INTRUSION DETECTION WITH MOBILE AGENTS

Wayne A. Jansen
National Institute of Standards and Technology
100 Bureau Dr., STOP 8930
Gaithersburg, MD 20899
(+1) (301) 975 5148
WJansen@NIST.Gov

Abstract

Implementing an effective intrusion detection capability is an elusive goal, not solved easily or with a single mechanism. However, we argue that mobile agent technology goes a long way toward realizing the ideal behavior desired in an Intrusion Detection System (IDS). This paper discusses various ways in which mobile agents could be applied to the problem of detecting and responding to intrusions. The paper looks not only at the benefits derived from mobility, but also at those associated with software agents in general. After exploring these benefits, we outline a number of ways to apply mobile agent technology in addressing the shortcomings of current IDS designs and implementations, and delineate the associated security issues involved. We also look at several new approaches for automated responses to an intrusion, once detected.

Keywords: Intrusion Detection, Mobile Agents, and Computer Security

Background

Intrusion detection systems (IDSs) were conceived of as a form of expert system that observes patterns of activity in user accounts and notifies a system administrator if anything unusual is detected. The concept, first proposed by James Anderson in 1980 [1], did not blossom until 1987 when Dorothy Denning published her seminal intrusion detection model [9]. Early IDS implementations employed a monolithic architecture under which data collected at a single host was analyzed at a central point, at or adjacent to the point of collection [21, 27, 28]. Because monitoring account activity on a single host does not reveal attacks involving multiple hosts, IDS designers subsequently developed network-based IDSs that use a model of the network traffic to infer anomalies or misuses from low-level network packets traveling among hosts [13]. Network-based IDSs can be characterized as a change in perspective from host-centric to network-centric detection. A network-centric approach resolves a number of performance and integrity problems as well as problems associated with the reliance on audit trails [25].

IDSs can be further characterized by the technique used to discover an intrusion. An intrusion can be detected based on deviations from a user's or a system's historical pattern of behavior. The behavior can range from characteristics of entered keystrokes to command profiles, to time of day usage. Behavior occurring outside some acceptable threshold triggers a notification. An intrusion can also be detected based on an exact correspondence to a known pattern of intrusive behavior. This is a more direct means of discrimination that typically involves a rule-based approach, whereby the rules codify patterns of intrusion known as signatures. An event or event sequence that matches a signature triggers a notification.

The first generation of IDSs followed a two-component architecture. The collection process gleans information either from audit logs and internal interfaces at the host, or from monitoring packets on attached networks. That information then feeds into a centralized analysis process, which employs one or more different detection techniques. While this architecture is effective for small collections of monitored hosts, centralized analysis limits the ability to scale up to handle larger collections. Subsequent generations

of IDSs address scalability mainly by introducing intermediate components that preprocess and consolidate information obtained by the collection process for input into the analysis process [6].

Nearly all present-day commercial IDSs follow a hierarchical architecture, such as that illustrated in Figure 1. Information gathering occurs at leaf nodes, network-based or host-based collection points. Event information is passed to internal nodes that aggregate information from multiple leaf nodes. Further aggregation, abstraction, and data reduction can occur at higher internal nodes until the root node is reached. The root is a command and control system that evaluates attack situations and issues responses. The root typically reports to an operator console where an administrator can manually assess status and issue commands.

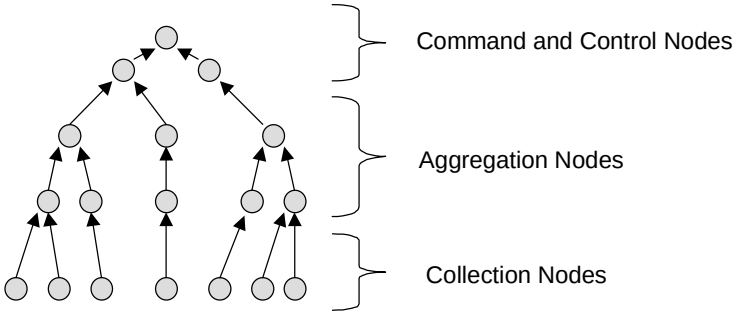

Figure 1: Hierarchical IDS Architecture

In general, hierarchical structures result in efficient communications, whereby refined information filters upward in the hierarchy and control downward. The architecture is excellent for creating scalable distributed IDSs with central points of administration, but somewhat rigid because of the tight binding between functionality and lines of communication that tend to evolve. While IDS components tend implicitly toward a hierarchy, this tendency is not strict. Communications can occur, in general, between any type of components and not solely on a one-to-one or master/slave basis. For example, to improve notification and response, a collection unit may directly communicate a critical event to the command and control node, as well as an aggregation node. Moreover, peer relationships among command and control nodes are needed when different administrations manage portions of an enterprise network, or distinct and separate networks [11].

