Cooperative Intrusion Traceback and Response Architecture (CITRA)

Dan Schnackenberg
Harley Holliday
Randall Smith
The Boeing Company, Phantom Works
daniel.d.schnackenberg@boeing.com
harley.s.holliday@boeing.com
Randall.Smith@PSS.boeing.com

Kelly Djahandari
Dan Sterne
NAI Labs, Network Associates
kelly_djahandari@nai.com
dan_sterne@nai.com

Abstract

The Cooperative Intrusion Traceback and Response Architecture (CITRA) was originally developed as an infrastructure for integrating network-based intrusion detection systems, firewalls, and routers to trace attacks back to their true source and block the attacks close to that source. Prototype implementations of CITRA have proven useful for integrating other security mechanisms in support of automated response to both intrusions and other changes in security status of a system. This paper provides an overview of CITRA policy mechanisms and how CITRA integrates diverse security technologies to improve system defense.

1. Introduction

The Cooperative Intrusion Traceback and Response Architecture (CITRA) is an architecture enabling intrusion detection systems, routers, firewalls, security management systems, and other components to interact to (1) trace intrusions across network boundaries; (2) prevent or mitigate subsequent damage from intrusions; (3) consolidate and report intrusion activities; and (4) coordinate intrusion responses on a system-wide basis. The CITRA concepts and the initial prototype software were developed under DARPA funding[1] by researchers at Boeing's Phantom Works, Network Associates' NAI Labs, and University of California Davis' Computer Security Lab.

CITRA was developed to help automate intrusion analysis and response tasks that are ordinarily performed by human administrators. Automating these tasks is critical for two reasons. First, human analysis of a single intrusion can take hours, days, or longer. During this period, the costs and damage caused by an intrusion, e.g., "down time" or corruption of data, can rapidly mount. Second, the analysis is complex; it requires expert network administrators, who are chronically in short supply. If such experts are not immediately available when an intrusion occurs, determining how to respond may be delayed, allowing damage to increase further. With appropriate automation, it may be possible to respond more quickly and at times when expert administrators are not available.

To analyze and respond to intrusions, administrators gather information from a variety of sources on an as-needed basis. Typical sources include traffic sniffers, network management systems, operating system audit trails, router/switch diagnostics, and specialized security subsystems. Automating the analysis and response processes similarly requires the ability to obtain and merge information from diverse devices and control the type and volume of information they supply. It also requires dynamically controlling the extent of network access available to potential intruders.

CITRA and its software libraries constitute an integration infrastructure for automating these processes. CITRA was specifically designed to facilitate low-cost integration of independently developed components (e.g., commercial intrusion detection systems) and flexible adaptation of these components' behaviors. An integral part of CITRA is the Intruder Detection and Isolation Protocol (IDIP), which defines the information that CITRA-enabled components may exchange. [1] presents

[1] This research was supported by DARPA/Rome Laboratory contracts F30602-96-C-0318, F30602-97-C-0217, F30602-97-C-0309, F30602-99-C-0181, and F30602-00-1-0602, and DARPA/SPAWAR contract N66001-00-C-8602.

an overview of CITRA and IDIP[2]. This paper explains key CITRA features that were beyond the scope of [1]. It also describes our recent experience using CITRA to integrate and adapt new types of components. This experience suggests that CITRA is an infrastructure that enables synergistic integration of diverse components into a cooperative, self-protecting network.

This paper provides a discussion of how CITRA can be used with other security technologies to enable better intrusion detection and response. Section 2 provides a brief overview of CITRA traceback, response, and reporting mechanisms. Section 3 discusses the issues related to controlling automated response, and describes how CITRA uses response policy mechanisms to ensure that only desired responses are taken. Section 4 describes how the CITRA mechanisms and IDIP protocol can be integrated with other security technologies to enable flexible intrusion response. Section 5 discusses how CITRA is being used in ongoing work related to intrusion detection and response. Section 6 discusses related work. Section 7 provides a summary.

2. CITRA Background

CITRA uses two levels of organization. First, CITRA communities are administrative domains controlled by a management component called a Discovery Coordinator. Second, CITRA communities consist of interconnected neighborhoods. A CITRA neighborhood is the collection of CITRA devices that are "adjacent" in the sense that no other CITRA nodes are positioned between them. (See Figure 1.) CITRA utilizes the neighborhood structure to trace and respond to intrusions. CITRA uses the IDIP [1] protocol for centralized reporting of intrusion-related events, attack traceback, and automated response.

IDIP traceback is accomplished by auditing network traffic at CITRA-enabled devices. When an attack is reported, CITRA uses this network audit trail to track the packets involved in the attack back as close to the attack source as possible. Figure 2 illustrates how the traceback is accomplished. If a CITRA-enabled detector detects an attack (step 1), the detector sends a traceback message to each CITRA neighbor (step 2). Each boundary controller and host along the potential path of an attack uses the network audit trail to determine if the packets associated with the attack passed through it. If so, the device sends a traceback message to its neighbors (step 3). This continues until either the attack is traced back to the source of the attack or to the edge of the CITRA system. At each CITRA component along the attack path, responses are taken using the CITRA policy mechanisms described in Section 3. For example, at boundary controllers, the policy may specify that the service being attacked at a host should be denied for all requests originating from the attacker's address or network for a specified time duration. For hosts, the policy may specify that the offending process should be killed or the offending user account disabled. The current implementation uses the "narrowest" network response that stops the current attack. That is, the response will attempt to install filtering rules that prohibit the attack source from reaching the target while allowing other users of the service to continue access. This helps to minimize the negative impact the response may have on legitimate system users.

