Identifying Malicious Code Infections Out of Network

Ken Dunham
Identifying Malicious Code Infections Out of Network

GIAC (GCFA) Gold Certification

Author: Ken Dunham, kend@kendunham.org
Advisor: Kees Leune

Accepted: July 25, 2011

Abstract
Best practices have evolved within the forensic industry over the past few years to address an emerging need for organizations to properly handle malicious code incidents. While this area of forensics is increasingly strong, the industry at large struggles with how to approach forensic analysis of images that are not from one's own network (e.g. image sent to consultant for analysis). Furthermore, many forensic practitioners lack tools and tactics to exhaustively research and report on malicious code infections that may exist on such media. Real-world case studies (sanitized) are used in this report to identify challenges that forensic analysts face given such tasks and best practices for researching malicious code events on Windows computers.
1. Introduction

Forensics is a complex subject, where details matter greatly. Even more complicated are investigations where forensic methods are used to further understand, identify, capture, and mature and understanding of a malicious attack that may have taken place on a computer. This is increasingly common in the commercial sector where images of an infected drive are made during incident handling, then analyzed post-incident to fully understand an attack.

Such investigations often focus first on identification components, such as files created and egress events, to aid in post-incident identification and isolation of any possible related infections on a network. Increasingly, such investigations also seek to further understand the full scope of an integrity compromise. This frequently includes deeper research into malcode functionality, especially in relationship to what sensitive data may have been exfiltrated from a network. Regulatory changes in the US and other countries appear to be driving some of the momentum in this area of ongoing research.

Companies are now increasingly contracting with commercial organizations to obtain remote forensics, where an image is shipped to the company for analysis. This type of arrangement is inherently more complicated than that of forensics within a known network. External companies now have the challenge of identifying how media is captured and processed before it is received, what the original operating context may have been for the image, and/or details potentially related to a compromise. Analysts working within a company normally have a wealth of information and personal relationships with key individuals to answer all such questions, while contractors and remote commercial organizations do not.

1.1. Common Procedures for Malcode Forensics

1.1.1. Bag and Tag

Traditional Bag and Tag methods used by law enforcement are proven, defendable, and hold up in court. With advancements in malicious code, some threats are
now *ONLY* resident in volatile RAM. Traditional methods will collect little to no data on such threats. One method for countering this emergent risk for investigations is to perform a physical dump of memory to a file before following up with traditional methods for evidence seizure and processing (Malin, 2008). Normally a dump of what the system reports, using common tools, is also performed along with a physical dump of RAM, for comparison later to prove rootkit functionality subverting a system. For example, on Windows Netstat and Fport can be used to dump data to a file about processes and images in memory. This data is later compared to the data analysis found in a physical dump of memory, to identify possible hidden or injected processes manipulated by rootkits.

Memory analysis is more advanced and may not be as easily defended in court. Acceptance of such tools and tactics is still underway in US courts at the time of this writing. Advanced memory analysis will likely become an accepted standard for forensic investigations over time if such research is performed with well documented and reliable tools and tactics and handling of evidence.

1.1.2. Chain of Custody

Traditional courses regarding forensics are centered around law enforcement techniques and what is known as "Chain of Custody". Chain of Custody refers to how evidence is collected and handled throughout the process of an investigation (Oriyano, 2010). Several key questions are identify by Oriyano related to Chain of Custody, around how evidence is collected, obtained, when collected, individuals that handled the evidence, reasons for each person to handle the evidence, and how evidence travels and/or is stored.

Chain of Custody events are always recorded within a log file to document all aspects of chain of custody, with no gaps in handling or the chain of custody through an investigation. This is a critical component to defend in court a verifiable collection, processing, and handling of data. Due to expenses and extra effort involved, this forensic procedure is only performed when legal actions are to follow a forensic investigation of data collected and analyzed.

Ken Dunham, kend@kendunham.org
An out of network malware related event chain of custody is rarely used. If chain of custody is involved for the client a copy of the original data is made for the contractor or outside party analyzing the data. Normally chain of custody investigations involve a known and rigid set of evidence handling, log documentation, and access controls all managed by the company performing the investigation. In the case of working with an out of network malware event, the remote contractor often has little to no context on where the data came from or how it was handled before it was received by the contractor. Contractors and outside parties assisting in such an investigation often have very little information about chain of custody or any collection, analysis, possible changes done to a drive, people involved, or context of an investigation. Naturally this can significantly sidetrack an investigation or introduce major challenges in how to properly contextualize and analyze such data.

1.1.3. Images – Data Collection & Processing

How an image is collected and handled, and possibly repackaged, is one of the first challenges an out of network consultant faces when working with data provided by a client. Even if questions are asked of the client, often little to no data is provided to adequately answer questions related to data that may have been handled in a variety of creative or unexpected ways.

Write Blockers are normally used as part of a forensic operation to ensure that no write actions take place against a drive of interest (NIST, 2008). In some cases clients do not use a write-blocker, resulting in anecdotal writes to a drive that may impact interpretation, question the integrity of data, or cause confusion for an out-of-network contactor analyzing such data.

Data collection is normally done with standardized and proven (defendable) tools such as FTK, Helix, Backtrack (Wiles, 2007), or Linux DD. In some cases an image is made of an operating system using other solutions such as an Acronis TrueImage backup, Norton Ghost, AccessData, Macintosh DMG, or other utility. Naturally every tool and image type has configurations and usage specific to itself. Instead of having to deal with one image tool and type in an organization, contractors

Ken Dunham, kend@kendunham.org
may leverage a wide range of tools to properly handle all the image data types submitted by clients.

