Towards Decentralized Network Management and Reliability*

Lynn W. Jones and Tamitha L. Carpenter
Stottler Henke Associates, Inc.
1107 NE 45th St., Suite 427
Seattle, WA 98105

Abstract - We depend more and more on computer networks, yet the growth of networks and their heterogeneous composition make ensuring network reliability a daunting task. SHAI introduces a novel alternative approach that pushes monitoring and response out into the network in support of better scalability and decision making. Using autonomous agents, the system detects and responds to network degrading events, even those not previously observed, and even when parts of the network have failed. This paper describes our motivation and system architecture and gives an example of how this system might perform.

Index Terms - Anomaly detection, case based reasoning, intelligent agents, machine learning, network management, network security.

1. Introduction

As computing power has grown, so has our reliance on computers and networks. Communications, transportation, just-in-time inventory and production, energy and water supply – the very foundations of our economic, personal, and political safety all depend on reliable network availability. And yet, computer networks are fraught with vulnerabilities. Computer "crackers" continue to find new ways to exploit weaknesses in both hardware and software. The Internet itself permits scripts, viruses, and other attack tools to be disseminated as rapidly as their solutions. A more widespread but less publicized problem is that of computer misuse or attack by "insiders" such as disgruntled employees or spies [1]. These incidents might include the planting of viruses or launching of denial of service (DoS) attacks just like ones mounted from outside the network, or they might be much more subtle and harder to detect, such as unauthorized viewing or theft of information or falsification of data.

There are a number of tools aimed at particular aspects of network security, such as policy management consoles ([2], [3]), virus scanners ([4], [5]), and intrusion detection systems (IDSs) ([6], [7]). Many virus scanners and IDSs are signature-based; that is, they match suspicious files or activities against previously identified patterns. Some of these systems are "brittle" and can easily be circumvented by making small changes in the attack sequence or even in the subject line of emails delivering the virus [8]. Virus scanners are fairly successful at detecting known viruses before they can cause damage, because the behavior of a given virus is relatively fixed. Intrusion and attack signatures are more difficult to match because there is no definitive window of time during which attack precursors (such as gaining administrative privilege) must occur. Exploits of this nature, while perhaps detectable, are more difficult to predict and prevent.

Other weaknesses trouble signature-based systems. They are always a step behind the attackers, since they use profiles of attacks that have already occurred. They cannot protect the first victims of a new exploit, and other networks remain vulnerable while a signature definition for the attack or virus is being produced. New vulnerabilities and exploits are constantly being discovered, and as the signature libraries grow, searching them consumes too many resources for real-time protection. Signature-based systems also tend to match similar patterns that do not indicate viruses, intrusions, or attacks, generating

* Support for this work was provided by the Defense Advanced Research Projects Agency through contract number DAAH01-00-C-R220.

false alarms that make them more difficult and time consuming to administer. Some newer IDSs are combating these shortcomings by developing anomaly detection or hybrid approaches ([9], [10]).

Network management tasks are not confined to security. A computer network, like any other complex system, has its share of equipment failures that can lead to downtime and loss of service. Unlike many other systems, however, computer networks are evolving extremely rapidly and are comprised of many different hardware and software components, utilizing different protocols, developed at different times by different vendors and to different specifications. In addition to routine faults, constant upgrading of hardware, software, and drivers presents opportunity for conflict between various components. Meanwhile, backward-compatibility requirements allow security and other flaws to persist. Because of the complexity and opacity of network behavior, interoperability and configuration errors may be difficult to detect and diagnose. Thus, even in a world free of malicious computer users, building and maintaining computer networks requires considerable expertise and ongoing effort. Various network monitoring and management tools do exist, but many are device- or vendor-specific. Integration, however, is improving ([11], [12], [13]).

The task of maintaining network health is compounded by the difficulty of accurately diagnosing problems once symptoms are observed. There is no exact correspondence between network problems and their underlying causes [14]. Faults, misconfigurations, attacks and misuse may manifest themselves in a variety of ways, and observable symptoms may have a number of possible causes. Problems may be intermittent and difficult to consistently reproduce. Relatively minor faults can persist undetected, exacerbating and masking the causes of larger events that might occur.

