AFRL-IF-RS-TR-2003-45
Final Technical Report
March 2003

CYBER PANEL EXPERIMENTATION PROGRAM

BBNT Solutions, LLC

Sponsored by
Defense Advanced Research Projects Agency
DARPA Order No. N176

APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED.

The views and conclusions contained in this document are those of the authors and should not be interpreted as necessarily representing the official policies, either expressed or implied, of the Defense Advanced Research Projects Agency or the U.S. Government.

AIR FORCE RESEARCH LABORATORY
INFORMATION DIRECTORATE
ROME RESEARCH SITE
ROME, NEW YORK

This report has been reviewed by the Air Force Research Laboratory, Information Directorate, Public Affairs Office (IFOIPA) and is releasable to the National Technical Information Service (NTIS). At NTIS it will be releasable to the general public, including foreign nations.

AFRL-IF-RS-TR-2003-45 has been reviewed and is approved for publication.

APPROVED: Mary L. Denz

MARY L. DENZ
Project Engineer

FOR THE DIRECTOR:

WARREN H. DEBANY, Technical Advisor
Information Grid Division
Information Directorate

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 074-0188

1. AGENCY USE ONLY (Leave blank)	**2. REPORT DATE** MARCH 2003	**3. REPORT TYPE AND DATES COVERED** Final May 02 – Sep 02

4. TITLE AND SUBTITLE
CYBER PANEL EXPERIMENTATION PROGRAM

5. FUNDING NUMBERS
C - F30602-02-C-0106
PE - 62301E
PR - N176
TA - 01
WU - 04

6. AUTHOR(S)
Joshua Haines, Dorene K. Ryder, Laura Tinnel and Stephen Taylor

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
BBNT Solutions, LLC
10 Moulton Street
Cambridge Massachusetts 02138-1119

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES)

Defense Advanced Research Projects Agency
3701 North Fairfax Drive
Arlington Virginia 22203-1714

AFRL/IFGB
525 Brooks Road
Rome New York 13441-4505

10. SPONSORING / MONITORING AGENCY REPORT NUMBER
AFRL-IF-RS-TR-2003-45

11. SUPPLEMENTARY NOTES

AFRL Project Engineer: Mary L. Denz/IFGB/(315) 330-2030/ Mary.Denz@rl.af.mil

12a. DISTRIBUTION / AVAILABILITY STATEMENT
APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED.

12b. DISTRIBUTION CODE

13. ABSTRACT ***(Maximum 200 Words)***

This CyberPanel Experimentation Program final report describes two experiments: (1) an experimental validation of correlation systems, as well as (2) a validation of autonomic response systems. The goal of this effort was to assess the overall progress in the field by separately measuring the aggregate correlation efficacy and response capabilities of seven distinct research technologies. The network supported planning activities critical to a hypothetical military mission and supported modeled user activity such as email and web browsing. Each enclave was defended by security devices implementing a defined security policy and by various intrusion detection sensors. A variety of multi-step cyber attacks were perpetrated against the target network, each of which typifies a current-day real-world attack. The preliminary results presented here represent those available at conclusion of the experiment process by BBN.

14. SUBJECT TERMS
Correlation Experiment Attack Description, Response Experiment Attack Description, Correlation Metrics, Response Experiment Metrics, Correlation Experiment Results, Response Experiment Results

15. NUMBER OF PAGES
32

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	**18. SECURITY CLASSIFICATION OF THIS PAGE**	**19. SECURITY CLASSIFICATION OF ABSTRACT**	**20. LIMITATION OF ABSTRACT**
UNCLASSIFIED	UNCLASSIFIED	UNCLASSIFIED	UL

Table of Contents

List of Figures

List of Tables

1. Introduction

The last several years have seen substantive investments in sensor alert correlation and autonomic response technologies through the DARPA CyberPanel program. These technologies are intended to improve network defense by increasing the security analyst's awareness of ongoing attacks and their targets as well as thwarting cyber-attacks automatically in real time.

This CyberPanel Experimentation Program final report describes two experiments: a first experimental validation of correlation systems, as well as a validation of autonomic response systems. The goal of this effort was to assess the overall progress in the field by separately measuring the aggregate correlation efficacy and response capabilities of seven distinct research technologies. The experiment was conducted in the context of a target network architecture composed of approximately 60 hosts segregated into four sites with one or two local area networks each. The network supported planning activities critical to a hypothetical military mission and supported modeled user activity such as email and web browsing. Each enclave was defended by security devices implementing a defined security policy and by various intrusion detection sensors. A variety of multi-step cyber attacks were perpetrated against the target network, each of which typifies a current-day real-world attack.

Quantitative metrics were used to measure the effectiveness of the five research correlation systems in the face of these attacks. Collectively, the correlators were able to recognize 95% of the attack steps for which underlying sensors produced alerts and were able to identify 80% of the targets of those attack steps. Individually their performance varied significantly with attack recognition rates varying between 13% and 80% and target identification rates varying between 9% and 51%.

In the second experiment, the two autonomic response engines were able to provide a reasonable response in real-time, consistently at speeds less than one millisecond when faced with multiple independent attacks.

The preliminary results presented here represent those available at conclusion of the experimentation process by BBN. Subsequent analysis of the results led to extensive sensor tuning that generally reduced the stated performance figures. The final results were reported in the warm-wash presentation slides presented on Nov 1st, 2002.

2. Overview

The significant investments in state-of-the-art sensor alert correlation systems and autonomic response systems have yielded several strong state of the art technologies.

The correlation technologies are intended to provide high-level reasoning that goes beyond the capabilities of low-level sensors such as system monitors, firewalls, and network and host-based intrusion detection systems (IDS's). Correlators achieve this through several approaches: combining the information from multiple sensors, relating sensor alerts from different enclaves, threading together sensor alerts about different components of an attack, and/or weighting the criticality of sensor alerts using a mission model[1]. By virtue of these different approaches, each correlator has the capability to reason about a distinct portion of the overall attack space as illustrated in Figure 1.Here five notional correlators C1 to C5 each cover, possibly overlapping, portions of the attack space. Collectively they are able to provide useful information over a broader array of attacks than each understands individually. Correlators have only recently reached the level of maturity where it is conceivable to assess the collective progress in the field.

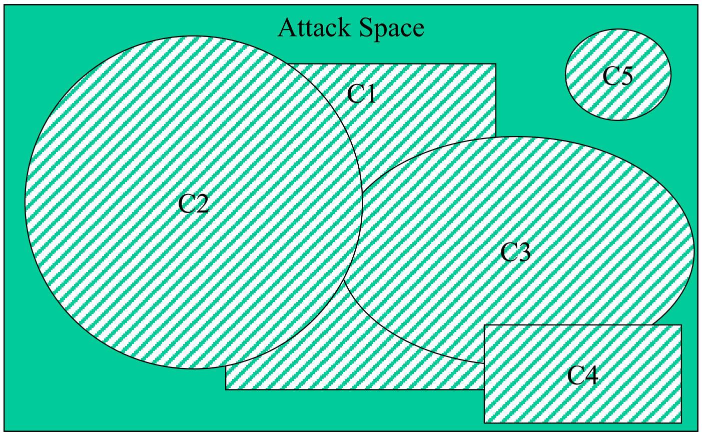


Figure 1: Covering the Space of Possible Attacks

The response technologies are designed to provide autonomic real-time responses to detected cyber attacks. Of the two technologies, one generates network based autonomic responses from correlated attack data, and one generates host based autonomic responses based on input from a highly instrumented server. Current practice requires a human in the loop to both analyze alert data, and initiate a manual response often times hours after the malicious activity is detected. By demonstrating autonomic real time response capabilities in a laboratory experiment, progress in the field will be validated.

This report describes two independent experiments executed on the same network infrastructure. The first experiment was an experimental validation of five research correlation systems with the express goal of quantifying their ability to *recognize cyber attacks* and correctly *designate their targets*. The second experiment was an experimental validation to assess the *speed* and *utility* of two autonomic response technologies.

[1] A mission model is a model of the network and host assets and their use or value to humans in accomplishing some real world mission.

To de-emphasize the inherent competitive nature of a validation effort and focus rather on the cumulative progress of CyberPanel Program research as a whole, the names and specifics of the individual correlation and response systems are not presented herein.