At least one IDS design, Cooperating Security Managers [31], uses a network structure, where information flows from any node to any other node, by consolidating the collection, aggregation, and command and control functions into a single component residing on every monitored system. Any significant events occurring at one system that stem from a connection originating from another are reported back to the system manager of the originating system by the security manager at the system where the event occurred. In situations where the originating system of the connection is an intermediate node in a communication chain, the system manager is obliged to report onward to the next system manager in the chain. Because of the potential for unconstrained communication flow, network structures, in general, tend to suffer from communications inefficiency when taken to the extreme (i.e., everyone directly communicating with everyone else). However, they compensate for this inefficiency with flexibility in function.

Current IDS Shortcomings

Present-day IDSs are less than perfect [22]. Developers continue to address shortcomings through the improvement and refinement of existing techniques, but some shortcomings are inherent in the way IDSs are constructed. The most common shortcomings include the following items:

- *Lack of Efficiency:* IDSs are often required to evaluate events in real time. This requirement is difficult to meet when faced with a very large number of events as is typical in today's networks. Consequently, host-based IDSs often slow down a system and network-based IDSs drop network packets that they do not have time to process.

- *High Number of False Positives:* Most IDSs detect attacks throughout an enterprise by analyzing information from a single host, a single application, or a single network interface, at many locations throughout the network. False alarms are high and attack recognition is not perfect. Lowering thresholds to reduce false alarms raises the number of attacks that get through undetected as false negatives. Improving the ability of an IDS to detect attacks accurately is the primary problem facing IDS manufactures today.

- *Burdensome Maintenance:* The configuration and maintenance of intrusion detection systems often requires special knowledge and substantial effort. For example, misuse detection has usually been implemented using expert system shells that encode and match signatures using rule sets. Upgrading rule sets involves details peculiar to the expert system and its language for expressing rules sets, and may permit only an indirect specification of the sequential interrelationships between events. Similar considerations may apply to the addition of a statistical metric, typically used for detecting unusual deviations in behavior.

- *Limited Flexibility:* Intrusion detection systems have typically been written for a specific environment and have proved difficult to use in other environments that may have similar policies and concerns. The detection mechanism can also be difficult to adapt to different patterns of usage. Tailoring detection mechanisms specifically to the system in question and replacing them over time with improved detection techniques is also problematic with many IDS implementations. Often the IDS needs to be completely restarted in order to make changes and additions take effect.

- *Vulnerability to Direct Attack:* Because of the reliance on hierarchical structures for components, many IDSs are susceptible to attack. An attacker can cut off a control branch of the IDS by attacking an internal node or even decapitate the entire IDS by taking out the root command and control node. Typically, such critical components reside on platforms that have been hardened to resist direct attack. Nevertheless, other survivability techniques such as redundancy, mobility, dynamic recovery, etc. are lacking in current implementations.

- *Vulnerability to Deception:* A network-based IDS evaluates network packets using a generic network protocol stack to model the behavior of the protocol stack of the hosts that it is protecting. Attackers take advantage of this discrepancy by sending specially adapted packets to a target host, which are interpreted differently by the IDS and by the target. This can be done in various ways such as altering fragmentation, sequence number, and packet flags [26]. The attacker penetrates the target while the IDS either is blind to the attack or fooled into interpreting that the target resisted the attack.

- *Limited Response Capability:* IDSs have traditionally focused on detecting attacks. While detection serves a useful purpose, oftentimes a system administrator is not able to immediately analyze the reports from an IDS and take appropriate action. This gives an attacker a window of opportunity in which to freely operate before being countered by the actions of the administrator. Many IDSs are beginning to implement automated response capabilities to reduce significantly the time available for attackers to extend their grasp on a network. However, they are limited in their ability to adapt dynamically to an attack.

- *No Generic Building Methodology:* In general, the cost of building an IDS from available components is considerable, due in large part to the absence of a structured methodology. No such structuring

insights have emerged from the field itself. This may be partly a result of a lack of common agreement on the techniques for detecting intrusions.

Besides these shortcomings, IDSs are continually faced with new obstacles that they must surmount. Some recent obstacles include the following issues:

- *End-to-end Encryption:* With security improvements in communications protocols, the ability to encrypt traffic on an end-to-end basis is on the rise. Besides thwarting an eavesdropper, encrypted content keeps a network-based IDS from peeking into packets and analyzing their contents for intrusions.

- *High Speed Communications:* Higher communication traffic rates directly affect the processing speed needed to analyze packet content, potentially resulting in lost packets. The trend toward switched communications over broadcast also increases the difficulty for a network-based IDS to monitor multiple communications streams.

- *Breadth of Attacks:* As new attacks are conceived, IDSs must be updated to discover them. While new attacks are added frequently, old ones can seldom be dropped. Typically, the greater the attack coverage, the more processing time that is needed by the detection algorithm.

- *Technology Limits:* It is impossible to construct a program to detect with certainty the presence or absence of harmful code within arbitrary programs or protocol. As existing services evolve and new services are introduced, intrusion detection techniques are faced with the prospect of diminishing returns – greater investments are needed over time for smaller gains in effectiveness.

