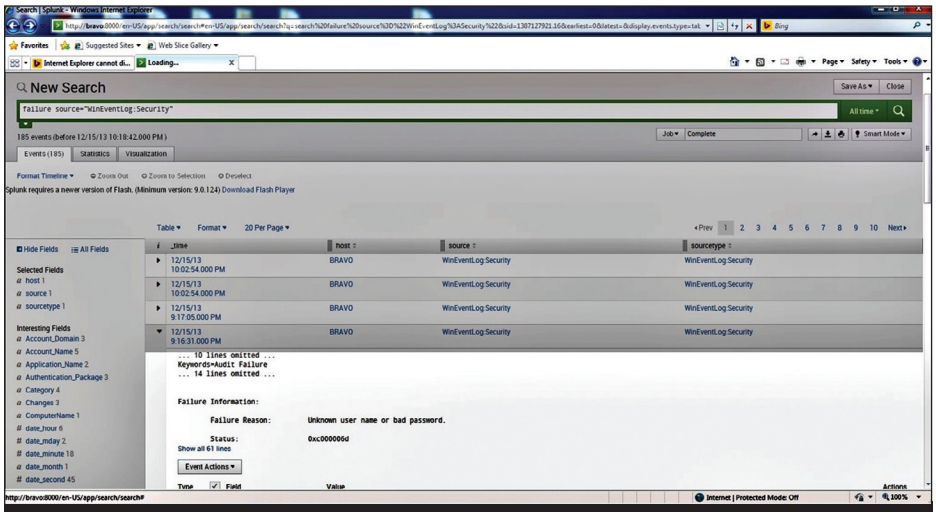


Figure 2: Failed log-on attempt history.

Experimentation and results

In this section, we describe a few test cases and experiments that we designed and ran to show proof-of-concept validation of our framework. The experiments described in this section are based on different methods used to attack systems. These experiments are not exhaustive – a heterogeneous mix of experiments would be required to build a data bank of attack scenarios.

Initially, we start with an empty blacklist database. As and when we obtain an attack scenario and the methodology to analyse that attacks, we store the pertinent procedure adopted to resolve the attack into the blacklist database. Once we obtain several attack scenarios and our blacklist database is populated with the methodologies to resolve those attacks, then it would help automate the triage process. Our blacklist database can provide lots of benefits, such as data sharing, performance improvement and assistance in digital investigations. However, our blacklist database would only be helpful to analyse those cyber-attacks that have been resolved previously where the sequence of steps used to analyse those attacks is stored in the database. Given the novelty of each new cyber-attack, it would always remain a challenge to fully automate the triage process. Furthermore, different experiments listed in this section were tested on different software and tools since no single tool can handle all sorts of attacks.

Experiment 1: System breach attempt

In this experiment, we tried to mimic the scenario where someone repeatedly tries to logon to a system by password guessing. Traces of failed login attempts were found in a Windows security log file. A number of tools are available for log file analysis. We used Splunk version 6.0 (open source software) for this purpose. In this scenario, our system was attached to the network as we wanted to perform analysis of the victim machine by using another machine on the network.

First, we added the Windows Event Logs by selecting the add data option and then by using the option 'collect event log from another machine' provided by Splunk. Although Splunk shows various types of logs available on the system, we only selected the System, Security and Application logs. We performed searching on these logs by tagging the keyword 'failure' and Splunk provided us with all kinds of failures that had occurred during a stipulated timeframe. Through this mechanism, we obtained a detailed view of the failed login attempts as well as the corresponding cause of their failure, such as invalid username or password. A snapshot of the image of failed login attempts in Splunk is provided in Figure 2.

Splunk software also provides us an interface for detailed analysis when the unauthorised attempt was made and whether the system was on domain or workgroup. The additional information includes: account name, event id, authentication package, failure reason