The experiment was carried out on a testbed network constructed at the DARPA-sponsored Information Assurance and Survivability (IA&S) Lab in the Technology Integration Center (TIC) in Arlington, Virginia. The experimental network was protected by typical security policies, populated with a broad array of network sensors, and used in a notional military mission. Simple cyber attack scenarios were designed that embody the characteristics of typical current-day attacks in use on the Internet. Each attack was scripted and perpetrated against the target network from an attacker host notionally located on the Internet. Sensors distributed around the target architecture generated both alerts corresponding to detection of the attacks and other alerts corresponding to typical user or system behavior. Alerts were deposited into a data repository [1] and consumed by the correlators in near real-time. Results produced by the correlators were then analyzed with respect to a set of numerical metrics to determine their performance. The content of the repository was labeled and archived to enable further research in the field.

3. Background

Both intrusion detection systems (IDS) and correlation systems attempt to provide security analysts with an improved understanding of cyber attacks, while suppressing false alarms. Correlation systems however operate at a higher-level and their performance has not been quantitatively tested until now. In comparison to existing second and third-generation intrusion detection testing efforts, the correlation system validation effort described here is a reasonably simple and straightforward first effort to apply attack and target identification metrics to correlation system performance.

This effort has been modeled after existing work in the intrusion detection testing field and many ideas from those evaluations have shaped this effort. Several aspects of data set creation were modeled after the MIT/LL [2][3][4] and AFRL [5] IDS evaluations. Data was generated using a testbed, but unlike the MIT/LL effort, there were no virtual hosts – all machines were instantiated with real hardware. Attacks were scripted to allow data sets to be re-created if (and when) problems were discovered after the run completed. Data set labeling and archiving will provide a lasting resource of sensor alerts similar to the MIT/LL repository of raw data [6]. Similar metrics were employed in the form of target identification metrics that relied on simpler attack recognition metrics. Future efforts to validate correlation system performance will likely want to use real, operational background datasets similar to the way Mueller and Shipley [7] did in their recent review of IDS's. The related efforts are described in more detail below.

The 1998 and 1999, MIT/LL experiments explored the capabilities of research IDS's. The 1999 evaluation effort used a testbed that generated live background traffic similar to that on an air force base containing 100's of users on 1000's of hosts. 200 instances of 58 attack types, including stealthy and novel attacks, were embedded in realistic background traffic. Detection and false alarm rates were measured, and Receiver Operating Characteristics (ROC) curves were generated for more than 18 research IDS systems. The attack categories included DoS, probe, remote-to-local, and user-to-super-user attacks. Attacks were counted as detected if an IDS produced an alert for the appropriate victim machine that indicated traffic or actions on a victim host generated by the attack. Attack identification metrics measured ability of IDS's to provide the correct attack name and other details including the IP source address, ports used, and beginning and end of the attack. In addition, detailed analyses were performed for a few high-performance systems to determine why specific attacks were missed. An important result of the MIT/LL evaluations was an intrusion detection data set that includes weeks of background traffic, host audit logs, and hundreds of labeled and documented attacks. This data set has been used extensively by researchers, used as part of a data mining competition [8], and recently posted to a public web site [6]. One example of how this data has been used was an evaluation of five commercial IDS's performed by Anzen Computing [7].

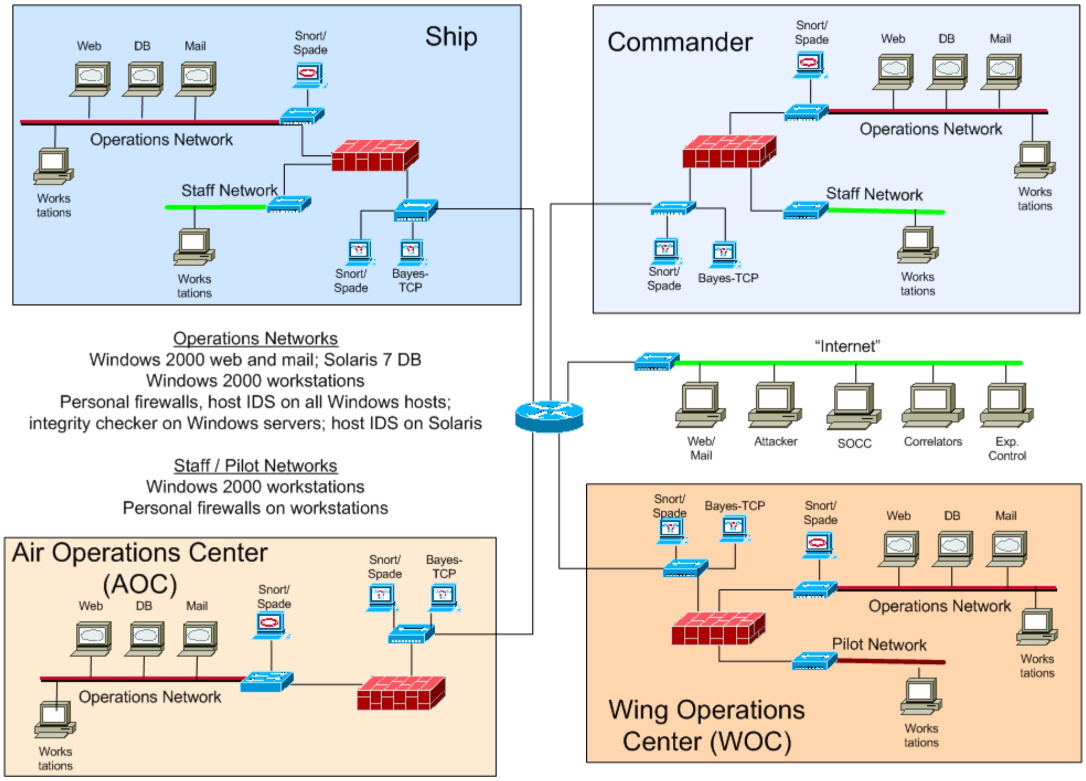
In parallel with MIT/LL, the Air Force Research Laboratory (AFRL) performed real-time testing of research IDS in both 1998 [5] and 1999. IDS systems were installed in a testbed, four hours of background traffic was generated, and attacks were launched against hosts in the midst of this background traffic. AFRL simulated a large network by developing software to dynamically assign arbitrary source IP addresses to individual network sessions running on testbed computers.

A recent review of IDS's sponsored by Network Computing and performed by Neohapsis [7] included 13 commercial IDS's and the open-source Snort IDS [10]. The review assesses performance under high, realistic traffic loads that have grown in complexity over the years. Qualitative results focus on practical characteristics including ease-of-use of the management framework, stability, cost effectiveness, signature quality/depth, and ease of customization. Quantitative results include the number of attacks detected [7]. Realistic background traffic was created by mirroring traffic from DePaul University in Chicago onto an isolated testbed network. This traffic was from a backbone network with 5,000 to 7,000 packets per second. It was found that only seven of the IDS's tested could operate at these high traffic loads and that one crashed after only a few minutes. No careful analysis was made of false alarm rates. Nine recent attacks were launched against eight network IDS's. Each IDS was scored in the areas of management framework, signature quality/depth, stability of engine, cost-effectiveness, and customization.

As shown in the results of these evaluations, no one current intrusion detection system detects all cyber attacks. IDS research continues, however researchers have also turned their attention to higher-level correlation systems to gather and combine evidence from many different intrusion detection systems and make sense of this broader base of evidence for better attack detection. Current correlation systems have taken steps toward this goal and can collect this broad array of evidence and intelligently group and discard alerts to identify cyber-attacks. This validation effort sought to measure the ability of these systems to assist a human administrator in analyzing the vast quantity of sensor alert data seen when analyzing many different intrusion detection sensors.

4. Experiment Network Architecture

The experiment was conducted on a network testbed composed of 48 hosts segregated into four sites of one or two networks each. A central network provided emulation of Internet web and email servers and served as home for the five correlation systems, two autonomic response systems, the alert collection/distribution system, the attacker host, and the experiment controller. In all, 61 hosts were used in the experiment.