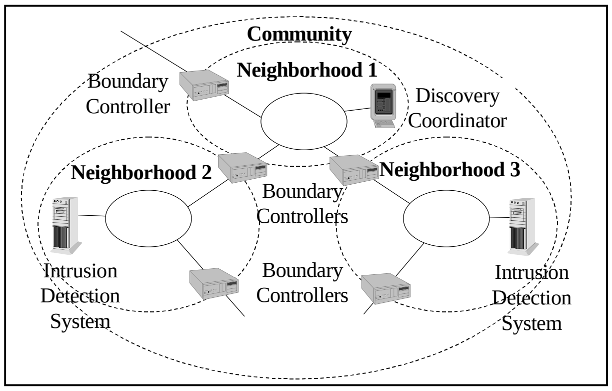

Figure 1. CITRA community

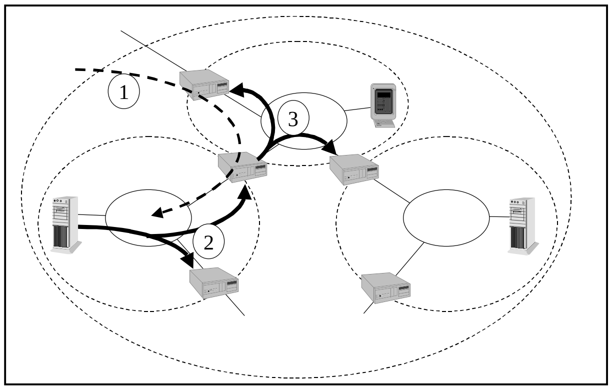

Figure 2. IDIP initial intrusion response

In addition to forwarding a trace message to adjacent CITRA components, the detector sends a report to the Discovery Coordinator. As each CITRA device responds to the attack, the device sends a report to the Discovery Coordinator describing the responses taken. Because the

[2] In [1] and earlier reports, IDIP is used to refer to both a protocol and an architecture. In this paper, for terminological clarity, this paper uses IDIP to refer to the protocol, but introduces CITRA to refer to the architecture and current implementation.

Discovery Coordinator receives the initial attack description and the response of each CITRA agent along the attack path, the Discovery Coordinator is able to gain a global picture of how the attack proceeded through the system. Using this global picture and the system topology information available to management components, the Discovery Coordinator can determine the optimal system response. CITRA supports execution of this optimal system response by enabling the Discovery Coordinator to direct changes to the CITRA agent responses. For example, the Discovery Coordinator can remove redundant responses along the attack path to minimize the impact of the automated response on system performance and functionality.

By collecting intrusion detection and response information at the Discovery Coordinator, CITRA supports community-wide aggregation and correlation of attacks, enabling the Discovery Coordinator to provide a more complete picture of the system security state.

3. Response Policies

An objective in the development of CITRA has been to take a response only if the response causes the system less harm than the attack. In some cases this might mean leaving the response to the administrators, and simply provide them with the necessary information. In other cases, the response is clear and the automated response system should take the appropriate action. For example, users who appear to be abusing their privilege should be logged off and their accounts suspended as their accounts have likely been compromised. Because mission objectives and administrator preferences affect the determination of the desired response to a given situation, CITRA has a number of controls available to administrators to ensure that only the desired responses are taken. This allows the administrator to trade the response benefits in terms of blocking the attack against the cost to the organization in terms of lost services caused by the response.

Adding the capability for automated response to a system enables it to respond quickly to attacks, potentially reducing the attack's impact on system users; however, these automated responses also introduce new risks to the system. If the adversary knows that automated response mechanisms are in place, then these responses can potentially be used to accomplish the attacker's objectives. The primary risk area is denial of service. If the attacker's objective is to deny remote system users access to some service, the attacker can launch exploits against the service and the automated response system may close down remote access to the service.

The CITRA policy mechanisms use two fields for the core policy mechanisms: certainty and severity. Certainty represents the likelihood that the reported event is an attack and severity represents the potential damage that may be caused by the attack. Although some detectors provide certainty data, most do not. The CITRA agent adds certainty data when missing. Severity data is dependent on site policy (e.g. the value of the attacked resource and the type of attack) and is always inserted into the attack report by the CITRA agent.

The CITRA response policy comprises multiple sections. Different policy sections cover different concerns, such as detection aggregation, limits on blocking rule duration, or control of host-based response. Most sections may have multiple policy statements, where each statement includes a set of parameters that must match the current situation (e.g., attack type or target), plus some fields that determine the appropriate response. The fields against which matches are performed can be wildcarded so that a single rule can match a number of situations. When a CITRA agent processes a policy section, the agent uses the first matching statement.

The following sections describe CITRA policy mechanisms and how they can help reduce the likelihood of an adversary using IDIP to cause denial of service attacks on the system.

3.1. Detection Control Policies

CITRA provides policy mechanisms to aid in controlling detection reports by removing false positives, reducing repetitive reports, and aggregation of reports. These CITRA policy mechanisms are used by CITRA agents that execute on the detector platform. The CITRA agents read detector alerts and augment the alerts with data required to support IDIP. In cases where the agent has information needed that is not contained in the detector's report (e.g., detector identity), the agent adds this data.

False positive reduction. Administrators can configure CITRA agents residing on detection platforms to reduce the certainty rating to reduce the certainty of specific detection reports known to be false positives in an environment. This can be used in conjunction with other policy mechanisms to cause the CITRA agent to not report these known false positives to the Discovery Coordinator. These event reports are still available locally at the detector. Because some detectors have high false positive rates in certain environments, this mechanism can greatly reduce the load on the system administrator.

Reducing repetitive reports. Repetitive reports can also cause administrator overload. In some instances, intrusion detection systems may generate a large number of reports that are all related to the same event. For example, a full port scan of a subnet can cause detectors to emit thousands of essentially identical port scan reports. These reports can be used to flood the IDIP messaging and thereby inhibit automated response. These reports also flood the administrator's console with unnecessary information. CITRA provides administrator controls over this situation by allowing the administrator to set time and event count thresholds for repeated detection reports. Figure 3 shows one such policy. On initial detection of the event, the CITRA agent uses the standard policy mechanisms to determine the correct response actions. After this initial response, repeated reports are collected until either the time or event count threshold is exceeded. At that point, an aggregate report is issued that specifies the number of repeated reports collected for that event type. In the example of the port scan for a subnet, these mechanisms can reduce the thousands of reports to a hand-full depending on the thresholds used and the scan duration. For the example policy in Figure 3, the CITRA agent sends the first port scan report issued by a detector, and then sends an updated report after 20 detector reports or 20 seconds, whichever comes first. This would reduce the number of reports in a full port scan by a factor of 20.

```
[THROTTLE]
    [THROTTLE/1]
        attack_code = PORT_SCAN
        thresholdnum = 20
        thresholdtime = 20
```

Figure 3. Sample CITRA throttle policy

CITRA throttling mechanisms cannot completely eliminate the threat of the adversary using IDIP to flood the network or the Discovery Coordinator. If the adversary attempts a large number of distinct exploits, then each is reported; however, such an adversary should be easy to trace, and once the source has been found and a suitable response taken close to the source, the flood ceases.