In some cases a company handling an investigation may modify a drive of sensitive data and then repackage an image in a different manner from an original image technique and then send that to a contractor for analysis, further complicating data collection, formats, and handling thereof.

1.1.4. Hash Values
Computing hash values for evidence is a key starting point for all investigations (Wiles, 2007). When handling an investigation internally to a company network the employee can access images used for production machines to compute hash values, to then compare against hash values collected from a forensic investigation of a compromised host. When out of network this is not possible as access to the image used for production is normally not provided. Naturally out of network contactors can perform more generic hash computations and comparisons against common Windows file components but will have no true baseline for the machine being analyzed and what it should have had on it before a breach of integrity may have taken place. As a result hash comparisons are more difficult for out of network contractors.

The National Software Reference Library (National Software Reference Library (NSRL) , 2010) and NDIC are detailed databases containing hashes of software packages, useful in deconflicting hash values from a computer of interest. Free downloads of hash sets are available on the website.

Similarly, the US Department of Justice (USDOJ, 2011) also maintains a hash list that can be used federal agencies and law enforcement for hash comparisons. This dataset is not freely available and must be requested on official letterhead as specified on their website. Some software, such as that authored by Access Data, automatically eliminate from a hash list all known good hash values correlated to whitelists such as the NSRL, NDIC, and USDOJ.

A wealth of other sources may also help to supplement whitelist and/or blacklist values for MD5s. FileAdvisor (Bit9, 2009) is a popular site with over 7 Billion hashes

Ken Dunham, kend@kendunham.org
entered to date. Bit9 also makes available a Windows desktop utility to quickly check a file of interest against their database, with an account required to perform such actions. This account is free, making it easy for investigators using a copy of an image in a live environment to install this application to quickly spot check files of interest against the Bit9 hash database.

For hashes specific to malicious code another great source is Team Cymru (Team Cymru, 2011). This free service enables users to query their database for an MD5 or SHA1 value of interest. If found it is malicious code that Team Cymru has in their database and has analyzed, with results that include anti-virus detection details.

Because hash values are an important part of a forensic investigation to locate unknown or questionable files of interest in a malcode investigation, scripts within Perl and Python are commonly used to help automate the process in conjunction with one or more hash databases.

1.1.5. Live Forensics

Live forensics can be performed in two manners, via an agent running on an enterprise computer or locally (Adelstein, 2006), or by running a possible compromised machine in a lab environment. A number of providers now have agent deployed solutions to install an agent on a computer prior to an incident. This enables the agent to baseline the system, and help with incident response and triage. For example, EnCase Enterprise contains such an agent making it intuitive and fast to identify which processes and images are of a known MD5 and which are not. A remote analyst is then able to quickly focus on processes and images on a disk that are of interest.

An out-of-network contractor will not normally be provided remote live forensics access. As a result the only other option is for the contractor to attempt to run the image in a lab environment. This can be done within a virtualized environment or a native environment. When investigating a malicious code environment the more failsafe method for analysis of such an image is on a native machine, as some malicious codes behave differently when run within a virtualized environment. Other checks may also

Ken Dunham, kend@kendunham.org
need to be subverted, such as modifying the MAC address to match that of the original image being analyzed, so that code behaves as it should.

When working as an out-of-network contractor tools like MountImage Pro (GetData, 2011) can be very useful. MountImage Pro is capable of mounting the following images: EnCase .E01, .L01, AccessData FTK .E01, .AD1, Unix/Linux DD and RAW images, Forensic File Format .AFF, SMART, ISO (CD and DVD images), VMWare, ProDiscover, Microsoft VHD, and Apple DMG.

1.2. Challenges of Malicious Identification

Analysis of malicious code is a complex subject. A variety of materials now exist for free, and commercially, to help individuals learn how to analyze malicious code. Lenny Zelter's work (Lenny Zeltzer, 2011) with SANS and on his website are one of the most well known benchmarks of the basics related to malcode analysis training available at the time of this writing.

Experienced malcode analysts know what to look for when analyzing certain types of malcode and/or handling incidents, such as a drive-by exploitation event. This is, in part, because they have experience that helps them create a scaffolding approach in their thinking and referencing of behaviors and other data related to malcode. Essentially, an experienced analyst is able to inspect code of interest within a malicious context that enables them to quickly identify suspect or known malicious behaviors and/or relationships to formerly known malicious attacks.

1.2.1. Relational Analysis

Looking for connections between computers is an excellent way to identify maliciousness, especially for network aware worms. Relational analysis can be performed to identify trust relationships between an infected drive being analyzed and other computers (Malin, 2008). Looking for administrative accounts and remote logins, mounted network drives, and IPs that have had connections with the computer are all valuable data points for identifying relationships to other computers, actor aliases, or IPs. This may also involve some reconstruction of data from a drive to further piece together disparate data points and other data that may be on a network, of interest to an

Ken Dunham, kend@kendunham.org
investigation. In such cases an out of network contractor must work with the client to obtain additional data of interest and hope that data collection and handling is done properly to support additional research.

2. Mounting the Image

Mounting an image involves taking a copy of evidence and then mounting it on a lab analysis computer. In this example a laptop is used for analysis of an image to aid in mobile forensic needs. As a result an eSATA solution is used to mount a drive through a PCMCIA card solution onto the laptop.

2.1. Physical Drive Mounting

For SATA drives being connected to a laptop an eSATA device is used to physically mount the drive.

Figure 2.1: eSATA solution for quickly mounting a SATA drive to a laptop.

Using an eSATA solution to mount a drive potentially puts the host computer at risk for malicious code infection. Always mount on a safe lab machine with no production assets in case mistakes are made in handling or unexpected malicious events follow mounting of a suspect drive.