Operations Networks
Windows 2000 web and mail; Solaris 7 DB
Windows 2000 workstations
Personal firewalls, host IDS on all Windows hosts;
integrity checker on Windows servers; host IDS on Solaris

Staff / Pilot Networks
Windows 2000 workstations
Personal firewalls on workstations

Figure 2: Target Network Architecture

The target architecture was designed to support a notional military task force during a conflict. The task force was composed of four organizational units (enclaves), each with its own network:

- A Commander: The location where the Task Force commander resides.
- One deployed destroyer (ship).
- An Air Operations Center (AOC).
- One Wing Operations Center (WOC).

Figure 2 shows the underlying enclave structure of the task force. Mission critical network traffic was scripted and infused into the network in line with the military scenario, which revolved around updating and distributing mission plans throughout the four enclaves. Cooperative communications between all four enclaves included updating target information in the databases via the plans server and disseminating it to the other plans servers throughout the architecture. Email traffic was relayed to share the updates with critical user workstations. Web client software also resided on user workstations and modeled real users browsing the Internet by creating realistic yet automated HTTP sessions to retrieve real content from the modeled Internet. This modeled user activity along with the scripted background mail and web traffic was critical to create a realistic network environment in the testbed.

Each enclave was defended by a local security policy enforced by restricting to a minimum access to critical resources, web and email access privileges, etc. The network was populated with a total of 69 intrusion detection sensors distributed over the enclaves as shown in Table 1. Two network based sensors, EBayes-TCP [11] and Snort [10], were used together with three host-based sensors: USTAT [12], Tripwire [13], and WinStat [12]. Linux IPTables [14] firewalls enforced the defined security policy at enclave gateways and provided alerts for attempts to violate this policy. The ZoneAlarm [15] personal firewall provided the same functionality at the host level.

Sensor Name	**Totals By Enclave**				**Total**
	Cmdr	**AOC**	**WOC**	**Ship**	
Snort	2	2	2	2	*8*
Ebayes-TCP	1	1	1	1	*4*
IPTables	1	1	1	1	*4*
ZoneAlarm	8	4	8	8	*28*
Tripwire	4	4	4	4	*16*
WinSTAT	2	2	2	2	*8*
uSTAT	1	0	0	0	*1*
Total Sensors	*19*	*14*	*18*	*18*	*69*

Table 1: Sensors By Enclave

5. Sensor Configuration

Each sensor's configuration required tuning to achieve specific alert set goals. An initial look at correlation research efforts showed that much of the current research is focused on alert reduction and prioritization by clustering alerts using common attributes. Hence the validation goal was to determine how well correlators identified and strung together attack steps and not to test their ability to deduce the likely occurrence of steps for which there is no evidence. This led to the primary alert set goal of having at least one alert corresponding to each attack step.

Prior to running the experiment, several system tests were conducted to verify alert set contents. These tests revealed that multiple events can occur for each attack step and that sensors of differing types and configurations were capable of alerting on different aspects of the steps, providing a broad range of evidence for attack step identification. These observations resulted in a secondary alert set goal of obtaining more than one alert for each attack step. It was further desirable that the alerts be generated from different sensors. This is because human analysis showed that the combination of information from disparate sensors provides much better insight into what is going on at a local point while correlation of similar alerts from same-type sensors distributed across a set of networks might provide better evidence of a wide spread attack. Both are important.

A final alert set goal was to include a high volume of realistic non-attack related alerts so that the alert reduction capabilities of the technologies could be sufficiently tested and to uncover scalability issues such as a technology's ability to keep up with high alert volumes in near real time.

Scripted mission and background traffic included anomalous user activity as often observed on live networks. Sensors were configured to alert when activity from these scripts varied from defined system use or violated defined security policy (e.g., the user `ping`ing the external router to debug a network connectivity issue, adding an O/S user account to a database server that should never be accessed directly, or attempts by the administrator to telnet into a server to fix a problem.) These alerts, together with those generated by the scripted cyber attack alerts described below, form the sensor alert data sets on which the correlation systems operated.

In order to achieve all these goals, significant time was spent in placing and configuring the sensors. The configuration of each sensor is summarized in Table 3.

Sensor	***Configuration***	***Alerting Frequency***
Snort/Spade	Standard rule sets with reporting enhancements required for two correlators.	Near real-time
eBayes-TCP	Standard configuration	Near real-time
IPTables	Local policy set to enforce gateway security policy. Alerts generated for any policy violation.	Near real-time
ZoneAlarm	Local policy set to allow only the traffic required per the defined use of the machine. Alerts generated for violation of the local policy.	Near real-time
Tripwire	Integrity check on the system registry, O/S files, and "critical" files as defined by the use of the machine (e.g., /Inetpub subdirectory for an IIS web server.) Checks run every 5 minutes. Duplicate alerts issued on subsequent runs.	Upon conclusion of each integrity check.
WinSTAT	Use similar file lists as Tripwire when matching attack scenarios.	Near real-time
uSTAT	Monitor database files and critical O/S files when matching attack scenarios.	Near real-time

Table 2: Sensor Configurations

6. Correlation Experiment Attack Description

The cyber attacks used in the correlation evaluation represent those commonly found on the Internet and are of particular importance for correlation systems. It was explicitly *not* the charge of this study to push the envelope in developing novel new coordinated cyber attacks. Each attack was composed of a sequence of distinct *steps,* each step representing an atomic attacker activity. For example, a typical attack might involve a network *surveillance step*, followed by an *intrusion step* through a known vulnerability, followed by a *privilege escalation step* to improve access to the target, and finally achieve some *goal step* such as the theft of information or denial of some system service. These individual attack steps were designed to be relatively simple and to explicitly trip sensors so as to create alerts to provide evidence of each step; the goal being to assess correlation capabilities as distinct from sensor capabilities.

Network surveillance, such as scanning and probing, was achieved using the *nmap* tool [16] in combination with a banner-grabbing program. Scan-only attack runs used a custom wrapper for nmap to yield stealthier scans with fewer packets and longer inter-packet delays. Privilege escalation was accomplished using either the Microsoft IIS Unicode vulnerability [17] or the IIS .printer buffer-overflow [18]. Malicious software was downloaded to a victim host via *ftp* or a custom hex-encoded tunneling mechanism. Attacker goals were typically carried out through binary programs, for example, denial of service was achieved with a custom *pingflood* tool that generates ping messages; common hacker tools like *samdump*, *l0phtCrack* [19], and *netcat* were also used. A generic worm attack was implemented with a custom binary program that uses the IIS-Unicode exploit and can self propagate, but only if given the command to do so at each hop by a central attack controller. The controller was used to guarantee that the worm could not “escape” the confines of the testbed.

16 basic categories of attack were planned based on attack spread method and goal. Attacks spread in one of four ways:

- ***Directed attacks*** - those in which the attacker’s motive is to achieve some singular goal on a particular host.
- ***Stealthy-directed attacks*** - directed attacks prefaced with a stealthy scan.
- ***Worm attacks*** - spread in a worm-like fashion attacking all possible victim hosts, not merely focusing on any one of particular interest.
- ***Stealthy worms*** - worms in which each spread is prefaced by a stealthy scan.

Each attack may have one of four primary goals:

- ***Denial of service (DoS)*** - sought to achieve a simple network denial of service
- ***Data theft*** - sought to steal or exfiltrate some critical information from a computer system such as the contents of a database or a password file.
- ***Defacement*** - sought to deface the webpage of a web-server

- ***Backdoor*** -sought to setup a very simple backdoor by which the attacker can return to a previously compromised host at some later time.

Each attack embodied a single mechanism of spread and attack goal. In addition, three stealthy-scan attacks were developed using the custom *nmap* wrapper in order to focus on this particularly important aspect of cyber defense. The scan attacks, *scan0, scan1,* and *scan2*, were comprised of single host scans with 11 TCP SYN packets that were sent to each of 11 TCP ports at one-minute intervals. This host scan was targeted against 2 hosts at one enclave, then two hosts at each of two enclaves, and finally four hosts at one enclave to compose scan0, scan1, and scan2 respectively.

	DoS	**Data Theft**	**Defacement**	**Backdoor**
Directed	*Directed1 DOS*	*Directed2 Web Deface*	*Directed3 Steal data*	*Directed4 Backdoor*
Stealthy-directed		*Sdirected2*		*Sdirected4*
Worm	*Worm1 DOS*	*Worm2Web Deface*	*Worm3 Steal Data*	*Worm4 Backdoor*
Stealthy-worm	*Sworm1*			

Table 3: Attack Categories

Eleven of the 16 basic attack types were used in the experiment in addition to the three stealthy scans for a total of 14 attacks. Table 3 shows which of the possible attacks were implemented (in addition to the stealthy scans) and provides a unique naming convention for each attack that are used for reference in discussing results.

