Application Level Network Access Control System Based on TNC Architecture for Enterprise Network

Zhen Chen, Fa-Chao Deng, An-An Luo, Xin Jiang, Guo-Dong Li, Run-hua Zhang, Chuang Lin
Research Institute of Information Technology
Department of Automation and Computer Science & Technologies
Tsinghua National Laboratory for Information Science and Technology (TNList)
Tsinghua University, Beijing 100084, China

***Abstract*—Traditional NAC system in enterprise network is in coarse granularity (e.g. IP or MAC address) and lack of flexibility. Recently the demand in tight control of the enterprise network to defense the misuse and security issues become more and more urgent. Based on the TCG TNC standard, an application level network access control mechanism is proposed and implemented. With TNC client/server model in hand, a client is designed to enhance TNC client with the function of host flow controller (HFC), and intercepts each application network access request(ANAR) and transfer it to PDP server to authorize the access request. When a sensor (i.e. intrusion detection system) detects any malicious traffic, host flow controller and network flow controller can identify the application that origins this traffic by querying Metadata Access Point (MAP) server and block this application's network access. A prototype system is implemented to demonstrate the design and can be used to defense the anomaly network behaviors. The prototype system demonstrates that the hosts, switches, firewalls and IDS can work together to detect, diagnose and protect enterprise network from the malicious applications attack initiated inside or outside of an enterprise network, quarantine unhealthy hosts and make the enterprise network more reliable and trustworthy.**

***Index Terms*—Network Security, Access Control, Trusted Network Connect, Application Level Access Control.**

I. INTRODUCTION

Users can access enterprise networks from anywhere in the world at any time, via kinds of access technologies and devices running any operating systems, operating environments, and applications. That has made network border increasingly blurred, and brought great difficulties on the management of enterprise networks. What's more, as the rapid development of Internet, more and more network applications emerge every day, and people install variety of applications on their computers, most of which have not been audited, so, Trojans, viruses, malicious codes and the backdoor are likely hidden in these applications. A survey of security professionals conducted by CSI/FBI shows that half of the attacks on enterprise networks start from inside [1].

Traditionally, network access systems use the username/password mechanism or X.509 certificate to authenticate users and let the corresponding IP or MAC traffic pass through. In order to ensure that each computer in the network is healthy, NAC technology considers not only the user's identity but also the health status of endpoint devices. But this scheme still can't restrain Worms, Trojans and other malicious programs from spreading in enterprise network, and can't restrict P2P applications, E-games and online videos.

This paper presents an application level network access control mechanism based on TNC architecture for enterprise network and implements a prototype system. The system intercepts each application network access request (ANAR) through host flow controller, authenticates and authorizes these network access requests through enhanced PDP, reports these requests information to MAP server, controls these network accesses through host flow controller and network flow controller, inspects communication of applications through network sensors.

This paper is organized as follows: Section 1 introduces the research background and application scenarios. Section 2 presents the basic TNC technology which is the foundation of TNC system. Section 3 explores the system framework and components, and also describes the prototype implementation, and the application access procedure in detail. Finally, the section 4 concludes the paper.

This work is supported by Natural Science Foundation of China No. 90718040, National High-Tech Program No.2007AA01Z468, Hosun Tech.

II. TNC OVERVIEW

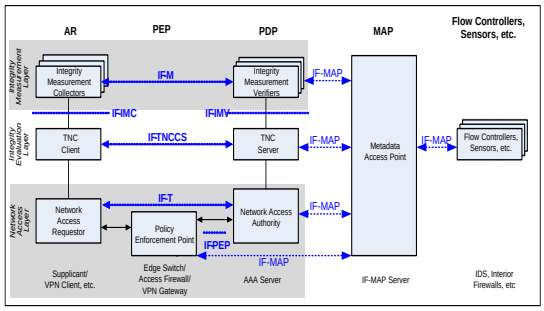

Figure 1. The TNC Architecture

TNC is an open standard network access control architecture, which is defined and promoted by Trusted Computing Group (TCG) [2-4] (See Figure 1). TNC defines entities and several standard interfaces between components, indicated by arrowhead lines in the architecture diagram. The architecture, as specified in [4], consists of Access Requestor (AR), and the Policy Decision Point (PDP). The optional entities are the Policy Enforcement Point (PEP), the Metadata Access Point (MAP), and Flow Controllers and Sensors. Interfaces in the TNC architecture are included as follows: Integrity Measurement Collector Interface (IF-IMC[5]), Integrity Measurement Verifier Interface (IF-IMV[6]), TNC Client-Server Interface (IF-TNCCS[7]), Vendor-Specific