Report aggregation. CITRA aggregation mechanisms enable an administrator to aggregate a set of reports into a new report. For example, an administrator can specify that after some number of unserviced ports are accessed on a firewall, that this collection of reports should be considered a port scan. This mechanism configures the CITRA agent to either (1) send each report and the aggregate or (2) send just the aggregate reports. This type of simple correlation at the CITRA agent reduces the load on the administrator, and also reduces the impact that the intrusion detection and response mechanisms have on the network.

3.2. Core Response Policies

CITRA uses a simple cost model to determine basic responses. This cost model uses attack severity and detection certainty, along with administrator-specified thresholds to determine which response should be taken: (1) take no response; (2) report to Discovery Coordinator; (3) trace the attack; (4) increase auditing; and (5) block the attack. Using these mechanisms, one can specify that very low certainty alerts should have no response, while low severity alerts with moderate certainty are traced and reported, and only high certainty and high severity alerts cause blocking rules to be installed.

The detector can specify attack certainty, but as discussed in Section 3.1, administrators can change this detector-specified value for selected attacks. The administrator also defines severity in terms of the value of the resource under attack, the attack type, and time of day. Thus, different actions may be taken for different attack targets.

Another set of controls allow the administrator to specify limits on duration of auditing or blocking rules based on the attack target, service under attack, and time of day. This would allow response policies such as block email from an attacker's address for no more than 10 minutes between 8AM and 5PM, but block for up to an hour otherwise. This policy is shown in Figure 4. In Figure 4, the second rule matches any time not specified previously, so that the two rules combine to provide the desired policy.

```
[BLOCK]
    [BLOCK/1]
        protocol = TCP
        source = *
        destination = 10.1.2.3:25
        time = 0800-1700
        action_max = 10min
        action_min = 2min
    [BLOCK/2]
        protocol = TCP
        source = *
        destination = 10.1.2.3:25
        time = *
        action_max = 60min
        action_min = 2min
```

Figure 4. Sample CITRA blocking policy

These mechanisms still do not prevent the adversary from using CITRA to deny access, but they do provide enough control that an administrator can limit the responses to only those the administrator would perform manually. The advantage to the administrator is that the response is immediate.

These immediate response mechanisms can be used to stop the effects of an attack. The response agents can also tune the response based on the type of attack. The CITRA response agent on intermediate boundary controllers understands that an alternative method of blocking a flood is to install a rate limiting rule, rather than a rule blocking all matching traffic. This stops the flood, while allowing some matching packets to continue to flow through the boundary controller. Where legitimate users require traffic that is indistinguishable from the flood, this mechanism allows the application to continue operation. This was demonstrated in a recent experiment with distributed denial of service attacks against a streaming audio/video application that used the User Datagram Protocol (UDP). By flooding the audio/video server with UDP datagrams, the attack was able to make the service unusable. The automated response installed rate limiting rules at boundary controllers along the attack path, allowing the audio/video service to be restored. A blocking rule at the intermediate boundary controllers would have stopped the attack, but it would also have stopped the UDP datagrams used for control from the clients to the server. Installing a rate limiting rule is more desirable than installing a blocking rule, since both clients and attackers were sending UDP to the server. This allows some legitimate traffic to make it to the server.

The structure of the CITRA policy specifications and the policy enforcement mechanisms allows easy addition of new policy elements as the need arises. In addition to the administrator controls for network-based responses, we found it beneficial to add controls for both local end-system responses and use of vulnerability scan results. These policies are discussed in Section 4.2.

3.3. Boundary Controller Management Policy

The Department of Defense uses a notion called Information Operations Condition (INFOCON) to represent the current cyber situation and the risk of attack on information assets. As INFOCON changes, security policies change. The CITRA implementation has mechanisms to broadcast INFOCON changes to all nodes within a CITRA community. The CITRA agent at each node changes response policies on INFOCON change. In addition, these agents can be programmed to change other local security policies. For example, on INFOCON change, boundary controllers can change the set of installed filtering rules. This can be used to restrict the services accessible to remote users as INFOCON increases, eliminating access to non-critical services. This same mechanism can be used to provide centralized control of the filtering policies used at each boundary. When the CITRA agent receives a new policy from the Discovery Coordinator, the agent removes filtering rules previously loaded by the agent, and loads those specified in the policy file for the current INFOCON.

3.4. Vulnerability Policy

A risk that is introduced by automated intrusion response is that the adversary can attempt to exploit the responses to cause self-inflicted denial of service. The adversary can attempt exploits that need not succeed in themselves if the objective is for the intrusion response system to deny access to the attacked service. For example, if the adversary attempts one of the buffer overflow attacks against a web server, the response system might block the Hypertext Transfer Protocol (HTTP) port to the targeted host.

One can reduce this risk by understanding the exploits against which the system is not vulnerable. If the system is not vulnerable to that specific buffer overflow attack, then there is no need to install filtering rules at the boundary controllers to block the attack. To support this notion, CITRA policy mechanisms enable an administrator to specify a list of exploits to which various system components are not vulnerable. These mechanisms also allow the administrator to specify the desired response to such exploits. This response policy is used instead of the standard response policy described in Section 3.2 for exploits specified in the vulnerability policy. Figure 5 shows one such policy.