Ken Dunham, kend@kendunham.org
2.2. **Image Mounting**

Images may need to be converted prior to mounting, depending upon what software is available to work with the existing format of data submitted by a client. For example, an Encase image (E01) files may need to be converted to another format, such as DD, if Encase is not used by a contractor in analysis of images.

2.2.1. **FTK Imager – Free and Flexible Mounting**

FTK Imager (AccessData, 2011) is free and works with a large number of formats.

As shown in the image below, FTK Imager is able to interpret the format of a large number of image types, making it easy to mount and manage archives of an unknown format. If the format is not supported by FTK Imager a variety of header and file inspection techniques may be utilized in an attempt to discover the format and tools required for mounting or conversion.
Figure 2.2: File data in a numerical sequence with no known image extension.

To mount data using FTK Imager mount the image as a **logical** drive. Another mounting method is to also perform a **physical** mount to then browse files on the file system using a variety of other tools and tactics. Once mounted, the analyst browses files on the disk, such as .001 as shown in the image above. Notice that FTK defaults to physical drive, when logical is needed for mounting and working with the drive on the local system for the tactics explained in this report:

![Select Source](image)

Figure 2.3: FTK options for mounting evidence type (select logical).

Once the image is mounted the analyst is able to browse the mounted image easily.

Ken Dunham, kend@kendunham.org
Identifying Malicious Code Infections Out of Network

3. Static Triage for Malcode

Looking for malicious code on a compromised system starts with a static analysis of the operating system, files, and data within files accordingly. The process must be systematic, following general forensic principles such as a hash list of files, and then diving deeper into areas of interest. The process begins with a positive identification of the operating system to ensure it's what is expected from the client and to fully understand the context of what is being investigated.

Figure 2.4: FTK viewing a mounted image.

FTK Imager also makes it easy to right-click on a mounted partition to export a hash list and files from the mounted image.

Ken Dunham, kend@kendunham.org
3.1. **Identifying the Operating System**

Forensic software frequently attempts to automatically identify an operating system (OS). If such solutions fail to identify the OS properly, manual inspection of content may be required. Manual inspection may include analysis of the file extension, viewing of image files and their headers using a hex editor, analysis of found partitions, and analysis of file systems and volumes related to the incident. In some cases the operating system received by a third party to analyze is not what was expected or reported by the client, leading to further qualification between both parties as the investigation begins.

3.2. **Manual Inspection for Malcode Process (Funneling)**

By the time manual inspection of a system begins to take place the analyst commonly has several suspect files or locations on the drive to focus upon based upon hash values queries against all files on the system. Some in the private malcode analyst community are starting to call this process funneling, where an analyst performs a systematic analysis and elimination of legitimate files from those that are suspect on a system. Normally analysts group files of interest into categories, such as suspect persistent hooks (code that starts up after restart of the computer), data or keylogger files, possible exploit or downloaded files, and so on.

When profiling files several questions should be answered as part of the deductive process (Malin, 2008):

- What type of file is it?
- What is the intended purpose of the file?
- What is the functionality and capability of the file?
- What does the file suggest about the sophistication level of the attacker?
- What affect does this file have on the system?
- What is the extent of the infection or compromise on the system or network?
- What remediation steps are necessary because the file exists on the system?

Ken Dunham, kend@kendunham.org
Manual inspection of a file system for malcode involves several common paths of research (Carvey, 2004), commonly performed within a specific user account of interest on a computer before broadening research to an entire computer:

- Inspection of all suspect files (even if the filename appears legitimate)
- File attributes (hidden or system or read only)
- Icons
- Changing views to reveal hidden and system files, as malcode may abuse these default settings to conceal themselves on a system.
- Linux "File" command analysis of files for rapid content triage. For example, using a script and Cygwin on Windows to see if all images in an image directory are truly images or possibly application type instead.
- Additional file analysis to identify if it is packed or encrypted, which may help define a malicious context.
- MAC (modification, access, and change) time changes for a file and correlation to found malicious files. If a malicious file is found, correlate all other files with similar MAC times.
- On Windows using NTFS, looking for Alternate Data Stream (ADS) data.
- On Windows, looking for possible hidden data in the Windows registry.
- Log file analysis to correlate events to suspect files of interest.
- Prefetch file data indicating what files may have been run, even if later deleted by malcode, such as tmp.exe.

Changes to MAC times is one of the most foundational and critical components of a forensic malcode investigation. It is very common to identify a malicious file and then find other related files based upon MAC times or those that are out of place for the local system. Looking for out of place last written times is also helpful, such as in the Windows registry.
Experience naturally helps analysts quickly identify suspect files. Common tricks used by malcode historically that can quickly be found by an experienced analyst include but are not limited to the following:

- Files that have very long names with many spaces or extra extensions, such as "filename.jpg.exe".
- Files that have an icon that does not match their extension type, such as an executable with a Microsoft Word icon.
- Files found in the Windows System32 directory that are confirmed hostile with MAC times that match up with suspect files in the user directory. DLL and SYS files are common in the Windows System32 directory for rootkit related payloads.
- ADS that are linked to an executable that is questionable or unknown, like that of PoisonIvy Trojans.
- Windows registry entries that correlate to confirmed malicious files on the file system.
- Questionable files with zero bytes or encoded content, possibly indicating a keylogger or theft of sensitive data stored in a file.
- Executables in the Temp directory that may have been used as part of an infection routine.
- A new ADMIN user is added to the machine on a given date, as shown in logs, may help reveal payloads also created on that same date.

