Evaluations of Host-Based Intrusion Prevention Systems (HIPS): Sana's Primary Response and Cisco's Cisco Security Agent[1]

Dr. Edwin R. Coover
Duncan Thomson
Center for Advanced Aviation System Development
The MITRE Corporation

Introduction

As part of the Federal Aviation Administration's (FAA's) "Intrusion Quarantine" project,[2] The MITRE Corporation's Center for Advanced Aviation System Development (CAASD) conducted an evaluation of two products, Sana's Primary Response and Cisco's Cisco Security Agent (CSA).[3] These two products were selected as examples of Host-based Intrusion Prevention System (HIPS)[4] technology that showed promise of meeting the Intrusion Quarantine project goals. It is important to note that the purpose of the evaluation was not to test these specific products against a well defined set of customer requirements, nor to make purchasing recommendations regarding these specific products. Rather, the intent was to use MITRE's evaluation of these products to understand the current state of technology at the time (spring of 2004). It is important to note that products in this category should be expected to evolve rapidly; organizations considering investing in these products may wish to contact the vendors or conduct their own testing to determine whether issues identified in this paper have been addressed or significant new functionality has been added.

CAASD's evaluation methodology included Internet-based testing, laboratory-based testing, and analysis. Internet-based testing consisted of installing the product on a computer directly connected to the Internet, making observations on the installation and configuration process and employing the product to protect the Internet-exposed computer. The Internet-based testing also included analyzing the alerts produced to observe the frequency of Type I ("false positives") and Type II ("false negatives") errors. Laboratory testing consisted of installing the product on

---

[1] NOTICE: This work was produced for the U.S. Government under Contract DTFA01-01-C-00001 and is subject to Federal Aviation Administration Acquisition Management System Clause 3.5-13, Rights In Data-General, Alt. III and Alt. IV (Oct. 1996). The contents of this document reflect the views of the authors and The MITRE Corporation and do not necessarily reflect the views of the FAA or the DOT. Neither the Federal Aviation Administration nor the Department of Transportation makes any warranty or guarantee, expressed or implied, concerning the content or accuracy of these views.

[2] Mehan, Daniel, "The Android Model of Cyber Defense," Keynote Address, Information Technology-Information System Security Research and Development (IT-ISS R&D), March 2-4, 2004, Workshop Proceedings.

[2] "Primary Response" is an anomaly-based HIPS product from Sana Security, 2121 South El Camino Real, Suite 700, San Mateo, California 94403. See www.sanasecurity.com. "Cisco Security Agent" (CSA) is a rules-based, artificial intelligence HIPS product from Cisco Systems, 170 West Tasman Drive, San Jose, CA 95134-1706. See www.cisco.com. The software versions tested were Primary Response 2.1 and CSA 4.0. The Primary Response testing took place between March 29 and April 12, 2004. The CSA testing occurred between April 6 and April 24, 2004. These software versions are no longer current; observations on the relative merits of the products described are germane only for the versions tested.

[4] In the literature, the terms "Host Intrusion Prevention System (HIPS)" and "Host Intrusion Detection System (HIDS)" are often used interchangeably.

computers in a laboratory environment and running a series of testing scripts, including running exploits against known vulnerabilities on the target machines and observing the ability of the products to detect or prevent the exploits.

The analysis performed included addressing a set of evaluation criteria. CAASD based its evaluation on: (1) experience gained in the experimentation, (2) information from the product vendors' documentation, (3) other information received directly from the vendors, as well as (4) other information gleaned on the design and implementation of the products, such as product reviews published by other parties.

Internet-Based Testing

The configuration used for the Internet-based experiment is shown in Figure 1. Both local area networks (LANs) (shown with solid black lines) were implemented with 10 Megabits per second (Mbps) Ethernet hubs.