7. Response Experiment Attack Description

The response experiment attempted to measure the time to choose a response and to assess at a high-level the suitability of the response chosen. The experiment objective was not extensive analysis of the harm done by response to false detections nor detailed comparisons between the two systems being tested. Differences in the operation and use of the systems being tested led to use of simple, high-level metrics. The Responder1 system received input from one of the correlation tools and specified responses across the entire test network. The Responder2 system received input from a single, highly instrumented Linux web server and generated specific responses for that host only. Thus, the experiment involved two sets of attack input for the two systems, with response time being measured and response accuracy subjectively assessed. Responder1 was tested using data from a correlation tool corresponding to 5 attack experiment runs in the correlation experiment. Responder2, on the other hand, was tested using 5 simple attack scripts adapted from the correlation experiment for use with the Linux web server.

Responder1 was tested by connecting the system to the database of alerts and "playing" each set of the associated correlator's output for Responder1 to read as input. The correlator had previously been setup and run during the correlation experiment, and its output was collected by the database for subsequent replay in the response experiment (and for future analysis). Data corresponding to the following attack runs was used in the response experiment:

1. Directed DOS
2. Directed Backdoor
3. Stealthy Deface
4. Worm Deface
5. Worm Steal

See the correlation experiment attack section for description of these attacks. Any background (non-attack) alerts issued by the correlator in the original data run were left in the dataset when used as input to Responder1. After each "set" of correlator output (corresponding to one correlation experiment attack run) Responder1's output log of events was collected. These logs were analyzed by manually parsing them to identify the "best" response chosen and to calculate the time it had taken to compute the response as well as assign a score denoting the quality of the response.

Responder2 was tested by running five simple, scripted attack scenarios against the Linux web server that it protected. The web server was setup in a standalone network segment with no background traffic either generated internally or externally to impinge against the server. The Responder2 detector was installed and calibrated by the developers themselves. The following five attack scripts were created and run for the experiment.

1. *Guess FTP password*: In this attack the adversary does a portscan of the server and then performs 10 attempts to guess the login/password on the ftp server.
2. *Guess Telnet password*: In this attack the adversary does a portscan of the server and then performs 10 attempts to guess the login/password on the telnet server.
3. *DOS*: This scenario is modeled after the correlation experiment directed attack. Here the attacker scans, probes, breaks-in using a remote-to-root wu-ftp buffer overflow attack, and attempts to deny service by filling using up file system inodes and storage space in /tmp.
4. *Web Deface*: This scenario is modeled after the correlation experiment directed attack. Here the attacker scans, probes, breaks-in using a remote-to-root wu-ftp buffer overflow attack, and defaces the servers webpage by writing a new index.html.
5. *Steal data*: This scenario is modeled after the correlation experiment directed attack. Here the attacker scans, probes, breaks-in using a remote-to-root wu-ftp buffer overflow attack, and steals the host's password file.

The entire experiment was carried out with an automated script situated on the attacker's machine. For each attack run:

- First a control connection (telnet session) to the Responder2-protected web server was established via a third machine to avoid the control connection being mistaken by the sensor as part of the attack.
- Responder2 was started, via this control connection.
- Then the attack was launched from the attacker host.
- A few minutes after the attack finished, the control script shutdown the sensor and converted the output log files from binary to ASCII text and archived them for later analysis.

8. Correlation Experiment Alert Set Characterization

Alert sets were comprised of all alerts from all sensors created during an attack run. An attack run consisted of running the previously mentioned user activity scripts and one attack script on the testbed. Each run took between approximately 15 and 90 minutes to complete. All sensor alerts were expressed using the Internet Engineering Task Force (IETF) Intrusion Detection Working Group (IDWG) Intrusion Detection Message Exchange Format (IDMEF) 0.3 draft standard [20]. Because IDMEF only defines the container for expressing intrusion activity and not the actual semantics of the events, a common sensor alert ontology was created and used in the experiment. This facilitated digestion and processing of the alerts by the correlation technologies.

Table 4 provides a general overview of the gross characteristics of each alert set. The columns list the number of attack alerts generated (percentage of the total number of alerts in each run shown in parentheses), the number of non-attack related alarms, and the total number of alerts generated.

Attacks	Attack Alerts	Other Alerts	Total Alerts
Directed1_DoS	354 (0.32%)	110986	111340
Directed2_Web_Deface	5947 (5.7%)	98951	104898
Directed3_Steal_Data	207 (0.13%)	161019	161226
Directed4_Backdoor	11010 (10%)	94793	105803
SWorm1	1631 (1.8%)	88331	89962
SDirected2	34 (0.039%)	86723	86757
SDirected4	3487 (3.8%)	87776	91263
Worm1_DoS	1595 (1.3%)	117813	119408
Worm2_Web_Deface	1908 (1.4%)	135573	137481
Worm3_Steal_Data	2611 (2.2%)	115648	118259
Worm4_Backdoor	1703 (1.0%)	167806	169509
Scan0	39 (0.027%)	146956	146995
Scan1	59 (0.03%)	177487	177546
Scan2	65 (0.033%)	198112	198177

Table 4: Alert Set Characterization

WinSTAT generated a large proportion of the alerts. These were created when WinSTAT detected Tripwire performing integrity checks on files it was also watching. Issues resulting from a known problem in the Windows 2000 event log prevented the configuration of WinSTAT to ignore Tripwire activities. This illustrates a typical interoperability issue associated with the use of multiple IDS sensors and shows that careful configuration and testing must be carried out when deploying sensors for operational use.

Figure 3 shows the percentages of the attack alerts for each type of sensor across all 14 alert sets. Three sensors, WinStat, Snort, and IPTables provided the majority of alerts, however all sensors provided valuable attack related alerts.

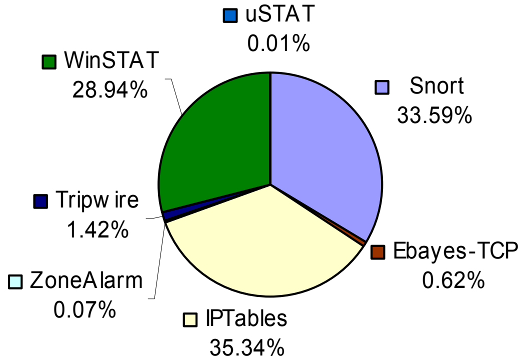


Figure 3: Composite Attack Alert Characterization For All Alert Sets

9. Correlation Metrics

Correlators are expected to *provide a higher level of reasoning than individual sensors.* To measure this improvement, three primary dimensions were identified. Each dimension has direct utility to analysts attempting to cope with the high volume of alert activity emanating from a comprehensively monitored, large-scale network. The most basic dimension, *Multi-Sensor Correlation*, is concerned with taking diverse information from at least two different sensors, and combining it to form an integrated picture of the attacker's activity. Another dimension, *Multi-Step Correlation*, is concerned with piecing together individual steps of an attack, from a collection of sensor activity, to build a picture of the attacker's chain of actions. A final dimension, *Prioritization*, is concerned with weighting alerts, clustering them, and assigning priorities based on the likelihood that the alerts indicate true attacker activity or activity that is harmful to a mission critical resource.