Mobile Software Agents

A software agent is loosely defined as a program that can exercise an individual's or organization's authority, work autonomously toward a goal, and meet and interact with other agents and its environment. A software agent comprises the code and state information needed to carry out some computation, and requires an agent platform to provide the computational environment in which it operates. Agents may be static or mobile. Stationary agents remain resident at a single platform, while mobile agents are capable of suspending processing on one platform and moving onto another, where they resume execution of their code. Mobile software agents provide a new and useful paradigm for distributed computing. Unlike the client-server computing paradigm, relationships among entities tend to be more dynamic and peer-to-peer, stressing autonomous collaboration.

Figure 2 depicts the movement of an agent among several agent platforms. The platform where an agent originates is referred to as the home platform, and normally is the most trusted environment for an agent. One or more hosts may comprise an agent platform, and an agent platform may support multiple locations or meeting places where agents can interact.

Mobile agent technology has benefited from the work done on intelligent agents, which emphasizes static autonomous agents capable of applying application domain knowledge, and the development of software systems capable of supporting mobile code on heterogeneous hardware (e.g., Java technology). Intelligent agents embody the ability to decompose and solve problems in a collaborative fashion. Agents observe their environment, reason about their own and other agent's actions, interact with other agents, and execute their actions concurrently with other agents. Interactions may convey facts or beliefs via an agent communication language and may depend on an ontology to reach a common understanding of a situation. A significant number of mobile agent systems have been developed at universities and by industry. Although mobile agents retain the characteristics of autonomy and collaboration as with intelligent agents,

emphasis is on mobility characteristics, often relying on simple straightforward algorithms for reasoning and collaboration through less elaborate interpretation of messages.

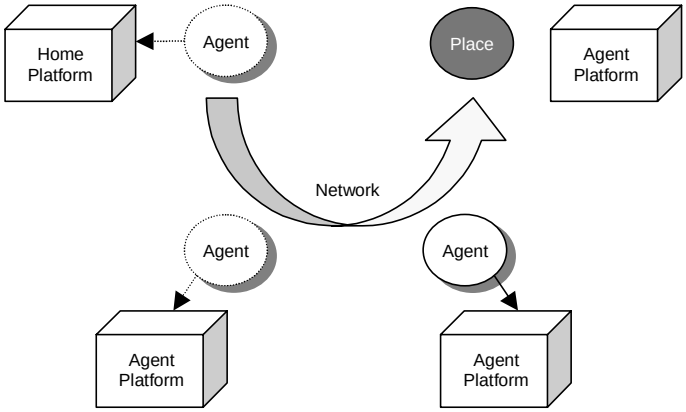

Figure 2: An Agent System Model

There has been considerable debate on the merit of applying mobile agent systems in lieu of client-server, transaction processing, and other well-established technologies (e.g., [7]). That debate continues today. Ultimately such decisions must be made on the particular characteristics of an application and the appropriateness of the technology for the engineered solution. We believe intrusion detection and response to be an application area well suited for software agents. In particular, the ability to move a computation among various nodes offers advantages over present day IDSs that rely on statically placed computations.

For mobile agents to be applied to intrusion detection, participating nodes (i.e., hosts and network devices) must have an agent platform installed. Since many agent systems operate over a wide range of hardware and software, this requirement is not as difficult to fulfill as it may first appear. With mobile agent technology, the collection nodes, internal aggregation nodes, and command and control nodes do not have to be continuously resident on the same physical machine. For example, a mobile agent may function as an aggregation node and move to whatever physical location in the network is best for its purposes. The mobile agent paradigm also offers specialization – the agents may be different for different functions, each looking for distinct attacks and processing data accordingly.

Benefits of Mobile Agents

Mobile agent technology can potentially overcome a number of limitations intrinsic to existing IDSs that employ only static components. For example, mobility and autonomy make them ideal for detection schemes that follow a "cop on the beat," "immune system," or other real-world analogy. This is not to say that the characteristics of mobile agents in themselves are sufficient for achieving improvements in IDSs. When applying mobile agents to this application domain, careful design choices are still required to take advantage of their traits [5]. In particular, the kind of knowledge level coordination required for detecting and responding to intrusions places many demands on agents, including locating other agents with needed capabilities, effectively communicating with them using an common mutually understood vocabulary, and coordinating the actions to be taken to jointly address a given situation.

A number of advantages of using mobile code and mobile agent computing paradigms over their static counterparts have been identified in the past [18, 29] and are relevant for intrusion detection systems.

- *Overcoming Network Latency:* Mobile agents can be dispatched to carry out operations directly at the remote point of interest, allowing them to respond in real time to changes in their environment. In addition to detecting and diagnosing potential network intrusions, mobile agents can provide appropriate response mechanisms. Such actions include gathering attack information sent to or emitted by the target of an attack, shutting down or isolating a system under attack to protect it from further damage, tracing the path of an attack, and shutting down or isolating an attacker's system if the attack is launched from an internal host

- *Reducing Network Load:* Instead of transferring the data across the network, mobile agents can be dispatched to the machine on which the data resides, essentially moving the computation to the data, instead of moving the data to the computation, thus reducing the network load. A side benefit where confidentiality is a concern is the efficiency of moving an encrypted agent and its refined data versus moving all of the raw data in encrypted form.