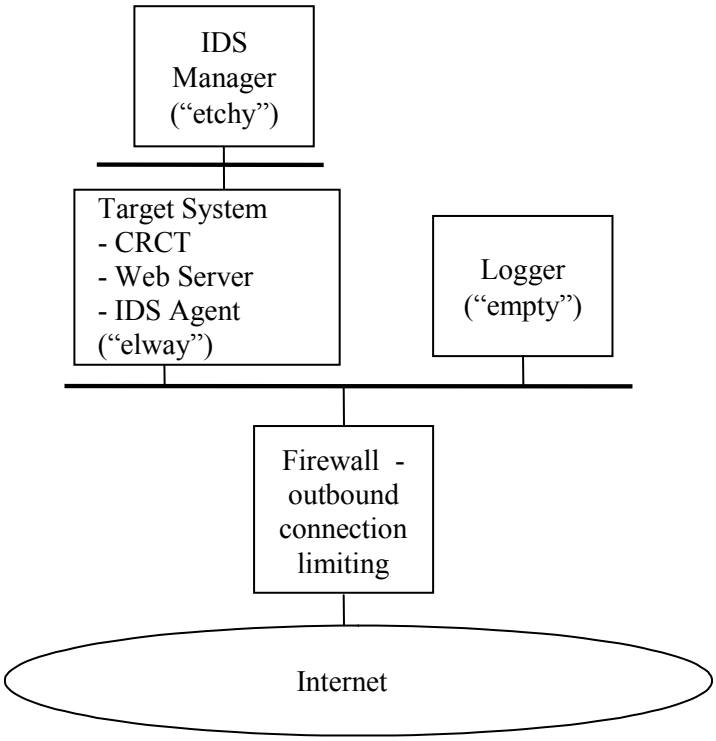

Figure 1. Internet-Based Configuration

The setup consisted of a target system (named "elway") running the agent portion of the HIPS product, configured to protect a target application. The target application included an application called the Collaborative Routing Coordination Tools (CRCT). CRCT is a CAASD-developed advanced Traffic Flow Management (TFM) tool that is currently being used as a prototype in two Air Route Traffic Control Centers (ARTCCs). The CRCT application was driven from recorded National Airspace System (NAS) data that had been previously cleared for public release. Also running on the target system was a Web server configured to serve Web pages showing air traffic and NAS status displays created by the CRCT application. The HIPS agent was configured to protect the CRCT application, the Web server, and other operating-system related executables

running on the Target System. The IPS manager was installed on a machine ("etchy") that was connected to the target system via a separate LAN.

All traffic to and from the target system was recorded via the *tcpdump* utility on a separate logger machine ("empty") that passively and invisibly sniffed the LAN between the target system and the Internet. The *snort* network intrusion detection tool was used on the logging system to detect attacks against the target. The Tripwire™ tool was also used on the target to detect any compromise of the target system.

The Internet connection exposed the target system to live attacks. There were two caveats concerning the direct connection to the Internet. First, in the event the Target System was taken over by a hacker, it was necessary to prevent the system from being used as a platform for further attacks; therefore a firewall was used to limit outbound traffic. Outbound traffic was allowed only to machines that had previously connected in to the CAASD system. In this way, a hacker could get back to the machines they used to launch the attack, but could not attack any other machines. Second, the CRCT application used the unix remote shell (*rsh*) facility for internal process control. Since *rsh* has known security vulnerabilities, CAASD felt it would not be reasonable to allow remote access to the *rsh* service. Therefore, CAASD also blocked inbound and outbound connections for he *r services* (Transmission Control Protocol [TCP] ports 512, 513, and 514).

The HIPS manager was installed (on "etchy") and the agent was installed (on both "etchy" and "elway") following the installation instructions in the vendor-provided documentation. The HIPS product was configured to monitor the CRCT application and Apache Web server (on "elway"), following the vendor-provided manuals.

HIPS Installation

The Sana install process was extremely easy. The manager and agent software were both installed in less than half an hour. For the Sana product, this installation also included exercising the application to create profiles. This included the following actions:

- Run CRCT repeatedly through five days of recorded NAS data
- Capture CRCT traffic and congestion monitor displays and make them available via Apache Web server
- Exercise the Web server by retrieving displays with a Web browser.

There are some aspects of the installation process that could become an issue in a large scale deployment. When the manager was installed it created a certificate that must be installed on each agent machine; the agent employed this certificate to authenticate the manager during operations. If the manager was re-installed, or if its Internet Protocol (IP) address changed, a new certificate must be transferred to the agent machines and the agents must be reinstalled to use the new certificate (which requires a reboot under Solaris). A large scale deployment of the Sana product would have to include procedures for certificate deployment and management, as well as management of the agent software on hosts that are being monitored. (Sana reported plans to address this issue in future releases of the product. Installation of the Cisco manager and agent software on the windows management machine was somewhat complex, requiring installation of

multiple applications and three reboots of the operating system.[5] Installation of the Cisco agent software on the Solaris target machine was relatively simple, although it did require a reboot.

