MAST – A Mobile Agent-based Security Tool [1]

Marco Carvalho, Thomas Cowin and Niranjan Suri
Florida Institute for Human and Machine Cognition
40 South Alcaniz. Pensacola, FL 32502, USA

ABSTRACT

One of the chief computer security problems is not the long list of viruses and other potential vulnerabilities, but the vast number of systems that continue to be easy prey, as their system administrators or owners simply aren't able to keep up with all of the available patches, updates, or needed configuration changes in order to protect them from those known vulnerabilities. Even up-to-date systems could become vulnerable to attacks, due to inappropriate configuration or combined used of applications and services.

Our mobile agent-based security tool (MAST) is designed to bridge this gap, and provide automated methods to make sure that all of the systems in a specific domain or network are secured and up-to-date with all patches and updates. The tool is also designed to check systems for misconfigurations that make them vulnerable. Additionally, this user interface is presented in a domain knowledge model known as a Concept Map that provides a continuous learning experience for the system administrator.

Keywords: MAST, Mobile Agents, CmapTools, network security, host security, automated system updates.

1. INTRODUCTION

The growth and globalization of the Internet have made computer and network security a paramount challenge. In 1988, the Internet Worm caused much havoc and effectively shutdown the Internet. However, the systems that were affected were mostly at universities and other research centers. Today, circumstances are very different from 1988. Many businesses rely on networks of systems connected to the Internet. Network attacks result in downtime that cost companies significant time and money due to lost productivity and/or loss of data. Moreover, critical infrastructural systems are being connected to the Internet thereby raising the potential for disruption through network-based attacks. Unfortunately, this same characteristic attracts attackers who wish to make highly-visible statements.

Another key difference between 1988 and today is the sheer number of systems that could come under attack and the ratio of expert system administrators to the number of systems. Administrators are often poorly trained and do not have the tools necessary to properly perform their jobs. Moreover, the workload of administrators prevents them from keeping up with the latest vulnerabilities and security advisories and updates. They often do not have the time to monitor each system in their network for suspicious activity and to make sure that the system is completely secure.

For example, in a distributed denial-of-service attack, the attacker first breaks into a number of geographically and topologically diverse systems to setup launching points for the attack. After securing access to these hosts, the attacker launches a coordinated attack on a target host or network. If the break-ins or attempted break-ins are caught early-on, launching distributed denial-of-service attacks would be much more difficult. This is one small example where the security of individual hosts and networks can be seen to be dependent upon the security of the Internet as a whole.

In recent work, the authors of [1] have estimated the potential threats of wide-spread propagation of worms on the Internet today and how that could effectively give virtually unlimited power to attackers. The understanding of this threat has prompted the authors to suggest the implementation of a Cyber equivalent of the “Center for Disease Control” (CCDC) [1] to monitor, at a national or global level, the propagation and evolution of viruses, worms or other potential security threats.

We believe that a broad effort such as the CCDC proposal is necessary and that it would greatly benefit from a coordinated effort to provide mechanisms to improve the efficiency and levels of training of the security officers and systems administrators of small to medium networks. These types of networks make up the vast majority of networks deployed. By efficiency, we mean the capacity of the systems administrators to quickly interpret the security advisories, understand how they affect their network and precisely determine what systems or sub-networks could be vulnerable. By level of training, we refer to an adequate level of understanding of the conceptual implications of the announced threat. In the complex and highly dynamic networks of today, it is critical to understand how the vulnerabilities and the suggested fixes will interact with the various systems and applications. It is important to provide means for a quick understanding of the inner-workings and exploit mechanisms of the reported threat, so that the threat can be properly addressed and a fix or patch can be correctly applied, in accordance with the local characteristics of the network.

In this paper, we discuss some of the issues that we believe are important for the improvement of overall network security at a local level. We also introduce our research in the MAST project, a mobile agent-based software tool that integrates advanced monitoring and management capabilities with an easy-to-browse knowledge model, semi-autonomously updated

[1] The MAST project is sponsored by the National Science Foundation (NSF) under the Strategic Technologies for the Internet program, award number 0230927.

by security advisories and experts from different domains of expertise.

