ANTI-SNIFFING IN UNBOUNDED NETWORKS

Xiu-guo Bao
Dept. of Computer Harbin Institute of Technology
No.92, West Da-Zhi Street, Harbin, Heilongjiang, China,150001,

Ming-zheng Hu
Dept. of Computer Harbin Institute of Technology
No.92, West Da-Zhi Street, Harbin, Heilongjiang, China,150001,

Xiao-chun Yun
Dept. of Computer Harbin Institute of Technology
No.92, West Da-Zhi Street, Harbin, Heilongjiang, China,150001,

Ri Ding
Dept. of Computer Harbin Institute of Technology
No.92, West Da-Zhi Street, Harbin, Heilongjiang, China,150001,

ABSTRACT

Along with the trend in networked computing environments towards largely unbounded network infrastructures, the traditional anti-sniffing issue meets a great challenge. This paper presents a distributed anti-sniffing approach in which varied anti-sniffing methods including protection methods, detection methods and countermeasure methods are integrated smoothly. This idea is also valuable for solving other security problems of unbounded networks.

KEYWORDS

Anti-sniffing, Unbounded network, XML, Message, Policy

1. UNBOUNDED NETWORKS

Most security technologies derive from a fortress model in which there is a clear distinction between trusted insiders and other potential users and intruders (Bob Blakley, 1996). In the fortress model some aspects about the nature and structure of networks are assumed. Generally, these include assumptions that systems are closed, that they are under central administrative control, and that administrators have the ability to observe any given activity within the system. These assumptions may have been appropriate when networks systems were isolated to the environment and has clear boundary. Today, however, networks systems are open, and no one person or organization has full administrative control. Observers, whether they are inside or outside the system, have only limited visibility into the structure, extent, or topology of the system. Currently, these kinds of systems are called unbounded systems or unbounded networks (Ellison, 1999). It is believed that solving security problems associated with unbounded networks need new angles of views since the assumptions based on the fortress model have destroyed.

Packet sniffing is a method of tapping each packet as it flows across the network; i.e., it is a technique in which a user sniffs data belonging to other users of the network. Packet sniffers can operate as an administrative tool or for malicious purposes. It depends on the user's intent (Ansari, 2003). Crackers install sniffers to obtain usernames, passwords, credit card numbers, personal information, and other information

that could be damaging to a person, a corporation or even a nation. Due to this, sniffing attack has been one of the most popular and easiest ways to penetrate a computer system.

Anti-sniffing research deals with all aspects to prevent, detect or countermeasure sniffing activities to networks. The sniffing prevention aspect deals with repelling attacks methods through mechanisms like firewalls or encryption blocks. The sniffing detection aspect includes techniques like intrusion detection or system logging. The sniffing countermeasure aspect concerns restoration of compromised system data or replacing system binaries, and also concerns system adaptation through system patches or intrusion signature timely updates to enhance system survivability. Although many anti-sniffing methods have been proposed in network and information security research society (Graham,2000), each of them is based on some assumptions, and most of these assumptions are derived from the fortress model. In the situation of unbounded networks, some assumptions may be unreasonable which will significantly affect their performances. This paper presents some different ideas for anti-sniffing in unbounded networks.

The rest of the paper is organized as follows. Section 2 gives a brief analysis to traditional anti-sniffing techniques, specially focusing on their assumptions. Section 3 proposes our fundamental idea dealing with security issues in unbounded networks, and the main result is a common security control platform (called FSCP, Flexible Security Control Platform). Section 4 further gives a new anti-sniffing approach over the FSCP. Section 5 is a brief discussion between our solution and related works, and finally is the conclusion.

2. ANTI-SNIFFING

As to any network security issue, generally speaking, there are three ways to deal with: protection approach, detection approach and countermeasure approach. The protection approach tries to increase the hardness of networks to prohibit attack actions. The detection approach tries to discover attack events in the networks immediately, then to stop them. The countermeasure approach tries to increase the survivability of networks even after some kinds of attacks happen. Anti-sniffing approaches also could be classified in this way. Our aim is not to offer details for each anti-sniffing approach but to show that any traditional anti-sniffing method only is effective under corresponding assumptions are promised. The following examples claim this idea.