As these issues indicate, computer networks are analogous to other complex systems, such as automobile engines and manufacturing processes, requiring automated analysis and control in order to remain functional and reliable. In this paper we present a work in progress, the Multi-Agent System for Network Resource Reliability (MASRR). This system is designed specifically to address the following needs and shortcomings of currently available approaches:

Scalability. Cooperating agents each monitor a part of the network and share information as needed, reducing redundant messages and eliminating the processing and data transfer bottleneck of a centralized system.

Interoperability. Agents themselves are platform-independent and offer a consistent interface to the administrator. Platform and device specific modules allow an agent executing in one environment to monitor or act upon network elements of different types and having proprietary management interfaces.

Survivability. The decentralized system allows agents to continue to act even when parts of the network have been compromised or have become unavailable. Localized response and reasoning under uncertainty enable MASRR to preserve network operation as much as possible, even in the face of faults or attacks. Agents protect themselves from triggering a denial of service effect by taking progressive action, making small corrections while gathering more information and increasing certainty as to the real cause of a problem.

Concurrent diagnosis. Network problems and symptoms have many-to-many relationships. Agents can follow several lines of reasoning at once in order to gather more information and take appropriate response. Information fusion by cooperating agents gives a "big-picture" view of the network to an individual agent monitoring only a part of the network.

Integration of security and management. MASRR agents recognize any number of network-degrading events and are able to respond to faults and misconfigurations as well as intrusions and attacks. While MASRR is not itself an IDS, agents can reason about the output of an IDS (or several different IDS systems) for improved diagnosis and response.

Anomaly detection. Signatures of attacks, intrusions, and faults are useful for diagnosing symptoms, but they do not aid in the recognition of events not previously observed. MASRR makes use of a novel anomaly detection component, also under development. This modeling of normal network behavior and usage promotes detection of anomalous insider usage as well.

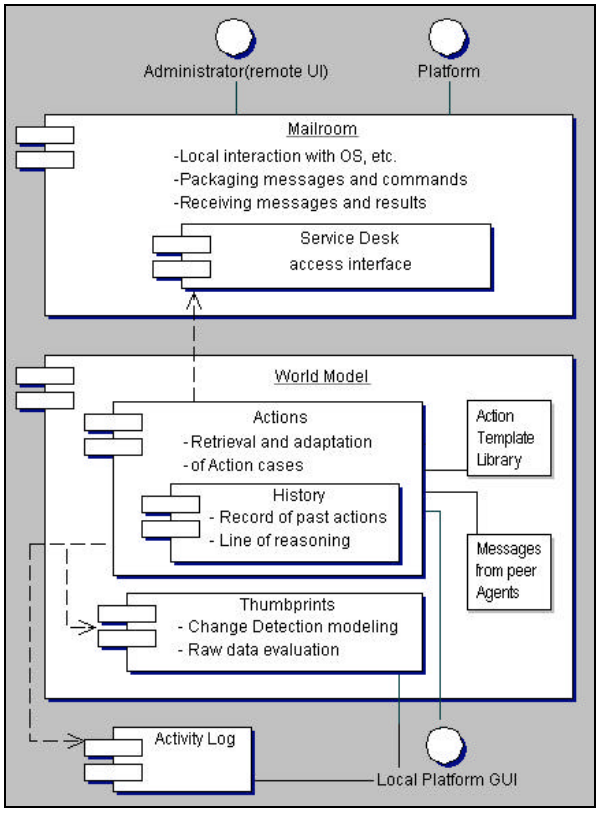

Figure 1. MASSR agent architecture.

Accurate alarm rate. Computer network usage is dynamic. There are normal periods of activity, similar to highway rush hours, but even these will drift over time. Moreover, no two networks will have identical characteristics. MASRR agents and their anomaly detection components learn what is "normal" for the specific parts of the network they monitor and then detect departures from expectation. They adapt to changes when appropriate. Thus MASRR agents stay closely "tuned" to the network to present low false alarm and low false negative (missed alarm) rates.

Low overhead for real-time operation. Agents will perform routine monitoring using standard Managed Information Base (MIB) data. This information is small, widely supported, and easy to collect and process. When suspicions warrant, higher cost data, such as packet and network flow traces, may be collected. In addition, the ability to process IDS output maximizes the utility of both the IDS and of MASRR.

The remainder of this paper presents some details of the MASRR system architecture, including "thumbprinting" of the normal network behavior and selection of actions in response to symptoms. We give an example of how the system might behave in a particular, real-world situation. Our conclusions describe the project status and outline some future work.