2. NEW CHALLENGES IN NETWORK SECURITY

There have been repeatedly successful attacks on the systems that reside on the Internet, often exploiting known vulnerabilities and causing significant slowdowns as well as loss of data and productivity. Some of these more prominent attacks - such as Code Red and Nimda against Microsoft's Internet Information Server (IIS) and the MS-SQL Slammer Worm, exploited known weaknesses in these software subsystems for which patches were readily available well before the attacks.

As our systems become more capable and interoperable, they become more complex and more difficult to maintain and manage. Typical system administrators are charged with supporting a large user population with a range of differing requirements. These often involve accessing different central resources or servers, with various departments often relying upon different types of software systems, which may entail different operating system support and specifically tailored server environments. Faced with this potential array of different environments, and attempting to keep up with each user's needs, general security related tasks are often prioritized down the list. This is accompanied by a common adage adopted by the computer cognoscenti - "if it works, don't fix it." This to the fact that often one software patch may have unintended consequences, changing the system subtly and causing something else to stop working, either by affecting a complementary software subsystem, or by uncovering an old bug whose patch wasn't included in the more recently applied update.

While some of this can certainly be addressed through better software engineering practices at these companies upon whom we all rely for crucial business applications, it is a problem that won't ever be totally resolved. Not only is it difficult for the administrator to find the time to apply the necessary patches, these patches are treated with some skepticism, often deservedly so, due to unexpected side effects that can arise due to special-purpose configurations and peculiarities of each network.

In addition, the management in most companies generally under-appreciates the tasks involved in day to day system administration. A large part of the typical system administrator's day is spent in a reactionary mode, responding to various user requests and fighting periodic fires that pop up and require immediate attention. Very little time is left for strategic tasks and such preventive maintenance as software patch and version management. The overriding aim in almost any enterprise is the bottom line and growing the business and often tasks ancillary to this goal are sacrificed in its pursuit. Security comes to the fore only after there has been an incident and, during the postmortem analysis, efforts are made to understand how and why it could have happened and then changes are made to prevent that particular attack from reoccurring.

The large number of services and applications as well as heterogeneity of networks plays a big role in the problem. It is hard to keep up with each particularity and vulnerability of each platform, OS, and application (and their combinations). Even just to understand the actual vulnerability and its implications on the network could take the administrator a long time, researching information and potential side-effects that could occur under specific network configurations.

All these issues end up resulting in a large number of systems and networks that simply fall behind the reported vulnerabilities and remain, usually for a long time, susceptible to potential attacks. The problem takes on huge proportions when a fairly large number of these networks are compromised and used in coordination to launch massive attacks on other critical systems.

Overall network security thus depends on addressing these problems at the local level on small to medium networks. We have identified three main points of paramount importance for the system administrators to achieve higher levels of effective security:

a) Timely access to accurate information about new threats and vulnerabilities for different platforms, systems, and applications.

b) The capacity to conceptually assimilate the implications of the reported vulnerability. That is, to classify the severity of the exploit within the context of his or her network, and the side effects that the vulnerability (or even the suggested patch for fixing it) could cause in the system.

c) The capacity to quickly identify which hosts on the network are susceptible to the vulnerability. In business and corporate networks for example, the hosts and even servers will often be modified or reconfigured at some point without direct knowledge of the systems administrator. It is imperative that when potential threats arise, the administrator is able to accurately and efficiently map the network and detect hosts that could potentially be affected by the reported threat.

In a broader picture, the problem must be solved by providing to the system administrators the capability to (1) quickly and effortlessly assess the level of risk posed in their environment by each new threat or vulnerability, and (2) take steps to either eliminate or reduce that risk. We propose to do this by providing software applications that would increase efficiency and access to system information and security-related knowledge bases, and provide a toolset to apply patches or make system level modifications to protect against a given type of threat.