With the policy specified in Figure 5, for an HTTP_PHP_OVERFLOW exploit used against the host at Internet Protocol (IP) address 10.33.4.4, the CITRA system attempts to trace the attack back to its source and reports the attack to the Discovery Coordinator; however, no blocking actions are taken.

```
[NONVULNERABLE]
    [NONVULNERABLE/1]
        attack_code = HTTP_PHP_OVERFLOW
        protocol = TCP
        ip = 10.33.4.4

[NONVULNERABLE_POLICY]
    [NONVULNERABLE_POLICY/1]
        action = REPORT TRACE
```

Figure 5. Sample CITRA vulnerability policy

3.5. Vulnerability Scanning Policy

Another potential problem for automated response is that detectors generally report vulnerability scans, as they are typically precursors to attack; however, organizations often perform "friendly" scanning to help identify and close security vulnerabilities. It is desirable that the system be able to respond differently to friendly scans than to unfriendly scans. For example, an organization may still want to report friendly scans, while both reporting and tracing unfriendly scans. Figure 6 shows the CITRA friendly scan policy for this case. For unfriendly scans, the CITRA agent uses the core response policy mechanisms described in Section 3.2.

```
    [SCAN_POLICY]
        [SCAN_POLICY/1]
            action = REPORT
```

Figure 6. Sample CITRA scan policy

4. Integration with Security Tools

Although initially designed as an infrastructure for integrating intrusion detection systems and boundary controllers, CITRA has evolved into a general framework for integrating a wide variety of components to support effective intrusion response.

The following sections provide an overview of how CITRA integrates selected technologies.

4.1. Sensors

CITRA enables the use of heterogeneous sensors within an enterprise. Any component that can provide information related to system intrusions or anomalous activity can be integrated into a CITRA community using the standard CITRA agent software. The detection agent is used to integrate detectors, providing the CITRA policy mechanisms and implementing IDIP. The current implementation has been effective in minimizing the cost of detector integration. This implementation uses either a file or socket interface to send strings describing attacks between the detector and the CITRA agent. This generally requires a simple program to convert the vendor-specific detector output to a standard format for input to CITRA.

Intrusion Detection Systems. Intrusion detection systems provide the initial attack reports that initiate CITRA automated responses. CITRA supports both host and network-based detectors. When the intrusion detection system reports the details of the network flows that contributed to the intrusion attempt, the CITRA agent can initiate a traceback, as described in Section 2. The event is also reported to the Discovery Coordinator. When the network flow information is not available, then the intrusion is reported to the Discovery Coordinator without initiating traceback.

Using the generic CITRA agent software, the cost of developing the software to integrate a new detector is approximately two to five days. The largest part of this effort is the mapping from the detector's attack names to the standard Common Intrusion Detection Framework (CIDF) attack names [2] used by IDIP.

One integration issue is that some vendor-specific reports do not map to a CIDF attack name. For those cases, one can either add new attack names to the CIDF list, or the integration software can insert a vendor-specific attack name within the report. Although the CITRA response engines do not recognize the attack name, they can still trace the attack and take default responses based on the attack information provided (e.g., source addresses to block).

System Integrity Validation Systems. System integrity validation mechanisms (e.g., Tripwire [3]) can be used as an indicator that an attack has occurred. Within the CITRA context, reporting changes to critical system files aids in situation understanding. System validation mechanisms require minimal integration effort as they map to a single CIDF attack code. (In one instance, approximately one hour of effort was needed for integration.) While these mechanisms do not provide the information needed to trace the attack, a system administrator may want to isolate a host that has been penetrated. CITRA automated response mechanisms can be used to perform this function. If no automated response is needed, the CITRA agent can be configured to simply report these events.

Virus Detectors. Virus detectors provide another sensor for CITRA, and like system integrity validation mechanisms, use only a single CIDF attack code. Because macro viruses can be used to install Trojan horses, which can be used to further attack the system, providing centralized reporting of virus detection helps in situation assessment.

4.2. Response Mechanisms

CITRA supports both host and network-based response mechanisms. Typical network responses include traceback and installing filtering rules at boundary controllers. Host-based responses include disabling users or services. The CITRA policy mechanisms described in Section 3 control which responses are taken for specific intrusion attempts.

Boundary Controllers. Boundary controllers provide the perimeter defense for most organizations. Because network-based attacks must travel through these devices, they provide the primary mechanism for automated traceback. In addition to tracing attacks, these boundary controllers can block selected network traffic, providing a primary response mechanism for network-based attacks. The CITRA prototype software uses these boundary controllers to locate the source of network-based attacks and block or limit the attack close to the source.

Integrating a new boundary controller with the CITRA software involves either utilizing native network auditing, or porting the CITRA network auditing software to the new platform. In addition, the boundary controller filtering mechanisms must be integrated with the CITRA software. This involves implementing a small set of functions that CITRA uses as the generic boundary controller API.

The CITRA auditing mechanisms are designed to support tracking an intruder (once identified) back to the attack source through intermediate boundary controllers and hosts regardless of the nature of the attack. This type of traceback requires traceback of-

- Attacks that use spoofed source addresses.
- Attacks that use single datagrams.
- Attacks with multiple sources (e.g., distributed denial of service).
- Attacks that cross network address translation (NAT) devices.
- Attacks through hosts and firewalls, where a new connection is created with different source and target addresses or ports.
- Attacks that flood the network.

Table 1 briefly describes how CITRA auditing achieves these goals.

Boundary controllers can also serve as detectors. Attempts to use unserviced ports can be an indication of port scanning. Also, firewall proxies may be able to detect specific anomalous events. For example, a File Transfer Protocol (FTP) proxy may be able to detect an attempt to retrieve a sensitive file such as the password file. If the boundary controller logs these events, a simple integration program can be used to send these events to the local CITRA agent. The agent can then trace the network flow, report the event, or store the event for aggregation.

Host Operating Systems. When an attack is detected by a host-based intrusion detection system, the system has an opportunity for immediate attack containment if an appropriate reaction can be taken at the victim host. For example, when an unauthorized attempt to transition to root occurs, if the process can be killed before the privilege is abused, the system can avert further damage. The CITRA implementation addresses this with mechanisms to perform local responses on local detection reports. When a local host-based intrusion detection system reports an attack, the local CITRA agent may use local operating system mechanisms to kill the offending process, kill the process's session, disable the user account for the process's user ID, reboot the host, or halt the host.