2. Description of MASRR

The MASRR system consists of a collection of semi-autonomous agents deployed throughout a computer network. Decentralized monitoring and response fulfills requirements of scalability and survivability, as described in the preceding section. The agent architecture, shown in Figure 1, illustrates how MASRR addresses some of the other requirements.

The component we have called the *Mailroom* provides interoperability. It separates agent reasoning from platform-specific details of execution. Our prototype agents are implemented in Java, which provides a standard interface to some operations. Other operations, however, will interact directly with a proprietary element such as an operating system or a router. The agent can ask for and reason about, for example, the IP address of its default gateway, and the Mailroom abstracts away details such as whether to use `ipconfig` (on Windows®) or `route` (on Linux™), and how to parse the desired information from the command output.

MASRR agents evaluate the network by building thumbprints of normal behavior and noting when the current behavior deviates from what is expected. An important part of the *Thumbprints* component is SHAI's innovative Change and Anomaly Detection (ChAD) data mining system [15]. ChAD's unique, adaptive approach allows it to report on changes in behavior, transition between learned normal periods of behavior, and to adapt to the changes as needed. Thumbprinting includes heuristic evaluation of raw data as well as the output of the ChAD system, using encoded expertise from network security and management domains.

The *Actions* component further integrates network management and security. Using domain knowledge, it links symptoms to responses. The prototype MASRR system includes a library of action cases, which are composed of one or more steps and may contain their own logic for handling ordering of steps, time-outs, and contingencies. Action cases are indexed by evaluations of network behavior,

including the Thumbprint and messages from peer agents. One or more action cases are selected using the current set of evaluations. The agent can pursue multiple lines of reasoning, using a recorded history, and can take intermediate action while gathering more information to decrease uncertainty about possible causes of problems (concurrent diagnosis). Certainly the library cannot contain actions for every conceivable event, but there are sufficient representative cases that can be adapted to respond to and improve many situations, even those not previously observed.

The Thumbprints and Actions modules combine to provide effective response to network-degrading events, whether caused by fault or attack. False alarms are reduced by a thumbprinting method that adapts to changes even as it reports them, and by encoding actions and their selection indices that promote remediative response as the agent obtains diagnostic information and reassesses the effects of its actions.

3. Example Scenario

We recount a network misconfiguration scenario and describe how MASRR would perform in similar circumstances. A *broadcast storm* is a "state in which a message that has been broadcast across a network results in even more responses, and each response results in still more responses in a snowball effect. A severe broadcast storm can block all other network traffic, resulting in a network meltdown. Broadcast storms can usually be prevented by carefully configuring a network to block illegal broadcast messages" [16]. Not all broadcast traffic is bad – many routers and services use broadcasts to advertise availability of network resources. However, misconfigurations can allow broadcasts to "run amok," degrading network performance. Storms can subside on their own but may recur frequently, as in this real-world example [17]:

In a Novell IPX network, thousands of broadcast packets were seen in short periods of time. The packets included Routing Information Protocol (RIP) and Service Advertising Protocol (SAP) messages, from both routers and servers, announcing services and routes that were no longer reachable. The packets that appeared to be the beginning of the storm were from routers announcing that they had lost contact with hundreds of networks and SAP services. All the other routers and servers then propagated the announcements. Here are the steps the network analysts took to identify and correct the cause:

- examined the intervals at which the storms were occurring,
- filtered the data to isolate traffic at one router,
- learned that the first packet of the storm always announced the unreachability of the same network or service lost during the prior storm,
- found that every five minutes, a router was announcing that a particular server was no longer reachable,
- compared these messages with SAPs advertising availability of that server.

They found that the advertisements occurred over two-minute intervals and then were absent over three-minute intervals and suspected that a router had been configured to supply RIP and SAP across a WAN link every five minutes, which they found to be true. However, the receiving router was configured to expect updated RIP and SAP messages every minute. When updates weren't received, the router assumed the networks and services were unavailable and broadcast that announcement, thus causing the broadcast storm. When the receiving router was set to expect five-minute RIP and SAP updates, the broadcast storms went away.