In this paper, we introduce our project for the design and implementation of a security tool that serves two purposes: (a) help system administrators find vulnerabilities and detect possible attacks on systems and networks and (b) teach students and novice administrators about security concerns. In the first use case, administrators will be able to use the tool to check for known vulnerabilities on their systems, to check whether the latest security updates and patches have been installed, and to monitor their systems for different forms of attacks or break-ins. In the second use case, the tool can be used in a classroom, laboratory, or in the field to help students and novice administrators learn about the security domain, the latest known vulnerabilities, and how to fix them.

3. THE MAST APPROACH

In the MAST approach we advocate the development of an integrated software tool that would provide both the flexibility for powerful management of network security as well as meaningful learning and understanding of security related issues and their implications.

The tool provides a flexible way to update and navigate extensive knowledge bases from which associated active components can be used for simple exploration, monitoring, and management of remote hosts. The knowledge base is represented through concept maps. We have chosen to use software agents for the remote monitoring and management of hosts.

Concept maps

Concept maps are tools for organizing and representing knowledge. They have been widely used in the many different domains since their inception in the 70's by Novak [2].

Concept maps are basically characterized by a graphical representation of a set of concepts and their relationships, providing a strong and concise description of the specific domain of knowledge. The concepts are hierarchically arranged and linked to created propositions, which are usually referred to as semantic units or units of meaning [3]. Figure 1 shows an example of a concept map. The propositions in a map provide a meaningful description of the relationship between two concepts. The concepts, perceived as regularities in events and objects, or records of events and objects [3] are associated to each other from top down on the map, where the more general concepts are on the top of the map and the more specific concepts located down at the bottom or at the leaves.

Concept maps can be used as powerful tools for elicitation and representation of an expert's understanding of a domain and as effective browsers, providing easy and quick navigation of the knowledge model by both experts and novices. A very important characteristic of concept maps, which is highly applicable to this work, is the notion of cross-links. These links build propositions across different branches of the concept map, indicating relationships across different sub-domains of the map.

More recently, software tools such as CmapTools [4][5] have been designed to facilitate the online manipulation of maps and extend their capabilities. CmapTools allows maps to be linked with each other through common or related concepts. This creates a multi-layered set of maps, providing an extensive representation of the domain. Repositories of concept maps (referred to as knowledge models) can be created and shared over the Internet.

