Proceedings of the 5th WSEAS Int. Conf. on APPLIED INFORMATICS and COMMUNICATIONS, Malta, September 15-17, 2005 (pp245-250)

A platform independent approach for Mobile Agents to monitor Network Vulnerabilities

GHULAM ALI MALLAH, DR. ZUBAIR A. SHAIKH
National University of Computer & Emerging Sciences
Shah Latif Town, National Highway, Karachi
PAKISTAN
http://www.nu.edu.pk

Abstract: - Mobile Agents are an effective choice for many research and application areas due to several reasons, including improvements in latency and bandwidth of client-server applications, reducing network load and threat assessment. Intrusion Detection Systems and Vulnerability assessment systems are used to monitor network traffic, to measure and prioritize the risks associated with network and host based systems. These systems monitor suspicious activities and alert the system or network administrator. All inbound and outbound traffic can be monitored either in whole network or individual host.

Amalgamating two technologies, i.e. Mobile agents and Network security systems will provide many benefits to the administrators where agents will autonomously roam and assess the network and the system.

Though many Agent models are available to provide agent management services, but they are mostly server/platform dependent. These models may fail when intended host is to be targeted for its security assessment and the supporting compatible environment, i.e. server, is not available there.

This paper surveys some server dependent agent models, implements and tests its results. It also points out the flaws of different server dependent agent models. We also propose and implement a Server-independent Agent Architecture to monitor intrusion detection and check the weaknesses of target host that normally attacker exploits to harm the network.

Keywords: - Mobile Agents, Platform independence, Web services, Computer Network, Agent Models.

1 Introduction & background study

The central organizing principle of today's computer communication networks, Remote Procedure Calling (RPC), was conceived in the 1970's and viewed computer to computer communication as enabling one computer to call procedures in another. Two computers whose communication follows the RPC paradigm agree in advance upon the effects of each remotely accessible procedure and the types of its arguments and results. Their agreements constitute a protocol. [1] Each call involves a request send from a user to a server and a response sent from the server to the user.

RPC essentially requires that each interaction between the user computer and the server consist of two acts of communication: one to ask the server to perform a procedure and another to acknowledge that the server did so.

An alternative to RPC is Remote Programming (RP). The RP paradigm view computer to computer communication as enabling one computer not only to call procedures in another, but also to supply the procedures to be performed. Two computers communicating with the RP paradigm agree in advance upon the instructions that are allowed in a procedure and the types of data that are allowed in its state.

Their agreements constitute a language. The language includes instructions that allow the procedure to make decisions, examine, and modify its state, and, importantly call procedures provided by the receiving computer. Such procedure calls are local rather than remote. The procedure and its state are termed as Mobile Agent to emphasize that they represent the sending computer even while they reside at and operate in the receiving computer. [2]

Mobile Agent (MA) mechanism has the following benefits that are advantageous over traditional RPC:

Communication efficiency: Instead of back-and-forth communication between distributed processes, the only remote communication in MA systems is agent mobility, which is expected to generate less network traffic by reducing the number of interactions and the amount of transmitted data. This advantage is especially significant when intensive remote communication would be present, such as information querying and analysis.

Distributed process control simplicity: Distributed process communication, synchronization, scheduling and resource sharing that are much cumbersome and require sophisticated algorithms in distributed systems now can go back to the mature techniques for centralized system.

Network fault tolerance: Advantageous in situations of slow unreliable network Connection (such as wireless) and mobile users. MA does not require continuous network connection. It relies on servers when available but functions autonomously if needed.

Higher concurrency and asynchrony: More than one agent can be created and sent to different places for a divisible task, so that concurrent asynchronous computation is well supported.

Intelligence: The mobility of agents with intelligence such as learning ability gives application programmers simpler, clearer and more object-oriented view to system logic and flexibility to system design.

Ability for heterogeneity: Expected to be more efficient for computations in heterogeneous environments. [3]

2 Security threats

Computer network environment is being rapidly changed. New programs, services, and architectures are being introduced.

