An Extendable Authenticating Scheme for Windows Vista KMS†

Ling-En Kao (高翎恩)[1], Chang-Shang Chen (陳昌盛)[2,*], Chen-Hsien Sun (孫承憲)[1]

[1]Campus Computer Communication Association

[2]Computer and Network Center,

National Chiao Tung University, Hsinchu, Taiwan

E-mail: {lekao, cssuen}[1]@ccca.nctu.edu.tw, cschen[2]@mail.nctu.edu.tw

摘要

微軟 Vista 作業系統與金鑰管理系統(KMS)引進許多新的使用設限機制，也相應形成不少嶄新課題與挑戰。本論文，主要闡述如何設計與實作一套複合式 KMS 身分驗證系統，並使用於交通大學校園網路。本方案可適用於現有校園大量授權合約的網路環境，而不必受限於微軟所提的認證作法。經本校半年多來實際使用，成效良好，可大量簡化用戶與 KMS 系統兩端的設定與管理。只要稍加修改，也可很容易運用到企業大量授權用戶環境。

關鍵詞: 身分驗證, 金鑰管理系統, Vista 作業系統, 軟體大量授權

Abstract

Microsoft Windows Vista introduces a new activation system, Key Management Service (KMS), for campus and enterprise environments. Thus, authenticating valid users for accessing the KMS becomes an important issue for campus agreement service providers. In this paper, we design and implement a hybrid server system solution (i.e., with both Windows Server 2003 and GNU/Linux), which largely reduces the complexity of both end-user and server-side configuration tasks. With relative simple client-side configuration, this scheme provides for the Microsoft KMS authentication, with the integrated abilities to authenticate valid users, to avoid abusively usage, and to help track for each user account's usage.

Keywords: authentication, campus agreement service, KMS, VPN, Windows Vista

† This work was partially supported by National Science Council of the Republic of China under Grant No. NSC96-2221-E-009-169.

*Corresponding author

1. Introduction

Microsoft Windows Vista [1] utilizes the KMS [2][3] as its underlying activation system for campus and enterprise environment, with its own Active Directory[4] service for backend authentication routine. However, this might not be an appropriate approach for many sites, especially for the campus and enterprise agreement service providers. As shown in Table 1, there are many different authentication strategies and issues that need addressing. More details will be described in Sec.2.2 later.

Table 1: KMS authentication issues

Item/Description	Remarks
1. Using Microsoft Active Directory service for backend authentication routine	(a) Active Directory service is not widely available. (b) It is infeasible to limit the network access to the KMS by only permitting the hosts located inside the campus network because there are too many different requirements.
2. Using VPN [5] for backend authentication routine	(a) It cannot stop a valid user from abusively accessing the KMS (b) This has the complexity of configuration and compatibility issues for the VPN server/clients.

In this paper, in order to overcome these shortcomings, we design and implement a hybrid server system (i.e., with both Windows Server 2003 and GNU/Linux) by integrating several tools to provide an effective authentication

scheme for the Microsoft KMS authentication, with the ability to authenticate valid users, to avoid abusively usage, and to help track for each user account usage with relative simple client-side configuration.

Furthermore, using the proposed scheme, it is also very easy for us to integrate the authentication process with many different server systems (i.e., faculty/student mail servers) under the campus or enterprise environments. This is because the proposed scheme uses the standard POP3 service (interface). On the other hand, the type of the account server is not restricted to use POP3 service. For example, many other interfaces provided by PHP libraries (i.e., LDAP, MySql, NIS, etc.) can be easily used to extend our scheme by some minor modifications of the PHP scripts.

The organization of the rest of the paper is as follows. Section 2 gives background overviews for Microsoft Key Management System. In Section 3 we describe the system architecture and the design principles. Section 4 describes the working algorithm of our proposed authentication scheme. Section 5 contains some implementation details and discussion issues. Finally, in Section 6, a concluding remark and some points are highlighted for future research.

2. Microsoft Key Management System (KMS)

2.1. Overview of the KMS

According to [2], the Microsoft Key Management Service (KMS) allows campus or enterprise network users to perform local Windows Vista activations without connecting to Microsoft authentication site individually. The KMS can run on either Windows Vista or Windows Server 2003 installations, with a specialized activation key. Once the KMS host is deployed, a Windows Vista client can create a TCP connection to it and then send an activation request with destination port 1688. The clients can find the KMS host responsible for their environments automatically if the DNS and the client environments are configured properly. For example, a sample DNS configuration for KMS could be illustrated as shown below:

```
KMServ.nctu.edu.tw.  IN  A  211.76.240.198
_vlmcs._tcp  SRV 0 0 1688 KMserv.nctu.edu.tw.
```

After activation, the client will try connecting to the KMS host every 7 days to renew their activations, which can be controlled by the “Renewal Interval” variable on the KMS host. A valid activation lasts 180 days, if it expires then the client has 30 more days of grace period to renew their activations, or their Windows Vista installation will go into the Reduced Functionality Mode (RFM) directly.