- *Autonomous and Asynchronous Execution:* For large distributed systems, the ability of the system to continue to operate when portions of it are destroyed or become isolated is essential. Mobile agents can exist and function independently from the creating platform, making them useful as IDS components, since agents that survive an attack may be able to reconstitute damaged components (e.g., by cloning) and restore functionality.

- *Dynamic Adaptation:* The ability for mobile agent systems to sense their environment and react to changes is useful in intrusion detection. Agents may move elsewhere to gain better position or avoid danger, clone themselves for redundancy and parallelism, or marshal other agents for assistance. Agents can also adjust to favorable situations as well as unfavorable ones. When combined with autonomous and asynchronous execution, these characteristics facilitate the building of robust and fault-tolerant systems.

- *Platform Independence:* Agent systems provide an abstract computing environment for agents, independent of the computer hardware and software on which it executes. These characteristics make it a suitable broad-based environment for network management applications in general and intrusion detection in particular, allowing relatively unfettered movement of agents within a domain. This is especially beneficial to response mechanisms, since when an intrusion is detected, remedies can be applied at or initiated from nearly any place in the network. Similarly, detection mechanisms also benefit from widespread mobility with the potential to acquire and fuse data readily from different network sources.

- *Protocol Encapsulation:* In conventional systems, the host owns the interface between communicating entities, requiring any changes to be synchronized for continued interoperation. Mobile agents can incorporate the protocol directly and bring about an upgrade in the interface with the movement of an agent to another host.

Besides these advantages, mobile agents allow a natural way to structure and design an IDS. The agent orientation and mobility considerations provide an effective maxim for organizing data and functionality. Agents inherently tend toward designs having the sought after properties of high cohesion and low coupling of modules.

Although our interest is in applying mobile agents to intrusion detection, it is unlikely that full mobility of all components would ever be effective in practice, due to the associated overhead. Therefore, some IDS components either are designated as static agents or remain static once deployed. Doing so allows application of the mobile agent paradigm, yet relies on mobility only where appropriate. Other practical factors such as trust relationships, performance capabilities, and physical location may also restrict mobile agents to a subset of available agent platforms.

A Mobile Agent Based IDS

While mobile agents do not directly improve the techniques for detection, they can reshape the way the techniques are applied, thereby improving efficiency and effectiveness. One potential area of use is reducing the massive amount of distributed log data moved among the inner nodes within the hierarchy of a conventional IDS. Having agents visit data repositories and mine results is an ideal alternative, well suited to the ability of mobile agents to transfer the computation to the data. Besides reducing network load, the approach is conducive to having specialized agents focused on specific classes of intrusions, such as coordinated attacks that occur over long periods of time from multiple sources.

Another area for use is in minimizing the ability of an attacker to deceive an IDS through discrepancies between the IDS protocol model and the protocol stack of the target. Because agents can replicate themselves and reside on multiple platforms, they potentially can eliminate such ploys. Moving away from a network-based IDS to multiple host-based detection agents running concurrently also reduces the potential for dropped packets occurring, while maximizing the potential for triggering a quick response to a detected intrusion. In addition, having resident components at the host provides the only means for the IDS to view the packets in cleartext, in situations where the host is using network level encryption (e.g., Internet Protocol Security (IPSec)).

Mobile agents can facilitate the implementation of robust, attack-resistant IDS architectures [23]. Agents can relocate when sensing danger or suspicious activity, clone for redundancy or replacement, operate autonomously and asynchronously from where created, collaborate and share knowledge, and be self-organizing (e.g., dynamically reconfiguring relationships to compensate for failure of key components). Moreover, agents are amenable to genetic diversity, which also helps to avoid attacks aimed at circumventing the known and stable detection mechanisms of an IDS.

The greatest potential for mobile agents lies with response to an intrusion rather than its detection. Because responses can be initiated from nearly anywhere in the network, mobile agents can deal with attacks in a more optimal fashion than in a conventional IDS. Mobile agents enhance an IDS's ability to trace an attacker through the attacked network, to respond at the target, respond at the source, to collect evidence about the attack from the host and network components, and to isolate the source and target. The following items describe some of the advantages of applying mobile agents to responding to an intrusion:

- *Tracing an Attacker:* Attackers often log into a chain of many hosts before attacking a target and sometimes spoof their source address. To find the attacker the IDS must trace back along the chain and locate the actual host launching the packets. In order to perform such a trace, the IDS needs the capability to sniff on every Ethernet segment and to analyze every host. Ordinarily, the infrastructure required would be prohibitively expensive, but not with a widely installed base of agent platforms available.

- *Responding at the Target:* When an attack is detected, it is vital to automatically respond at the target host. A quick response can prevent the attacker from establishing a better foothold and using the penetrated host to further compromise the network. It can also minimize the effort needed to recover damage done by the attacker.

- *Responding at the Source:* Responding at the attacker's host gives an IDS much greater power to restrict the attacker's actions. Without using mobile agents, it is unlikely that an IDS would have sufficient access to an attacker's host in order to take corrective action. While this option has limitations, since it requires an agent platform be active on the attacker's host and the attack to come from within the management domain, it also has the potential to be a very effective part of the IDS arsenal.