IMC-IMV Messages (IF-M[8]), Network Authorization Transport Protocol (IF-T[9]), Platform Trust Services Interface (IF-PTS[10]), Policy Enforcement Point Interface (IF-PEP[11]) and Metadata Access Protocol (IF-MAP[12]).

III. System Prototypes

3.1 Application Access Control Requirements

Best to our knowledge, there are no application level access controls using current TNC architecture. In current TNC specification [4], PDP authenticates user and hosts for network access control based on identification and integrity report. After the PDP's decision and authorization, the access granularity is quite coarse, such as L2 on-off access control based on 802.1x, while more and more application–level network access control require a more fine granularity in network access control from TNC architecture.

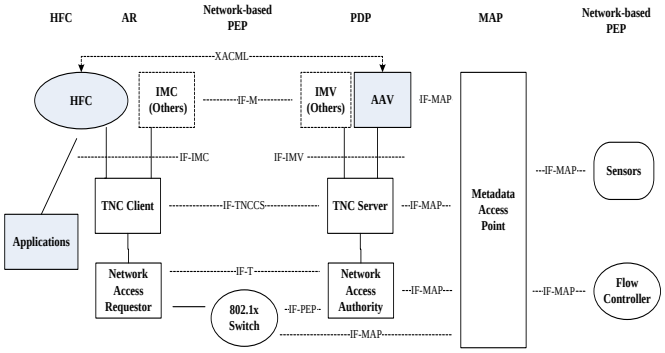

Figure 2. System Framework

- Capture of application network access request

In order to control the network accesses of applications, we have to capture the network access request first. There are several ways to capture the request, but no more than the following two means: one is capturing the request at the endpoint, which needs to install agent software, such as host firewall software; the other is capturing the request at network gateway, which needs protocol recognition technology. The second mean is more difficult to implement, and what's more, the effect is always not good enough. On the contrary, the first mean is easier to implement, but additional agent software is needed. Generally, agent software is also needed by TNC system, which can be extended with the help of interface of IF-IMC. And for enterprise network, it is feasible to ask every employee to install a agent software in their computers.

- Authentication and authorization of application network access request

After we capture the network access request of applications, we also need a mechanism to check the validity of this request, i.e., determining whether this request is compliant with network administrator's policies. This process is called authentication. After that we need another mechanism to authorize the request. We hope these authentication and authorization mechanisms must be standard, widely accepted, and suitable for current systems.

- Application level network access control

After authenticate the application access request (AAR), we need a scheme to control the application access in a host. Such control scheme can be implemented in host or in gateway. The control scheme will became easier to be deployed in host, e.g. most PC firewall can control the application's network access easily.

To achieve the goal of application-level access control, we need to deploy a host flow controller (HFC) in host to intercept the applications access request (AAR), extract AAR's detailed info, and generate a XACML (eXtensible Access Control Mark Language)[13] access request to PDP. This AAR is evaluated by PDP's XACML policy evaluation engine based on the access policy. After evaluation, PDP sends back a XACML response to HFC of the host. When the HFC received the corresponding XACML responses from PDP, it will deploy the corresponding policy (allow or deny rule) for this application.

3.2 System Framework

According to requirements proposed above, we design the system framework and the key components wherein. The oval block is the PEP, and there are three sub-components of PEP, i.e., HFC, 802.1x Switch, Network Flow Controller.

- TNC Entities

Existed entities in TNC architecture are in the gray blocks in Figure 2.

- Host Flow Controller (HFC)

HFC is deployed in host, similar to host firewall, but enhanced with TNC-compatible PEP capabilities. HFC intercepts every application access request (AAR), and controls the application network access on-off based on PDP's decision. The HFC also consists of a subcomponent as IMC in TNC architecture, and interacts with corresponding IMV module in PDP which is shown in Figure 3.