HIPS Configuration

While the initial installation of the Sana product was extremely easy, configuration was more difficult. Primary Response only monitors or protects applications that have been explicitly identified, and the system administrator must explicitly identify the set of executables that make up each application to be monitored. Executables that are spawned as sub-processes of the explicitly identified executables can be automatically detected, but others are not. For some applications this is trivial, for example the Apache Web server consists of a single executable (*httpd*). However, for other applications this is non-trivial. For the CRCT application, CAASD needed to identify each of the executables that needed to be manually identified to the Primary Response manager to define the CRCT application. There were nine such executables. If CAASD had not had access to the application programmers, identifying these executables could have been non-trivial.

Primary Response needs to observe the "normal" behavior of the application over a period of time in order to build a profile which allows it to generate alerts on, or block, unexpected behavior. For simple applications, e.g., the Apache Web server with a very simple Web page, this learning process required no user intervention. However, for the CRCT application user intervention was required. After the initial learning process, multiple "false positives" were observed as a result of different features of the application being exercised that had not been exercised during the learning period. The user of the Primary Response management console must be able to judge whether an alert is a false positive as a result of the application being used in a new way. If so, the user can create an "exception" which will prevent this behavior from generating an alert in the future. This may work well for applications that have a small set of functionality and limited user control, such as Web servers, Web proxies, or mail servers. However, even for these applications, the security administrator operating the Primary Response management console must have knowledge of the application being monitored and how that application is being operated. For example, the security administrator would have to know when the Web content being provided by a monitored server has been updated with new directories, in order to know when to put the manager back into learning mode ("readapt" mode) or to mark alerts as exceptions. For applications that have broad functionality and extensive user interaction that exercises different application functionality at different times, the "unexpected behavior" monitoring or blocking capabilities of the Sana product are not expected to be useful, although its buffer overflow protection capabilities would still apply.

---

[5] CAASD was unable to install CSA in 32-bit mode; ultimately, CSA had to be installed in 64-bit mode. Therefore, the performance tests for Sana PR and Cisco CSA were run in different environments, which CAASD recognizes was less than ideal. Nevertheless, the performance testing for Sana was done consistently in the 32-bit environment (with and without Primary Response installed), and similarly the CSA testing was done consistently in the 64-bit environment. Therefore, CAASD's tests provide valid data on the performance impact of each product within the environment in which they were tested, and it is reasonable to expect that the performance impact will not vary considerably based on execution environment.

The CRCT application includes a great deal of user-controllable functionality. After some experimentation, CAASD concluded that taking the application through all of the possible modes in order to fully create a Sana "profile" would be beyond CAASD's capacity. In an operational environment, this may be less of a problem, as the product could simply be left in "learning mode" over the course of several days while the operators performed their normal functions. CAASD's solution was to simply to narrowly limit the use of the application during its experiment to a small set of operator actions needed to create the real-time traffic and NAS status monitoring displays, and to restart the application as needed when the five days of scenario data were completed.

The challenge with configuring the Cisco CSA product was to understand how the product works enough to create an appropriate rule set for protecting the application server. CAASD created a new group named "CRCT with Apache Server" by cloning the "Apache servers – dedicated" group. This worked well. When CAASD started CRCT a burst of application alerts (111 "warning" alerts) were generated due to a potentially dangerous coding practice (use of the *strcpy* function rather than the safer *strncpy* function). Adjusting the rule set to eliminate these alerts was a fairly simple task, accomplished using the "wizard" to create a special exception rule that allowed this program to make this call without generating a warning.

Installing and configuring the Cisco CSA manager and agent for the CRCT application took two experienced network and security engineers approximately two full work days, plus about one additional day spent reading the product manuals and understanding how the product works. Clearly, this is not a task that an inexperienced user should be given. On the other hand, given a short training course, a typical system administrator should be able to become proficient in installation, configuration, and operation of the product.

Internet-Based Testing

Because the Sana product only monitors specific executables (in this case the Apache Web server and the CRCT application executables), no true positives were expected in the experiment unless the Apache Web server or the CRCT application was successfully hacked. Because the Web server was the latest version, the underlying OS was fully patched, and the CRCT application was largely unknown to hackers, CAASD did not expect the box to be successfully hacked. CAASD's *snort* and *tcpdump* logs, together with the Tripwire™ tool, confirmed that, although "elway" was probed and attacked, it was not successfully hacked during the experiment.