- *Evidence Gathering:* Currently, it is impractical to automatically gather evidence for an attack from many different sources. The problem is having the right software running at the right place at the right time. Mobile agents offer the ability to run anything, anywhere, at any time, making it conceivable that evidence may be gathered from different hardware platforms, different operating systems, and even different applications such as web servers. Mobile agents can also intelligently audit the network by dynamically reconfiguring the audit capabilities of relevant hosts to strongly audit suspicious or important network locations.

- *Isolating the Source and Target:* Since actions to respond automatically at the target and source may fail, ultimately a response at the network level is needed to limit an attacker's actions. Three general strategies exist: block the target's communications, block the attacker's communications, and block communications between the target and the attacker. The ability for mobile agents to travel to all network elements to carry out remedial actions is what enables them to perform these strategies.

Potential Drawbacks

While mobile agents can help improve IDSs in many areas, they offer no help in others. Mobile agent technology cannot enhance the ability of an IDS to detect attacks from a single event source (e.g., network interface) or reduce false positive rates. Mobility, if used with indiscretion, can also impair the ability of an IDS to process events thereby actually decreasing its detection ability. Compared with present day IDSs these are not serious drawbacks and relate more to the strength of available detection techniques. By far, the main obstacle to applying mobile agents to intrusion detection is security. That is, the use of mobile agents may introduce vulnerabilities that can be exploited by an attacker to propagate an attack or subvert detection by the IDS.

The security threats for the mobile agent computing paradigm can be classified into four broad categories: agent-to-agent, agent-to-platform, platform-to-agent, and other-to-platform. The agent-to-agent category represents the set of threats in which agents exploit security weaknesses of or launch attacks against other agents residing on the same agent platform. The agent-to-platform category represents the set of threats in which agents exploit security weaknesses of or launch attacks against an agent platform where they reside. The platform-to-agent category represents the set of threats in which agent platforms compromise the security of agents that reside there. The other-to-platform category represents the set of threats in which external entities, including agents and agent platforms situated elsewhere on the network, threaten the security of an agent platform, including its underlying operating system and network communications service.

For the most part, these threats are not unlike those faced by any distributed application and addressable through conventional security techniques, including isolation of agents from other agents and the agent platform, language and code safety, controlled access to resources, audit by both agents and platforms, and authenticated and protected communications. There is one exception, however. The platform-to-agent category is extremely difficult to defend against in applications that require unrestricted movement of agents across the Internet [15]. Fortunately, intrusion detection is quite different in nature from typical Internet applications, involving a closed domain and a small set of known users. These same characteristics hold true in general for other network management applications, and allow adequate defenses to be put into place. Table 1 summarizes the common threats an agent platform poses to a management agent and the applicable countermeasures available.

Table 1: Summary of Platform-to-Agent Threats and Countermeasures

Threat	Countermeasure
Attacker attempts to introduce a malicious agent	All management agents are signed by a system security officer and must pass validation before being executed. Agent platforms must mutually authenticate successfully before communicating (includes inter-platform agent communications).
Attacker attempts to gain access to a host supporting an agent platform	Detection of the attack by the IDS and possible initiation of a response. Hardening of the platforms commensurate with their use by network management agents.
Attacker gains access to a host and attempts to corrupt an agent's state information	Cryptographic encapsulation of the partial results collected, allowing detection of any modification. Privilege management of an agent's access rights, thereby limiting its activities to a minimum set.
Attacker gains access to a host and terminates or indefinitely retains an agent	Detection of the lost or tardy agent by the IDS and possible initiation of a response.

Our objective is to ensure that the use of mobile agents in intrusion detection maintains a level of protection equivalent or superior to a conventional IDS. The basic principle is to allow agents to move among comparable trustworthy hosts within their security domain, and have each platform place the agent under the same security policy, granting or denying the same privileges it had on its previous platform.

Within classical hierarchical IDSs, trust relationships are strong in the downward directions (i.e., subordinates trust superiors), but weaker in the reverse (i.e., superiors do not trust subordinates) [11]. By successfully subverting a node high in the hierarchy, an attacker can isolate a control branch of an IDS or completely disable it by taking out the root. Mobile components have an obvious advantage in being able to move away from danger. As mentioned earlier, to resist direct attacks, conventional IDSs rely on hardened systems for their critical components. Similarly, mobile agents can also take advantage of hardened systems by restricting the movement of the critical components they embody to such systems. That is, the trustworthiness of an agent platform can be a factor in the itinerary of an agent.

A key protection mechanism is digital signature. The code of all management agents must be signed by a system security officer and successfully pass validation before the agent platform allows execution. This countermeasure prevents an attacker from modifying an agent's code or forging a bogus management agent. In combination with the ability of most agent platforms to mutually authenticate one another before allowing an agent to move, this measure makes it difficult to attack the mobile agent paradigm directly.