Table 1. Meeting trace requirements

Requirement	Approach
Spoofed source addresses	Traceback based on audit trail
Single datagrams attacks	Recording all flows
Multi-source attacks	Recording inbound and outbound interfaces for all flows
NAT devices	Recording translation information and appending this to trace request
Hosts and firewalls traversal	Recording translation information and appending this to trace request
Attacks that flood the network.	Recording packet counts to determine if enough datagrams passed through the device to contribute to a flood Use of reliable transfer over UDP for IDIP to ensure delivery of traceback requests

A significant risk in this type of response is that the response can have negative affects on the system if the attack is falsely reported. Another issue is that the response must match the reported event. For example, one would not want to kill a process that is detected behaving in an anomalous fashion if the behavior could be legitimate. This requires mechanisms that decide, based on attack type, what an appropriate response would be. In the case of a reported unauthorized transition to root, an appropriate response would be to kill the offending process's session and disable the user account that owns that process as that account was apparently compromised; however, other reported actions that have less severe consequences might only require reporting so that the system administrator can investigate. Also responses may differ depending on system usage. For example, in a closed system, anomalous actions may not be tolerated, while in a general-purpose system, they may be tolerated.

To enable the CITRA system to behave appropriately for the different environments, CITRA policy mechanisms allow administrators to specify the response for specific situations.

In an experiment using a red team, the policy was set to only kill the offending process. This delayed the red team's activities as they repeatedly made small amounts of progress before the process was killed. If the host response had been to disable the user account, the red team would have been forced to use a more costly attack. The Discovery Coordinator was notified of the root escalation, and could also have initiated another response.

Operating System Wrappers. Operating system wrappers [4] provide additional host-based responses. Because these wrappers intercept system calls, they can be used to enforce very flexible policies. Wrappers also provide an alternative method of accomplishing responses that can be done through standard operating system mechanisms. For example, a wrapper developed for one experiment effectively disabled a user account by returning an error on each operating system call made by any process owned by the specified user. Other more complex responses are feasible, including generating a false environment for a malicious process that allows monitoring of the process without compromising the system.

4.3. Vulnerability Assessment Tools

The CITRA vulnerability policy described in Section 3.4 provides a mechanism by which administrators can control the CITRA response to exploits for which the system is not vulnerable and the scan policy described in Section 3.5 provides a mechanism to limit responses to friendly scans. To make optimal use of these policy mechanisms requires both that the CITRA system can (1) learn the set of exploits to which the system is not vulnerable and (2) differentiate between friendly and unfriendly vulnerability scans. Integration with vulnerability assessment tools provides this knowledge.

To investigate the benefits of providing this capability, NAI's CyberCop Scanner was integrated with CITRA. CyberCop Scanner is a network-based vulnerability assessment tool. Figure 7 shows the event flow for performing scans using the CITRA mechanisms.

The Discovery Coordinator schedules scans by sending a scan message (step 1, above) to the CITRA scan agent. Scan messages include the list of vulnerabilities to be scanned, the scan targets, and how often the scan is to be repeated for periodic scans. Note that the scan agent randomizes scan start time to help prevent an adversary from knowing when friendly scans are occurring, which would allow the adversary to hide in the noise of the friendly scan.

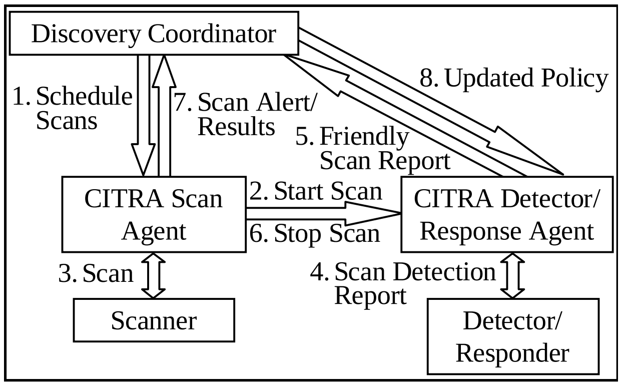

Figure 7. CITRA-controlled scans

When the scheduled time to perform a scan is reached, the scan agent issues a "start scan" message (step 2) to CITRA nodes in the neighborhood of the scan target. These nodes use this "start scan" message to determine if reported scans are friendly.

The scan agent sends the scan command to the scanner (step 3), which performs the scan and writes results to a file.

During the scan, it is likely that detectors along the path of the scan will report the scans (step 4). The detection agent uses information from the start scan message to determine if the detected scan is a friendly scan. Depending on the scan reporting policy, this agent may report the friendly scan to the Discovery Coordinator (step 5) or may attempt to trace the scan to its origin. If an unfriendly scan is detected during the friendly scan, then the standard CITRA policy mechanisms are applied to determine the appropriate response.

When the scan completes, the scan agent sends a "stop scan" message to neighboring CITRA nodes (step 6) so that they will no longer be looking for friendly scans.

The scan agent then reads the scan results and sends them to the Discovery Coordinator (step 7). If the scan results indicate that new vulnerabilities have been introduced since a previous scan (an indication of potential compromise), the agent also sends a scan alert to the Discovery Coordinator. The Discovery Coordinator generates an updated CITRA response policy that specifies exploits to which each target system is not vulnerable and sends the updated policy to each CITRA agent (step 8). The agents use this information in determining appropriate responses. CITRA policy statements allow an administrator to specify what response action to take when an exploit to which the system is not vulnerable is detected.

With this integration, the CITRA system can now learn the set of events to which the protected system is not vulnerable, and from that information automatically generate the vulnerability policy statements described in Section 3.4. Because a large number of detectable exploits are patched in many sites, this greatly reduces the number of blocking rules that the CITRA implementation would use in an enterprise, greatly reducing the negative impact that automated responses could have. When integrated with vulnerability assessment tools, CITRA can apply harsher responses only where needed: when the system under attack is vulnerable to the exploits being used.

Besides improving system response capabilities, this integration provided another detection sensor: when an attack modifies the system to introduce a vulnerability for further use by the attack, the vulnerability scanner may detect a change in the vulnerability results. Reporting this to the Discovery Coordinator enables system administrators to respond. An automated response that isolates the vulnerable system is also possible.

4.4. Network Management Systems

Integration with network management systems within CITRA occurs at the Discovery Coordinator. It is expected that the Discovery Coordinator is co-located with the management tools for the enterprise. To use network management mechanisms required developing an interface between the Discovery Coordinator components and the network management engine. For the experiments performed, the Scotty network management system [5] was used. This is a public domain network management system, which made integration relatively easy.