Attack Recognition Metrics: A correlation system's performance was measured along the three dimensions with the following three attack recognition metrics**.** As described in section 6, each attack is comprised of individual *attack steps* and each step yields one or more corresponding sensor alerts in the resulting alert set. Based on the notion that each attack step was an *observable event*, three metrics were defined:

- **DM1: Multi-sensor Correlation**
  - *true* iff at least 2 alerts from different sensors are combined to recognize an attack step

- **DM2: High-level Reasoning**
  - *true* iff 2 or more attack steps are combined

- **DM3: Prioritization**
  - *true* iff an attack step is evidenced in the set of correlated sensor alerts, defined by correlator assigned priority and at most 1% the size of the original alert set

To provide a gross overview of the collective recognition capability for a particular correlator, a general attack recognition metric DM was constructed where DM is true iff any of the above metrics are true:

$$\mathbf{DM = (DM1 \lor DM2 \lor DM3)}$$

In the few cases when an attack step was not visible in the sensor alert set, that step was not counted in the calculation of these metrics. Table 5 demonstrates how these metrics are applied in practice to a given correlator for a single attack. The attack is composed of five distinct steps numbered 1 through 5, and a 1 is listed in the table for each step the correlation system recognizes when judged with a particular metric. Based on multi-sensor correlation (DM1), two steps (1,2) are correlated, meaning that multiple sensor alerts were combined in detecting each step. Based on multi-step correlation (DM2), three steps (1,3,5) where threaded together. Based on prioritization (DM3), the alerts associated with two steps were assigned priority for consideration by humans or a higher-level system. Overall, the correlator would be rated as 80% effective (4 out of 5 steps) in correlating this attack.

Attack	DM1	DM2	DM3	DM
Step 1	1	1	1	**1**
Step 2	1		1	**1**
Step 3		1		**1**
Step 4				
Step 5		1		**1**

Table 5: Example Attack Recognition Rating

Considering DM3, it is clear that the use of 1% *aggregation threshold* is arbitrary. This was selected as an interesting level of achievement for correlators as it signifies a 2-order of magnitude reduction (100-fold fewer) in alerts to be reviewed by the security analyst. If the bound is successively tightened, correlators are required to make increasingly accurate predications and must draw out the most important sensor information. Thus in presenting results, detailed plots are provided that show how the effectiveness of correlation varies as a function of the aggregation threshold; however, for distilling the overall performance of correlators into a single rating, a specific threshold of 1% was used.

To obtain an overall rating for the correlator collective, the "or" operation is applied across all correlators at each step of each attack. If one or more correlators recognized a given step of an attack, the combined system is credited with having recognized that step.

Target Identification Metrics. In addition to measuring the ability of the correlators to identify individual attack steps, it was deemed important to measure their ability to identify the targets of the attacks. Attack target information is critical to helping security analysts formulate responses to the attack. Simple attack step recognition metrics are not sufficient to measure the correlation system's abilities in this respect since there are often *multiple targets* associated with a single attack step, especially in the case of worm-based attacks. Target metrics were based on the identification of the host IP addresses targeted by each attack step. This yields the following metrics for target identification:

- **TM1: Multi-sensor Correlation**
    - *true* iff DM1 $\wedge$ target IP available from the associated correlated alert

- **TM2: High-level Reasoning**
    - *true* iff DM2 $\wedge$ target IP available from the associated correlated alert

- **TM3: Prioritization**
    - *true* iff DM3 $\wedge$ target IP available from the associated correlated alert

For each target metric, the attack step must be recognized prior to target analysis. As with attack recognition metrics, a broad overview metric TM was formed from the individual component metrics:

$$\mathbf{TM = (TM1 \vee TM2 \vee TM3)}$$

Table 6 shows how these target metrics are applied to an attack continuing the example from Table 5. A "1" is given in the table if that target is identified in correlator report(s) that identified that attack step. In assessing the target identification performance using multi-sensor correlation (TM1), the second step of the attack is recognized (by virtue of DM1 in Table 5), but only two of the 3 target IP addresses are present in the corresponding correlated alerts. Notice however, that the missing IP address is present when considering TM3. Thus of the 8 target IP addresses used in the attack, only 6 are actually correlated in one way or another, and thus the overall effectiveness on target identification is rated at 6/8 or 75%. This effectiveness measure can be applied over attacks and correlators to provide a gross overall measure, as is done in the attack recognition metrics.

Attack	Target IP	TM1	TM2	TM3	TM
Step 1	192.168.0.2	1	1	1	**1**
Step 2	192.168.0.4 192.134.0.2 192.156.2.4	1 1		 1 1	**1** **1** **1**
Step 3	192.168.0.3		1		**1**
Step 4	192.155.2.3				
Step 5	192.157.0.6 192.155.2.9		 1		 **1**

Table 6: Example Target Identification Rating

It is acknowledged that the metrics applied here are very generous, however they represent a first attempt to classify the varied aspects and approaches of correlation technologies and to develop a common set of metrics that can be applied to any correlation system. These metrics give a general idea of correlation performance but do not sufficiently measure unique capabilities of specific correlation systems nor do they help in understanding the overall purity and classification of correlator outputs.

10. Response Experiment Metrics

Prior to the experiment, each response system was instrumented by the system developers to output a log of internal operations. This log listed receipt of input events/alerts, response specification events and timestamps corresponding to each event. For each attack run, results were calculated by collecting these logs and computing the time difference between when input was received and the time that the response was chosen based on that input. In the case where multiple input events may have contributed, the timestamp of the most recent input event was taken. This avoided penalizing the response systems for delays inserted within attack scripts or in system operation, and allowed measurement only of the "time to choose a response". Further, response systems only specified the response to be taken but did not (usually) enact it, thus a single attack run yielded multiple instances in which response were specified as the attack progressed.

For each attack run only the "best" instance of the system choosing a response was considered for this experiment. Usually this was the response that would have most effectively stopped the attack, with fewest false alarms.

Results were presented in a table for each system listing the attack run, the time to respond, and a manually generated assessment of the quality of the response. The latter was a score from 1-5, with 1 being an "ideal" response and 5 being either totally ineffective or even damaging. Response system developers were not told ahead of time exactly how the response time or response quality were going to be calculated but did instrument their own systems to produce the log files.

11. Correlation Experiment Results

Table 7 summarizes the results for the attack recognition metric DM for each attack listed in Table 3 using five CyberPanel correlation systems labeled 1 through 5. For each correlation system and for each attack, the table gives the number of attack steps detected when combining that system's results across the three detection metrics. The Visible Steps column gives the total number of attack steps for which alerts occurred in the alert set. The Overall column combines the results for each attack across all correlation systems as described in section 9 above. The attack-step percentages represent the total number of attack steps each correlator recognized across all attacks. Since the primary goal was to assess progress in the field rather than compare correlators directly, the correspondence between correlators and experimental results are purposely hidden. In combination, the correlators were 95% effective in overall attack step recognition, using a DM3 filtering threshold of 1%. Individual correlators rated between 13% and 80%.

Attack Name	Visible Steps	Correlator1	Correlator2	Correlator3	Correlator4	Correlator5	Overall
Directed1:DoS	7	7	6	7	6	1	7
Directed2:Deface	4	3	2	4	1	2	4
Directed3:Steal	8	7	0	0	1	0	7
Directed4:Backdoor	6	6	3	4	5	2	6
SDirected2:Deface	4	2	0	0	1	0	2
SDirected4:Backdoor	5	4	0	3	2	0	5
SWorm1:DoS	10	9	0	0	8	0	10
Worm1:DoS	10	9	8	4	8	0	10
Worm2:Deface	10	0	9	9	8	0	9
Worm3:Steal	10	9	9	7	7	0	10
Worm4:Backdoor	10	10	9	5	8	0	10
Scan0:Intel	2	2	0	2	0	2	2
Scan1:Intel	3	3	0	2	0	2	3
Scan2:Intel	4	4	0	2	0	4	4
Attack Step Count	93	75	46	49	55	13	89
Attack Step Percent		80.65%	49.46%	52.69%	59.14%	13.98%	95.70%

Table 7: Combined Attack Recognition Metric DM

It is important to recognize that although the first four correlators have very general capabilities, Correlator5 is specialized by design to recognize very specific attack steps and was not expected to recognize the breadth of attack steps used in the experiment.

Table 8 summarizes the corresponding results for the target identification metric TM. The Targets column gives the total number of targets for each attack, and the number of targets identified for each attack is given for each correlation system. In combination, the correlators provide an overall target identification rating of 80%, with individual correlators operating at ratings between 9% and 51%.