Once suspect files are identified they are normally extracted and analyzed further for confirmation.

### 3.1. Analysis of Suspect Files

Diving deeper into file analysis of suspect files is critical for a good understanding of the malicious context and functionality that may exist with such a file.
3.1.1. Anti-virus Scan

Once an image is mounted anti-virus software can be used to scan and identify any files that are believed to be malicious. Using more than one anti-virus software package may also help in cases where few results turn up or more robust analysis is required. This process can be streamlined, if desired, by exporting out only those files from the image that are believed to be suspect, to then focus on those files with multiscanners like VirusTotal (VirusTotal, 2011). This is not very feasible though, if large numbers of file are suspect after manual inspection of a file system. In such cases local updated anti-virus programs should be used to scan larger file sets.

3.1.2. Fuzzy hashes

By this stage of processing an analyst has already performed hash lookups for all files, identified suspect files, and performed anti-virus scans of suspect files to identify possible detection names and inferred functionality from such results. A fuzzy hash is used to then compare samples against one another to see how similar they are. The more a fuzzy hash matches, the closer the files are to one another in terms of minor variants of a code base.

Use a program like ssdeep (Jesse Kornblum, 2010) to generate fuzzy hashes and then compare them. For example, several PDF files that all exploit a GetIcon vulnerability were tested to see how similar they are to one another for overall content (which includes the exploit itself). Results are below:

<table>
<thead>
<tr>
<th>Filename</th>
<th>Fuzzy Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>pdf</td>
<td>96:kwc8nTykkngzVPTmMlbN8MrxmuW8vhYFnOBzAS6MaUoTqEXNLMjW:D3nOckngzNTDmuxmuW86F4TaUmqu1</td>
</tr>
<tr>
<td>pdf</td>
<td>96:kwc8nTykkngzVPTmMlbN8MrxmuW8vhYFnOBzAS6MaUoTqEXashU7XGJW:D3nOckngzNTDmuxmuW86F4TaUmq2hU7l</td>
</tr>
<tr>
<td>PfMDvb05Grpd</td>
<td>96:kwc8nTykkngzVPTmMlbN8MrxmuW8vhYFnOBzAS6MaUoTqEXyi3pJW:D3nOckngzNTDmuxmuW86F4TaUmqvyu</td>
</tr>
</tbody>
</table>

Ken Dunham, kend@kendunham.org
Table 3.1: Ssdeep fuzzy hash comparative analysis.

<table>
<thead>
<tr>
<th>Sample 1</th>
<th>Sample 2</th>
<th>Sample 3</th>
<th>Sample 4</th>
<th>Sample 5</th>
</tr>
</thead>
<tbody>
<tr>
<td>geticon</td>
<td>48:PqodEWXe4F/GduEkkPbNHgfInUZcHR7MSkiEMoV6go:/C odEs2uExRAflecZ4u/</td>
<td>ametWhenThose</td>
<td>1536:k4zg8x3fNuekfNpLSSxABvhD321t/ez/H32koplScKz+/wE: XJxPNuRfeSxAv5icDXiecKzA</td>
<td>aldusToThis</td>
</tr>
</tbody>
</table>

The first two samples have the same filename but are different in terms of their content, but closely related. The first three samples have the same computed fuzzy value of 96 but differ in their fuzzy hash value, ever so slightly (look near the end of each string value). These samples are all very closely related to one another. In this case these are files extracted from various exploit frameworks, showing that the very exploit file used in one kit may also be borrowed and slightly modified for another exploit kit. In the case of forensics on a file system, many minor variants of a malcode may be installed on a computer. Fuzzy hashes of select files, or the entire file system, can show relationships more clearly for which files may be closely related to one another.

3.1.3. Family Analysis

When reviewing anti-virus scan results, look for family names that may aid in understanding functionality or malicious context. For example, select detections taken from a random VirusTotal report for a sample reveals several family names:

- Avast5 5.0.332.0 2010.08.07 Win32:Tiny-ADY
- AVG 9.0.0.851 2010.08.07 BackDoor.PoisonIvy.AD
- BitDefender 7.2 2010.08.08 Trojan.Keylog.ZKT
- DrWeb 5.0.2.03300 2010.08.07 BackDoor.Poison.685
- Emsisoft 5.0.0.36 2010.08.07 Trojan.Trash!IK

Experienced analysts quickly identify generic names like "Tiny", "Keylog", and "Trash". In this case two possible names exist, BackDoor.Poison and

Ken Dunham, kend@kendunham.org
BackDoor.PoisonIvy. Queries can then be performed generically for these terms to identify more information about this family of code. Queries can also be made more specific on vendor sites, such as navigating to an AVG website to perform a search for the exact detection or family name, such as "BackDoor.PoisonIvy.AD", when such options exist for a given vendor of interest.

Some detection names are so specific and rare that they are excellent for abuse queries also. In such cases searching for the entire name inside of quotes on popular search engines often yield outstanding results for related abuse queries, sandbox analysis, and other data available on public websites related to the code of interest.

3.1.4. Abuse Lookups

Abuse lookups related to malcode of interest can take several interesting directions including but not limited to the following common vectors of interest:

- MD5 query to locate other reports directly related to a sample of interest.
- Family names, correlated to date and time, to locate other incidents that may be related to the incident under investigation.
- Research into malicious servers, via domain and IP that are related to the sample of interest. In some cases passive DNS (BFK edv-consulting GmbH, 2011) and public data may reveal a wealth of data. Additionally some providers (DomainTools, 2011) may have historical data of interest, such as related historical IPs for a hostile domain or server of interest.