To circumvent the agent system, an attacker would need to successfully penetrate the host system supporting the agent platform, using the same techniques as with any conventional IDS. While the level of vulnerability of a conventional system or one based on mobile agents is arguably the same (i.e., similar hardening and IDS mechanisms to be overcome), the potential for damage in the latter case is higher, since

potentially more than one IDS component could be affected. An attacker may delay or terminate incoming management agents, falsify information to them, or modify their state information. The latter is a particularly insidious form of attack, since it can radically change the agent's behavior or the accuracy of the computation.

The problem an attacker faces with delaying or terminating agents is that the IDS may note their absence (e.g., through "heartbeat" messages), raise the level of suspicion of an intrusion, and launch a replacement agent or response agents to deal with the situation. Deceit is a better option, but not without its problems. The degree of difficulty in modifying the agent platform to deceive, but not otherwise hamper agents could vary widely. For example, the agent platform might be incorporated into the kernel requiring a high degree of skill to subvert. Moreover, if the IDS periodically checks agent platforms for tampering using agents containing genetically diversified tests, the stakes are raised higher.

To counter the impact of a highly skilled intruder successfully taking control of a host and tampering with an agent's state information, many agent systems [16, 24, 30] provide a means to limit an agent's capabilities through privilege management. Authorizations, conveyed either internally within some data structure or externally along with the agent, are encoded into a protected object and bound to an agent, serving in effect as a kind of agent passport or visa. For example, a signed digital certificate, issued by the domain authority (e.g., system security officer) and containing the needed authorizations, can be cryptographically bound to the agent's code and serve this purpose. Any attempts to violate those authorizations are raised to the agent platform to take appropriate action. While not all agents performing management tasks need a high level of authorization, some agents may need to run with administrative or root privileges.

For additional protection, a management agent can encapsulate collected results to prevent tampering, such that each result entry obtained at a platform is cryptographically bound to all previous entries and to the identity of the subsequent platform to be visited [17]. Each platform digitally signs its entry using its private key, and uses a secure hash function to link results and identities within an entry. Besides forward integrity, the encapsulation technique also provides confidentiality by encrypting each piece of accumulated information with the public key of the originator of the agent. The technique also ensures that a platform is unable to modify its entry in the chain, should the agent revisit it, or to collude with another platform to modify entries, without invalidating the chain. Other techniques having different properties also exist for encapsulating partial results [20. 32].

Related Work

Applying software agents to intrusion detection is not entirely new. However, much of the related work outlined below has emphasized static agents as opposed to mobile ones. Nevertheless, collectively they give a good indication of the range and richness of solutions that are possible.

Colleagues at NIST within the Mobile Agent Security project have developed an attack resistant architecture for defending intrusion detection systems from denial of service attacks [23]. The architecture uses mobile agent technology, in combination with network topology features and communication restrictions between different types of IDS components identified, to make those components invisible from an attacker's normal view of the network. In the event of a successful attack, the architecture mitigates its effect by allowing IDS components to relocate from attacked hosts to operational hosts.

Researchers at the Université Claude Bernard Lyon (UCBL) are experimenting with applying a social insect-based, mobile agent framework known as ANT (Artificial Network Termite colony) to the problem of intrusion detection [10]. ANT relies on the raising and lowering of pheromone fields, which represent criteria for agents to satisfy, to guide simple agents toward collectively exhibiting complex problem-solving behavior. The design of their intrusion detection system involves static internal agents and

components at the agent platform as well as mobile agents. Static components include a pheromone server, which supplies and receives information from visiting agents, and a watcher, which detects the likelihood of an intrusion from its available information sources and emits pheromones appropriately coded for diffusion throughout the network. Pheromones are spread to pheromone servers via short lived mobile agents, while other defensively minded agents prowl the network performing system checks, sensing the gradient of the pheromone field, and deciding whether to take a defensive action or move onward in a direction where the field is stronger.

The Intrusion Detection Agent (IDA) system [2] at The Information-technology Promotion Agency (IPA) in Japan, relies on mobile agents to trace intruders among the various hosts involved in an intrusion. IDA works by focusing on specific events that may relate to intrusions, referred to as "Marks Left by Suspected Intruder (MLSI)." If an MLSI is found, IDA gathers information related to the MLSI, analyzes the information, and decides whether an intrusion has occurred. The system follows a hierarchical structure, with a central manager at the root and a variety of agents at the leaves. A sensor agent statically resides at a node in search of an MLSI, and upon discovery, notifies the manager who dispatches a tracing agent to the host. The tracing agent initiates an information-gathering agent to collect related information at the host and continues onto any site identified as a suspected point of origination. The information-gathering agent returns to the manager with its results and logs them on a bulletin board, used for integrating the information collected about the intrusion from the various agents involved. The tracing agent eventually returns to the manager when it exhausts all routes or ends up at the final point of origination. Possible duplication caused by multiple sensors detecting the same intrusion is resolved through a message board at each monitored host. The developers indicate that the resulting system is an efficient and effective way for detecting intrusions.