Attack Name	Targets	Correlator1	Correlator2	Correlator3	Correlator4	Correlator5	Overall
Directed1:DoS	24	20	8	7	3	3	21
Directed2:Deface	9	1	0	6	2	3	8
Directed3:Steal	15	6	2	0	1	0	9
Directed4:Backdoor	11	4	3	4	6	3	9
SDirected2:Deface	6	3	0	0	0	0	3
SDirected4:Backdoor	10	8	0	3	0	0	9
SWorm1:DoS	25	15	0	0	20	0	24
Worm1:DoS	26	18	12	4	15	0	19
Worm2:Deface	26	0	13	14	16	0	19
Worm3:Steal	25	10	0	10	14	0	16
Worm4:Backdoor	29	29	13	6	16	0	29
Scan0:Intel	3	0	0	1	0	3	3
Scan1:Intel	6	0	0	1	0	3	3
Scan2:Intel	7	0	0	1	0	7	7
Attack Target Count	222	114	51	57	93	22	179
Target ID Percent		51.35%	22.97%	25.68%	41.89%	9.91%	80.63%

Table 8: Combined Attack Recognition Metric TM

Notice that the correlators provide overlapping and mutually beneficial recognition characteristics, each correlator performing well on some attacks and poorly on others. This result lends credence to the theory that the technologies may be combined to produce better attack understanding.

Although Table 7 and Table 8 provide a broad overview of system performances, they do not provide insights on each system's performance with respect to a specific metric. Graph 1 shows the overall performance of each correlation system for each metric. Results for DM3 are given at each of 1, 1.5, 2, 2.5, and 3 orders of magnitude alert reduction thresholds. This gives a more accurate picture of the correlator performance and illustrates that, as expected, the greater the threshold, the fewer attack steps recognized.

Of the three metrics, the correlators generally performed most poorly on DM1 in combining multiple sensors. Prior to the experiment, each of the correlators worked with a limited and often non-overlapping, unique set of sensors, often reflecting availability or individual preference. For the experiment however, a common base of sensors was necessary. Thus, each research group was provided with the common sensor ontology described previously and given opportunity to encode knowledge necessary to utilize the full set of experiment sensors. Time restrictions imposed on the experiment limited the degree to which each group was able to utilize this ontology, and we would expect the DM1 performance results to increase considerably with additional time and refinement of the sensor models.

Correlator5 is highly specialized, taking its input from this one sensor type, and can only score with respect to DM1 if it correlates alerts from different copies of the sensor. In the experiment, some unresolved configuration problems with the several copies of that one sensor made it difficult, if not impossible, for Correlator5 to score well on this metric. We believe if these configuration problems had been resolved, the DM1 scores for this correlator would have increased dramatically.

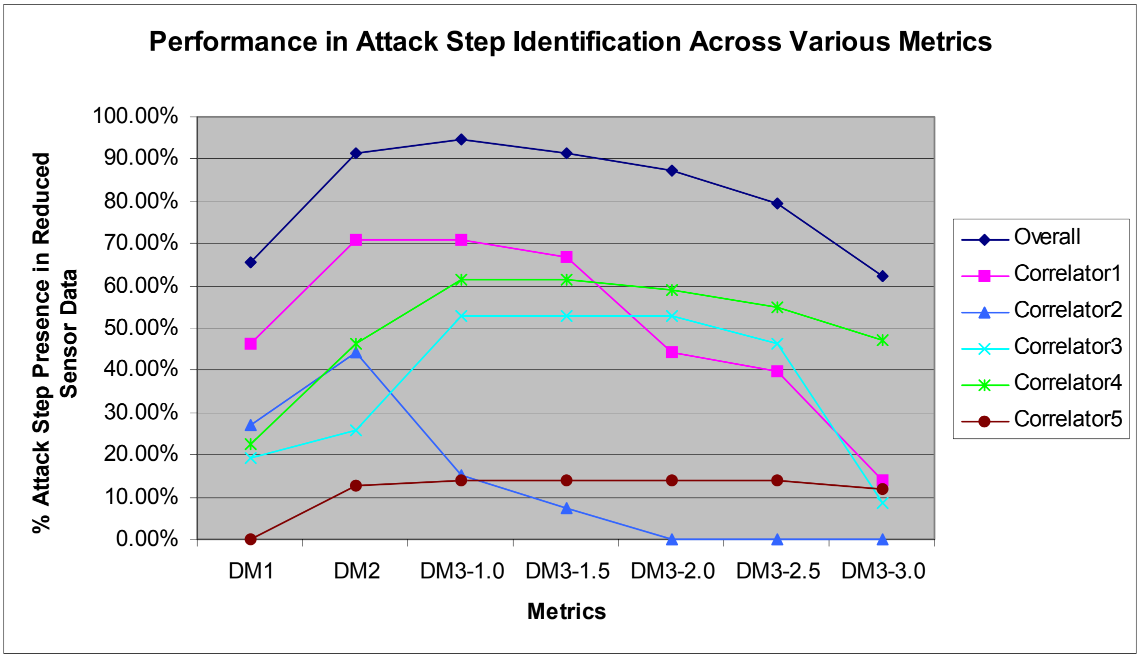


Figure 4: Attack Recognition Performance by Metric

The results on DM2 were encouraging: overall more than 90% of the attack steps where threaded together by one or more correlators. All of the correlators were able to achieve some degree of high-level reasoning, with respect to chaining together multiple steps of an attack to gain a high-level overview of the attacker's activity. This is a particularly important metric and one that distinguishes correlators from individual sensors.

DM3 results were similarly encouraging with greater than 90% of attack steps being recognized. As expected however, performance on DM3 dropped as the aggregation threshold was successively tightened. At a 500-fold reduction in alerts, performance fell to below 80%. Obviously this result is highly dependent on the overall mix of sensor alerts and the quality of sensor tuning. Several weeks of sensor tuning were performed prior to the experiments; however, clearly this result will vary sharply for a particular configuration of a network and its sensors.

Graph 2 shows the corresponding target identification performance with respect to the individual metrics. Obviously, since the target identification metrics pre-suppose attack recognition, one would expect these results to follow the general trends of attack recognition. The TM2 metric gives an overall rating of slightly less than 75%, and the TM3 results fall off dramatically as the aggregation threshold is varied. The target identification metrics use only the IP address of the target. Obviously this is a simple metric; it would be much more useful from the viewpoint of formulating an attack response to also include the *port* or *service* under attack or the exact file or resource being targeted. Re-scoring on this additional level of detail would significantly impact the results.

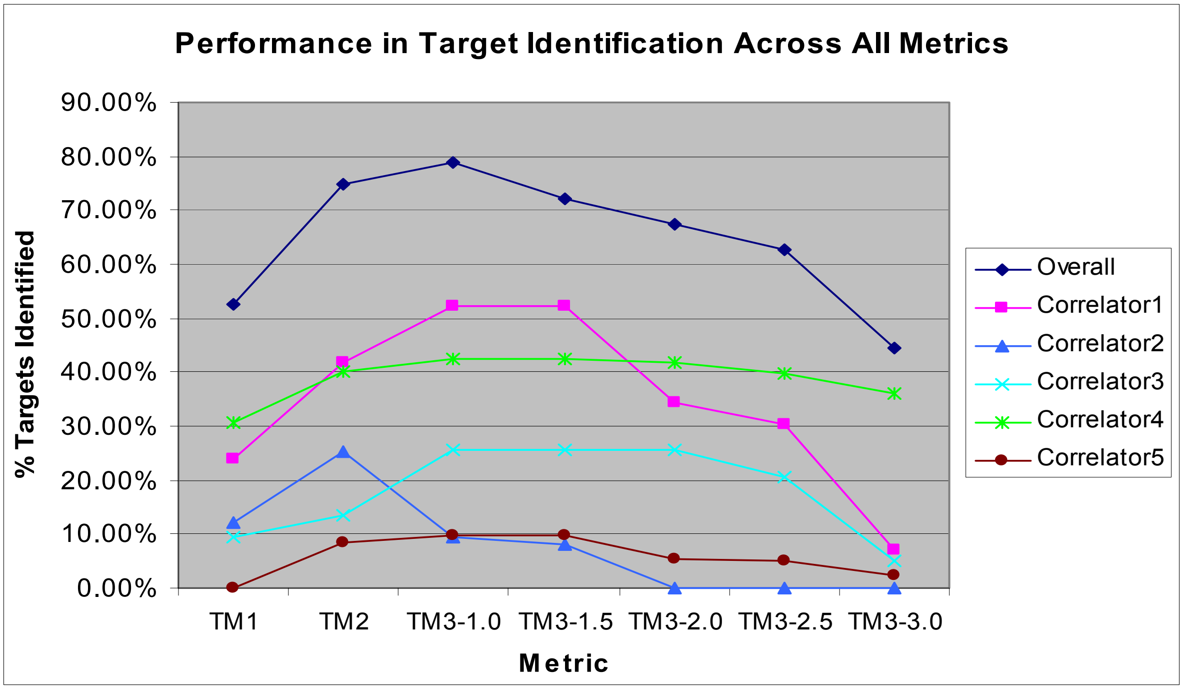


Figure 5: Target Identification Performance by Metric

12. Correlation Experiment Detailed Case Studies

The metrics used to measure overall correlator performance are simple and illustrate the general reasoning ability of these systems. Unfortunately, they do not adequately characterize the details of the correlator output or provide detailed error analysis. To provide some insight into these issues, let us consider two sets of representative results taken from the experiment and examine the detailed performance of the correlators.