As to the protection approach, replacing Ethernet hubs with Ethernet switches is recommended as a way to protect networks against sniffing, since it claims that switches have the ability to segment a network traffic and prevent every system on the network from being able to "see" all packets (Graham R., 2000). Obviously, this method assumes the Ethernet switches are effective against sniffing compared with Ethernet hubs. But the real case may be different when a cracker uses a tool like Dsniff (Dug Song, 2000). Many methods have been proposed in anti-sniffing detection approach. MAC detection is one of common methods in sniffing detection techniques (D. Wu, 1998). However, this way requires that the machine running the detector be on the same Ethernet segment as the host that is suspected of running a sniffer. However, whether the suspected node is in the same segment with anti-sniffing detector is often unknown in an unbounded network. Besides, we all know that the operation system types and NIC (Network Interface Cards) types of suspected nodes behave differently to the same ICMP or ARP probing packet. Another well-known example is DNS method detection method that is entirely depending on the assumption that the sniffing tools used by attackers will do automatic reverse-DNS lookups on the IP addresses they see.

A seemingly attractive idea is sniffing countermeasure approach which claims that network will be secure even some attackers get packets with sniffing tools. The TAP protocol in (Gookwhan, 2000) is such an effort. However, it assumes that each network node runs TAP protocol. Unfortunately this assumption is definitely impractical, we think. Anther direct idea is to encrypt your data, so that while attackers can sniff it, they cannot read it. VPNs (Virtual Private Network) provide encrypted traffic across the Internet. However, if a hacker compromises the end-nodes of a VPN connection, they can still sniff the traffic. A typical scenario is that an end-user who surfs the Internet normally and gets compromised with a Remote Access Trojan (RAT) that contains a sniffing plug-in. When the user establishes the VPN connection, the sniffing program is able to see not only the encrypted traffic that can be seen on the Internet, but also the unencrypted traffic before it gets sent through the VPN.

In a word, each of traditional anti-sniffing methods has its assumptions. If these assumptions are not promised completely, their performance would become worsened even lost. In the situation of unbounded networks, many of such preconditions or assumptions become unreasonable from the above brief analysis.

Therefore, a different approach is needed to meet this challenge, and currently, similar scenarios also arise from other network and information security issues.

3. FSCP ARCHITECTURE

FSCP (see Fig. 1) is our common framework or platform for solving unbounded networks security issues. The details of the FSCP are beyond the scope of this paper. Here, from the view of application angle, the basic idea about it is given. Generally speaking, the FSCP architecture is a distributed eXtensible Markup Language (XML) messages handling system. It consists of two types of nodes. One is called as policy agency, and another is called as security manager. Everything exchanged among policy agency nodes and security manager nodes are messages in XML format. Here, the word *message* has a wide meaning. Thanks to a message in XML format may have several appendices, they could store anything like an executable file, a Cisco router configuration script, even an entire system to be remotely mounted at somewhere. Due to the flexibility of XML in dealing with complicated data structure, our policy agency and security manager could be dynamically constructed to finish any special security tasks.

Another important item in FSCP design is that characteristics of unbounded networks have been fully considered. We think that only several nodes are controllable in unbounded networks, and that the majority of nodes are un-controllable for network administrators, and that any node is unable to know entire information about the network, and that the data collected no matter how much effort is enforced remains inexact, incomplete. Therefore, as to security issues of unbounded networks, we make following assumptions:

1) It is unable to know exact security status of networks.

2) Any security devices or software is not absolutely reliable no matter how high quality is promised.

3) A network is an open system not only without clear boundary but also without fixed and closed requirements.