Abuse research is time consuming and more difficult than file system analysis. For example, passive DNS settings may reveal a large number of domains that used the same server as a known hostile server related to an investigation. This does not, however, prove that those domains were used maliciously, just hosted on the same server. When analyzing such data the analyst must exercise caution and wisdom in connecting the dots to prove malicious context and relationships to build out a case of defendable relationships. Most importantly, well qualified abuse research may help in the investigation underway, to know what files or payloads to look for or other changes to a system that should be present.

Ken Dunham, kend@kendunham.org
3.2. Advanced Triage

There are exceptions to normal file-based malicious codes that may infect a Windows computer (or other OS). One is a file-less payload that runs only in memory. If such a payload exists, a forensic investigator may only have trace clues as to what was in memory at the time of evidence collection. Analysis of prefetch and log files may aid in identifying possible payloads that existed in memory but are not found on the file system.

Another exception to the rule is Master Boot Record (MBR) and slack space storage malicious codes. Mebroot is one of the more infamous in this space, able to modify the MBR of an infected disk to then hook startup on the disk level. Data in the MBR then loads a kernel-level rootkit that is stored encrypted in slack space at the end of the drive, before the operating system even loads. Such advanced threats are very difficult to deal with compared to traditional methods as the MBR requires advanced skills to capture and identify compared to a traditional file system based payload. Additionally, unique techniques using specialized forensic tools and tactics may be required to capture and then also decrypt payloads stored in slack space.

4. Behavioral Analysis of Suspect Code

Use automated analysis tools, whenever possible, to quickly identify core functionality of code. This may aid in giving a second opinion or results that might vary from those seen on a computer. Such analysis may also provide clues for the analyst to then locate additional files or system changes of interest related to a malcode event.

4.1.1. Sandbox

A number of free public sandboxes exist on the Internet for analysis of malicious code. Some of these run on virtualized machines that are hardened against detection by malicious code. Others run on native machines that are quickly re-imaged or reset for each test. Some have feature such as allowing for interactive scripts or events to take place while others may provide netflow PCAPs or other details of interest. Realize that any samples uploaded to such a system normally provides legal rights to that provider to

Ken Dunham, kend@kendunham.org
then share that code with others and also make the analysis public. Never use a public sandbox if working with samples that are should not be disclosed to others.

The table below highlights common free sandbox scanners and unique elements of value as identified by the author. Annotations are not designed to be comprehensive nor comparative.

<table>
<thead>
<tr>
<th>Sandbox</th>
<th>Annotation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Norman</td>
<td>One of the first free sandbox scanners on the market.</td>
</tr>
<tr>
<td><a href="http://www.norman.com/security_center/security_tools/">http://www.norman.com/security_center/security_tools/</a></td>
<td></td>
</tr>
<tr>
<td>Sunbelt</td>
<td>Native system analysis.</td>
</tr>
<tr>
<td><a href="http://www.sunbeltsecurity.com/sandbox/">http://www.sunbeltsecurity.com/sandbox/</a></td>
<td></td>
</tr>
<tr>
<td>ThreatExpert</td>
<td>One of the easiest reports to read quickly.</td>
</tr>
<tr>
<td><a href="http://www.threatexpert.com/submit.aspx">http://www.threatexpert.com/submit.aspx</a></td>
<td></td>
</tr>
<tr>
<td>Anubis</td>
<td>Includes PCAP output when egress traffic is present.</td>
</tr>
<tr>
<td><a href="http://anubis.iseclab.org/">http://anubis.iseclab.org/</a></td>
<td></td>
</tr>
</tbody>
</table>

Table 4.1: Public sandbox comparative analysis.

4.1.2. Malicious Context

When analyzing a system, common correlations for understanding related files, changes, and configurations are date and time checks in addition to persistence and events. Systematically reviewing all such relationships helps to establish a malicious context for one or more events on a computer for one or more incident(s).

4.1.2.1. Persistence Checks

In order to survive a reboot most malicious codes include some sort of persistence hook. If malicious activity is found in log files on a file system of interest, a persistence method must exist somewhere. Once this method is found confirmation of malicious payloads and their relationships are also proven. Common areas to check include but are not limited to the following:

- Changes to Windows registry to run upon login or similar events.

Ken Dunham, kend@kendunham.org
• File created in startup folder.

• Installed as a Browser Helper Object (BHO) to run in memory with a browser.

• Changes to a configuration file to run upon Windows startup.

Naturally the list of possible persistence is long and creative and can be time consuming! If a known primary payload is found or suspect, such as Trojan.exe, queries can be made in the Windows registry, the file system, and configuration files to quickly locate any possible configurations pointing to that file.

If persistence is difficult to discover on a computer a live test of the system may prove to be more efficient, using tools like Autoruns (Mark Russinovich, 2011) to look for startup hooks on the system.

4.1.2.2. Event Correlation

Detailed analysis of all log files, a common forensic practice, can also prove to be invaluable in a malicious code investigation. Information such as MAC Time correlation to web events and other file changes similar to known/found malcode may be discovered. Questionable or known hostile accounts on the system may also exist with logs related to when a bad actor accessed a system. All such events are then correlated to help identify files within the system, in a recursive manner, until the full functionality and extent of known payloads for a system have been accounted for. For example, if a downloader Trojan is discovered that is found in the lab to also install an adware BHO, the analyst may then search the image for the adware BHO that should exist on the system.

4.1.3. Abuse Correlation

Once automated methods are exhausted for understanding functionality and families of code related to an investigation, abuse queries may provide additional context under which to interpret and/or probe within a forensics image. For example, a file associated with a known drive-by site domain may be confirmed in a forensic investigation. Once this is performed by the analyst a query for all exploits and payloads related to the drive-by exploitation server may reveal additional payloads that may exist on the forensic image of interest.