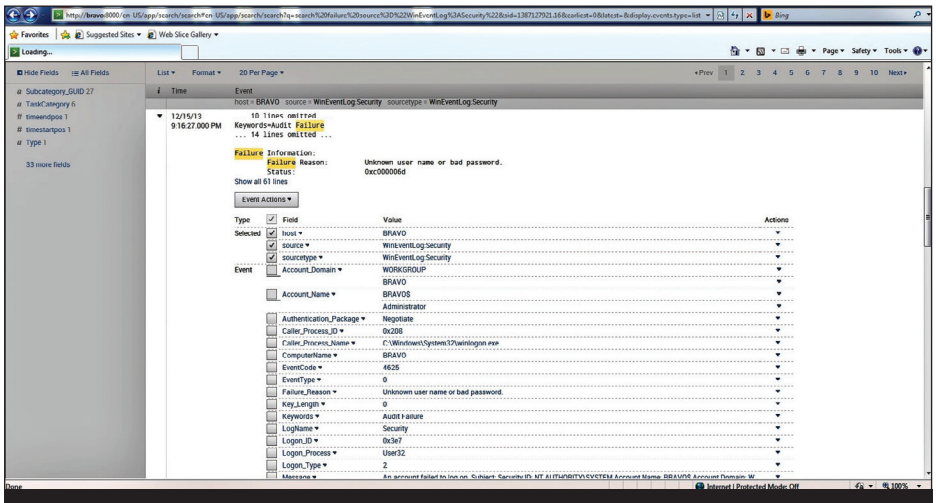


Figure 3: Attributes of the unauthorised attempt.

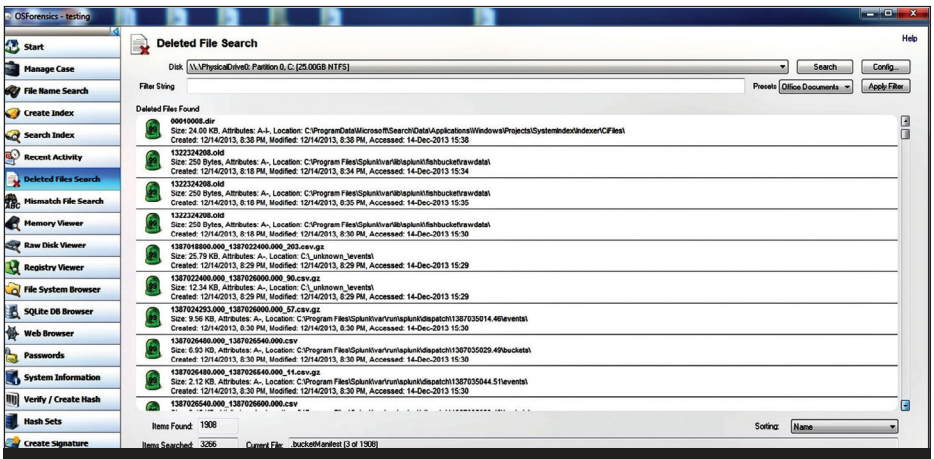


Figure 4: History of file deletion in OS Forensics.

and detailed description about the occurred event. A snapshot showing the key attributes related to an event is shown in Figure 3.

The stepwise findings obtained through the experiment were recorded in the blacklist database.

Experiment 2: File/folder deletion

We conducted another set of experiments to trace the deletion of files and folders. We launched an attack to delete the file 'Thesis.docx' containing valuable information. Traces of the file deletion activity were observed by using the OS Forensics version 2.0 tool. Traces of a deleted file or folder can be observed in OS Forensics by clicking on the 'deleted file search' tab followed by selecting the relevant disk (such as HDD or USB flash storage device). Next, we selected the option 'Office document' from the preset option as our deleted file was a Word document. By clicking on the search button, OS Forensics provides us list of all the deleted files found on the system. We can sort this output by date, name, size, folder etc (see Figure 4).

By double clicking the required deleted file, OS Forensics showed us its detailed information such as creation, modification and access dates along with path of the file where it was residing before deletion. We can also save or export search results of the deleted files/folders in case we need these for evidence collection.

Experiment 3: Remote login attempt

In this experiment, we created an environment to find traces of an illegal attempt to remotely login to a system.

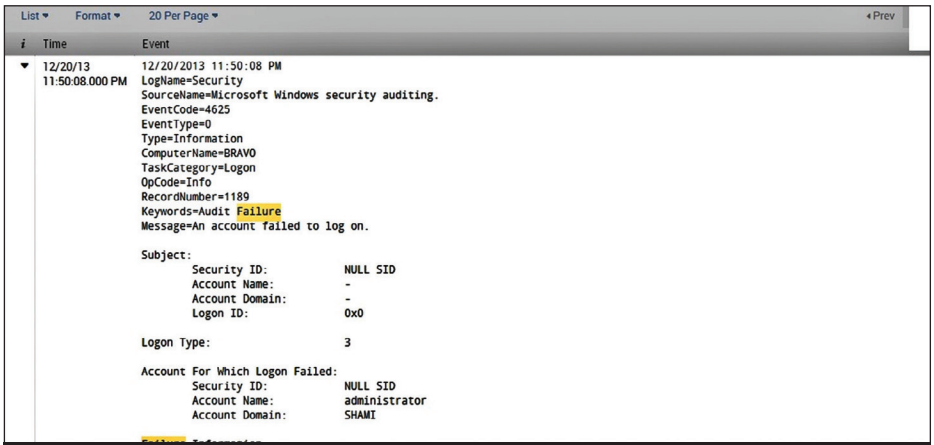


Figure 5: Remote login attempt details.