Limiting the number of false positive events during this experiment required some management intervention. After the first three days of operation, the Sana management console showed 184 events, all caused by one of the CRCT executables accessing different weather data files. Creating an "exception" to suppress these events was quite easy using the Primary Response management console. Multiple events were generated indicating "unexpected file access" or "unexpected behavior" of various CRCT applications and the Apache Web server. CAASD classified the CRCT events as "false positives" and, when possible, marked them as "exceptions" to suppress further events. For some events, for unknown reasons, the Primary Response management console did not provide the capability of doing this. Also, in some cases, the events continued even after creating an exception.

Analysis of the *tcpdump* and *snort* logs confirmed that the Apache Web server events resulted from the server writing messages to its error log as a result of an unsuccessful attack from the

Internet. On one hand, this was an actual attack that was detected by the Sana product. On the other, it was unclear that writing to an error log should be considered "unexpected behavior." Further, it would certainly be undesirable for this action to be blocked, which is what would have happened if Primary Response had been configured in "block unexpected" mode.

Over the course of the experiment, Cisco CSA generated a total of 917 events on the host elway. Most of these events were "Notice" events, of low significance, including notices that CAASD's evaluation license was going to expire in a certain number of days. Filtering out these events is easily accomplished. There were 128 events of severity "Warning" or above. The bulk of these were the warnings generated by coding practices within the CRCT application that the Cisco agent considered questionable, discussed previously. The remainder of the "Warning" events concerned malformed TCP headers received (probably as a result of a malicious scan). There were five events at the "Alert" level. These were due to actions CAASD took at the console of "elway" as "root" which the agent considered questionable, such as modifying files within system directories.

Snort produced a total of 2361 events and detected 745 port scans. This represents a significantly higher rate of attack than occurred during the Sana test. The reason for this is unknown, but CAASD speculated that, over time, the attack rate increased as the target system's IP address gradually became known to more and more automated scanners, thereby generating a higher attack rate during the Cisco CSA experiment, which was conducted last.

Laboratory-Based Testing

The laboratory-based experimentation included running exploits against applications protected by Sana and Cisco products, measuring the impact of Sana and Cisco agents on Central Processing Unit (CPU) performance, and measuring network traffic transmitted between the Sana and Cisco agents and the respective management stations.

The configuration used for the laboratory experiments is shown in Figure 2. The setup consists of a target machine ("essay") running an unpatched version of the Solaris 8 operating system. The HIPS agent was installed on the target machine, and a manager was installed on another Windows machine ("early"). A third machine ("enemy"), running a Linux operating system, was used to launch attacks against vulnerable applications running on the target machine.

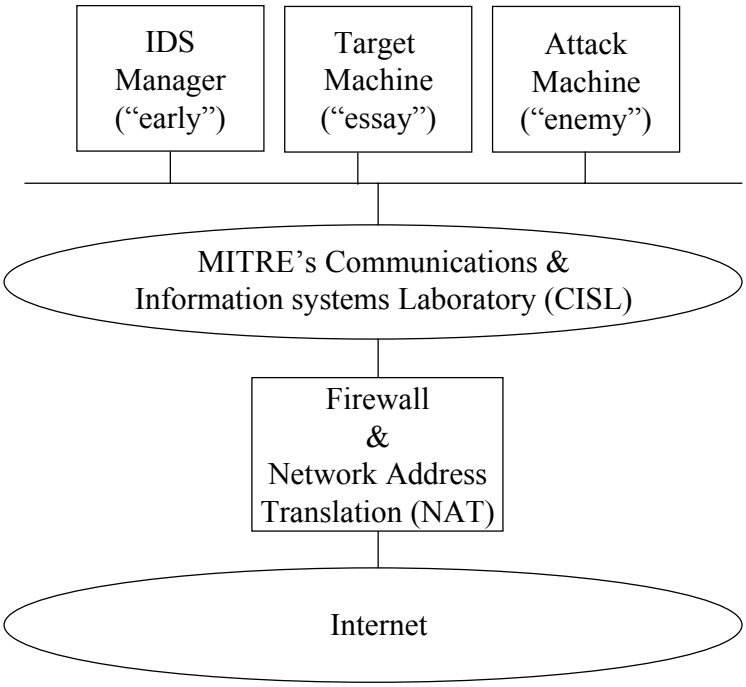

Figure 2. Laboratory-Based Experiment Configuration

For the Cisco tests, the Cisco CSA agent was configured on "essay."