Correlator 4 on the Sworm1 Attack. The Correlator4 result for the Sworm1 attack was 80% for both DM2 and DM3 (100x reduction). This was accomplished by outputting 256 reports, all of which referenced parts of the attack.

The Sworm1 attack involves a worm that uses the IIS Unicode directory-traversal attack to propagate successfully to web servers at two enclaves and unsuccessfully to web servers at the other two enclaves. After propagating the worm launches a ping-flood denial of service attack across the network. Sensor alerts provide ample evidence for the correlation system to recognize the attack: At each propagation step Snort sensors emit *WEB-IIS cmd.exe access* and *ATTACK RESPONSES file copied* alerts in response to the initial compromise of each web server. It also generates multiple *WEB-IIS Unicode2.pl script* and *ATTACK RESPONSES* alerts as attack related files are downloaded to the server and the next stage of propagation is launched. Both Tripwire and WinSTAT issue numerous alerts as the worm writes attack-related files to the file system. ZoneAlarm sensors issue alerts corresponding to the blocked adversary actions. Finally, in response to the ping-flood, the Snort sensor produces *MISC Large ICMP Packet* alerts.

The sensor alert dataset for Sworm1 contains 89,962 alerts, of which 1,631 alerts corresponded to the attack. Correlator 4 produced 256 reports as output, of which nearly all referenced multiple alerts related to the attack. Its average report referenced about 9.16 sensor alerts. A total of 2,253 references were to snort alerts, while 92 references were to ZoneAlarm alerts. Report purity was high; 95% of the sensor alerts referenced in the reports were attack related. Commonly mis-correlated alerts were from ZoneAlarm and appear to pertain to background traffic in which the Windows Telnet program was invoked in background traffic scripts and blocked by ZoneAlarm. 141 of the reports referenced only Snort sensor alerts, while the others referenced Snort and ZoneAlarm alerts. About 98% of the Snort alerts referenced were attack related, while about 90% of the ZoneAlarm alerts referenced were non-attack related.

Correlator 4 detected 8 out of 10 attack steps, scoring 80% in DM2 and DM3. This can be attributed to its ability to "put together" multiple worm actions. No one report put the entire worm together, but many reports correlated multiple steps: 9 reports correlated 6 steps each, while 89 correlated 2 steps, and the rest correlated 3, 4, or 5 steps.

Correlator 3 on the Directed1_DoS Attack. The Correlator3 result for the Directed1_DOS attack was 71% for DM2 and 100% for DM3 (100x reduction). This was accomplished by outputting 233 reports, of which only a handful referenced parts of the attack.

The Directed1 DoS attack is similar to the Sworm1 worm attack, except that the attack is directed only at one web server and the ping-flood is launched from only that server. Attack-related alerts are similar in sensor and type to those described for Sworm1, above.

The input sensor alert set contained 111,340 alerts, of which only 354 were related to the attack. Correlator3 issued 233 reports as output, each referencing an average of 5.2 sensor alerts. Correlator3 combined alerts from multiple sensors in 38 of the 233 reports. In those 38 reports, either Tripwire and WinStat alerts were correlated, or various combinations of network sensor alerts were correlated. 76 of the 233 reports were marked high-priority, and of those 76, two reports stand out:

- One report correlated Snort alerts from steps 2, 3, 4, 5, and 6 of the 7-step attack by referencing 10 underlying Snort alerts and was “pure” in that no non-attack alerts were included.

- The other report correlated 3 of correlator3’s own outputs to produce a single, pure, alert that detected scan and probe activity from attack steps 0 and 1. The 3 sub-reports contained: 21 portscan-related alerts from Snort and eBayes, 15 *NMAP ICMP_PING* alerts from Snort, and 9 Snort/Spade and Iptables alerts, respectively.

Overall, Correlator3 scored 100% on DM3 with a 100 fold alert reduction, detecting all 7 steps of the attack with the 2 reports described above providing the coverage. In total about 7 of the reports contained references to multiple attack-related alerts. Other reports may have contained valid correlations of non-attack behavior but these were not analyzed in greater detail.

13. Response Experiment Results

Results from the Responder1 response system are shown in Table 9:

Attack	**Time (uSec)**	**Expected Response**	**Actual Response**	**Rating 1-5**
1. Directed DOS	35	Block IP	Kill conn,Blk IP	1
2. Directed Backdoor	29	Kill Backdoor	Kill conn,Blk IP	2
3. Stealthy Deface	30	Kill PID	Kill conn,Blk IP	2
4. Worm Deface	27	Block IP	Kill conn,Blk IP	1
5. Worm Steal	36	Block IP	Kill conn,Blk IP	1

Table 9: Responder1 Results

Responder1 was easily able to achieve the "sub-one-second" response time, choosing responses on the order of microseconds. Response "quality" results were obtained much more subjectively and should be interpreted as such – the actual utility of any given response will depend directly on the environment in which the response is taken and the effect on mission/background traffic is. We did not directly measure that effect, but merely hypothesized what a reasonable response would be, and assessed the reasonableness of the specified response in comparison.

Responder1 issued mainly network responses in regards to the attack. Detections of scanning, probing, and break-in from the correlator were met with responses to kill the offending TCP connection and block that attacker's IP address. The two actions were specified in the same atomic response event. Obviously the rating of this response as a "2" for several of the examples is purely subjective and is based on the fact that in the case of the backdoor and webpage defacement attacks, a more specific response to take would have been to kill a host process. Blocking the IP address and Killing the connection could also stop the attack but could have greater risk of unwanted side effects.

Responder1 did issue quite a number of false responses due both to false alarms and erroneous reporting of Source IP address by the correlator, detailed analysis of which was not performed. For example from the "Worm-Steal" data run, Responder1 recommends blocking 39 different IP address, of which at most three are relevant to the attack. Other attack runs show similar behavior.

Table 10 shows experiment results from Responder2:

Attack	**Time (uSec)**	**Expected Response**	**Actual Response**	**Rating 1-5**
1. Guess FTP pswd	> 1	Block Port/IP	Kill service (ftp)	4
2. Guess Telnet pswd	> 1	Block Port/IP	Kill login	2
3. DOS	> 1	Kill shell	Escalate, Kill shell	1
4. Web Deface	> 1	Kill shell	Kill shell	1
5. Steal data	> 1	Kill shell	Kill shell	1

Table 10: Responder2 Results

Responder2 was easily able to achieve the "sub-one-second" response time, choosing responses faster than one microsecond. Response "quality" results were obtained much more subjectively and should be interpreted as such – the actual utility of any given response will depend directly on the environment in which the response is taken and the effect on mission/background traffic is. We did not directly measure that effect, but merely hypothesized what a reasonable response would be, and assessed the reasonableness of the specified response in comparison. Since Responder2 is a host-based system we did expect host-based responses, killing of processes or blocking access to the host from an IP address using some host-based firewall.

Responder2 performed well during the attacks that involved a remote-to-root exploit. In these cases we observed that Responder2 tracked the UNIX process that had been exploited and gradually raised the severity of the response specified at each new event until finally the malicious process was specified to be killed. In those cases responses ranged from, "Increase monitoring", to "Alert the Administrator", to "Kill the process". Generally "Kill Process" responses would be issued when the subverted process tried to do something that it did not normally do, like overwrite web pages or read /etc/passwd. In the case of the ftp password guessing attack however, Responder2 specified to kill the ftp service daemon. This response would have stopped the attack but would also have denied ftp service to other legitimate users. In the case of the telnet password guessing a slightly different response was chosen: the login-process that resulted from the login-attempt was killed. This did stop part of the attack, but allowed future password guessing attempts from the same source.