Network management systems provide two potential benefits to automated response within CITRA: control and information gathering.

Where Simple Network Management Protocol (SNMP)-based control of system components is enabled, CITRA can use SNMP to affect some responses. A capability developed for one experiment uses SNMP to shutdown the Ethernet interface of a compromised host at the switch to which the host is attached. This makes sense when either the host-based mechanisms are not present or they have been disabled by the attack. By commanding the switch to disable the host's Ethernet interface, CITRA can limit the damage caused by the attack to the compromised host.

Use of network management tools for topology discovery also supports the automated response mechanisms at the Discovery Coordinator. The Discovery Coordinator requires topology information to determine the optimal location for response. Acquiring this data through automated mechanisms built into network management systems reduces the effort required to configure the Discovery Coordinator's automate response engine.

4.5. Correlation Tools

Correlation tools can be used to reduce false positives, eliminate duplicate reports, and detect that reports are related. Related events may indicate a larger threat than might otherwise be detected, such as a distributed coordinated attack. Within CITRA, correlation tools are placed at the Discovery Coordinator. The IDIP protocol ensures that all intrusion reports are received at the Discovery Coordinator, which enables correlators co-located with the Discovery Coordinator to receive this data.

In experiments performed using CITRA, a number of correlators have been integrated at the Discovery Coordinator to perform various correlation tasks:

- merger, which merges duplicate reports of the same event from different sensors into a single report.
- Graph-based Intrusion Detection System (GrIDS [6]), which combines reports based on graph algorithms to locate coordinated distributed attacks.
- A Perl-based component developed by Silicon Defense that filters out false positives by looking for corroboration of attack reports for events known to represent false alarms.
- The Stanford Complex Event Processor [7], which was programmed to filter out false positives.
- A version of Event Monitoring Enabling Responses to Anomalous Live Disturbances (EMERALD) [8] designed to perform duplicate report recognition.

The integration with these tools was accomplished with minimal effort using the socket interface supported by the current CITRA implementation.

5. Ongoing Work

In addition to the work described above, CITRA is being used and extended in ongoing projects concerned with automated intrusion traceback and response.

5.1. Multicommunity Cyber Defense

While providing these capabilities within a single administrative domain is a major step forward, operation in the context of the Internet, extranets, and very large intranets requires operation across administrative domains. For example, tracing attacks launched from other Internet sites generally requires cooperative interactions among the Internet service providers.

A CITRA community corresponds to an administrative domain. As described in Section 2, each community includes a Discovery Coordinator, which acts as the locus of administrative control for that community. The current CITRA implementation only supports a single (potentially very large) administrative domain; however, to support traceback and response across the Internet, mechanisms are required to allow mutually suspicious domains or multiple domains with different detection and response policies to cooperate. The Multicommunity Cyber Defense (MCCD) project is extending CITRA so that it can be used in such multi-domain contexts and can support cross-domain correlation of intrusion information. The MCCD project is extending CITRA and its support libraries to provide the following capabilities.

- Cross-community trust – In the commercial arena, organizations are frequently mutually suspicious as they may be competitors in some areas or they may be partners in other areas, but even as partners they must protect their proprietary interests. In either case, there are problems in both releasing data to and accepting response directives from remote domains. There is a need to protect domain-sensitive data that could increase the risk to the local organization if shared with other organizations. Supporting multiple communities requires policies to control information sharing and cooperation between communities. For each community, these policies specify the extent to which other communities can be trusted, that is, the extent to which cooperation should be extended to and expected from other specified communities. A policy might specify that traceback and traffic blocking requests from unknown communities are ignored while requests from communities with which business relationships exist are honored fully or partially. A partial trust policy might specify that traffic blocking requests should be honored only if the requester and recipient concur that the intrusion poses a severe threat to their organizations. Alternatively, a partial trust policy might specify that organization-sensitive information, such as private IP addresses, must be removed from a traceback request before it can be forwarded to certain communities.
- Community-aware boundary controllers and Discovery Coordinators – Supporting cross-community trust policies requires that new functionality be incorporated in boundary controllers at the edges of a community and in Discovery Coordinators. This functionality includes the ability to (1) discard or sanitize outbound traceback messages before they are sent to other communities; (2) transform inbound messages before they are processed (e.g., to normalize data fields); and (3) escalate inbound or outbound message processing to an administrator for approval or dynamic policy change.
- Cryptographic Support – CITRA uses cryptography to provide secure communications including authentication, integrity, and privacy protection. Supporting secure communications across communities has required restructuring the existing CITRA cryptographic and key management facilities so that Discovery Coordinators can use jointly trusted external certificate authorities and establish new trust relationships "on the fly" during a security incident.
- Correlation – Multicommunity operation provides a new opportunity to pool and correlate intrusion data on a larger scale than is currently possible. Pooling all low-level intrusion data for correlation is impractical because of sheer volume: the data would clog network pathways between communities and overwhelm correlation systems. Consequently, a key challenge is determining which data should be pooled. The MCCD project is developing new statistical techniques that pool and correlate only the data that are most anomalous. Preliminary results suggest that these techniques are effective in detecting stealthy port scans [9]. Beyond developing correlation techniques that scale to the multicommunity environment, the MCCD project is developing the distribution mechanisms needed to share the necessary data between communities. Because CITRA aggregates the data within a community at the Discovery Coordinator, the multicommunity environment requires communication of correlation results between Discovery Coordinators in different domains. This also requires policy mechanisms to enable system administrators to constrain what information is shared.
- CITRA Survivability – Within CITRA, the Discovery Coordinator is a potential single point of failure. Although other CITRA components continue to operate without a Discovery Coordinator, loss of the Discovery Coordinator causes the overall system to be less capable. In the multicommunity case, even more is lost, as the Discovery Coordinator has a greater role when responding across community boundaries, so increasing Discovery Coordinator survivability becomes more important. Increasing Discovery Coordinator survivability involves replication of Discovery Coordinator functionality and data, plus fail-over mechanisms that cause functionality to shift on loss of a Discovery Coordinator component. The current CITRA implementation supports distributing Discovery Coordinator functionality across multiple platforms, with each platform receiving a copy of the incoming intrusion detection and response reports. Thus, the primary CITRA extensions for survivability are monitoring liveness of Discovery Coordinator

components, and moving Discovery Coordinator functionality on component failure.