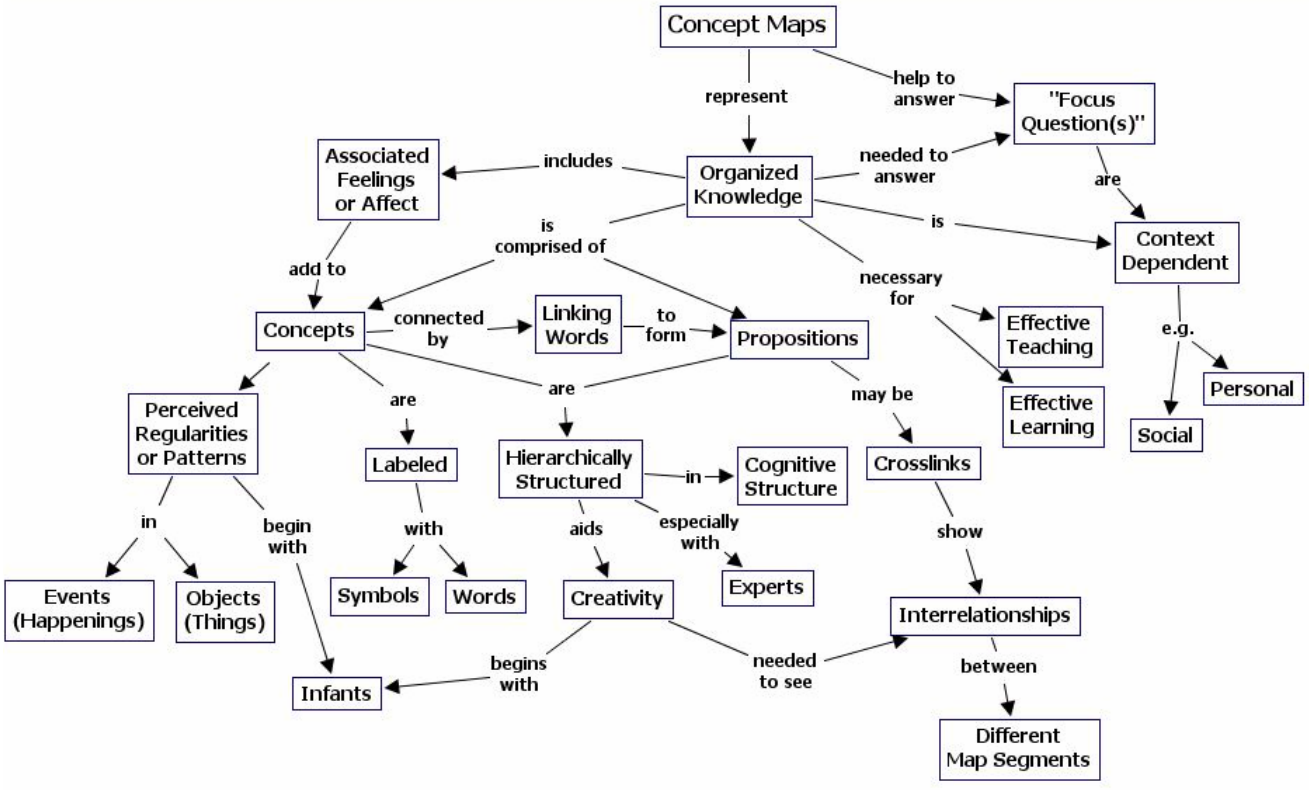

Fig. 1 – Example of a concept map, extracted from reference [3]

Among many other features, CmapTools creates an environment that greatly facilitates and enhances the browsing and construction of concept maps in collaboration with other users. Synchronous collaboration, Case-Based Reasoning search of related maps [6] and automatic search of the web for

concepts and documents related to a map [7][8][9] are some of the many valuable capabilities added to CMapTools, that allow a continuous evolution of the maps. By allowing the user to easily find related maps and other information, the tools offer an environment where the represented knowledge can be easily adapted and complemented to incorporate new trends or concepts. The tools also provide means to insert or link a variety of resources in the maps. Small icons attached to a concept represent different resources that can be accessed directly from the map, as shown in figure 2