The fundamental principle different to the traditional fortress model in our research is that we model FSCP as a non-linear, feedback control complicated system. Control theory provides a vocabulary for reasoning about how to keep systems operating as desired and for structuring information-based mechanisms to effect such control (Kevin, 1999). Obviously, the networks are the targets that FSCP controls. Policy agency looks like a data collecting unit (sensing unit) and executing unit (enforcing unit), and the security manager looks like a control unit. Here, we remind you that over one security manager may exist in a network. To an unbounded network, how many security manager nodes or policy agency nodes are running may be unknown for an administrator. The basic control flow is as follows: a security manager aggregates network knowledge from data collected by policy agency nodes, then makes out a decision (a control policy) which is usually implemented through selecting from a group of policies, then a series of operations commands which achieve the control policy are delivered to the related policy agency nodes to be executed. When further information is acquired, new proper control polices are produced again. From the general view of point, this procedure is similar to feedback control process deeply studied in control theory. The main elements of FSCP are given as follows.

Component supermarket: a component in FSCP is anything that contains data or executable code to achieve some specific independent security objectives. In FSCP, any object no matter functions it completes is abstracted as a component. Our component is analogy to a DCOM (Microsoft, 1999) component, but it is not like a DCOM component tightly coupled in Microsoft technology. A component usually consists of two parts: component description and component entity. The component description is a XML file through which, a XML parser could automatically understands what a component is and how to manipulate the component entity. The component entity is one file or a group of files for achieving some specific functions. All movable components in FSCP form a component database that is called as component supermarket. The component supermarket has two parts: a LDAP (Wahl, 1997) directory database that keeps all component XML description files, and a distributed file system that keep component entities. Though a file system to store component descriptions files is feasible, Using LDAP database to store large quantity of XML files is proved to be more effective.

Policy agency: located at the controllable nodes. Its main function is to complete whatever the security manager requires. Since the task of a policy agency node is assigned by one security manager node, it is

similar to any ordinary travel agency organization. It receives the message, then parses the message and does whatever operations defined in the received message. Usually, some necessary components are copied here from the components supermarket accordingly. A policy agency may do some local decision when it gets more knowledge about network security status and more components from the component supermarket.

Decision unit: it is the core module of the security manager that processes the messages from policy agency nodes, and determines the operations in the next step. The result of decision unit produces is a control policy usually expressed in XML format. Some control polices may directly defined and enforced by administrators through *Administrator UI* (see below). A FSCP allows several decision units to meets complex security requirements.

Administrator UI: provides operation interface for administrators to add, delete, modify components or control policies.

Message service: a demon process in the security manager that is used to receive all messages from policy agency nodes, and disseminate messages to different decision units according to messages types.

Policy service: another daemon process in the security manager that generates control messages according to a control policy, and then dispatches these messages including corresponding components entities towards related policy agencies.

Control message: message (flows along dotted lines) to implement control task delivered by the policy service process.

Feedback message: the message reflects network security status delivered by policy agency nodes. The feedback messages usually are originated from the results when corresponding components are enforced to networks elements.