Ken Dunham, kend@kendunham.org
Abuse queries can take place through three primary avenues: MD5, domain, and IP queries. While others can also take place, such as a different hash query like that of SHA1, the three primary values most commonly reveal abuse queries of interest when searching private and public data sources.

Interpretation of abuse queries can be straight forward, such as a security website identifying a specific MD5, anti-virus scans, and/or behavior (sandbox) seen on a specific date. Others are more difficult, such as lengthy forum posts with a wide variety of unqualified and/or unverifiable data points. All such data needs to be qualified accordingly, related to confidence and accuracy, as it relates to the data that can be verified, and then applied to the forensic investigation.

A general search engine query, such as Google.com, may render a wide variety of results. When looking into malcode abuse related issues, several other specific sites may be useful in perform individual searches to gather historical or new data not yet cached by search engines. Samples of such sites are in the table below as a starting point for such sources for the reader:

<table>
<thead>
<tr>
<th>Website</th>
<th>URL</th>
</tr>
</thead>
<tbody>
<tr>
<td>McAfee Site Advisor</td>
<td>siteadvisor.com/sites/domain/IP</td>
</tr>
<tr>
<td>Malware URL</td>
<td><a href="http://malwareurl.com/listing.php">http://malwareurl.com/listing.php</a>?</td>
</tr>
<tr>
<td>Malc0de.com</td>
<td><a href="http://malc0de.com/database/index.php">http://malc0de.com/database/index.php</a>?</td>
</tr>
<tr>
<td>Malware Domain List</td>
<td><a href="http://malwaredomainlist.com/mdl.php">http://malwaredomainlist.com/mdl.php</a></td>
</tr>
<tr>
<td>Spamcop</td>
<td><a href="http://spamcop.net/w3m?action=checkblock&amp;ip=domain/IP">http://spamcop.net/w3m?action=checkblock&amp;ip=domain/IP</a></td>
</tr>
<tr>
<td>Spamhaus</td>
<td></td>
</tr>
</tbody>
</table>

Ken Dunham, kend@kendunham.org
4.1.4. Lab Analysis and Reverse Engineering

More detailed analysis of a sample may also be performed by skilled analysts to validate content, interact with code, and/or investigate it further than what automated analysis tools reporting can provide. This is where a wide variety of tools and techniques may be employed, such as identified in free and commercial training (Lenny Zeltzer, 2011). This is not normally a path taken beyond simple virtualized testing of malcode as it takes a long period of time, beyond that of a forensic investigation already underway, and is not necessary in most cases to identify core functionality and associated risks with malcode tested through automated methods aforementioned.

4.2. Analysis of Suspect Scripts

Many malicious code events involve drive-by attacks and hostile scripts. Forensic investigators often correlate archives for Internet browsing to questionable files or known malcode found on a file system. In some cases analysis of scripts is required to better qualify the vector of attack, such as exploitation of a given vulnerability. This may be core to such an investigation if the client is attempting to identify how the compromise took place to then identify other infections and/or mitigate against ongoing attacks against a network.

4.2.1. Query for notable strings if related to exploitation, CLSID, etc.

When manually inspecting a script an analyst may look for notable strings of interest that may help to identify that script. The example script below, just the header, has several invaluable strings of interest:

```html
<html>
<body>
<script language="JavaScript">
    var m = new Array();
</script>
</body>
</html>
```

Ken Dunham, kend@kendunham.org
Identifying Malicious Code Infections Out of Network

Ken Dunham, kend@kendunham.org

```javascript
var mf=0;
var url="hxxp://69.50.173.195/h2sv3b1/load.php";
var fn="c:\winJm5TI.exe";
```

Clearly a link of interest, pointing to load.php, is highly suspect. Additionally a possible file name and location exists in the script above. An analyst may then locate and analyze such a file, provided through such a script analysis. The analyst may also be able to perform abuse queries and capture data related to the URL, or even perform a lab analysis drive-by test of such a URL.

Analysis of a suspect script becomes more difficult when encoding or encryption is used. Take for example additional data found in the same exploit script aforementioned:

```javascript
function ms(spl){
    var plc=unes(
        "%x43\x43\x43\x43\x43\x43\xEB\0F\5B\33\C9\66\B9\80\x01\80"+
        "%x33\xEF\43\x42\xFF\EB\05\xFF\xFF\xFF\xFF\7F\x8B\x4E\xDF"+
    );
```

This script suggests a function that is related to "ms" which may stand for Microsoft. "spl" may stand for something like "sploit" or "exploit". The text "unes" may stand for unescape, a common tactic used with JavaScript based statements. The format of the script then reveals strings inside of quotes with a concatenation of strings. Analysts must be script savvy in a variety of languages in order to properly analyze such scripts.

4.2.2. JSUNPACK

Jsunpack (Blake Hartstein, 2011) is a free JavaScript unpacker that works wonders on a wide variety of scripts. For private samples this tool can also be implemented locally, for free. Sometimes other analysts who have already used this tool to analyze a script may have analyzed a hostile script that is related to an incident under investigation by the forensic analyst. To find such related content perform queries from an engine like Google, such as "inurl:jsunpack.jeek.org ScriptString", where ScriptString

Ken Dunham, kend@kendunham.org
is something unique within the script of interest or the domain or IP suspected of performing a drive-by attack.