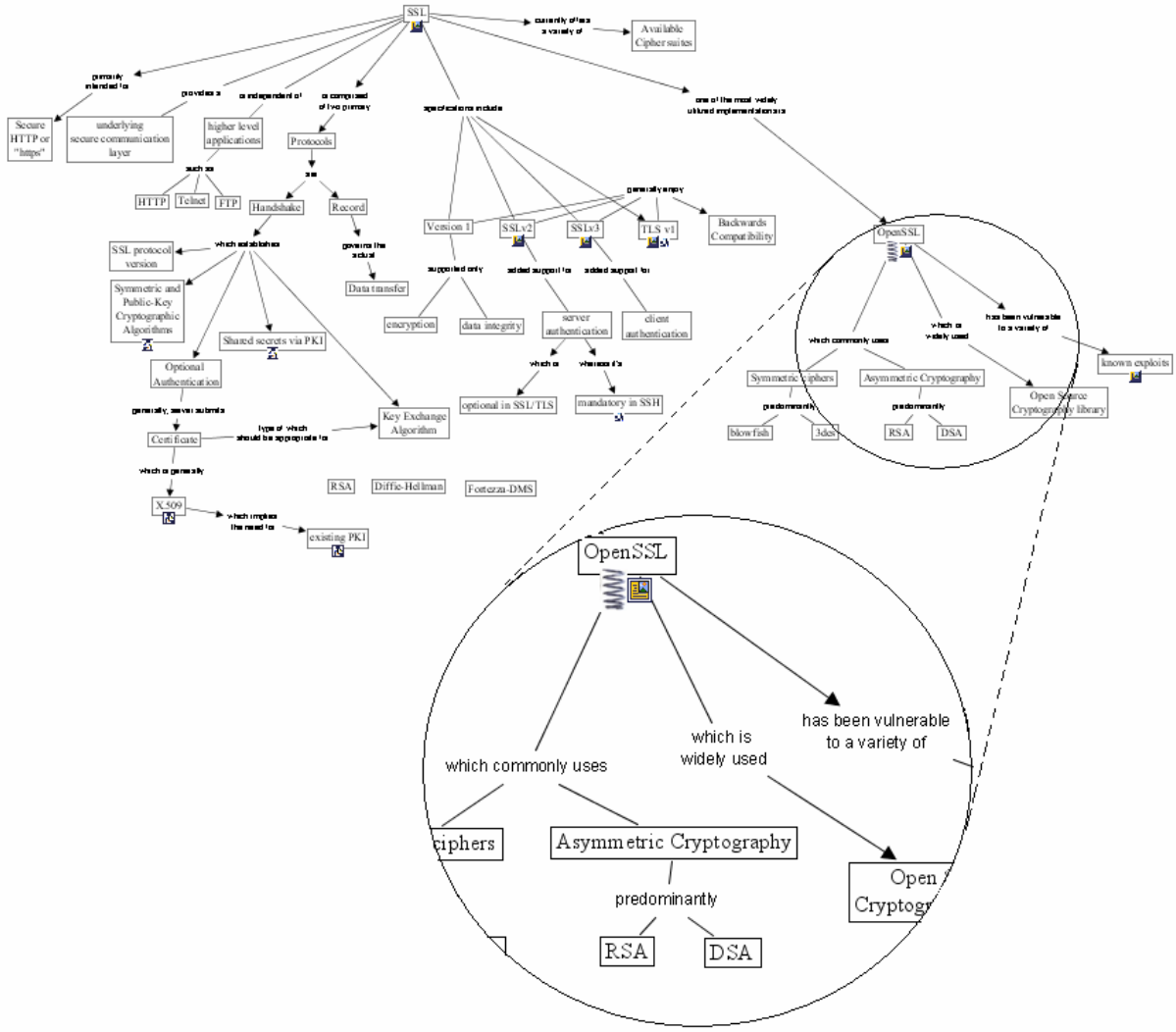

Fig. 2 – Exploratory mobile-agent attached to "OpenSSL" concept in CmapTools

Mobile agents

Mobile Agents are programs that, with varying degrees of autonomy, can move between hosts across a network. It combines the notions of mobile computation and mobile code.

Mobile computation allows a process to execute on a remote host with no dependencies on the originating host. That is, mobile computation extends remote computation (such as what we find in remote procedure calls) to allow disconnected operation. Code mobility builds upon the notion of deploying code to a remote host to enable, on demand, capabilities on that machine that didn't exist before. This is a very important feature for the MAST tools, since code to check for specialized vulnerabilities can be written and deployed for execution on every host of the network as needed. This capability associated with the capacity to maintain state across systems (characteristic of mobile agents) allows the creation of highly sophisticated checks and comparisons between hosts.

The use of mobile agents for network monitoring and management could be seen as analogous to a security guard on a building. As security guards, specialized mobile agents would proactively roam through the network, evaluating system binaries, configurations, and possible vulnerabilities.

Mobile agents can check the system from inside, by directly accessing files and calling upon local OS resources and, if necessary, move back to the administrator's console (or to a trusted peer host) to perform external probing tests. This approach allows for the development of highly customized and complex checks of systems and applications.

MAST allows the association of special-purpose software agents directly into the knowledge model. Agents can be automatically launched by the tool, or by the system administrator, through the administrator's console. Once the agents move to the systems, they will operate locally where they will be able to perform local monitoring and maintenance tasks. They will be able to check the status of the system, fix the configuration, and update the system. Other types of security agents will live on a target system continuously, periodically monitoring the system for anomalous behavior.

The mobile code capability will be used to dispatch updated security agents to systems. These updated agents can carry with them new security checks for the latest vulnerabilities, as well as knowledge about new updates and configuration changes. Exploiting mobile code will ensure that the security agents themselves are always up to date.