Work done by the Army on the Advanced Telecommunications/Information distribution Research Program (ATIRP) [8, 14] addresses computer vulnerability assessment, not intrusion detection. However, intrusion detection modules could easily be substituted for vulnerability assessment modules to create a rudimentary IDS. A central dispatcher launches agents to one or more target nodes to test for known vulnerabilities and report back results. Agents are composed dynamically using a genetic algorithm, which continually attempts to maximize the likelihood of discovering existing vulnerabilities. The gene pool from which agents evolve consists of code fragments that correspond to a detection technique and have been designed for composition with other fragments. The architecture also has significant security capabilities, based on cryptographic signatures and public key certificates.

The Autonomous Agents for Intrusion Detection (AAFID) effort at Purdue University [3] employs a hierarchical architecture of agents. At the root of the hierarchy are monitors, which provide global command and control and perform analysis of information flowing from lower-level nodes. At the leaves are agents that collect event information. The agents are static and reside on special purpose agent platforms, called transceivers. Transceivers perform command and control of locally running agents and the analysis or reduction processing of the information received from the agents. Transceivers feed processed information onto monitors. AAFID is in many ways a classical IDS hierarchy with agents used mainly as a means for structuring the intrusion detection collection component into a set of lightweight software components, which can be easily reconfigured.

Researchers at the Institut EURECOM in France are exploring the use of applying Distributed Artificial Intelligence to intrusion detection in the form of a multi-agent system [4]. Their architecture, called Multi Agent Network Intrusion Detection (MA-NID), provides a flexible means of integrating intelligent agents into a network environment. Rather than a centralized decision manager approach, NID agents, located in specific network entities, work autonomously and cooperate to perform intrusion detection. Agents having both knowledge-based cognitive abilities for reasoning and stimulus-response reactive capacities for rapid reaction are envisaged.

The Java Agents for Meta-Learning (JAM) project [19] at Columbia University applies meta-learning to distributed data mining using intelligent agents. The design has two key components: local agents that learn how to detect fraud and provide intrusion detection services within a single corporate information system, and a secure, integrated meta-learning system that combines the collective knowledge acquired by individual local agents. Data mining, like neural network and other single-point learning applications, does not engender knowledge sharing among agents. The meta-learning approach attempts to overcome this limitation by integrating a number of separately learned classifiers embodied as remote agents.

A project at Iowa State University [12] involves an IDS based upon intelligent agent technology, in a manner somewhat similar to JAM. Mobility is used to allow various types of intelligent agents that employ classifier algorithms to travel among collection points, referred to as data cleaners, and uncover suspicious activities. The architecture is hierarchical, with a data warehouse at the root, data cleaners at the leaves, and classifier agents in between. A classifier agent specializes on a specific category of intrusion and is capable of collaborating with agents of another category to determine the severity level of an activity deemed suspicious. Moving the computational analysis to each collection point avoids the costly movement of information to an aggregation unit. The resulting arrangement is also less likely to be vulnerable to attack, since static aggregation units are avoided.

Conclusions

The preceding sections indicate a variety of ways mobile agents could be applied to intrusion detection to evolve new designs that are more efficient, scalable, and robust. While not a perfect solution, mobile agent technology goes a long way toward being able to realize the ideal behavior wanted from an IDS. Not only do aspects of the detection side of the equation benefit, but also, and perhaps more significantly, the response side of the equation is improved significantly. Because present day IDSs do not inherently involve mobile agent technology, we do not expect a wholesale transition to this paradigm. However, the technology lends itself to gradual adoption and use. Because of the noted advantages, particularly with respect to responding to an intrusion, mobile agent technology has the potential for gaining an initial foothold and expanding its reach over time.

References

[1] James P Anderson, “Computer Security Threat Monitoring and Surveillance,” Technical Report, James P. Anderson Co., Fort Washington, PA, April 1980.

[2] Midori Asaka, Shunji Okazawa, Atsushi Taguchi, and Shigeki Goto, “A Method of Tracing Intruders by Use of Mobile Agents,” INET’99 Conference, June 1999.

[3] Jai Balasubramaniyan, Jose Omar Garcia-Fernandez, David Isacoff, E. H. Spafford, and Diego Zamboni, “An Architecture for Intrusion Detection using Autonomous Agents,” Department of Computer Sciences, Purdue University; Coast TR 98-05, 1998.

[4] Karima Boudaoud, Houda Labiod, “MA-NID: A Multi-Agent System for Network Intrusion Detection,” Eighth International Conference on Intelligent Systems, June 1999.

[5] Giacomo Cabri, Letizia Leonardi, Franco Zambonelli, “The Impact of the Coordination Model in the Design of Mobile Agent Applications,” Twenty-second Computer Software and Applications Conference (COMPSAC), August 1998.

[6] S. Staniford-Chen, et al., “GrIDS – A Graph Based Intrusion Detection System for Large Networks,” Nineteenth National Computer Security Conference, pp.361-370, October 1996.

[7] David Chess, Benjamin Grosof, Colin Harrison, David Levine, Colin Parris, Gene Tsudik, "Itinerant Agents for Mobile Computing," IEEE Personal Communications, 2(5), pp.34-49, October 1995.