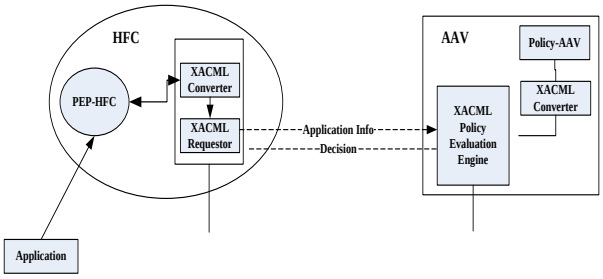

Figure 3. Inner structure of HFC and AAV.

- Application Access Verifier (AAV)

An IMV is customed for HFC's IMC functionality. The application's access policy and evaluation procedure engine are all implemented in this module.

3.3 XACML based IMC-IMV Scheme for Application Level Access Control

3.3.1 The eXtensible Access Control Markup Language - XACML

XACML which is designed to support the needs of most authorization systems, is a general purpose policy system. At its core, the syntax for a policy language and the semantics for processing those policies are defined by XACML. There are also semantics for determining applicability of policies to request, and a request and response format to query the policy system. The later represent a standard interface, between a PDP that presents standard behavior when processing policy and a PEP that issues requests and handles responses.

3.3.2 Application Level Access Control on XACML[14]

This use case explores the possibility of applying XACML policy for verifying the given Application Access Request against PDP, and demonstrates how XACML policy engine generates verification results (see Figure 4). XACML can return " VALID", "INVALID", or "UNVERIFIED" according to the verification criteria.

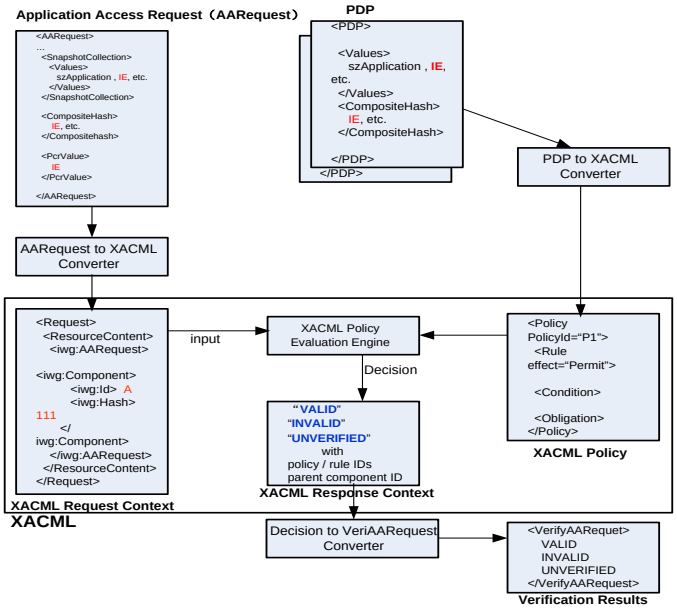

Figure 4. XACML-based Validation Framework

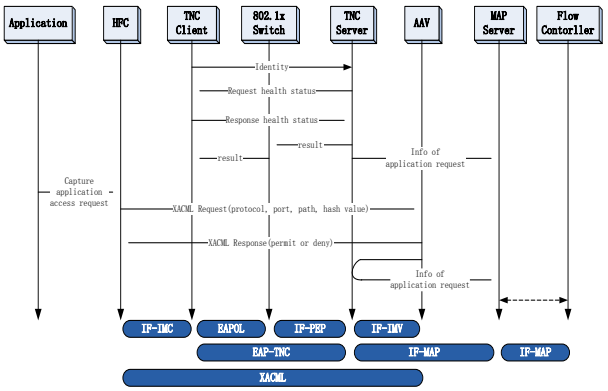

Figure 5. Message Flow diagram

3.4 Application Level Access Control Procedure

Message flow in Application level Network Access Control is divided into two stages: the first is the stage of authentication and authorization when hosts request for a network connection, and the second which is called AAC(Application Access Control) is the stage of authentication and authorization when applications request to access network. And the whole messages exchange is shown in Figure 5.