The trace of this activity was found in Windows Event Logs. We again used Splunk and performed a sequence of steps to analyse the unauthorised attempt made on the system.

First, we added the Windows event logs through the 'add data' option in Splunk. We searched Security Logs and then we tagged the keyword 'remote login failure', and as a result Splunk provided us all kinds of failures that occurred during a specific timeframe. Through this approach, we obtained a detailed view of the failed remote login attempts that were made as well as name of the systems from where those attempts were launched, along with the causes of their failure such as invalid username or password. The additional information provided by Splunk includes: account name, event ID, authentication package, failure reason, detailed description about the occurred event and the domain name from where such attempts were made. A snapshot illustrating the key attributes related to failed remote login activity is shown in Figure 5.

Experiment 4: Opened port history

In this experiment, we launched an attack to open the HTTP/UDP port number 20. This port was originally closed to restrict data transfer using FTP. To find the traces of the opened port, we used an open source tool Zenmap which is a GUI version of Nmap. We ran Zenmap on the network and entered the IP address of the machine whose port was opened. The IP address used in this case was '192.168.0.2'. Zenmap provided us output of the scanned IP. A snapshot illustrating this output is shown in Figure 6.

The key features of Zenmap are that it probes computer networks and provides host discovery and service, protocol, domain and operating system detection. The Zenmap output consists of OS name, workgroup,

computer name and OS version of the scanned system. Zenmap also provides list of all the ports showing the following attributes:

- Port number.
- State (open or closed).
- Service running on the port.

Experiment 5: Retrieving flash storage device history

We conducted another experiment to trace the usage history of a USB flash storage device on a system. These traces were found in the Windows registry. We used the open source tool USBDeview version 2.27 for this purpose. It provided us with a detailed view of device name, description, device type, serial number, last plugin time with date, drive letter assigned to the USB storage device, its version and device description etc (see Figure 7).

USBDeview also provides an XML report about the connected devices. The detailed description about a device can be viewed by clicking 'property' tab of this tool. A snapshot showing properties of a device is illustrated in Figure 8.

We performed the above-mentioned five experiments for validation of our proposed framework. The tools and software used in the experimentation did not affect memory contents while producing the desired output. Through this set of experiments, we also built our blacklist database for future reference.

Experiment 6: SQL injection traces

In this experiment we mimicked a scenario in which we discovered traces of SQL injection. We prepared a database server using Oracle 11g R2 and connected to SQL Plus as SYSDBA. We verified that the initialization parameter AUDIT_TRAIL is set to DB_EXTENDED and the AUDIT_SYS_OPERATIONS parameter is set to TRUE (as shown in Figure 9).

Then we executed the following set of commands to enable the audit trial for creating, altering and dropping the table in our database server and applied the policy as shown below.

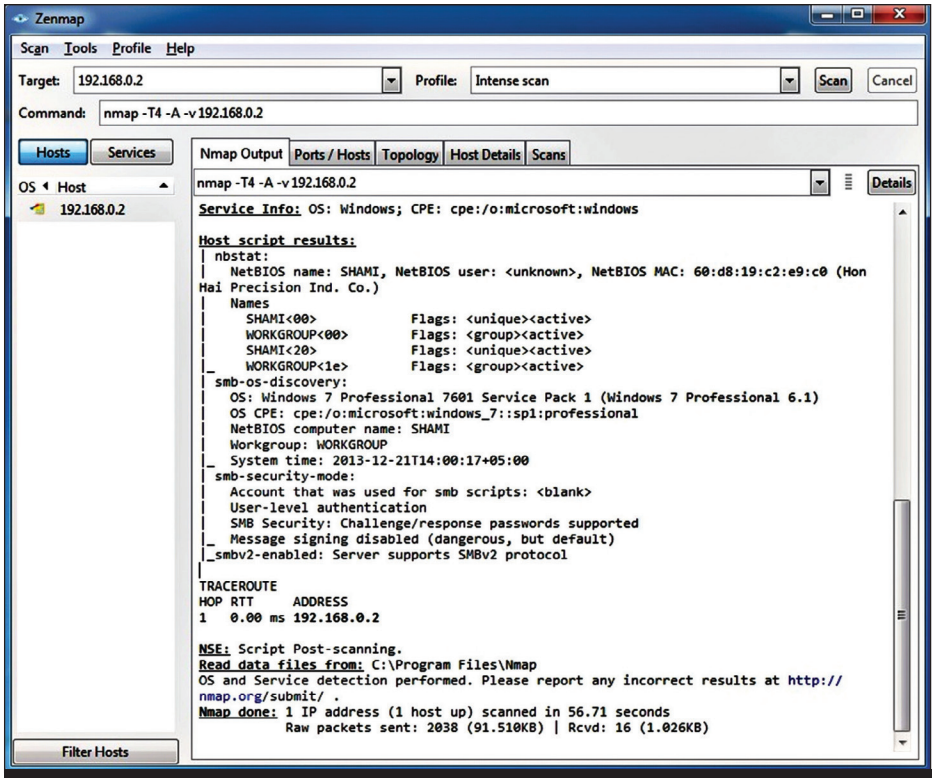


Figure 6: Zenmap output.