MAST relies on a modified version of the NOMADS [10] agent system. NOMADS is a mobile agent system for Java-based agents. NOMADS provides two implementations: Oasis and Spring. Oasis incorporates a custom Java-compatible Virtual Machine (named Aroma) whereas Spring is a pure Java implementation. The Aroma VM is a clean-room VM designed to provide the enhanced capabilities of execution state capture and resource control [11][12]. Building on top of the capabilities of Aroma, Oasis provides three capabilities:

- strong mobility where agents can move while preserving their execution state
- forced mobility where, completely transparent to them, agents may be moved from one system to another by an external asynchronous request
- secure execution of untrusted agents while protecting the host from denial of service and other forms of attack

The Spring implementation is fully interoperable with Oasis but does not provide the above features. Spring is well-suited for lightweight applications or environments that do not support Aroma (e.g., mobile phones or PocketPC). [13].

The NOMADS Agent system will provide the primary underlying infrastructure for the MAST project. The security kernel (explained in the next section) leverages from the Oasis implementation of the NOMADS system, hardened for security.

All the operations performed in the remote hosts will be implemented in the form of mobile agents and will be deployed through the NOMADS infrastructure.

4. THE MAST COMPONENTS

These two main technologies, CmapTools and the NOMADS Agent system are integrated in MAST through a set of components that realize the requirements for the system.

There are essentially four main components in MAST:

- *An administrator's console* that is the main user-interface component
- *A security kernel* that is installed on hosts (and possibly routers) and handles the execution of the security agents
- *A knowledge model* about security built using concept maps that is constantly updated as needed
- *A set of security agents* that are part of the knowledge model that travel to hosts to perform security-related tasks

The Administrator's console is the primary graphical interface that integrates all the remaining components.

The security kernel is the component responsible for remote execution and control of the security agents deployed in the platform. Besides providing a low level API to allow the agents to have access to the system, the security kernel will also be responsible for providing the necessary encryption and authentication services between the agents and the management console. The kernel will be based on the current execution environments of NOMADS (Oasis and Spring) to combine the advantages of high portability and extended features offered by NOMADS. Resource control and policies will regulate how the security and monitoring agents will execute in the remote hosts to avoid the compromise of the common activities of the hosts.

The security kernel will be mutually authenticated with the administrator's console to create a secure channel for communication and for the transport of agents. This approach secures remote hosts against malicious code being pushed to the security kernels.

Agent mobility is always single-hop, that is, the agent always moves between the administrative console to the kernels and back, never between kernels. This approach avoids the problem that a compromised host could modify the kernel to attack legitimate agents and spread itself through the network.

At the administrator's console, the integrity of every agent is checked upon return. This way, a compromised kernel might become un-trusted to the administrator but cannot propagate malicious code to other hosts in the network.

The Knowledge Model will be composed of a set of concept maps, populated with resources (such as reports, security advisories, and mobile agents). The agents, depending on the access level of the user, will be either exploratory agents, normally used for training or instruction purposes, or actual monitoring or correction agents that will search for specific vulnerabilities and apply the appropriate fixes for it.

5. CASE STUDY

We have begun development of the MAST Tool by focusing first on cryptography and in particular on the SSL [14] and SSH [15] applications that form the basis for existing efforts to secure Internet based communications. These may involve web browser, electronic mail, interactive session, or some other type of network traffic. These are all potentially vulnerable to network eavesdropping and are typically utilized when transferring sensitive data. In the past few years, there have been a variety of threats or vulnerabilities identified against some of the commonly used implementations such as OpenSSL [16] and OpenSSH [15]. These applications are typically

installed and configured by the system administrator and involve a moderate learning curve and initial familiarization period. As new vulnerabilities are discovered and the user community is apprised of these, there will typically be a time lag between the time that the vulnerability is identified and the time that a patch is applied.

The more involved the patching process is, the less chance of it being applied in a timely manner, as system administrators often are in a reactionary mode and struggling to balance all of their competing tasks and user requests. Stretches of uninterrupted time for careful patch application are typically few and far between.

With the use of the MAST tool, the vast majority of this effort would be offloaded: the sysadmin would have the ability to launch an agent whose task it is to sort through all new vulnerability/security announcements to see if there are any that might affect systems on his or her network. Such an agent could potentially be scheduled as a recurring task so that the results are ready for the sysadmin upon login, or sent via email or made available through a web page.