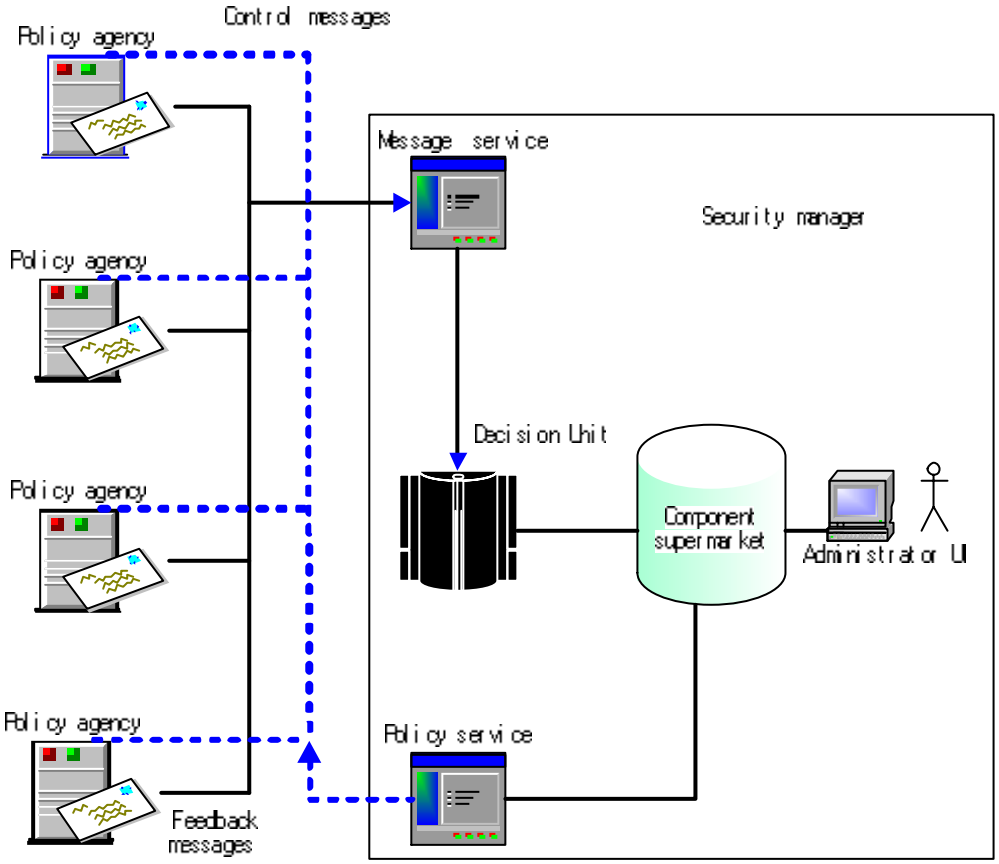

Figure 1. FSCP architecture

4. ANTI-SNIFFING OVER FSCP

In the case of anti-sniffing of large-scale unbounded networks, many aspects are different to traditional situations. The first aspect is about targets, and some are listed below. 1) Operation system types and quantity of nodes are unable to know ahead. Since most of nodes are out of control to administrators, they are usually unable to know node operation system types and NIC types in advance when a network permits new participating nodes come randomly. For instance, a network offering DHCP service belongs to this case. 2) Physical connection mode sometimes is unclear. In a large-scale network, to obtain the exact physical topology information is very hard even impossible, thus, the situation often happens that whether a node connected by a switch or by a hub is unknown. 3) Multiple IP addresses makes complicated to sniffing detection. Usually, in traditional sniffing detection approach, a scope of IP is given, and each of them is detected through a group of anti-sniffing methods. For example, the tool antisniff (L0pht Heavy Industry, 1999) works in this way. However, the checked IP may belong to a router interface or a switch but a host, and several IPs may go to one interface. These options result in that the performance of traditional anti-sniffing tools becomes significantly degraded. 5) It is difficult to be sure whether all nodes are covered. Since in large-scale networks or when security mechanisms like VLAN are widely used to avoid malicious attack including sniffing action, some subnet may be unreachable from one single anti-sniffing detector.

The second aspect is about anti-sniffing methods. According to section 2, no any anti-sniffing method is absolutely effective in all cases. Any method is applicable only under that its assumption is kept. So, in this case, we think that the only effective way is taking proper measurement according to practical situations. For example, if a switch is under control by one policy agency node of the FSCP and it provides Telnet service or SSH service for remote configuration, a component to achieve MAC binding should be sent to this node executed to prevent active sniffing in this case.