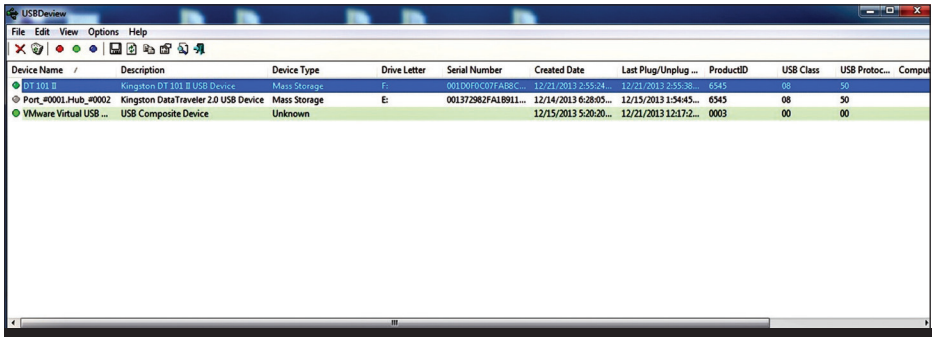


Figure 7: Details of USB storage device usage history.

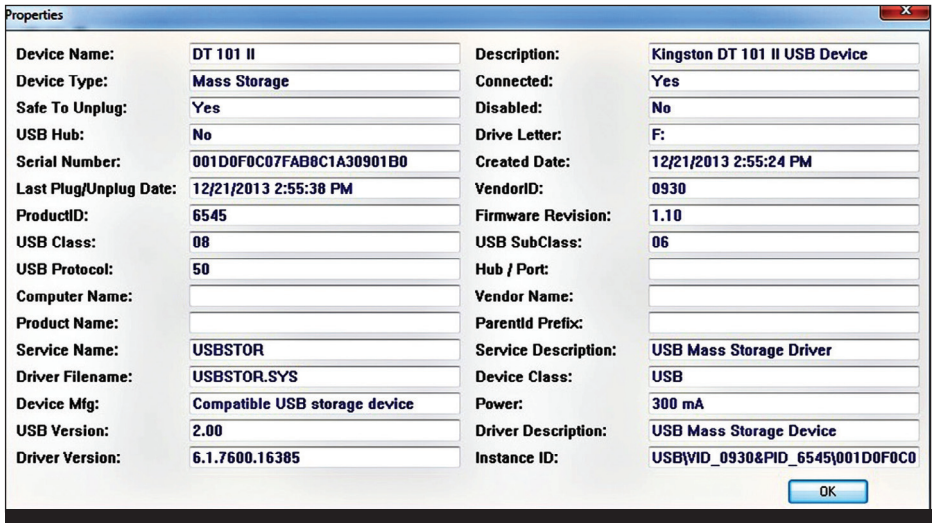


Figure 8: Device details in USBDeview.

SQL> show parameter audit		
NAME	TYPE	VALUE
audit_file_dest	string /u01/oracle/admin/db01/adump	
audit_sys_operations	boolean	TRUE
audit_syslog_level	string	
audit_trail	string	DB_EXTENDED

Figure 9: Database settings for the SQL injection experiment.