5.2. Active Networks Intrusion Detection and Response

The Active Networks Intrusion Detection and Response Project (AN-IDR) objective is to develop automated intrusion detection, traceback, and response mechanisms that are more powerful, adaptable, and effective by exploiting technology produced by DARPA's Active Networks Program. Active Network technology is a dynamic network infrastructure that allows administrators and users to reprogram and customize routers, switches, and other components to provide new network services. The AN-IDR project is using ideas and code from CITRA and the MCCD project to develop active-network-based intrusion detection, tracing, response, and recovery mechanisms that are self-deploying and self-adaptive, i.e., autonomous, migrating and self-configuring.

The initial AN-IDR prototype configuration demonstrates the use of active network technology to defend a streaming video server and its distributed clients against a distributed denial of service attack launched from the "Stacheldraht" hacker toolkit [10]. After the denial of service attack is detected and reported by a CITRA-enabled intrusion detector, the Discovery Coordinator responds by sending an active rate limiter program to the router nearest the video server. The rate limiter begins execution, then clones itself and migrates successively to all routers that are upstream along all attack paths between the server and the Stacheldraht flooding agents. At each such router, it restricts the volume of suspicious packets forwarded toward the video server by discarding packets that exceed a specified arrival rate. The effectiveness of the rate limiter increases successively as it migrates closer and closer to the flooding sources. As the rate limiter migrates, it allows streaming video sessions with additional clients to recover and continue. Other active components under investigation by the AN-IDR project include lightweight mobile vulnerability scanners, intrusion detectors, and repair agents. Ideas from the AN-IDR project may ultimately be fed back into CITRA.

5.3. Complex Event Processing for Cyber Command and Control

The Complex Event Processing for Cyber Command and Control project objective is to develop technology to help assess the current cyber situation. The strategy is to use the ePatterns Complex Event Processing system [11], which is based on the concepts developed at Stanford [7]. The objective of this project is to develop causal event models and define event patterns that can be used to correlate system events to help assess whether the system is under attack and determine the nature of the attack. These events include intrusion detection reports formatted in the evolving Internet Engineering Task Force (IETF) intrusion reporting format [12].

6. Related Work

Techniques for automated traceback to the real source of an attack have increased since the emergence of the distributed denial of service attacks in early 2000. These solutions, however, are limited only to flooding-type attacks and have no capability for a possible automated response. Also, the attack path traceback stops when a NAT device is reached because these traceback mechanisms do not perform the corresponding address translation required to trace through NAT devices.

Bellovin's Internet Control Message Protocol (ICMP) Traceback, or itrace, is currently active in the IETF [13]. With this method, a router generates and emits a new traceback message to the same destination as one of a low probability (1/20,000) sampled message it is forwarding. The traceback message includes information on previous and next hops. When a flooding attack occurs, the victim can reconstruct the attack path using these traceback messages; however, this mechanism only works when the flood packets are among the sampled packets. With most floods, the probability is sufficiently high that a flood packet is among the sampled packets. Authenticating the traceback messages is an area of current research. Without authentication, the adversary can trick the system into incorrect traceback results.

IP Packet Marking is another method for traceback during a flooding attack. Savage et al. [14], and Song and Perrig [15] use an approach where routers occasionally mark packets with partial path information. The victim reconstructs the attack paths after receiving a modest number of these packets. This approach also relies on sampling packets. That is, only a relatively small number of randomly selected packets are marked.

The DECIDUOUS project [16] uses IP Security (IPsec) mechanisms to determine the attack path by creating dynamic IPsec security associations to identify ongoing flooding attack sources. By strategically establishing IPsec Authentication Header (AH) tunnels between routers and the victim, the attack path can be deduced by looking at whether the attack packet has been authenticated or not.

The Source Path Isolation Engine (SPIE) [17] effort has proposed an approach where each router retains a hash of each packet that traverses the router. These hashes are discarded periodically. In that approach, when an attack is detected, a detector provides SPIE with the offending packets. SPIE then saves the records of the recently collected hashes. By comparing the hashes of the collected packets with those collected by the routers, SPIE can determine whether the packet likely came through the router. Both SPIE and CITRA can trace individual packets. SPIE requires much less storage than CITRA's audit trail, however, CITRA's audit trail can be used to trace back events that occurred several minutes or even hours earlier, while SPIE is limited to more recent events. Another difference is that within CITRA, events that involve a large number of packets (e.g., distributed denial of service attacks) require a single traceback message which describes the class of packets used in the attack. The CITRA audit trail can be used to determine if that that class of packets passed through the CITRA node. SPIE must trace back individual packets, requiring tracing a large number of packets for distributed denial of service attacks.

Flooding attack traceback using logging of packet flow through routers is suggested by Glenn Sager [18]. Cflowd allows a user to collect, store, and analyze router traffic flow information. Flow information includes source-IP-address, source-port, destination-IP-address, destination-port, IP-protocol, type of service (TOS), and input interface, which is similar to what is stored in CITRA's audit trail. CITRA's audit trail is also used on firewalls and NAT devices, where address translations occur, so the traceback can continue across NAT-device boundaries.

Limited commercial automated intrusion response solutions have been developed. Several commercial off-the-shelf (COTS) vendors supply automated response with their products (e.g., [19], [20], [21]), however these products use proprietary protocols and are limited by an architecture that requires all response decisions to be made at a central controller. The firewalls or routers are simply response mechanisms and not full participants in the response decision making process. Also, coordinated automated traceback and responses across different network administrative domains is not currently achieved.