In the event that there are some potential problems identified, the administrator can probe those potentially affected systems to see if they are indeed vulnerable. A specific agent would be dispatched to those identified hosts, where it would query the host for the specific weakness or package version that contains the weakness and then, upon return to the MAST Console, present the information to the sysadmin. An agent capable of patching the given system may or may not be readily available, depending upon the level of specificity required in the fix. We will have agents that are capable of disabling specific ports, or specific services or server processes (daemons) on most operating system environments. There will be agents that are capable of downloading and installing new versions of readily available precompiled software packages, such as Linux RPM's or FreeBSD packages. There will also be agents to fit the situation where there are just two discrete stages: retrieve patch file and apply patch. In cases where it is more complex, agents may need to be tailored to fit the sequence of steps required, and then made available to the users of the MAST system. Assuming an appropriate agent is readily available, the sysadmin will be able to specify the system and package he/she would like upgraded, or patch to be installed, and then dispatch the agent. Errors or feedback encountered during the installation of patches or packages would be carried back by the agent to the MAST console and presented to the sysadmin by the agent.

In our first case study, we have populated our domain model in the area of cryptography and implemented an agent to check the current version of OpenSSL. Notice the Spring like icon underneath the OpenSSL concept in Figure 2. This indicates that an active agent is available to launch to perform some function related to OpenSSL. We are in the process of implementing an agent to update OpenSSL to the latest version if it is out of date.

6. CHALLENGES AND FUTURE WORK

The MAST tool is partially implemented. The security kernel is being designed from the current implementations of the NOMADS Agent System, while the Administrator's console is being designed as the integration element between the two main components CmapTools and NOMADS.

Some of the main challenges that are still being researched by this group are:

a) One of the most complicated security matters in mobile agent systems is associated with protecting the agent against a compromised host. The MAST approach uses single-hop mobility to avoid the propagation of malicious code through the network but it still relies on information provided by the kernel to monitor and manage the system. As part of our future work we intend to evaluate possibilities that could allow the administrator to determine that a specific kernel could be compromised.

b) Yet another issue that must be addressed is in regards to ongoing population and evolution of the knowledge models. The system is designed to allow collaboration on the development of the knowledge models, allowing users from different levels of expertise to add or modify information on the knowledge models. We feel this is critical to developing a robust knowledge model of the security domain, and will strengthen the tool, by continuously stretching the model to cover previously unanticipated security problems. It is important to establish access control mechanisms that would allow verification of knowledge updates.

c) Resource control is yet another issue. Agents deployed to a remote host must operate and perform security checks (or updates) without disrupting normal work on the machine. The NOMADS agent system allows for fine-grained control of resource utilization on an agent basis. This feature will ensure that the resource utilization of the security agents can be limited so as to not interfere with the regular operation of the systems. Under special circumstances though, such as an ongoing attack, system-wide policies would override these restrictions and allow high-priority execution of the contingency agents.

d) Access to routers, switches and firewalls is limited to the capabilities of code execution on these devices. The NOMADS agent system is very flexible and provides different types of execution environments to run on a variety of platforms, and still maintain compatibility with the overall agent system [13]. For devices where the security kernel cannot be installed, monitoring will potentially be limited only to external probing and traffic analysis.

7. CONCLUSIONS

The MAST project is in the first year of funding and at this time we have finished integration of the CmapTools and the NOMADS mobile agent systems. We have three ongoing efforts in parallel: a) the final development of the security kernel and its authentication mechanisms with the administrator's console, b) the construction of the knowledge models and c) the automatic information retrieval engines that will help in maintaining the knowledge maps and keeping them current.

Of these efforts, the construction of the knowledge models and the update engines will probably be the most extensive tasks. Sets of standard security agents will be made available for free download and it is expected that customized agents will be developed and shared between security experts in different communities. While the MAST tools and access to the public knowledge models will be open to the general public, we expect that modifications and extensions on the knowledge base will have to be moderated by some group of experts using a voting mechanism. Much like the envisioned "Cyber Center for Disease Control," MAST will provide a repository for advanced security checking agents and a public forum where general security-related knowledge can be easily browsed, modeled, and shared amongst peers and interested parties.