Based on the above discussion, our anti-sniffing approach is an integration approach that is based on the FSCP. Apparently, the steps to construct a real anti-sniffing tool for unbounded networks over FSCP become easy and simple: 1) Build components to implement desired functions such as sniffing detection using ARP, MAC binding. 2) Define alternative control polices provisions for selection by the security manager. Because FSCP is a message handling system, a control policy is an expression to a group of operations about messages. Polices are usually made out by administrators described with an XML-based language which is not discussed in this paper due to space limits. 3) Determine mapping rules between feedback messages and available policy items. In FSCP, the control decision is based on experience. To one kind of specific feedback messages, after a control policy is enforced, the expected results may not appear, in this time, the best way is to try another policy. Thus, the mapping rules among polices and messages should be changeable on time. Some components examples in our system are listed as follows:

- Alive nodes discovery component, to poll an IP range to find out new alive nodes.
- ARP detection component, detect sniffing by ARP packets in an Ethernet segment.
- Load detection component, detect sniffing by checking the response time.
- Gateway control component, modify the filter rules of a gateway to countermeasure sniffing for a while.
- MAC binding component, operate a switch to fix the mapping rules among its ports, MAC addresses and IP addresses.

One important feature of our anti-sniffing tool is that more anti-sniffing components and policies allow to be plugged at anytime according to new requirement of the unbounded networks and experiences aggregated.

Our system is implemented with standard JDK1.4 to be portable among different running environments. The powerful JDK1.4 provides an excellent support for XML messages processing. The communication among policy agency nodes and security manager nodes follows the Simple Object Access Protocol, or shorts as SOAP (Don Box, 2000). SOAP is becoming a de facto standard for exchanging structured and typed data in many further Internet application fields. There are four major categories in SOAP definition: defining the way of using XML to represent data, an extensible message format, how to represent remote procedure calls (RPC) using SOAP message format, and bindings to Hypertext Transfer Protocol (HTTP). In our solution, a specific XML schema following SOAP message format is defined and we do not bind to HTTP but SSL to be secure.

5. DISCUSSION

The greatest distinguished difference between our solution and related works is that we take a different view to the anti-sniffing issue. The research on anti-sniffing has attracted many security professionals recently, much informal discusses about anti-sniffing are scattered around Internet community. A paper (Marco V. 1998) offered a solid foundation for understanding anti-sniffing issue. AntiSniff (L0pht Heavy Industry , 1999) is a proactive security monitoring tool by running a number of non-intrusive tests in a variety of ways network. It is based on polling mechanism and only involves sniffing detection aspect. Anti Sniff Toolbox (WebTECA – maggio, 2003) is a collection of three tools: ACiD, PMD and https2http. Anti Sniff Toolbox is more powerful than Antisniff in which includes the function of active sniffing detection. However, it is a centralized system and only provides few anti-sniffing methods. Our solution is a distributed system and provides an infrastructure to contain unlimited anti-sniffing methods. The authors (D. Wu , 1998)shares some ideas with us. It pointed out that any anti-sniffing method only is effective under its corresponding assumptions and only detection is not enough for anti-sniffing. However, except three categories of sniffing detection methods, other two aspects of anti-sniffing, sniffing prevention and sniffing countermeasure were not presented. Similar to our idea, the authors (Kewley, 2001) noticed difficulty in anti-sniffing issue. It provides a technique called as DYNAT(dynamic network address translation)to hide the real IP addresses and socket port numbers of critical servers and services since encryptions mechanisms can not be applied to IP addresses in standard TCP/IP stack to avoid sniffing. However, it is based on fortress model too, since if an adversary bypasses its DYNAT Gateway in server side or DYANT Shim in client side, their solution would become useless. Similar work has been done in (Syverson, 2003) which provides anonymous connections mechanisms called as the Onion Routing that are resistant to both eavesdropping and traffic analysis. (Ramakrishna, 2002) takes an integration approach too, but only active sniffing item involved, and no practical details are proposed. Our solution deals with not only passive sniffing but also active sniffing. The authors (Gookwhan, 2000) proposed a protocol named as TAP to prevent sniffing. However, their solution is not applicable to unbounded networks due to TAP has to be running in each network node. As a summary, we think there are some advantages in our solution compared to above works.