2.2. KMS Authentication Strategies and Issues

As shown in Table 1, there are many different authentication strategies and issues that need addressing. First, for example, since the Active Directory service is not widely available for many campus environments, the deployment of Active Directory services (i.e., only for supporting Windows Vista authentication) may not be a cost-effective solution. Next, it is also infeasible to limit the network access to the KMS by only permitting the hosts located inside the campus network. This is because there are too many different requirements. For example, there might be many addresses for public usages. Or, once in a while, we might have many ill-protected hosts exploited. Both of these hosts could be utilized to act as jump gates relaying traffic for adversaries outside the campus or enterprise environments. Moreover, there could also be lots of valid users that gain their network accesses via other non-campus dynamic IP addresses. Their rights should not be ignored.

In principle, the authenticating of valid users over the Virtual Private Network (VPN) [5] might be a solution; however, in real word, we cannot prevent any person with a valid user account from abusively accessing the KMS [3][6] by using VPN only. This is because, in order to make this scheme more secure against abusive usages, we would also have to restrict the number of successful logins of each username per week/month. Moreover, the complexity of configuration and compatibility issues (e.g., IPSec[7], etc.) for the VPN server/client also make this approach infeasible.

In this paper, in order to overcome these shortcomings, we design and implement a hybrid server system (i.e., with both Windows Vista and GNU/Linux) for the Microsoft KMS authentication, with the ability to authenticate valid users, to avoid abusively usage, and to help track for each user account’s usage with relative simple client-side configuration. More details will be given in Sec. 3.

3. System Architecture

As shown in Fig. 1, the system architecture of our proposed scheme can be achieved by three

parts: the KMS provider, the authenticator, and the account server.

The KMS provider is usually a Windows Server 2003 or Windows Vista, which runs the "Key Management Service" package from Microsoft Corporation. In our scheme, this server resides on a private network in order to enhance the overall security of the system.

Next, the authenticator comes with two network interfaces, one for the private network that the KMS provider resides, the other for the Internet. It provides destination-NAT service for authenticated users connecting to the KMS provider.

Finally, the account server is responsible for storing the usernames and passwords, and then the authenticator will request the needed information from this server.

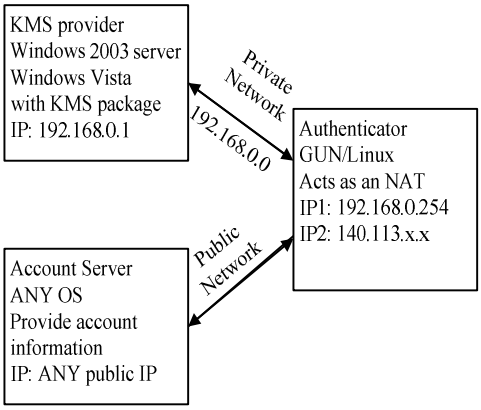

Fig. 1: Our Proposed Scheme

In our implementation, the KMS provider runs Windows Server 2003, and the authenticator runs GNU/Linux with NAT configuration via iptables[8], while asking the account server through POP3 (or MySQL) connections. The client-side only needs to assign the authenticator's Internet-side IP address via the "slmgr.vbs" command[3], and to interact with the web interface, which is rather easy as compared to the VPN approach.

Furthermore, using the proposed scheme, it is also very easy for us to integrate the authentication process with many different server systems (i.e., faculty/student mail servers) under the campus or enterprise environments since we are using the standard POP3/MySql interface. On the other hand, in principle, the type of the account server is not restricted to use the POP3/MySQL service for authentication. For example, many other interfaces provided by PHP libraries (i.e., LDAP, MySql, NIS, etc.) can be easily used to extend our scheme by some minor modifications of the PHP scripts.

4. Proposed Authentication Algorithm

In Sec.4, we will give the detailed descriptions on the operation principles of our proposed KMS authentication scheme.

4.1. Authentication Algorithm

As mentioned in Sec.3, the authenticator comes with two network interfaces, one for the private network that the KMS provider resides, the other for the Internet. It provides destination-NAT service for authenticated users connecting to the KMS provider.