There are several items of note. First, these broadcast storms had been occurring undetected, and were only found when the company held a training session led by independent network management consultants. The disruption of the network was too transient to have been otherwise investigated, yet the impact could have made other problems hard to find, or could potentially have compounded other problems, inducing enough stress to overwhelm the network. Next, the diagnostics concentrated on network traces, looking at evidence of the broadcast storm. The analysts looked at the message origins and intervals, and not at whether or not the service was truly unreachable. Finally, their suspicion (and conclusion) was likely drawn from a great deal of knowledge and experience. We are not

told if this was the first diagnosis that they pursued.

MASRR's response to such a situation. While the above scenario transpired on a network of hundreds of nodes, we illustrate the scenario with the small abstraction of a network as diagrammed in Figure 2. The scale of the problem can be reduced to the task of, rather than detecting the broadcast storm itself, observing the discrepancy in information between peer agents monitoring the routers. Initially, service V is available. The sending router S advertises the availability every 5 minutes. Router R announces availability every minute, and is also expecting receipt of an advertisement every minute.

Around time *t*, S and R advertise the availability of service V. A minute later, R's path to V has not been refreshed. R assumes that service V is no longer available, and broadcasts that message. This announcement goes only to "downstream" routers 3 and 4, and the broadcast storm is confined to R's subnet. MASRR agent M_R observes the broadcast but has received no "bad news" from peer agent M_S. Agent M_R queries M_S about the loss of service V; M_S replies that V is available. The two agents work together to determine why R thinks the service is unavailable. Cases selected during the reasoning process include an action to compare and reconcile the advertisement intervals configured in the routers. This case is retrieved by the description of symptoms and is adapted to correct router R's refresh interval to 5 minutes so that it will not incorrectly announce that Service V has been lost.

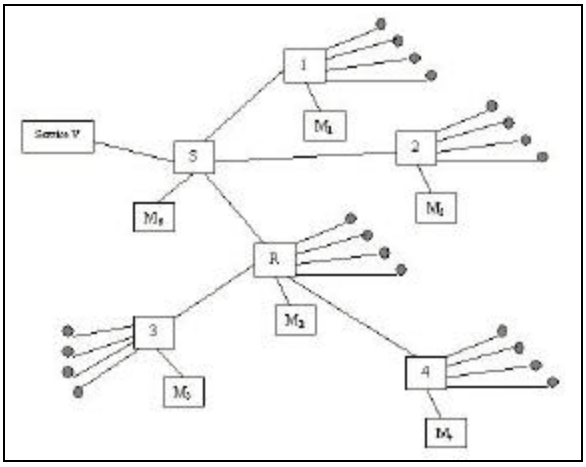

Figure 2. Small abstraction of a network topology. The "sending" router S advertises that it has a path to service V every 5 minutes. The "receiving" router R expects the advertisement to refresh every minute. MASRR agents M_i monitor the routers and switches in the network.

At the same time, agents M_3 and M_4 go through the same process. However, their upstream peer, M_R, informs them that the problem lies further upstream. Thus M_3 and M_4 respond by accepting that the service is effectively unavailable and deferring problem solving to M_R. Agents M_3 and M_4 can implement selective filtering of broadcasts to prevent the storms, which is the cause of the network degradation they observe. Similar events may or may not be occurring at M_1 or M_2, depending on the router configurations at 1 and 2. When M_R and M_S resolve the discrepancy between refresh intervals, M_R informs agents M_3 and M_4, which then remove the broadcast filters.

MASRR agents' ChAD models would have been trained on data that included the broadcast storms, so the corrected behavior would trigger anomaly reports. However, MASRR agents use records of their actions (the *History*, in Figure 1) along with performance metrics and heuristics to recognize that there is no improper activity taking place. ChAD models adapt to the new, normal characteristics, and MASRR remains in operation without intervention by the administrator.

Comparison. The scenario exemplifies what we consider a traditional approach to network management, with data analysis occurring at a centralized location (in this case, by human analysts). We see differences along the following measures:

MASRR's decentralized approach detects or corrects the situation sooner than the centralized approach. MASRR agents recognize the discrepancy in their observations before the broadcast storm symptoms are observed, and immediately assess that the problem is occurring between S and R. The centralized approach starts by examining the broadcast storm packet trace in the subnets and has to deduce or infer the location of the problem.