Local Buffer Overflow/Code Injection Tests

A series of well-known exploits were run against both products. The exploit results appears in Table 1 below:

Exploit	Sana Results	Cisco CSA Results
Local buffer overflow code injection (*mydateXploit*)	Not detected, exploit succeeded	Exploit blocked, event generated
Remote buffer overflow code injection (*snmpXploit*)	Not detected, exploit succeeded	Exploit blocked, event generated, vulnerable daemon killed
Remote "Fail Open" exploit (telnet/rlogin *TTYPROMPT* exploit)	Not detected, exploit succeeded	No alert generated, unauthorized login succeeded, however subsequent actions blocked.

Table 1. Summary of Exploit Results

As can be seen in Table 1, the Sana product was unsuccessful in detecting or blocking the exploits CAASD ran. This was surprising, and CAASD should note that the product vendor claims that their own testing and other independent evaluations have obtained much better results. At the time CAASD ran the experiments, CAASD did provide the vendor with an opportunity to check the installation to see if they could identify any problem that would have led to these

results. The only issue identified was the fact that CAASD was running an old version of the Solaris operating system against which the vendor had not fully tested their product. (CAASD had purposely selected this older version so that it would have known vulnerabilities to run exploits against.) CAASD does not dispute the vendor's claim that these and similar exploits can be detected and blocked by their product in other environments. These results do illustrate the point that HIPS ability to defend against specific exploits is a function of HIPS product version *x* and operating system version *y* at time *t*. To suggest that a HIPS will allow an organization to defer operating system patches ignores the interdependence of the two. An organization considering use of a HIPS product must be ensure that the product has been thoroughly tested against the execution environment they are using, especially if they are running in an older environment in order to support legacy applications.

Application Impacts

The purpose of this experiment is to determine the impact of the HIPS product on applications being monitored. The test consisted of performing an "SNMP MIB Walk" operation with the HIPS agent not installed and measuring the processing time dedicated to this operation as well as elapsed time. Then, after installing the agent and configuring the HIPS manager to monitor/protect the application, running the timing test again.

The Sana performance test results showed that an average elapsed time (wall clock time) to complete the operation with no HIPS agent installed was approximately 8.8 seconds. After installing the Sana Primary Response agent, the elapsed time increased to 9.97 seconds in "learning" mode, 10.22 seconds in "detect" mode, and 10.18 seconds in "block" mode.[6] This represents an increase of 13 percent, 16 percent, and 15 percent, respectively, over the baseline time without the Sana product installed.

With Cisco the average elapsed time (wall clock time) to complete the operation with no HIPS agent installed was approximately 8.5 seconds. After installing the Cisco CSA agent, the elapsed time increased to approximately 8.7 seconds in both "test" and non-test modes. This represents an increase of approximately two percent over the baseline time without the Cisco CSA product installed.

Network Impacts

The steps performed in this experiment consisted of (1) monitoring the traffic between HIPS manager and agent using Sniffer when no alerts are being generated; (2) measuring the bytes transferred per unit time when no alerts are being generated; and (3) measuring the bytes transferred when alerts are being generated.

With Sana, the following results were obtained:

- Six applications, comprising 17 executables, being monitored
- With no alerts being generated (once per minute "heartbeat" traffic only): 1,561 bits/sec

---

[6] The "blocking" mode used with Sana Primary Response was to block unexpected file access, and block all buffer overflows.

- When alerts are being generated:
  Trial 1: 2 alerts, 23,112 bytes captured → 11.5 Kbytes/alert
  Trial 2: 5 alerts, 101,368 bytes captured → 20.3 Kbytes/alert

With Cisco, the default polling rate for communications between the Cisco CSA agent and manager is once per 600 seconds. To generate more data for measurement, the polling rate for this test was increased to once per 30 seconds. At this rate, with no alerts being generated, and no changes to the policies, the average data rate for Cisco CSA traffic between the manager and agent was 250 bits per second, measured over approximately 37 minutes. At the default polling interval, this would be expected to reduce to only 12.5 bits per second.

The experiment was then repeated, this time measuring the traffic flow after a policy change. The "Restrictive iPlanet Module" was applied to the host "essay." During a brief measurement period (16 seconds) immediately after this change, 30,206 bytes were transferred.

Several experiments were conducted to measure the traffic sent when alerts were being generated. An average of 4,560 bytes were transferred per alert.