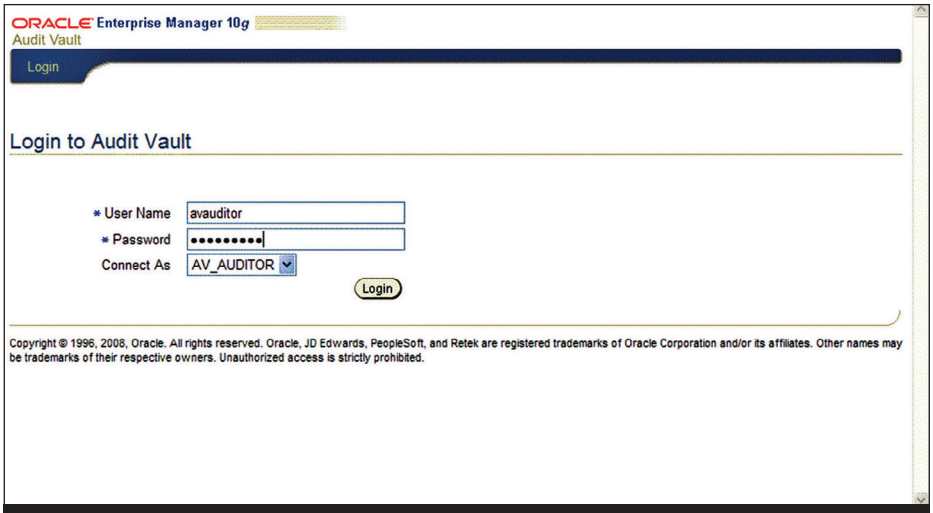


Fig. 10: Snapshot of the Audit Vault login form.

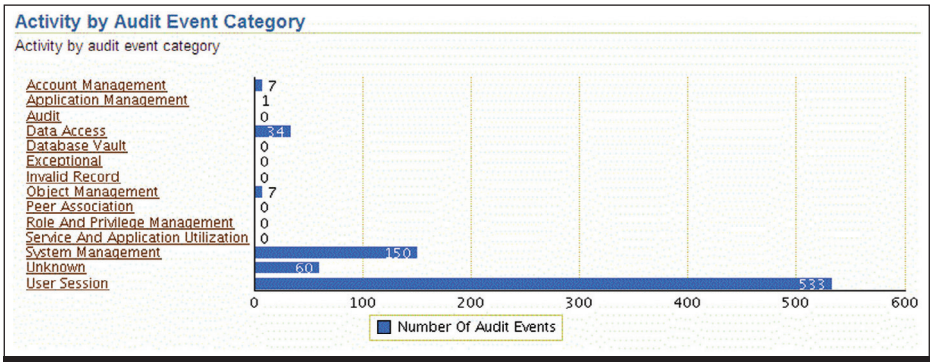


Figure 11: Activity by Audit Event Category.

```
SQL> Audit alter any table by access;
Audit succeeded.
SQL> Audit create any table by access;
Audit succeeded.
SQL> Audit drop any table by access;
Audit succeeded.
```

Next we mimicked the injection scenario and injected a SQL script that created two extra tables in our database server. We ran a script inject_audit.sql (shown below) to create audit records in the database server.

```
SQL> @inject_audit.sql
SQL> @change_schema.sql
SQL> connect system/oracle1
Connected.
SQL> create table hr.emp1 as select *
from hr.employees;
Table created.
SQL> create table hr.emp2 as select *
from hr.emp1;
SQL> exit
```

To find traces of injected SQL, we logged on to the Audit Vault console as the 'avauditor/oracle12#' user by selecting the value AV_AUDITOR in the 'Connect As' drop-down menu option (Figure 10). We scrolled to the far lower portion of the 'Overview' screen which contained summary entries for audit events. Figure 11 shows Activity by Audit Event Category. By clicking on the hyperlink for the Audit Event Category named 'Data Access', we can view the Data Access report as shown in Figure 12. By Logging to the Audit Vault console as the avadmin/oracle12# user and navigating to Management Warehouse, the 'Refresh History' tab shows start time, duration and status of the tasks performed by the injected SQL (Figure 13).

After performing this experiment, we were able to trace the successful and failed SQL commands that were performed through SQL injection. This event was also recorded in our blacklist database so that, in future, if any such attack reoccurs on our systems or we have to find traces of SQL injection, we would query our database and would be able to find the traces by performing the previously recorded steps.