Research efforts in the area of appropriate strategies for automated intrusion response are beginning to emerge. Ohta, et al. [22] discuss distributed Internet applications for detection, traceback, and cooperative defense, using correlated traffic pattern information from multiple security agents. Sekar, Cai, and Segal [23] are investigating methods of using operating system wrapper technology to perform intrusion detection and automatic isolation response within a host. Electronic quarantine by Brutch, Brutch, and Pooch [24] automatically responds to an intruder by confining the compromised host by modifying network access controls. Mountainwave's Adaptive Network Security Management System [25] is planned to have a software framework for detecting intrusions and automatically responding with countermeasures. That framework uses a more centralized architecture than CITRA and does not attempt cooperation between boundary controllers (i.e., firewalls and filtering routers) in locating and isolating intruders.

7. Summary

Network-based attacks will continue to become more sophisticated. Recent distributed denial of service attacks provide a glimpse of these future attacks, with increased coordination between attack agents and improved ability of these agents to change behavior in response to system responses. Automation of responses is critical to keeping pace with the speed of these attacks. This requires an automated response infrastructure and tools that enable system administrators to easily integrate new capabilities or change the behavior of existing responses.

CITRA provides a first step towards such an infrastructure. It has proven to be an environment amenable to easy integration of new tools and to modifying the behavior of existing tools to provide suitable responses to various attacks.

8. Acknowledgements

The approach for integration of vulnerability assessment tools with CITRA was made possible by major design and implementation contributions from William Babson and Jaime Spacco from NAI Labs and Laurence Rockwell from Boeing Phantom Works.

9. References

[1] D. Schnackenberg, K. Djahandari, and D. Sterne, "Infrastructure for Intrusion Detection and Response", Proceedings of the DARPA Information Survivability Conference and Exposition, Hilton Head, SC, January 2000.

[2] Rich Feiertag, Cliff Kahn, Phil Porras, Dan Schnackenberg, Stuart Staniford-Chen, Brian Tung, "A Common Intrusion Specification Language", http://www.gidos.org/, June 1999

[3] Tripwire.org, http://www.tripwire.org/

[4] T. Fraser and L. Badger, "Hardening COTS Software with Generic Software Wrappers", Proceedings of the DARPA Information Survivability Conference and Exposition, Hilton Head, SC, January 2000.

[5] Scotty - Tcl Extensions for Network Management Applications, http://wwwhome.cs.utwente.nl/~schoenw/scotty/

[6] S. Staniford-Chen, S. Cheung, R. Crawford, M. Dilger, J. Frank, J. Hoagland, K. Levitt, C. Wee, R. Yip, D. Zerkle, “GrIDS -- A Graph-Based Intrusion Detection System for Large Networks”, Proceedings of the 19th National Information Systems Security Conference, October 1996.

[7] Stanford University, Program Analysis and Verification Group, http://pavg.stanford.edu/cep/

[8] Ulf Lindqvist and Phillip A. Porras, “Detecting Computer and Network Misuse Through the Production-Based Expert System Toolset (P-BEST)”, Proceedings of the 1999 IEEE Symposium on Security and Privacy, IEEE, Oakland CA, May 1999.

[9] S. Staniford, J. A. Hoagland, and J. M. McAlerney, “Practical Automated Detection of Stealthy Portscans”, ACM CCS IDS Workshop, Athens, Greece, Nov. 2000.

[10] CERT® Advisory CA-2000-01 Denial-of-Service Developments
http://www.cert.org/advisories/CA-2000-01.html

[11] ePatterns Inc., http://www.eventpatterns.com/technology/index.htm.

[12] IETF Intrusion Detection Working Group, “Intrusion Detection Message Exchange Format (IDMEF) Data Model and Extensible Markup Language (XML) Document Type Definition “, http://www.silicondefense.com/idwg/

[13] Steven M. Bellovin, Editor, “ICMP Traceback Messages”, Internet Draft: draft-bellovin-itrace-00.txt, Mar. 2000.

[14] Stefan Savage, David Wetherall, Anna Karlin, and Tom Anderson, “Practical Network Support for IP Traceback”, Proceedings of the 2000 ACM SIGCOMM Conference, August 2000.

[15] Dawn X. Song and Adrian Perrig, “Advanced and Authenticated Marking Schemes for IP Traceback”, Report No. UCB/CSD-00-1107, Computer Science Division (EECS) University of California, Berkeley, California, June 2000.

[16] H.Y. Chang, P. Chen, A. Hayatnagarkar, R. Narayan, P. Sheth, N. Vo, C. L. Wu, S.F. Wu, L. Zhang, X. Zhang, F. Gong, F. Jou, C. Sargor, X. Wu, “Design and Implementation of A Real-Time Decentralized Source Identification System for Untrusted IP Packets”, Proceedings of the DARPA Information Survivability Conference & Exposition, January 2000.

[17] Alex C. Snoeren , Craig Partridge, Luis A. Sanchez, W. Timothy Strayer, Christine E. Jones, Fabrice Tchakountio, and Stephen T. Kent, “Hash-Based IP Traceback”, BBN Technical Memo 1284, February 7, 2001.

[18] Glenn Sager, “Security Fun with OCxmon and cflowd”, Presentation at the Internet-2 Measurement Working Group, November 1998.

[19] PGP Security, Active Security, http://www.pgp.com/products/activesecurity/.

[20] Internet Security Systems, SAFEsuite Decisions, http://documents.iss.net/literature/SAFEsuiteDecisions.ssd_ps.pdf.

[21] Symantec Enterprise Security, Intruder Alert, http://enterprisesecurity.symantec.com/products/products.cfm?ProductID=48.

[22] Kohei Ohta, Glenn Mansfield, Yohsuke Takei, Nei Kato, Yoshiaki Nemoto, “Detection, Defense, and Tracking of Internet-Wide Illegal Access in a Distributed Manner”, Proceedings of the 10th Annual Internet Society Conference (INET 2000), July 2000.

[23] R. Sekar, Yong Cai, and Mark Segal, “A Specification-Based Approach for Building Survivable Systems”, Proceedings of the 21st National Information Systems Security Conference, October 1998.

[24] Paul Brutch, Tasneem Brutch, and Udo Pooch, “Electronic Quarantine: An Automated Intruder Response Tool”, Proceedings of the 1998 IEEE Information Survivability Workshop (ISW’98), October 1998.

[25] Mountain Wave, Inc., “Adaptive Network Security Management”, http://www.mountainwave.com/darpa-report/.