On the other hand, the rapid increase in attacks on computer systems has made security concerns increasingly important in academic, corporate, and government networks. The ability to constantly monitor an organization's networks for both old and new vulnerabilities is critical to secure a system before it is attacked. [4]

The discouraging tasks that security professionals are facing with are how to effectively and efficiently resolve these security threats. Attacker always exploits the weaknesses of the system. That's why to design and implement an effective and efficient security policy Vulnerability assessment is an important element. [5]

Security threats fall into three major categories that include Vendor-supplied software such as bugs, missing operating system patches, vulnerable services, and insecure choices for default configurations; Network Administration such as insecure requirements for minimum password length or unauthorized changes to the system configuration; and third and the final category is User activity such as sharing directories to unauthorized parties, policy avoidance such as failure to run virus scanning software and other, more malicious, activities. These types of risks can be present in and apply to network services, architecture, operating systems, and applications.

3 Mobile Agents

Mobile Agents are the programs that move between computers, autonomously trying to fulfill some specific goals given by users. Agents are different from other applications in that they are goal-oriented: they represent users and act on their behalf to achieve some set goals in an autonomous manner – i.e. they control themselves, as in the decision where and when they will move to the next computer. Mobile Agents do provide a viable means of performing network security assessment and analysis efficiently and effectively.

The concept of a mobile agent sprang from a critical examination how computers have communicated since the late 1970s. Prompted by the difficulty of the Internet's then-current architecture to match the pace of the exponential growth of its users, a new approach was needed that would satisfy two seemingly contradictory needs: increasing the sophistication of the possible communication types without strangling the available bandwidth of the Internet's weaker components. [6]

Mobile Agent technology has been a very proficient research topic for some years now. Use of mobile agents in sophisticated applications offers an enriching advantage for constructing flexible and adaptable distributed wide-area systems. Indeed, as they can be retracted, dispatched, cloned or put in stand-by. Mobile Agents have the ability to sense network conditions, and to load dynamically new functionalities into a remote node [7].

3.1 Why Agent platform?

Agents are pieces of code that are able to perform complex, autonomous operations, and become truly powerful when they are mobile, meaning that they can travel from one machine or computer network to another. Agents must be both safe from being harmed and incapable of doing any harm to them before they are even sent out into the world. [8, 9]

They are pieces of code where a computer environment is needed for them to operate. This environment is provided by the habitat. The habitat is a software system or server running on a computer, and can also be thought of as the runtime environment. It provides the agents with the functionality needed for code execution, mobility, service management, communication and much more. Another analogy for the habitat is that it acts as an Agent Operating System (AOS). Without it, agents would not be able to exist. One or more habitats can run per machine and configure them individually.

Removing this environment would categorize them into a Virus; an Environment can be configured to stop the agents from harming the workstation.

3.2 Mobile Agents in Networking

Mobile agents offer several potential advantages that may overcome limitations that exist in static, centralized components:

Reducing Network Load: Instead of sending huge amount of data to the data processing unit, it might be simpler to move the processing algorithm (i.e. agent) to the data.

Overcoming Network Latency: When agents operate directly on the host where an action has to be initiated, they can respond faster than the tree based systems that have to communicate with a central coordinator located elsewhere on the network.

Autonomous Execution - When portions of the tree based systems get destroyed or separated, it is important for the other components to remain functional. Independent mobile agents can still act and do useful work when their creating platform is unreachable which increases the fault-tolerance of the overall system.

Heterogeneous Environment: The agent platform allows agents to travel in a heterogeneous environment and inserts an OS independent layer.

Dynamic Adoption: The mobility of the agents can be used to reconfigure the system at run-time by having special agents move to a location where an attack currently takes place to collect additional data.

Scalability - when distributed mobile agents replace a central processing unit, the computational load is divided between different machines and the network load is reduced. This enhances scalability and additionally supports fault-resistant behavior [10].

4 Proposed Model

We intend to amalgamate two models: Mobile agent technology and Security systems. A distributed security system is suggested that will be able to monitor, detect intrusions and respond accordingly.

This architecture will provide a fault tolerant solution. If a workstation had been compromised, the workstation would be temporarily isolated from the network thus limiting any further damage to the network.