Conclusion and future work

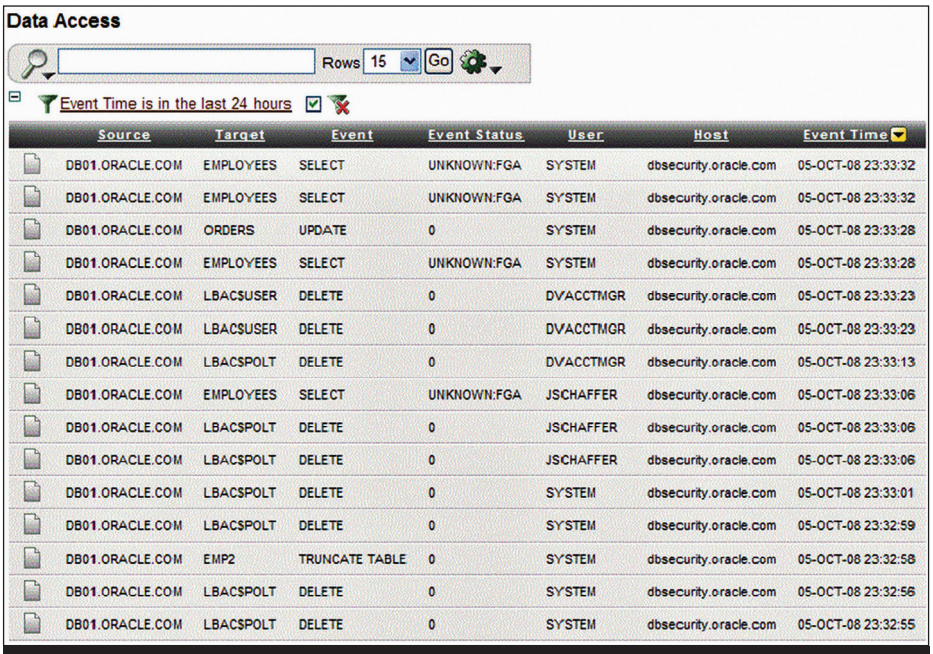
We have presented a framework that can help automate the triage process in digital forensics. Several tools and techniques were used to investigate the disk images, registry content, log analysis and deleted file history followed by recording these events in a database. We restricted our framework to a computer system that has been reported compromised by the user. Too much generalisation was avoided as it can lead to a system that is unworkable for most scenarios.

As a future dimension to this work, we intend to include analysis of mobile devices, which substantially differ from computer forensic analysis and have different storage and content media. We intend to investigate the mobile device by using our framework and by focusing on SMS history, call record data, viral attacks and deleted files/images.

About the authors

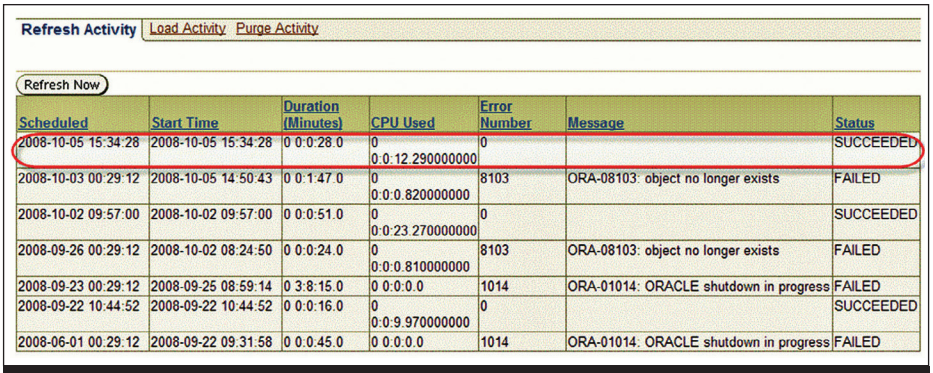
Muhammad Shamraiz Bashir obtained an MS degree in Computer Science from Shaheed Zulfikar Ali Bhutto Institute of Science and Technology (SZABIST), Islamabad, Pakistan. He has over six years of experience in the IT Industry. His research interests are in the fields of cyber-security and digital forensics.

Muhammad Naeem Ahmed Khan obtained a D.Phil. degree in Computer System Engineering from the University of Sussex, UK. His research interests are in the fields of software engineering, cyber administration, digital forensic analysis and machine learning techniques.



Source	Target	Event	Event Status	User	Host	Event Time
DB01.ORACLE.COM	EMPLOYEES	SELECT	UNKNOWN:FGA	SYSTEM	dbsecurity.oracle.com	05-OCT-08 23:33:32
DB01.ORACLE.COM	EMPLOYEES	SELECT	UNKNOWN:FGA	SYSTEM	dbsecurity.oracle.com	05-OCT-08 23:33:32
DB01.ORACLE.COM	ORDERS	UPDATE	0	SYSTEM	dbsecurity.oracle.com	05-OCT-08 23:33:28
DB01.ORACLE.COM	EMPLOYEES	SELECT	UNKNOWN:FGA	SYSTEM	dbsecurity.oracle.com	05-OCT-08 23:33:28
DB01.ORACLE.COM	LBAC$USER	DELETE	0	DVACCTMGR	dbsecurity.oracle.com	05-OCT-08 23:33:23
DB01.ORACLE.COM	LBAC$USER	DELETE	0	DVACCTMGR	dbsecurity.oracle.com	05-OCT-08 23:33:23
DB01.ORACLE.COM	LBAC$POLT	DELETE	0	DVACCTMGR	dbsecurity.oracle.com	05-OCT-08 23:33:13
DB01.ORACLE.COM	EMPLOYEES	SELECT	UNKNOWN:FGA	JSCHAFFER	dbsecurity.oracle.com	05-OCT-08 23:33:06
DB01.ORACLE.COM	LBAC$POLT	DELETE	0	JSCHAFFER	dbsecurity.oracle.com	05-OCT-08 23:33:06
DB01.ORACLE.COM	LBAC$POLT	DELETE	0	JSCHAFFER	dbsecurity.oracle.com	05-OCT-08 23:33:06
DB01.ORACLE.COM	LBAC$POLT	DELETE	0	SYSTEM	dbsecurity.oracle.com	05-OCT-08 23:33:01
DB01.ORACLE.COM	LBAC$POLT	DELETE	0	SYSTEM	dbsecurity.oracle.com	05-OCT-08 23:32:59
DB01.ORACLE.COM	EMP2	TRUNCATE TABLE	0	SYSTEM	dbsecurity.oracle.com	05-OCT-08 23:32:58
DB01.ORACLE.COM	LBAC$POLT	DELETE	0	SYSTEM	dbsecurity.oracle.com	05-OCT-08 23:32:56
DB01.ORACLE.COM	LBAC$POLT	DELETE	0	SYSTEM	dbsecurity.oracle.com	05-OCT-08 23:32:55