- Scalability The implementation based on SOAP makes our solution scalable since the policy agency nodes and security manager nodes in FSCP may be located in anywhere of unbounded networks, and there are no limits to their quantity.
- Adaptation Methods of sniffing prevention, detection and countermeasure may are enforced or disabled dynamically according to information obtained due to all various methods are wrapped into components.
- Intelligence Our system essentially is a distributed feedback system. The procedure close to mankind thinking, aggregating knowledge, then making out policy, then turning next round, fits to characteristics of unbounded networks and provides a room to achieve more intelligent anti-sniffing strategies.

6. CONCLUSION

The traditional anti-sniffing approach meets great challenge in the situation of unbounded networks. The above anti-sniffing solution takes a different angle in which anti-sniffing looks like a doctor to a patient. The protection, detection and countermeasure methods are integrated smoothly. The anti-sniffing procedure is modeled as a continuously distributed feedback control procedure: aggregating knowledge, then taking proper measurement, observing network feedback, then turning to next round again. Our research shows that the traditional approach only focused on detection intrusion behaviors is a narrow view in anti-sniffing research field This work gives a new angle of anti-sniffing issue, and is an initial effort to meet similar challenges in the security issues of unbounded networks.

ACKNOWLEDGEMENT

This work is funded by Chinese 863 high-tech project under contract number 2002AA142020. We deeply thank professor Fan Bin-xin and Phd. Zhang Hong-li for many discussions with them are very helpful to us. We also acknowledge Mr. Wang Yong-heng and Huang Shuo for their hard works in this project.

REFERENCES

Ansari S. et al, 2003. Packet Sniffing:a Brief Introduction. *In IEEE Potentials*, Vol. 21, No. 5, pp 17 -19.

Bob Blakley,1996. The Emperor's Old Armor. *Proceedings of the 1996 New Security Paradigms Workshop.* Lake Arrowhead, California, pp 2-16.

Don Box et al,2000. Simple Object Access Protocol (SOAP). Available: http://www.w3.org/TR/SOAP/.

Dug Song, 2000. dsniff. Available: naughty.monkey.org/~dugsong//dsniff/.

D. Wu et al, 1998. Remote Sniffer Detection. Available: www.cs.berkeley.edu/~daw/teaching/cs261-f98/projects/final-reports/fredwong-davidwu.ps.

Ellison, R.J. et al,1999. Survivability: Protecting Your Critical Systems. *In Internet Computing, IEEE.* Vol. 3, No. 6,pp 55 –63.

Gookwhan Ahn, et al, 2000. Tapping Alert Protocol. *IEEE 9th International Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises*. Gaithersburg, Maryland, USA, pp 159 –164.

Graham R., 2000. Sniffing (network wiretap, sniffer) FAQ. Available: www.robertgraham.com/pubs/sniffing-faq.html.

Kevin S. et al, 1999. Information Survivability Control Systems. *Proceedings of the 21 st International Conference on Software Engineering*. Los Angeles, USA, pp 184-192.

Kewley D. et al, 2001.Dynamic Approaches to Thwart Adversary Intelligence Gathering. *DISCEX II'01*. California ,USA. pp 176 –185.

L0pht Heavy Industry, 1999. AntiSniff. Available:http://www.l0pht.com/antisniff/.

Marco V. et al, 1998. Internet Security Attacks at the Basic Levels. *In ACM SIGOPS Operating Systems Review.*Vol.32, No.2, pp4-15.

Microsoft, 1999. DCOM. Available: http://www.microsoft.com/com/tech/DCOM.asp.

Ramakrishna P. et al, 2002. Detection and Prevention of Active Sniffing on Routing Protocol. *Student Conference on Research and Development Proceedings*. Shah Alarn, Malaysia,pp 498 –501.

Syverson P., 2003. Onion Routing for Resistance to Traffic Analysis. *DISCEX '03*. Washington D C, USA, pp 108 –110.

Wahl M. et al, 1997. Lightweight Directory Access Protocol (v3):Attribute Syntax Definitions. *RFC 2252.*

WebTECA – maggio, 2003. Anti Sniff Toolbox. Available: http://utenti.lycos.it/webteca/AntiSniffToolbox.htm.