We tried using existing Agent toolkits to design and implement our architecture to cope with all kind of security threats. Alternative approach would be to create our own Agent toolkit and then work on problem in hand. [11]

4. 1 Survey & implementations of different Agent models

The first major task was to choose the right technology. Many of the technologies were checked like Agent Development Kit (ADK), JADE and Aglet Software Development Kit (IBM).

These are well-known available platforms. The above mentioned technologies provide a platform for Agent development. ADK is still in its development phase and has lot of problems regarding agent movement which is the core essence of our research. JADE is FIPA compliant Agent development framework that's why it does not provide more standards for agents' mobility. Though most of the platforms have their own features and limitations but keeping in view the key property of our research, i.e. mobility, we chose ASDK. Aglet Software Development Kit is being widely tested, used and verified for Aglet development. It provided us with the control we needed to develop a low level application monitoring the OSI Layers. Mobility of the agents is the core essence of the research. Once agents roam around, the next task will be the security measures.

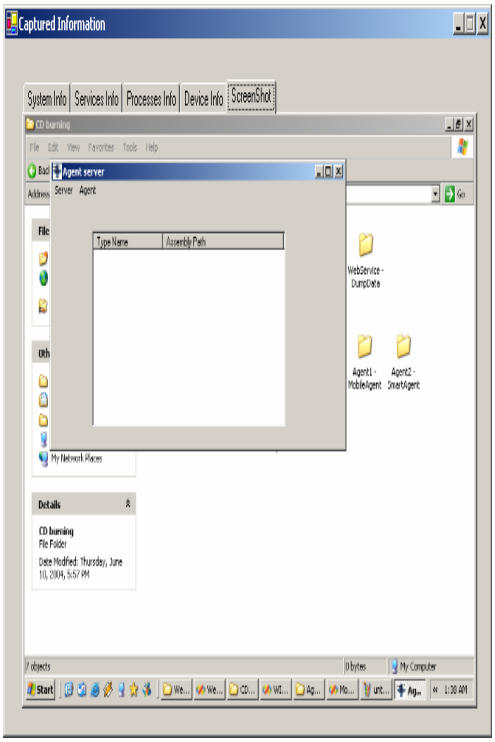

Fig.1: *Agent server execution for agent services management.*

4.2 Monitoring & Filtering

Once Agents' mobility tested, this will be easier to monitor remote workstations. The created mobile agents will target the destination machine. The mobile agent that goes to the target host is capable to capture information about the running processes, resources there were being used, open ports and system status information. On a remote system, our agent is able to capture, monitor and filter incoming packets into that system.

4.3 Problems of Platform dependency

In previous section we observed that when an agent migrated into another system, it needed a client on the host machine to receive it. If the client were to be compromised on the host machine, it would render the agent helpless to provide any sort of security.

This led to change in plans in choice of our Agent Toolkit. After further research we discovered that none of the Agent toolkits available were matching our requirements, i.e. to target the host and monitor and filter packets even if agent server is not installed there. These toolkits are effective in other scenarios but not in the proposed one.

4.4 Platform-independent Architecture

After studying various Agent toolkits, their flaws and our requirements, we developed Agent architecture, flexible enough to incorporate Security modules into it.

Major components of our architecture include *Master Agent, Slave Agent, Web Service and Web Client.*
Our Master Agent would be an intelligent agent, able to make decision and dispatch different types of Slave Agent. Slave Agent would respond to its duties for example monitoring a particular workstation. It would then email the collected information to the administrator's account in XML format making the analysis of data easier and would also send the information back to its Master.

Our Master Agent would have embedded Web Client in it; the information it receives from its Slave would then be dumped onto a Web Server through the help of our Web Service and Web Client. This would complete the monitoring part. The next step would be to analyze the data collected and act accordingly.