Figure 12: Snapshot showing the Data Access report.



Scheduled	Start Time	Duration (Minutes)	CPU Used	Error Number	Message	Status
2008-10-05 15:34:28	2008-10-05 15:34:28	0 0:0:28.0	0 0:0:12.290000000	0		SUCCEEDED
2008-10-03 00:29:12	2008-10-05 14:50:43	0 0:1:47.0	0 0:0:0.820000000	8103	ORA-08103: object no longer exists	FAILED
2008-10-02 09:57:00	2008-10-02 09:57:00	0 0:0:51.0	0 0:0:23.270000000	0		SUCCEEDED
2008-09-26 00:29:12	2008-10-02 08:24:50	0 0:0:24.0	0 0:0:0.810000000	8103	ORA-08103: object no longer exists	FAILED
2008-09-23 00:29:12	2008-09-25 08:59:14	0 3:8:15.0	0 0:0:0.0	1014	ORA-01014: ORACLE shutdown in progress	FAILED
2008-09-22 10:44:52	2008-09-22 10:44:52	0 0:0:16.0	0 0:0:9.970000000	0		SUCCEEDED
2008-06-01 00:29:12	2008-09-22 09:31:58	0 0:0:45.0	0 0:0:0.0	1014	ORA-01014: ORACLE shutdown in progress	FAILED

Figure. 13: Snapshot showing tasks performed by SQL injection.

References

1. ACPO good practice guide for digital evidence version 5'. ACPO (Association of Chief Police Officer), Metropolitan Police Service, UK, 2012.
2. Malin, CH; Casey, E; Aquilina, JM. 'Malware forensics: investigating and analysing malicious code'. Syngress, 2008.
3. Bolagh, S; Pondelik, M. 'Capturing Encryption Keys for Digital Analysis'. In Proceedings 6th International Conference on Intelligent Data Acquisition and Advanced Computing Systems(IDAACS), pp.759-763, Prague, 15-17 September 2011.
4. Dija, S; Balan, C; Anoop, V; Ramani, B. 'Towards Successful Forensic Recovery of BitLocked Volumes'. In Proceedings 6th International Conference on System of Systems Engineering (SoSE), pp.317-322 , Albuquerque, NM, 27-30 June 2011.
5. Chan, E; Venkataraman, S; Gutierrez, A; Campbell, RH. 'Characterizing Data Structures for Volatile Forensics'. In Proceedings of 11th International Workshop on Systematic Approaches to Digital Forensic Engineering(SADFE), pp.1-9, Washington, DC, US, 2011.
6. Balaz, A; Hlinka, R. 'Forensic Analysis of Compromised Systems'. In Proceedings of 10th Emerging eLearning Technologies & Applications (ICETA), pp.27-30, Stara Lesna, 8-9 Nov 2012.
7. Barrere, M; Betarte, G; Rodriguez, M. 'Towards machine-assisted formal procedures for the collection of digital evidence'. In Proceedings of 9th Annual International Conference on Privacy, Security and Trust (PST), pp.32-35, Montreal, QC, 19-21 July 2011.
8. Li, J; Gu, D; Luo, Y. 'Android Malware Forensics: Reconstruction of Malicious Events'. In Proceedings of 32nd International Conference on Distributed Computing Systems Workshops (ICDCSW), pp.552-558, Macau, 18-21 June 2012.
9. Marturana, F; Me, G; Berte, R; Tacconi, S. 'A quantitative approach to Triaging in Mobile Forensics'. In Proceedings of 10th International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom), pp.582-588, Changsha, 16-18 Nov 2011.
10. Yang, Chung-Huang; Yen, Pei-Hua. 'Fast Deployment of Computer Forensics with USBs'. In Proceedings of International Conference on Broadband, Wireless Computing, Communication and Applications (BWCCA), pp.413-416, Fukuoka, 4-6 Nov 2010.
11. Choi, J; Savoldi, A; Gubian, P; Lee, Seokhee; Lee, Sangjin. 'Live Forensic Analysis of a Compromised Linux System Using LECT (Linux Evidence Collection Tool)'. In Proceedings of International Conference on Information Security and Assurance (ISA), pp.231-236, Busan, 24-26 April 2008.
12. Lempereur, B; Merabti, M; Shi, Q. 'Information Flow Monitoring:

Model, Policy, and Analysis'. In Proceedings of International Conference on Developments in E-systems Engineering (DeSE), pp.227-232, Dubai, 6-8 Dec 2011.
13. Olajide, F; Savage, N; Akmayeva, G; Shoniregun, C. 'Extracting Forensically Relevant Information from Windows Applications'. In Proceedings of International Conference on Information Society (i-Society), pp.423-428, London, 25-28 June 2012.
14. Law, FYW; Chan, PPF; Yiu, SM; Chow, KP; Kwan, MYK; Tse, HKS; Lai, PKY. 'Protecting Digital Data Privacy in Computer Forensic Examination'. In Proceedings of 6th International Workshop on Systematic Approaches to Digital Forensic Engineering (SADFE), pp.1-6, Oakland, CA, 26 May 2011.
15. Mrdovic, S; Huseinovic, A; Zajko, E. 'Combining Static and Live Digital Forensic Analysis in Virtual Environment'. In Proceedings of 12th International Symposium on Information, Communication and Automation Technologies (ICAT), pp.1-6, Bosnia, 29-31 Oct 2009.
16. Wang, L; Zhang, R; Zhang, S. 'A Model of Computer Live Forensics Based on Physical Memory Analysis'. In Proceedings of 1st International Conference on Information Science and Engineering (ICISE), pp.4647-4649, Nanjing, 26-28 Dec 2009.
17. Khan, MNA; Chatwin, CR; Young, RC. 'A framework for post-event timeline reconstruction using neural networks'. Digital Investigation, 2007, 4(3), 146-157.
18. Khan, MNA. 'Performance analysis of Bayesian networks and neural networks in classification of file system activities'. Computers & Security, 2012, 31(4), 391-401.
19. Kim Y; Lee S; Hong D. 'Suspects' data hiding at remaining registry values of uninstalled programs'. In ICST Proceedings of the 1st international conference on Forensic applications and techniques in telecommunications, information, and multimedia and workshop, January 2008.

Not safe for work

Russell Horton, Elitetele.com

Many companies are introducing Bring Your Own Device (BYOD) schemes into their working practices. Despite the numerous benefits this offers, a significant number of businesses are still not equipped with the necessary protection to implement these schemes in the right way. Understandably, businesses want to give staff access to devices that can improve productivity and mobility, but they need to ensure these are equipped and updated with the latest technology to secure corporate networks from increasingly complex threats.

Russell Horton

Bad behaviour

According to a recent survey of 2,000 UK workers by Elitetele.com, 21% of staff had accessed pornography and/or adult websites on a personal device which was also used for work.[1] Alarmingly, 25% claimed they were unaware visiting such websites could lead to their device being infected by malicious viruses or bugs that could compromise the data on their device.

Opening such a gateway into a business can have severely damaging effects on its ability to operate, particularly since 24% of women and 21% of men admitted they would be too embarrassed to inform their employer if a breach of security had indeed happened as a result of accessing such content. With 7% of both sexes claiming they would only disclose this information after one week and 2% within a month, in this time a business could be severely compromised by cyber-criminals who have the opportunity to peruse poorly secured corporate networks at leisure because nobody has raised the alarm, and nothing is being done to stop them.

Coupled with Gartner's findings that a quarter of business users admitted to having had a security issue with their private device, it is clear more needs to be done to manage BYOD schemes effectively.[2] With the proliferation of mobile devices within the workplace and increased mobility among staff, businesses must address this issue now. The need to provide solutions to these new working norms doesn't mean businesses should jeopardise their security practices.

Confused about security

Many organisations are still confused about the security and cost implications of BYOD. They want to know how to ensure data protection and prevent malware and malicious apps being downloaded, as well as how to understand usage allowances. There is a real need for specialist advice and guidance to help organisations make the most of the benefits of BYOD and ensure the security of their information at the same time. The following tips can help any business seamlessly implement an effec-