The decentralized approach is less expensive, in terms of processing time and bandwidth consumption. Before the centralized processor can begin diagnosis of the root cause of the example situation, it must first correlate packet

trace data, which, in turn, requires that all the data be shipped to a centralized location. MASRR agents can concentrate on identifying and correcting the problem after transmitting a small amount of data across a local communication path. Information from uninvolved stations is not considered, and most, if not all, of the relevant information is locally available.

Our decentralized approach is less reliant on network connectivity. Because the centralized approach requires the data to be sent to one location, if the connection is lost or becomes too congested, the data will not arrive. With MASRR, only communication between S and R is required; if that fails, then the service is truly unavailable from R's point of view. In general, MASRR agents reason amongst themselves; if connections between peers are lost, they continue to monitor and act to improve or preserve network reliability locally.

MASRR's agent-reasoning approach provides meaningful reporting to the network administrator. In the example, detailed analysis revealed a network-degrading situation that had previously gone undetected. Even if the administrator had used some of the available network monitoring tools, substantial analysis would have remained in order to track down the source of the broadcast storms. MASRR agents would detect this problem without supervision and either resolve it automatically or alert the administrator to the misconfiguration. In either case, the system would provide a detailed explanation of its reasoning and the root cause of the problem.

4. Conclusions and Future Work

SHAI is developing a prototype of the MASRR system to meet the need for scalable network management and security. Key strengths of the system are its abilities to detect faults and attacks not previously observed, to recognize anomalous network usage, to adapt to changing network characteristics, and to continue taking corrective action even when parts of the network have failed. Much remains to be done. We recognize that this paper does not address security of the MASRR system itself. We have deferred that study in favor of developing anomaly detection and agent reasoning, claiming that we can use known approaches to solve these issues. The system requires significant knowledge engineering to create the action template library and indexing. Other issues, such as Quality of Service, administrative policy enforcement, installation and deployment, and more, have been included in our requirements analysis and design work but are not presented in this paper.

Other areas that we would particularly like to develop include machine learning and feature selection components. The learning component would assess the effectiveness of actions in order to improve both indexing and adaptation of actions. It would also employ post-analysis to identify event pre-cursors for prediction or earlier detection. A feature selection approach such as that of Cabrera, et. al. [1] would also improve action case indexing as well as give the agent better predictive abilities. These additions would make the agents much more proactive in their response, allowing them to steer the network to minimize or prevent problems. These components were considered in developing our architecture, but they will not be developed under our current contract.

We expect to conduct experiments with the prototype MASRR system this summer and to report results in the fall.

References

[1] J.B.D. Cabrera, L. Lewis, X. Qin, W. Lee, and R. K. Mehra, "Proactive Intrusion Detection: A Study on Temporal Data Mining," in *Data Mining in Computer Security*, Daniel Barbará and Sushil Jajodia, eds., Kluwer Academic Press. March 2002.

[2] Tripwire Policy Manager, http://www.tripwire.com.

[3] Microsoft Windows® Group Policy and Active Directory, http://www.microsoft.com.

[4] Norton AntiVirus. http://www.symantec.com.

[5] VirusScan. http://www.mcafee.com.

[6] RealSecure®, http://www.isi.net.

[7] Cisco IDS (formerly NetRanger). http://www.cisco.com.

[8] PC Computer Notes & Online Tutorials: Viruses. http://www.pccomputernotes.com/viruses/-viruses.htm.

[9] Mazu Networks. http://www.mazunetworks.com.

[10] Peakflow DoS by Arbor Networks. http://www.arbornetworks.com.

[11] SolSoft NP, http://www.solsoft.com.

[12] SPECTRUM suite, http://www.aprisma.com.

[13] StormWatch, http://www.okena.com.

[14] T. Oates, *Fault Identification in Computer Networks: A Review and a New Approach*, Technical Report 95-113, University of Massachusetts at Amherst, Computer Science Department. 1995.

[15] L. W. Jones, *ChAD: Change and Anomaly Detection - Discovering when a system is not behaving normally*, August 2001. http://64.81.14.30/ReliabilityWeb/.

[16] Webopedia: Online Dictionary for Computer and Internet Terms. http://www.webopedia.com.

[17] Bill Alderson and J. Scott Haugdahl, "A Broadcast Storm Becomes A Thunderstorm", in Network Computing online magazine. http://www.networkcomputing.com/611/-611alderson.html.