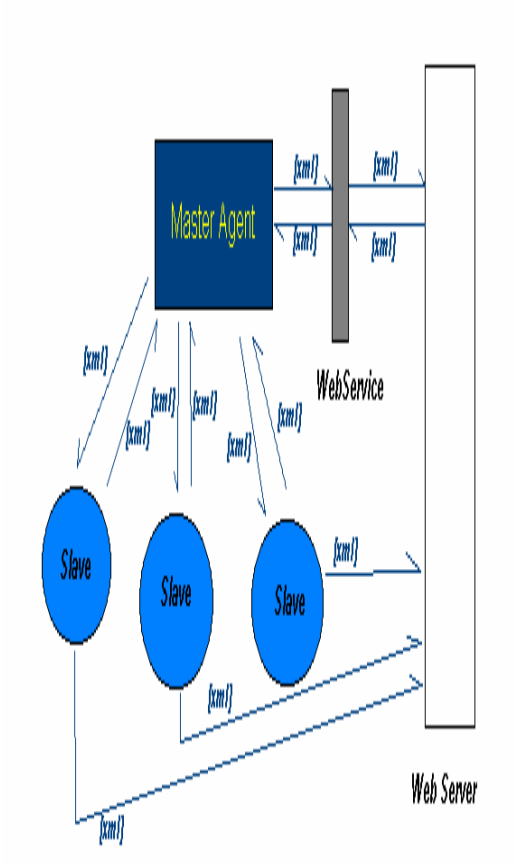

Fig.2: *Agent Toolkit in .NET Framework*

Working on current Agent toolkits, designing our own Agent architecture led us to believe that we could develop our own agent toolkit in the .NET Framework. Keeping our architecture in mind, we developed a primitive Agent Toolkit with a Stealth Client.

We chose this platform because it is truly platform independent, provides immense support for all sorts of new technology for example Web services, mobile applications, XML support etc.

5 Discussion & Future work

Existing systems have their limitations namely, flexibility, autonomy, adaptability and distribution. While there are several areas of work presented here that require further investigation. Two interested areas are still there to be concentrated: Firstly, we would like to assess the performance of our proposed solution in a large uncontrolled network, because so far all our testing has been in a

controlled laboratory. Secondly, we would like to develop more mobile agents that are more application specific and which deal with increasingly complex resources that are defined using XML.

6 Conclusion

In fact, the autonomy given to the agent reduces considerably the implication of the security manager in security management and makes its administration easier. Autonomous roaming of Mobile agents on network will periodically report about all vulnerabilities. A new architecture using intelligent mobile agents is outlined where traditional heavy agent server will not be needed. Here we are in position to say that mobile agents do provide a viable means of performing network security analysis as well as some other complex tasks. The next step would be to analyze the data collected and act accordingly.

7 References

[1] Jai Sundar Balasubramaniyan, Ganesh Krishnan, Eugene Spafford, Karyl Stein, Aurobindo Sundaram, *Software Agents for Intrusion Detection*, COAST Laboratory Technical Report, Department of Computer Sciences, Purdue University, May 15, 1997.

[2] J. Pikoulas, M. Mannion, W. Buchanan , 7th IEEE International Conference and Workshop on the Engineering of Computer Based Systems April 03-07, 2000 Edinburgh, Scotland.

[3] Berbers, Yolande, B.De Decker, and W. Joosen. "Infrastructure for Mobile Agents." 1996.

[4] Stallings, W., *Network Security Essentials: Applications and Standards*, Prentice Hall, Upper Saddle River, New Jersey, 1999.

[5] Christopher Kr¨ ugel, Thomas Toth, Applying Mobile Agent Technology to Intrusion Detection.

[6] Robby Fussel, Vulnerability Assessment: Network versus Host based, GSEC Option 1, December 2002.

[7] Barrus, J. and N. Rowe, "A Distributed Autonomous-Agent".

[8] J. Vitek, M. Serrano, and D. Thanols, "Security and Communication in Mobile Object Systems".

[9] Tony Bradley, Network Security, http://netsecurity.about.com/hackertools/a/aa030504_p.htm

[10] B. Blakley, "The Emperor's Old Armor," *Proc. New Security Paradigms Wksp.*, 1996.

[11] C. Kaufman, R. Perlman, and M. Speciner, *Network Security: Private Communication in a Public World*, Englewood Cliffs, NJ: Prentice Hall PTR, 1995.