Basically, as shown in Fig.2 and Table 2 below, the authenticator executes the following actions step-by-step:

Table 2: Our KMS Authenticating Algorithm

Step1. Prompt a client with a web-based interface for username and password and log the source IP address to *sIP* variable.

Step2. Connect to the account server and check for the provided username and password.

Step3. If the provided username and password are both correct (i.e., valid), and there are no other sessions from the same IP running, then this system will configure the NAT and forward the KMS traffic from *sIP* to the KMS provider within an allowed time period.

Step4. After the first TCP SYN packet from one particular IP/host is arrived, the system will drop any other TCP SYN packet from the same IP address during the time period to avoid multiple connections(i.e., to avoid abusive usage).

Step5. At the end of the time period (usually several minutes), we reconfigure the NAT to disable forwarding traffic to KMS provider, and log the username, *sIP*, system time to the system log.

In our scheme, Steps 1 and 2 are done by a PHP script. It reads usernames and passwords from web pages and creates connections to the account server using the POP3 (or MySQL) interface provided by the PHP language, and executes a shell script that manipulates the kernel routing table via the system() function call. The flow chart of this script is demonstrated as shown in Fig.2.

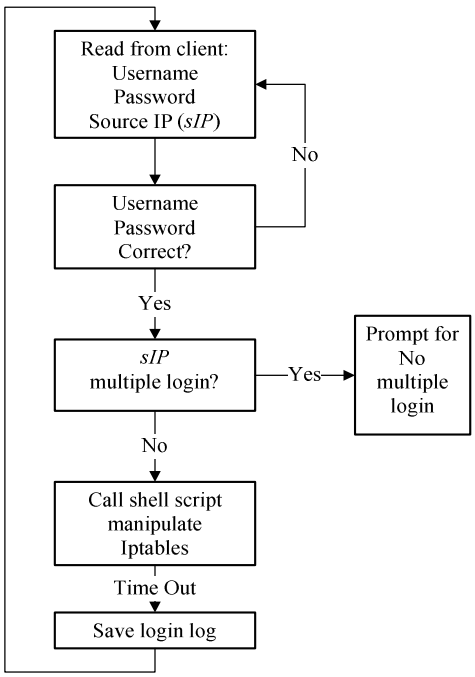

Fig. 2: The flow chart of our proposed KMS authenticating scheme

In Step 4 above, the shell script utilizes the iptables firewall and cooperates with kernel's "recent matching" module to avoids multiple abusive connections. It works as follows: after the first TCP SYN packet from one particular IP/host is arrived, the system will drop any other TCP SYN packet from the same IP address during the time period to avoid multiple connections (i.e., to avoid abusive usage). Fig.3 shows the general process flow of implementing the above idea.

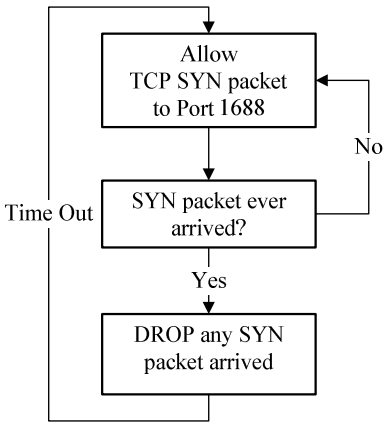

Fig. 3: The iptables configuration

Next, as shown in Fig.4 below, the following example code drops any other TCP SYN packets 200 seconds after the first arrival, where the $SIP variable stands for the source IP address for the client and TCP/1688 is the default port of the KMS:

```
iptables -A FORWARD \
        -p tcp -s "$SIP" –syn \
        --dport 1688 \
        -m recent --update --seconds 200 -j DROP

iptables -A FORWARD \
        -p tcp -s "$SIP" \
        --dport 1688 \
        -m recent --set -j ACCEPT
```

Fig. 4: The iptables code example

- Note: For insuring the successful working of this shell script, we need to enable and configure GNU/Linux kernel NAT function through the following commands, as shown in Fig.5, where $INTIF and $EXTIF representing the interface for private network and Internet respectively:

```
echo 1 > /proc/sys/net/ipv4/ip_forward
Iptables -A FORWARD
        -i $INTIF -o $EXTIF -j ACCEPT
```

Fig. 5: The pre-configuration of NAT

4.2. Avoiding abusive connections

Avoiding multiple connections to the KMS provider guarantees that there is only one Windows Vista installation being activated for each successfully login. On the other hand, by blocking multiple login from the same source IP address at the same time, it also guarantees no abusive usage for that IP address, and the jump gate problem as mentioned in Sec.1 will not show up.

Furthermore, in practice, we would also like to restrict the number of successful logins of each username per week/month. In this way, we can make this scheme more secure against abusive usages.