TNC stage message flow:

(1) User connects his computer terminal to the switch with the functionality of 802.1x.
(2) User supplies his user identity information (EAP-MD5 or WAP-TLS), and then requests for authentication initiatively. Before authentication, no packages but packages for EAPOL can pass because of the mechanism of 802.1x. The representative authentication flow in TNC as follows:

a) Switch sends the authentication information from AR to PDP server, and PDP server authenticates the user's identity.
b) After User Authentication succeeded, PDP publish user identity information, role information and related request information to MAP Server, and requests AR to validate the machine's healthy state information, such as the name, version number and mend condition of AR's operation system, name, version number, run state and virus base state of Anti-virus software, run state of HFC.
c) AR extracts the local host's healthy state information according to corresponding IMC, packs the information into message accord with TNC and sends it to PDP.
d) PDP authenticates AR's healthy state information, and makes corresponding decision according to healthy policy. PDP possibly makes the final decision after several handshakes. Decisions maybe permit user to access, maybe deny to access and maybe permit to access to certain VLAN. PDP will supplies mend policy for AR that is not accord with policy. Besides, PDP need to publish AR's healthy information to MAP Server.
e) Switch transmits the decision made by PDP to AR, and takes corresponding actions according to the decision, such as opens up or closes the corresponding port and comes under certain VLAN.

AAC stage message flow

(1) When certain application (such as IE7 browser) triggers a network attempt, PEP-HFC module of HFC captures the application's access request. HFC produces outline information of network access attribute (includes the type and port of protocol, application's path, application's characteristic and so on) about this access request, and converts the information into message based on XACML description by the XACML Converter inner HFC and then send it to PDP.
(2) The message is passed TNCS module of PDP to AAV

for authentication. Meanwhile, PDP publish the network access authentication information to MAP Server. Flow Controller of boundary obtains the information from MAP Server and makes policy for policy consistent with host.

If the application is permitted to access network after authentication, it not only can access inner network of enterprise, but also pass Flow Controller all right and access outer network or sensitive network of enterprise. If the application is prohibited to access network after authentication, it can not access network because of restrict of HFC, and it still cannot get out of the control from Flow Controller even if it can bypss the HFC's restrict by certain measures. Therefore, it effectively protects the enterprise's sensitive network resources, prevents the enterprise's confidential information from releasing, and limits the use of some network application.

If AR is prohibits to access network after authentication, the corresponding port of switch is closed and any network access request from the host's any program is invalidate, and the AAC stage is denied. However, AAC stage is taken when AR's application triggers a network connection attempt if AR is permit to access network after authentication.

3.5 System Implementation

3.5.1 Client Program

a) TNC Client (TNCC)

The client is TNC-compliant and running on Windows Platform, i.e., Windows XP and Windows Vista. We port TNC@FHH (for Linux) to windows platform because the windows platform is more popular. According to TNC specifications, the client program is organized in four layers: GUI, IML, IEL and NAL. The GUI is shown as Figure 6.

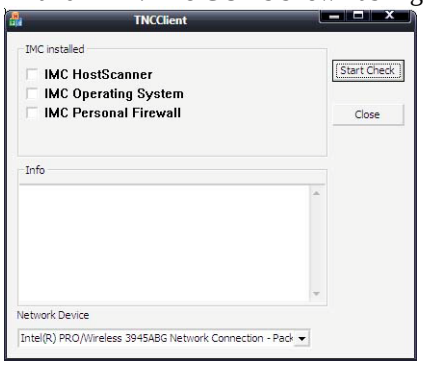

Figure 6. TNC Client GUI

b) Host Flow Controller

The host flow controller is complemented both in the kernel and application level. When setup the connection, the controller downloads the kernel filter rules from PDP, which is encapsulated by XACML through the TNC Client. When an application intends to access the network, the controller will capture the information about the application such as attributes of the application, then encapsulate them in XACML, at last send the encapsulation to PDP through TNCC.

The host flow controller is composed of three modules: report module, application level filter module and kernel level filter module.