4.2.3. De-obfuscation

Other de-obfuscation tactics may be necessary to analyze a script further. Clearly various documented techniques exist, such as changing write actions to display in a scrolling text box instead, as well as working with a variety of data encodings. Tools such as Malzilla (Boban Spasic, 2008) and Script Monkey (Siddique, 2011) may aid the advanced analyst in further analyzing obfuscated scripts.

5. Live Boot of a System & Memory Analysis

A live boot can sometimes speed up the analysis of a forensic malcode investigation. A variety of tools can be utilized to quickly help identify possible malicious processes in memory, autorun hooks, and questionable file activity on a disk. If an Internet connection is available egress communications may also take place with a remote command and control (C&C) server, in which case an analyst may use tools to map processes to network activity to identify possible injection or malicious files on a file system.

When booting into a live system credentials may be required and some challenges may exist with various changes in hardware, drives, etc. The client may be able to provide credentials to aid in analysis of a file system. A variety of hacking techniques can also be used to get around such limitations. If such operations are performed on a regular basis use of a tool like Kon-Boot (Kryptos Logic, 2011), which is able to bypass such limitations, is helpful.

Once a system is live advanced memory analysis may also aid in further understanding malicious code operations and context. For example, a dump of RAM to a file can be performed. This file can then be analyzed using a tool like the Volatility Framework (Volatile Systems, 2011), and various plug-ins, to quickly identify malicious code in memory.

Ken Dunham, kend@kendunham.org
6. Case Studies

6.1. Possible Worm

The client submits an image of a Windows operating system host that was generating TCP port 445 suspect traffic. The goal of this investigation is to locate a possible worm attempting to spread over TCP port 445. The image of the host was created using Ghost, easily extracted and mounted in the lab using Ghost and/or Ghost Boot (ISO).

Analysis of the netflow reveals that traffic generated by the questionable host appear to be semi-random, generating possible scanning probes via the fourth octet of IPs within the same subnet as the host computer.

A review of prefetch files for the user account of interest on the host computer reveals a large number of executables run on the machine. After funneling through possibilities two primary prefetch instances are suspect:

• CVTRES.EXE
  A search of this executable online reveals that it may be a legitimate component related to Microsoft® Visual Studio® 2005. It may also be a Trojan as identified online via Virscan.org, greatis.com, and auditmypc.com reports.

• LOGGER.EXE
  This filename is highly suspect. Three reports exist online related to anti-virus website and abuse threads that indicate it is likely a keylogger component.

A malicious HTML file was found in the system's cache which attempts to load a flash exploit file hosted on cdn4.specificclick.net (or a related domain). Internet cache revealed a suspect flash file, "clipF1[1].swf", that contains a link to hxxp://bp.specificclick.net/. A review of public and private archives for malcode reveals Vundo and Rogue Anti-virus payloads related to the specificclick.net domain and flash file in question.

Ken Dunham, kend@kendunham.org
The suspect flash file was scanned with anti-virus software and was detected as a click-fraud based Trojan, Trojan-Clicker.HTML.IFrame.aiw. This confirms the computer is compromised but does not necessarily prove the TCP 445 activity, the original focus of the investigation. A dump of the flash file reveals the link aforementioned.

A DAT file was found on the host that is similar to that of the filename of interest for the DLL linked to the SWF file. It is stored in the user "Application Data" directory as gdipfontcachev1.dat. A search for this unique filename reveals two malicious codes in the past that have used that exact name, according to Threat Expert\(^1\), which are both malicious.

Several other questionable files are located in Windows System32 through funneling but none show any signs of maliciousness when scanned with a multiscanner anti-virus solution.

In this investigation the confirmation of the malicious flash file and compromise was enough for the response team of the client to image the computer and cease the investigation. Additional research into the questionable TCP 445 activity was not required. If, however, it was required, booting the system in a lab environment with Internet connectivity (filtered and protected against worm spreading) is the next logical step to perform live boot and testing of the infected host.

### 6.2. Where is the Malcode?

In this response the client submitted two samples that turned out to be benign. This led to analysis of an image and anti-virus log files, between the client and the forensic consultant, to identify TDSS rootkit and DNSChanger payloads on the system installed via a Phoenix exploit kit.

One file submitted for analysis has the filename "I-hate-keyloggers". An extensive review of this file proved that it is a legitimate application that has not been modified for any malicious means. A second benign file was also submitted by the client, FTP Now, a legitimate FTP program.

---


Ken Dunham, kend@kendunham.org
Once the first two files were found to be benign a search began to funnel into additional suspect files that could help to explain suspect behavior on the system seen over netflow (IDS/IPS). AVG anti-virus was installed on the computer, and then upgraded from version 9 to version 10. To easily read these log files to identify dates, times, and events of interest, AVG 9 and AVG 10 were obtained and installed on a computer. Log files extracted from the suspect host were then placed into the appropriate log file locations for AVG 9 and AVG 10 to then read through the AVG log interface.

Analysis of the anti-virus log files revealed an infection that took place on or around the date of Nov. 26, 2010. An AVG 9 log file revealed a temporary Internet HTM file that was suspect and moved to the virus vault. This was related to a tracking cookie called "Doubleclick" A possible malicious sample was also found in the log files, in the TEMP directory (highly suspect), 555y5.sys. The SYS file is highly suspect as a possible rootkit. It was detected by anti-virus software as "Trojan horse Crypt.ZOJ".

Several older events also appeared related to possible compromised restore data points and executables that are suspect. AVG 10 log files had little to contribute to this investigation as the upgrade was done after the original suspect date and time of compromise (AVG 9 period). A full review of all such log data reveals that the end user likely suspected an infection and ran a variety of anti-virus programs, updates, and scans in an attempt to identify and remove any malcode on the computer.