5. Implementation and Discussions

5.1. Our Implementation

The KMS can run on either Windows Vista or Windows Server 2003 installations, with a specialized activation key. As mentioned in Sec.3, under our implementation, the KMS provider is running Windows Server 2003, and the authenticator is running GNU/Linux with NAT configuration via iptables, while asking the account server through the POP3 (or MySQL) connections. The hardware platforms that we used for our implementation are two 1U servers with Dual-Processor Xeon 2.8GHz and 1GB RAM for both the KMS provider and the authenticator, respectively.

We implement and test our scheme on the NCTU campus agreement KMS [6], with over

250 different Vista installations activated during last 6 months. This system processes the KMS requests correctly and it is dependable for more widely and even lager environments. For more details, interested users please connect to the NCTU KMS site, http://kmserv.nctu.edu.tw .

Fig. 6: NCTU KMS Authentication System

Fig.7: Rules to deny abusive KMS connections

5.2. Main Contributions

In this paper, we design and implement a hybrid system solution (i.e., using both a Windows Vista for KMS provider and a GNU/Linux for KMS authenticator), which largely reduces the complexity of both end-user and server-side configuration tasks. In Table 3, we give some brief descriptions on the advantages of our proposed approach over Microsoft's scheme(s).

First, as compared to the Active Directory solution, our scheme could reduce the cost and complexity of constructing the whole Active Directory only for KMS and it could provide more types of authentications.

Second, as compared to the approach that only permits the hosts located inside the campus network, our proposed scheme could not only eliminate the possibilities of jump gates relaying traffic for adversaries from outside but also preserve the rights of valid users that accesses from non-campus network environment.

Third, as compared to the VPN approach, our scheme simplifies both the configuration of server-side and client-side, with the additional ability to deny abusive usages of the KMS (i.e., as shown in Fig.7).

Table 3: Main contributions

Item/Description	Advantages of our approach
1. As compared to the Active Directory solution	Our scheme could reduce the cost and complexity of constructing the whole Active Directory only for KMS and it could provide more types of authentications.
2. As compared to the IP restricting approach	Our proposed scheme could not only eliminate the possibilities of jump gates relaying traffic for adversaries from outside but also preserve the rights of valid users that accesses from non-campus source addresses.
3. As compared to the VPN approach	Our scheme simplifies both the configuration of server-side and client-side, with the additional ability to deny abusive usages of the KMS.

6. Concluding Remarks

In this paper, we design and implement a hybrid server system (i.e., with both Windows Vista and GNU/Linux) by integrating several tools to provide an effective authentication scheme for the Microsoft KMS authentication, with the ability to authenticate valid users, to avoid abusively usage, and to help track for each user account usage with relative simple client-side configuration. Using the proposed scheme, it is also very easy for us to integrate the authentication process with many different mail servers (i.e., faculty/student mail systems) under the campus or enterprise environments since we are using the standard POP3 interface. On the other hand, the type of the account server is not restricted to POP3. For example, many other interfaces provided by PHP libraries (i.e., LDAP, MySql, NIS, etc.) can be easily used to extend our scheme by some minor modifications of the PHP scripts.

For the deployment of KMS authentication

process on larger user environments, the administrators might also want multiple KMS providers for redundancy. In principle, this could be done by using Linux Virtual Server (LVS) [9] or some other similar techniques to extend our scheme. The redundancy of the authenticator of the scheme could also be provided by using LVS, Application Layer switches, or even DNS round-robin, which is also a way for more decent deployments.

References

[1]. Windows Vista: Home Page, http://www.microsoft.com/windows/products/windowsvista/default.mspx. (Retrieved on 2007-07-28.)

[2]. Windows Vista Volume Activation 2.0 Step-By-Step Guide, http://technet.microsoft.com/en-us/windowsvista/bb335288.aspx

[3]. Windows Vista Volume Activation 2.0 Technical Attributes, http://technet.microsoft.com/en-us/windowsvista/bb335289.aspx

[4]. Microsoft's Active Directory Page, http://www.microsoft.com/windowsserver2003/technologies/directory/activedirectory/default.mspx

[5]. Virtual private network (VPN), http://en.wikipedia.org/wiki/VPN (Retrieved on 2007-07-28)

[6]. The NCTU KMS web site: http://kmserv.nctu.edu.tw/

[7]. IPSec, “Security Architecture for the Internet Protocol”, RFC 4301, December 2005.

[8]. The iptables manual pages, http://www.netfilter.org/

[9]. Linux Virtual Server Project, http://www.linuxvirtualserver.org/