The report module provides interaction between PDP and the flow controller, such as downloading the filter rules from PDP. It is also used for communicating with application level filter module and kernel level filter module. Its work flow is like this: downloading the kernel filter rules from PDP and transfer those to kernel level filter module when the connection is set up, sending the request to PDP and receiving the decision from PDP when an application attends to visit the network.

Application level filter module is used to capture the invocations of Winsock[15]. It checks the rights of each application that intends to access network with the application rules which are dynamically obtained from PDP through report module. It is built into DLL (Dynamically Link Library) and installed in the directory of Winsock. All the applications have to invocate the DLL service provider, so it can capture all the invocations of Winsock.

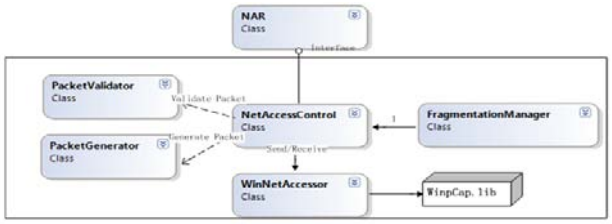

Figure 7. NAR

Kernel level filter module is a NDIS intermediate driver (IM driver). It deploys the kernel filter rules from report module that obtained from PDP when connection is set up. Intermediate driver between protocol driver and miniport driver can capture and filter all the packets. We develop the program on the basis of the Passthru which is an sample of IM driver in Windows DDK[16]. The report module communicates with PDP, fetches the application filter rules and sets application rules which are used to filter the application requests. DLL modules send the status of the network and the requests to report module. The report module sets the kernel filter rules through I/O control codes.

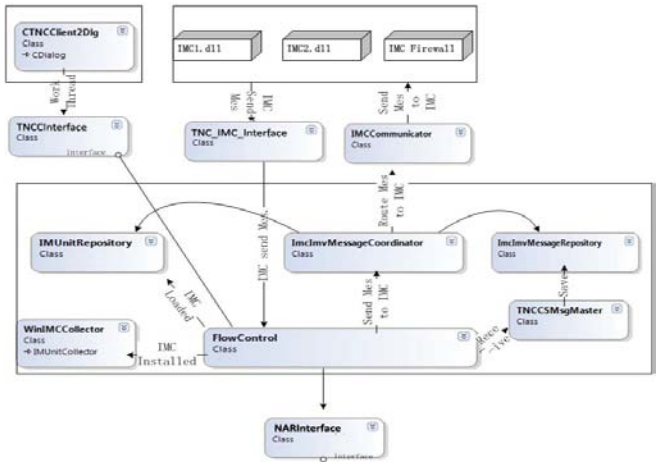

Figure 8. IEL.

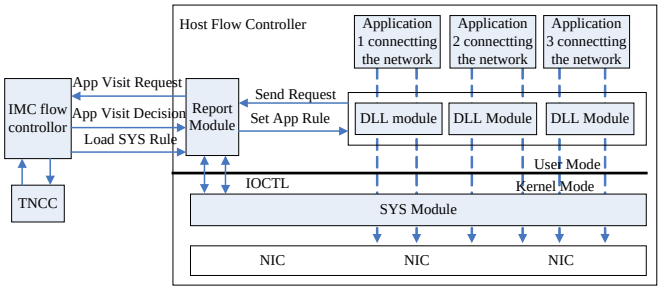

Figure 9. The components in HFC implementation.

From the Figure 8, it is more reliable that each application intending to access network will be filtered in both application level and kernel level.

3.5.2 Network Flow Controller

PDP publishes access requestor of authentication, VLAN, and other status to the MAP server. A TNC server publishes access requestor compliance status after performing an integrity check.

Both TNC Server and network flow controller are MAP clients. The TNC Server is also a publisher, while the network flow controller is a subscriber. Network flow controller (such as a layer 3 firewall) subscribes to notification of endpoint's application access request information. After performing the XACML access policy evaluation, TNC server will send back a XACML response to endpoint. In the mean time, TNC server publishes information about policy compliance of the application. Then network flow controller detects a previously unseen flow from an access requestor and queries an MAP server to obtain authentication and compliance status associated with this access requestor in order to make enforcement decisions about the new flow.