Evaluation Summary

Both products appeared to be compatible with the Windows and Solaris operating systems and a variety of different applications. CAASD did not encounter any cases where installation of the product "broke" any application or caused unacceptable performance problems.

CAASD found that Primary Response was easier to install, but harder to configure. CSA was harder to install but—employing the defaults—easier to configure. But there is no "free lunch" with HIPS; one has to be very application-aware.

CAASD did not perform any large-scale experiments to determine whether the products would work correctly if deployed to thousands, tens of thousands, or more hosts. From what was learned of how the products are architected, there do not appear to be any fundamental limitations that would prevent either product from being deployed on a large scale, given sufficient server performance and network resources. (The Sana Primary Response product appears to generate more network traffic per alert than the Cisco CSA product, and the ability to handle the network connectivity to large numbers of agents could be a limiting factor in both cases.) However, more importantly, there are human and operational factors which would limit the scale of deployment. With both products, notwithstanding the group management capabilities discussed below, management console operators will need to be aware of how each individual managed host is being used in order to interpret results and to allow normal system administration actions to be performed on those hosts. This means that the management console operators must either be the system administrators of the hosts being managed, or must closely coordinate their activities with the system administrators. This will naturally limit the number of hosts that can be effectively covered by one set of IPS console management staff.

Both products contain features that allow hosts to be organized and managed in groups, which is essential for large scale deployment, but the Sana product has some serious limitations in capabilities for managing groups of machines.[7]

[7] The vendor claims to have made improvements in this area since the CAASD evaluation was conducted.

Both products require Internet Protocol (IP) network connectivity between each managed host and the management console. Providing this connectivity creates another possible avenue for the spread of malware or attacks. The communication between agent and management console is encrypted, which prevents interception or modification, but which also means that it cannot be inspected by any firewall to ensure that it has not been compromised and used as an avenue for malicious connectivity.

Both the Cisco CSA management software install and the Sana Primary Response management software install opened an Hyper Text Transfer Protocol (HTTP) server port on the management console host for communication with agents. The Cisco CSA management software install also opened ten other server ports on the management console host. The function of these additional ports is obscure, but they are presumably used for internal communication among different components of the Cisco CSA management console. Each of these open server ports could, in theory, create vulnerabilities on the management console.

Both products provide a "shim" layer that intercepts application calls to operating system services and checks for code injection buffer overflow exploits. It should be noted that forms of buffer overflow protection are being introduced into modern operating systems such as Windows 2003 server. Solaris 2.6 and higher also allows blocking code execution from the stack, which defeats most (but not all) buffer overflow exploits (but also interferes with applications written in Java). It should also be noted that researchers have proposed techniques that could be used in malware to overcome various buffer overflow protection mechanisms.[8] CAASD did not have access to detailed design information on how the buffer overflow protection mechanisms in Cisco CSA and Sana PR work, but it is safe to assume that they are not undefeatable.

Both Cisco CSA and Sana Primary Response provide the capability to generate reports in a variety of formats. The Sana Primary Response product provides the capability to capture log data from the agent host system when an alert is generated, which could be valuable in incident analysis.

For purposes of cost comparison, CAASD estimated the costs for a single site with ten monitored IP appliances and two management servers, one on-site and one remoted to the Computer Security Incident Response Center (CSIRC) in Leesburg, VA.[9] Cost estimates using Sana Primary Response and Cisco CSA in this configuration are $37,500 and $33,633, respectively.

Conclusions

From the tests, product strengths and weaknesses were determined along with their potential application for an FAA-wide security "quarantine" countermeasure. The results from testing Sana's Primary Response showed that, while it incorporates some promising technology, as a product it was immature at the time CAASD ran the experiments. Sana Primary Response generated many false alerts and did not catch a number of known exploits.

---

[8] For example, see "Defeating the Stack Based Buffer Overflow Prevention Mechanism of Microsoft Windows 2003 Server," David Litchfield (david@ngssoftware.com), 8th September 2003, URL: http://www.nextgenss.com/papers/defeating-w2k3-stack-protection.pdf.

[9] The second management server at the CSIRC would only be appropriate to a trial environment; in a production mode the existing CSIRC manager would have to be adapted to accept additional SNMP inputs.

CAASD's testing of CSA showed it is a mature product that can automatically block hostile activity. The FAA should proceed with caution before utilizing this product, especially on NAS systems, due to the risk of inadvertently disabling FAA applications. CAASD recommends a pilot program to gain further experience with the CSA product in an operational environment employing FAA administrative systems.