The process of eliciting knowledge and constructing the maps will be crucial to the success of the tools and we will rely on CmapTools capabilities to facilitate the process for collaborative construction of these bases of knowledge that will support the daily work and the training of current and future network administrators.

More information about this research is available online at http://mast.ihmc.us and the CmapTools can be freely downloaded for non-profit use from http://cmap.ihmc.us

8. REFERENCES

[1] Stanifor, S., Paxson, V., Weaver, N. – *How to Own the Internet in Your Spare Time.* – Proceedings of the 11th USENIX Security Symposium. San Fransicso, CA. August 2002.

[2] Novak, D.B. Gowin, *Learning How to Learn*. Cambridge University Press (1984)

[3] Novak, J.D., *The Theory Underlying Concept Maps and How to Construct Them*. Cornell University.

[4] Cañas, A. J., J. Coffey, T. Reichherzer, N. Suri, R. Carff, G. Hill. *El-Tech: A Performance Support System with Embedded Training for Electronics Technicians*, Proceedings of the Eleventh Florida Artificial Intelligence Research Symposium, Sanibel Island, Florida, (May 1997)

[5] Cañas, A. J., K. M. Ford, J. D. Novak, P. Hayes, T. Reichherzer, N. Suri., *Using Using Concept Maps with Technology to Enhance Collaborative Learning in Latin America*, accepted for publication, Science Teacher.

[6] Canas, A., Leak, D.B., Maguitman, A., *Combining Concept Mapping with CBR: Towards Experience-Based Support for Knowledge Modeling*. AAAI 2001.

[7] Carvalho, M. Hewett, R. Canas, A. *Enhancing Web Searches from Concept Map-based knowledge Models.* SCI – World Multiconference on Systemics, Cybernetics and Informatics. July, 2001.

[8] Canas,A.J., Hewett, R., Carvalho,M., Carnot, M.J., *Knowledge Models for Organizing Search Information.* SSGRR - Advances in Infrastructure for Electronic Business, Science, and Education on the Internet. Aug, 2001

[9] Canas, A.J., Carvalho, M., Arguedas, M.. *Mining the Web to Suggest Concepts During Concept Mappinp: Preliminary Results.* XIII Simposio Brasileiro de Informatica na Edicuacao, - Porto Alegre, Brasil (Nov, 2002)

[10] Suri, N., Bradshaw, J.M., Breedy, M.R., Groth, P.T., Hill, G.A., Jeffers, R., and Mitrovich, T.S. *An Overview of the NOMADS Mobile Agent System*. Sixth ECOOP Workshop on Mobile Object Systems. (http://cui.unige.ch/~ecoopws/ws00)

[11] Suri, N. Bradshaw, J.M., Breedy, M.R., Ford, K.M., Groth, T., Hill, G.A., and Saavedra, R.: "*State Capture and Resource Control for Java: The Design and Implementation of the Aroma Virtual Machine*." White Paper. http://nomads.coginst.uwf.edu

[12] Suri, N., Bradshaw, J.M., Breedy, M.R., Groth, P.T., Hill, G.A., and Jeffers, R. *Strong Mobility and Fine-Grained Resource Control in NOMADS*. Proceedings of the 2nd International Symposium on Agents Systems and Applications and the 4th International Symposium on Mobile Agents (ASA/MA 2000). Springer-Verlag.

[13] Suri, N, Carvalho, M., Bradshaw, R., Bradshaw, J.. *Small Mobile Agent Platforms.* Autonomous Agents & MultiAgent Systems, Workshop on Ubiquitous Agents on Embedded, Wearable and Mobile Decives. July, 2002

[14] Freier, Alan O., Karlton, Philip, Kocher, Paul C. – *The SSL Protocol Version 3.0 Internet Draft*. Available from http://wp.netscape.com/eng/ssl3 Netscape. November 1996.

[15] Ylonen, T. – *The SSH Remote Login Protocol. Internet Draft.* (http://www.snailbook.com/docs/protocol-1.5.txt), Nov/1995

[16] *The OpenSSL Project*. Information and Software downloads available from http://www.openssl.org