Network flow controller subscribes to notifications from an MAP server about changes in authentication, compliance, vulnerability, or other status for an access requestor so the network flow controller can make appropriate enforcement adjustments to an existing flow.

When the TNC server detects that the endpoint is no longer policy compliant, the TNC server updates the information in the MAP server. The MAP server notifies the network flow controller and the network flow controller blocks the access to the network from the new non-compliant device.

3.5.3 Network Sensors

A sensor, i.e. an intrusion detection system, is deployed in enterprise network to detect the anomaly traffic and malice attacks. A sensor also publishes information related to an application access request or flows originated from an access requestor (vulnerability detection, flow classification, flow compliance, etc.) to the MAP server.

PDP queries the MAP server for metadata that a sensor has associated with an access requestor (e.g. flow classification or vulnerability information). The PDP uses the metadata to make appropriate policy decisions. The PDP subscribes to notifications from the MAP server about changes to the access requestor's metadata so the PDP can adjust the access requestor's access when the access requestor's metadata changes.

Network flow controller can also subscribe to notifications from the MAP server about the metadata that a sensor has collected from an access requestor (e.g. flow classification, misbehavior or vulnerability information). The network flow controller uses these metadata to make appropriate policy decisions to block or restrict network access.

IV. CONCLUSIONS

Based on TNC architecture, this paper presents an application-level network access control framework. An application access request is described as an XACML request and evaluated by PDP's XACML policy evaluation engine based on access policy. We extend the PEP in TNC architecture to Host-based PEP (i.e. Flow Controller) for application access control, and propose a holistic strategy to integrate the distributed security resources into coordinated network protection system under MAP scheme.

With the above application-level network access control framework, a prototype system is implemented and shows that the whole network access procedure is correct and non-ambiguous. The prototype system demonstrates that the hosts, switches, firewalls, IDS can work together to detect, diagnose and protect from the malicious application attacks initiated inside or outside of an enterprise network, quarantine the unhealthy hosts and improve the reliability and security of the enterprise network.

ACKNOWLEDGMENT

We thank the NSLAB and QoSLAB colleagues' generous help for this paper and prototype work.

We thanks for Hosun Tech. for their generous support.

References

[1] Computer Crime and Security Survey. CSI/FBI. 2005; http://www.cppe.umd.edu/Bookstore/Documents/2005CSISurvey.pdf

[2] R. Whiteley, Demystifying NAC: Going Beyond Basic Admission Control, tech. report, Forrester Research, Inc. Sept, 2006.

[3] INTEROP LABS. What is Cisco NAC. May, 2007

[4] Trusted Computing Group. TCG Trusted Network Connect TNC Architecture for interoperability Specification Version 1.3. April 2008.

[5] Trusted Computing Group. TCG Trusted Network Connect TNC IF-IMC. Technical report, 2008.

[6] Trusted Computing Group. TCG Trusted Network Connect TNC IF-IMV. Technical report, 2008.

[7] Trusted Computing Group. TCG Trusted Network Connect TNC IF-TNCCS. Technical report, 2008.

[8] Trusted Computing Group. TCG Trusted Network Connect TNC IF-M. Technical report, 2008.

[9] Trusted Computing Group. TCG Trusted Network Connect TNC IF-T. Technical report, 2007.

[10] Trusted Computing Group. TCG Trusted Network Connect TNC IF-PTS. Technical report, 2006.

[11] Trusted Computing Group. TCG Trusted Network Connect TNC IF-PEP. Technical report, 2007.

[12] Trusted Computing Group. TCG Trusted Network Connect TNC IF-MAP. Technical report, 2008.

[13] OASIS eXtensible Access. Control Markup Language. (XACML). XML Community of Practice, 21 June 2006.

[14] Markus Lorch, Seth Proctor, Rebekah Lepro, Dennis Kafura, Sumit Shah 'First Experiences Using XACML for Access Control in Distributed Systems," ACM2003

[15] Microsoft MSDN http://msdn.microsoft.com/en-us/library/ms740673(VS.85).aspx

[16] Microsoft Corporation. Windows driver kit(WDK)documentation, http://msdn.microsoft.com/en-us/library/aa469207.aspx