Several days following the Nov. 26, 2010 event an additional event of interest took place on Dec. 2, 2010 related to a generic Trojan horse. This information strongly suggests that the computer was compromised around the Thanksgiving holiday period in 2010 and was then further compromised despite efforts by anti-virus to detect and remove some of the malcode.

The forensic consultant was then able to acquire a copy of 555y5.sys from the makers of AVG software, detected as "Trojan horse Crypt.ZOJ". Analysis of this file proved that it was a rootkit driver related to TDSS that took place in November 2010, which correlates to the timeline of AVG and system logs on the compromised computer. This then led to the discovery of a related file, 31k9y1c9.dll, which then correlated back

Ken Dunham, kend@kendunham.org
to a command and control of interest, bestrico.com (62.122.75.42) which posts data via kx.php.

Bestrico.com then became the focus of the investigation as it was related to a large amount of abuse. Public and private data sets related to malicious code, malicious domains, and similar data revealed that this command and control server was responsible for a large number of TDSS and DNSChanger payloads during the date and time period of the compromised host. This then led to the discovery of several additional filenames, command and control domains and IPs, and a relationship with a Phoenix exploit kit at 188.65.74.26. Related data of interest was used to coordinate with the client to then check log files, IDS/IPS, and similar checkpoints to see if any related payloads existed on the suspect host and network. This helped the client to confirm the infection on the host of interest, which was in a remote lab, as well as aggressively audit their network for any other possible variants or compromises related to this original incident.

6.3. Imaging Handling and Full Data?

This investigation involved confirmation of a hostile server used in malcode attacks, hosted on a Linux system. Handling of the images prior to delivery to the forensic consultant proved to be interesting, along with what data was provided and was not. After data was successfully mounted and analyzed, a database on the system included many links of interest. These links were then correlated to public and private malcode tracking systems to prove that they were linked to malicious behavior.

Acquisition of SATA disks led to the forensic consultant using a write blocker hardware component along with an eSATA device to mount the physical drive on a Windows computer. Inspection of the original drive revealed two hidden directories that are created on drives when mounted by Windows: RECYLER and System Volume Information. Since a write blocker hardware component was used in mounting the drive this meant that the drive had to have been mounted on a computer using Windows prior to delivery to the forensic consultant. This does not mean that the actual file system on the drive is Windows, even though one might lean in that direction after such a finding.

Ken Dunham, kend@kendunham.org
Manual inspection of the drive contents revealed inside of an images directory a large number of files that appear to be an image split archive using a sequential numbering system for extensions (.001, .002, etc.). At the end of this directory is a file that contains a log of the imaging that was created on the drive, created by a Tableau forensic bay controller (hardware for making copies of drives). The log file reveals that it was likely performed off of a Macintosh computer. The files created by Windows on the drive of interest remain unanswered but are noted for the client as it may be relevant to their research if someone else mounted the drive on a Windows machine without authorization or proper write-blocking hardware.

Using FTK Imager to mount the image it is clear that it is not Windows or Macintosh but Linux based. An inspection of the partitions on the drive revealed the following data:

![FTK Imager interface](image)

**Figure 6.1:** FTK reveals a BSD operating system.

Ken Dunham, kend@kendunham.org
The text "BSD" are clearly visible along with five partitions. On Linux systems "/proc/version" and other locations and log files can also be used to help manually validate the version of the operating system on the file system. A little research into the "BSD" string and format reveals that the operating system in question is a format an older version of FreeBSD. At this point the forensic consultant confirms this as the file system on the drive and discusses the various aspects of imaging handling, mounts, and data encountered in the investigation to date. The investigation then focuses on malcode that may be on the system.

An inspection of the system reveals several user files of interest, along with system log files that are very revealing. Specifically, several remote admin sessions took place at specific dates and times of interest to the research in this case. This then correlates to file system data found on the system. The final area of interest found on the system is within a MySQL database that contains many links. These links were then correlated to abuse related to malcode to prove a relationship to maliciousness.

After analysis of the entire file system it is clear that several user directories that should exist on the drive are not present. This proves that the disk provided is only part of a computer system of interest. The forensic consultant then discusses this with the client to then identify and retrieve a second disk to analyze. This second disk has the same interesting mounting issues but does provide the missing data of interest to round out the investigation.

7. Conclusion

Analysts working on non-network images face a variety of challenges. Unfortunately efforts to communicate with the client may not provide information in an accurate or helpful manner. As a result analysts must develop best practices to quickly validate all received information against data analyzed in the lab. By systematically mounting, inspecting, and processing images from clients analysts are able to pursue malicious code investigations (funneling). Such investigations involve a combination of forensics, static and behavioral malicious code analysis, and investigative efforts to correlate events, abuse, and command and control servers of interest. Live testing of a

Ken Dunham, kend@kendunham.org
system may also greatly expedite a malcode investigation but was not the focus of this research.

8. References

Special thanks to individuals that contributed to a peer review of this research report: Shane Hartman, Drew Robinson.

Figure 2.1: eSATA solution for quickly mounting a SATA drive to a laptop.

Figure 2.2: File data in a numerical sequence with no known image extension.

Figure 2.3: FTK options for mounting evidence type (select logical).

Figure 2.4: FTK viewing a mounted image.

Table 3.1: Ssdeep fuzzy hash comparative analysis.

Table 4.1: Public sandbox comparative analysis.

Table 4.2: Search engine abuse queries.

Figure 6.1: FTK reveals a BSD operating system.


Ken Dunham, kend@kendunham.org
Identifying Malicious Code Infections Out of Network


Ken Dunham, kend@kendunham.org