[8] Michael Conner, Chirag Patel, and Mike Little, "Genetic Algorithm/Artificial Life Evolution of Security Vulnerability Agents," Army Research Laboratory Federal Laboratory Third Annual Symposium on Advanced Telecommunications & Information Distribution Research Program (ATIRP), February 1999.

[9] Dorothy E. Denning, "An Intrusion Detection Model," IEEE Transactions on Software Engineering, 13(2), pp.222-232, February 1987.

[10] Serge Fenet and Salima Hassas, "A Distributed Intrusion Detection and Response System Based on Mobile Autonomous Agents Using Social Insects Communication Paradigm," First International Workshop on Security of Mobile Multiagent Systems, Autonomous Agents Conference, May 2001.

[11] Deborah Frincke, Don Tobin, Jesse McConnell, Jamie Marconi, and Dean Polla, "A Framework for Cooperative Intrusion Detection," Twenty-first National Information Systems Security Conference, pp.361-373, October 1998.

[12] Guy Helmer, Johnny S. K. Wong, Vasant Honavar, and Les Miller, "Intelligent Agents for Intrusion Detection," IEEE Information Technology Conference, pp.121-124, September 1998.

[13] L.Todd Heberlein, G.V. Dias, K. N. Levitt, B. Mukherjee, J. Wood, and D. Wolber, "A Network Security Monitor," Symposium on Research in Security and Privacy, pp.296-304, May 1990.

[14] Stuart Jacobs, Dave Dumas, William Booth, and Mike Little, "Security Architecture for Intelligent Agent Based Vulnerability Analysis," Third Annual Fedlab Symposium on Advanced Telecommunications/Information Distribution Research Program, pp.447-451, February 1999.

[15] Wayne Jansen and Tom Karygiannis, "Mobile Agents and Security," NIST Special Publication 800-19, September 1999.

[16] Wayne Jansen and Tom Karygiannis, "Privilege Management of Mobile Agents," Twenty-third National Information Systems Security Conference, pp.362-370, October 2000.

[17] Günter Karjoth, N. Asokan, and Ceki Gülcü, "Protecting the Computation Results of Free-Roaming Agents," Second International Workshop on Mobile Agents, Stuttgart, Germany, September 1998.

[18] Danny Lange and Mitsuru Oshima, *Programming and Deploying Java Mobile Agents with Aglets*, ISBN:0-201-32582-9, Addison-Wesley, 1998.

[19] Wenke Lee, Sal Stolfo, and Kui Mok, "A Data Mining Framework for Building Intrusion Detection Models," IEEE Symposium on Security and Privacy, pp. 120-132, May 1999.

[20] Sergio Loureiro, Refik Molova and Alain Pannetrat, "Secure Data Collection with Updates," Workshop on Agents on Electronic Commerce, First Asia Pacific Conference on Intelligent Agent Technology December 1999, pp. 121-130.

[21] Teresa F. Lunt and R. Jagannathan, "A Prototype Real-Time Intrusion-Detection Expert System," IEEE Symposium on Security and Privacy, pp. 59-66, April 1988.

[22] Stefano Martino, "A Mobile Agent Approach to Intrusion Detection," Joint Research Centre-Institute for Systems, Informatics and Safety, Italy, June 1999.

[23] Peter Mell, Donald Marks, and Mark McLarnon, “A Denial of Service Resistant Intrusion Detection Architecture,” Computer Networks Journal, October 2000.

[24] Srilekha Mudumbai, Abdeliah Essiari, and William Johnston, “Anchor Toolkit - A Secure Mobile Agent System,” Mobile Agents ’99 Conference, October 1999.

[25] Biswanath Mukherjee, L. Todd Heverlein, and Karl N. Levitt, “Network Intrusion Detection,” IEEE Network, pp. 26-41, May/June 1994.

[26] Thomas H. Ptacek and Timothy N. Newsham, “Insertion, Evasion, and Denial of Service: Eluding Network Intrusion Detection,” Technical Report, Secure Networks, Inc., January 1998.

[27] Michael M. Sebring, Eric Shellhouse, Marry Hanna, and R. Alan Whitehurst, “Expert Systems in Intrusion Detection: A Case Study,” Eleventh National Computer Security Conference, pp.74-81, October 1988.

[28] Stephen E. Smaha, “Haystack: An Intrusion Detection System,” Fourth Aerospace Computer Security Applications Conference, pp.37-44, December 1988.

[29] Jonathan Smith, “A Survey of Process Migration Mechanisms,” Operating Systems Review, 22(3), ACM Special Interest Group on Operating Systems, pp.28-40, July 1988.

[30] Joseph Tardo and Luis Valente, “Mobile Agent Security and Telescript,” IEEE COMPCON ’96, pp.58-63, February 1996.

[31] Gregory B. White, Eric A. Fisch, and Udo W. Pooch, “Cooperating Security Managers: A Peer-based Intrusion Detection System,” IEEE Network, 10(1), pp.20-23, January/February 1996.

[32] Bennet S. Yee, “A Sanctuary for Mobile Agents,” Technical Report CS97-537, University of California in San Diego, April 28, 1997