14. Concluding Remarks

In summary, the correlators were able to collectively recognize 95% of the attack steps and correctly identify 80% of the targets, although individually their performance was much lower and varied significantly. The correlators exhibited complementary capabilities indicating that combining the technologies and concepts from multiple research projects is likely to provide effective new correlation results.

Most correlators consumed alerts in real-time, with the exception being Correlator 2, which proved unable to keep up with the data rates and had to be run off-line. Correlator 3 was run in real-time, however it did not generate real time correlations. Correlators 1 and 2 did not produce significant data reduction. Correlators 3 and 4 were able to produce two orders of magnitude reduction with no throttling on the data. Correlator 5 produced three orders of magnitude data reduction, however it was only processing alerts from one sensor type, resulting in alerts being discarded without processing.

The correlators performed poorly in combining alerts from multiple sensors largely due, we believe, to time-constraints imposed on the experiment. Further work is required on target identification. The results here are based only on target IP address, and significantly more accurate target information is required to facilitate effective responses.

Unfortunately, all of the correlators produce outputs in their own formats, making it extremely difficult to automatically combine their results and feed them to higher-level systems such as those focused on response recommendation. It is clear that a standard for correlation reporting is necessary before significant progress can be made in combining correlated alerts on a large-scale.

The most encouraging aspect of these results is the overall performance of the correlators in threading together multiple steps of an attack to provide a higher level of reasoning (DM2 and TM2). This complex task represents a significant step away from understanding individual sensors, and the correlators provided an unexpected result in this area.

In general, the metrics and results of this correlation study provide a base-line characterization of the progress made in correlation research. The study was, by design, based only on relatively straightforward attacks that could be expected in the general Internet community. The attacks were perpetrated against networks making no use of VPN's or other higher-level protective mechanisms. Further studies that push the envelope of strategically motivated, coordinated attacks on networks with increasingly realistic system usage models and security policies would challenge the correlation systems significantly beyond this study.

It is acknowledged that the metrics applied in the correlation experiment are extremely generous. Use of these metrics has helped in obtaining a general idea of correlation performance but has not fully evaluated all aspects of correlation system output. Specific areas for exploration include composition of correlated alert clusters and a correlator's ability to properly classify combined events at higher levels. Analysis of the results is still ongoing and a false-correlation analysis is expected in the near future.

In general, the response systems performed substantially faster that expected, with reasonable responses generated in less than one *milli*second for all attacks, and approaching the speed of measurement. The range and level of sophistication in the generated response appears to be evolving, and thus the experiments to date focus primarily on establishing the speed of response more than the overall breath and appropriateness. Both systems appear sufficiently fragile that they could not currently be employed directly into operational systems; however, this appears to be the next logical step in their evolution. Both systems have a strong coupling to underlying sensors and correlators that do not necessarily provide an accurate picture of an attack in progress.

The preliminary results presented here represent those available at conclusion of the experimentation process by BBN. Subsequent analysis of the results led to extensive sensor tuning that generally reduced the stated performance figures. The final results were reported in the warm-wash presentation slides presented on Nov 1st, 2002.

15. Acknowledgement

In addition to BBN Technologies, this research was conducted in conjunction with Teknowledge Corporation, MIT Lincoln Laboratory, and Dartmouth University. The same research team jointly authored this final report. This research was supported by the Defense Advanced Research Projects Agency under Air Force contracts F30602-02-C-0106, F19628-00-C0002, F30602-00-C-0057 and F30602-99-D-001-0016. Opinions, interpretations, conclusions, and recommendations are those of the authors and are not necessarily endorsed by the United States Government. Portions of this report were submitted for publication to IEEE Security and Privacy in October 2002.

16. References

[1] L. Tinnel, S. Allain, and L. Clough, "Cyber Mission Modeling for Information Survivability," submitted to the DARPA Information Survivability Conference and Exposition III (DISCEX-III).

[2] R. P. Lippmann, D.J. Fried, I. Graf, J.W. Haines, K.R. Kendall, D. McClung, D. Weber, S.E. Webster, D. Wyschogrod, R.K. Cunningham, and M.A. Zissman, Evaluating Intrusion Detection Systems: The 1998 DARPA Off-Line Intrusion Detection Evaluation, in Proceedings of the 2000 DARPA Information Survivability Conference and Exposition (DISCEX), Vol. 2, 12-26, 2000, IEEE Computer Society Press: Los Alamitos, CA.
http://www.ll.mit.edu/IST/pubs/discex2000-rpl-paper.pdf

[3] R. P. Lippmann, J.W. Haines, D.J. Fried, J. Korba, and K. Das, The 1999 DARPA Off-Line Intrusion Detection Evaluation. Computer Networks, 2000. 34(2), 579-595.
http://www.ll.mit.edu/IST/ideval/pubs/2000/1999Eval-ComputerNetworks2000.pdf

[4] R. P. Lippmann and J. Haines, Analysis and Results of the 1999 DARPA Off-Line Intrusion Detection Evaluation, in Recent Advances in Intrusion Detection, Third International Workshop, RAID 2000 Toulouse, France, October 204, 2000 Proceedings, H. Debar, L. Me, and S.F. Wu, Editors. 2000, Springer Verlag, 162-182.
http://link.springer.de/link/service/series/0558/bibs/1907/19070162.htm

[5] R. Durst, T. Champion, B. Witten, E. Miller, and L. Spagnuolo, Testing and Evaluating Computer Intrusion Detection Systems. Communications of the ACM, 1999. 42,(7), 53-61.
http://www1.acm.org/pubs/articles/journals/cacm/1999-42-7/p53-durst/p53-durst.pdf.

[6] MIT Lincoln Laboratory, DARPA Intrusion Detection Evaluation Data Sets, Jan. 2002.
http://www.ll.mit.edu/IST/ideval/data/data_index.html

[7] P. Mueller and G. Shipley, Dragon claws its way to the top. Network Computing, 20 August 2001, 45-67.
http://www.networkcomputing.com/1217/1217f2.html.

[8] C. Elkan, Results of the KDD'99 Classifier Learning Contest, September 1999, Sponsored by the International Conference on Knowledge Discovery in Databases.
http://www-cse.ucsd.edu/users/elkan/clresults.html.

[9] D. Song, G. Shaffer, and M. Undy, Nidsbench - A network intrusion detection test suite. 1999Recent Advances in Intrusion Detection, Second International Workshop, RAID 1999, West Lafayette, Indiana. http://citeseer.nj.nec.com/cache/papers/cs/19418/http:zSzzSzwww.monkey.orgzSz~dugongzSztalkszSznidsbench-slides.pdf/song99nidsbench.pdf

[10] M. Roesch, Snort - Lightweight Intrusion Detection for Networks, in USENIX 13th Systems Administration Conference - LISA '99. 1999: Seattle, Washington http://www.usenix.org/publications/library/proceedings/lisa99/roesch.html.

[11] A. Valdes and K. Skinner, “Adaptive, Model-Based Monitoring for Cyber Attack Detection,” in Recent Advances in Intrusion Detection, Third International Workshop, RAID 2000 Toulouse, France, October 204, 2000 Proceedings, H. Debar, L. Me, and S.F. Wu, Editors. 2000, Springer Verlag, 80-92

[12] G.Vigna, M. Eckmann, R. A. Kemmerer, “The STAT Toolsuite,” Proceedings of DARPA Information Survivability Conference and Exposition I (DISCEX-I) Vol. II, pg. 46-55.

[13] Tripwire: http://www.tripwire.com.

[14] Netfilter/IPTables: http://www.netfilter.org/

[15] Zone Alarm: http://www.zonealarm.com.

[16] nmap: http://www.nmap.org/

[17] Unicode vulnerability: http://online.securityfocus.com/bid/1806

[18] IIS .printer buffer-overflow: http://online.securityfocus.com/bid/2674

[19] l0phtCrack: http://www.atstake.com/research/lc/index.html

[20] Intrusion Detection Message Exchange Format (IDMEF): http://www.ietf.org/html.charters/idwg-charter.html.