International Journal of Science and Research (IJSR)
ISSN (Online): 2319-7064
Index Copernicus Value (2013): 6.14 | Impact Factor (2013): 4.438

Secure Package Manager for UTM

Tom K Sunny[1], Beatrice Ssowmiya J[2]

[1]M.Tech Student, Department of Information Technology, SRM University

[2]Assistant Lecturer, Department of Information Technology, SRM University

Abstract: ***Unified threat management system(UTM) is an integration of standalone security products such as Firewall, IDS, Proxy, Antivirus etc. The proposed paper is for implementing a package management system for the UTM. Whenever a update is available for the different modules in UTM such as, an update released for antivirus or Firewall or IDS the package management system can be used for the update. The problem with the current system is the updates have to be installed manually by the user for every UTM implemented. The proposed system uses linux's advanced packaging tool for managing the PACKAGE update, removal, etc. Additional modules are added for checking the integrity of the packages.***

Keywords: Package, Encryption, Hashing, Package Management System

1. Introduction

In our current world some security technologies came to the market top list within few years. UTM is one of those security technologies. UTM help to improve the network security and performance of an organization in reduced network complexity and reduced cost.

Nowadays the networking environment are changing, new threats and attacks appear daily. As a result almost all organizations struggle to provide secure access to the users. Traditional approach was to install new security products individually for the new threats like antivirus , firewall etc. So for each new technology a new device should be installed. This approach increases the network complexity and cost of the Organization.

Thus as a solution for the above problem UTM is introduced. UTM came to the top of the market within few years. The evolution of UTM has produced many benefits across a wide spectrum of systems and users.UTM creates an environment in which all network security technologies come in a single. UTM enables the consolidation of all traditional as well as next generation firewall functions into a single device. Since UTM consolidate these all technologies the software update should be maintained for all by the UTM correctly. That is each and every individual security product in the UTM may have update releases in future. As more threats and technologies are evolving day by day updates and patches should be released for the security products to protect the network from new attacks. Manually installing the updates for each product in a UTM is a complex process. The objective of the paper is to present a method for implementing a package manager system for the UTM. A package management system is a collection of software tools that automates the process of installing, upgrading, configuring and removing software packages for a system.

2. Review of Literature

1) Unified Threat Management

Unified threat management is a network security platform that represents the next stage of evolution for traditional firewalls. A verity of hardware and software elements protect networks, including IPS, application control; content filtering, anti-virus and anti-spam software, and more. UTM delivers all these forms of protection on a single platform. UTM can be monitored from a single interface as a collective system. This decreases the network complexity and save a lot of time. A UTM mainly consist of three interfaces, the platform in which the UTM is built say LINUX based OS, the different security applications like antivirus, IDS, anti spam and finally the user interface.

2) Package

Software packages can be said as a piece of software that provide certain functionality. A package is typically provided as compile code, with additional meta information such as package description, package version, or dependencies. The package management system can evaluate this meta-information to perform different several functions like searching a package automatic update to a newer version, to check for dependencies etc.

3) Debian packages

Packages contains files necessary to implement a set of related commands or features. A debain system consist of mainly two types of packages, binary packages and source packages. A binary package contains executables, control files, copyright information and other documentations. A source package contains the original unmodified source.There are tools that generate binary packages but that packaging is build-system specific . Source packages can be built to produce binary packages on any other machine and architecture.

4) Package Management Systems

A package manager is software that support in managing packages. It will automate the process of installing, upgrading configuring and removing software packages. Their are different open source package management systems available in the market such as APT, YUM , RPM etc. APT refers to Advanced Package Tool; It is a set of core tools inside debian. APT helps to install, upgrade, configure and removing software packages in debian operating system. Their are several other operating systems that uses APT as their package manager.

5) Working of apt

When the debianos was built the main problem faced was, the users have to manually compile and install all packages to the system. For managing this problem the package management system APT is introduced. APT consists of a file named source.list all the package repositories are listed in this file. So for each request from the user apt will check the file and it will check all the repository locations listed in the source.list for the packages requested. Thus it will automatically install the packages from the repositories. The user can also edit the source.list file for adding or removing repositories.

6) Building a Debian Package

Making a debian package consist of mainly 3 steps
1. Making upstream tarball
2. Making source package
3. Making binary package

The packaging work flow is usually like this
STEP1. Rename the upstream tarball
STEP2. Unpack the upstream tarball
STEP3. Add the debian packaging file
STEP4. Build the package
STEP5.Install the package

7) Hashing

A hash function is a one way encryption function that takes a variable size input plain text and generates a fixed size hash output. This hash function can be used for checking the integrity of information. Cracking a hash function is practically not feasible. So it is widely used for checking integrity. MD5 is one of the most secure hash function that is currently used. MD5 process a variable length text input into a fixed length output of 128 bits.

8) Encryption

Encryption is a method of encoding information in a way that only authorised persons can access the information. Encryption protects the confidentiality of the information. An Encryption process generate a cipher text output for a plain text input. It uses encryption algorithms for the encryption process. The generated cipher text can only be read if it is decrypted. Encryption are mainly classified into two types Symmetric key encryption and Asymmetric key encryption. In symmetric key encryption the key used for encryption and decryption are the same and in asymmetric key encryption uses a public key for encrypting the message and uses its private key for decryption.

3. Design and Working

The entire design consists of two modules, building a debian package and configuring the advance packaging tool. In the case of a UTM, It mainly consists of three layers, the platform on which it is built, the application in the UTM, and the user interface. So whenever a update or patch is released for an older version of softwares in these three layers the system should automate the update process. Figure 1 shows the different layers of a unified threat management system.

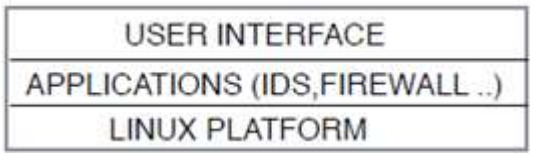

Figure 1: Layout of a UTM

In this case am considering a UTM that is linux based. The packages to the UTM should be a linux package. If apt is not available in the linux system that UTM uses it can be cross compiled and installed manually in the system. The packages that the Organisation developing should be stored in a repository in the server. The client systems source.list should contain the mirror location of the repository.

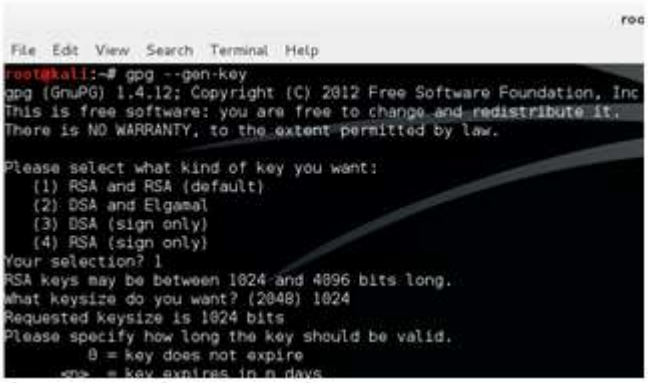

Figure 2: GnuPG key generation

APT will then take care of the packages that are released. Cronjob can be set in the UTM so that the UTM will check for updates and upgrades at a regular interval. For checking the integrity of the packages MD5 hashes are used. MD5 hash is used to generate a hash value for the new package made, and the hash value is added to a file.

Then the package and the file containing the hash is packed into a tar file. This file is then encrypted using RSA encryption algorithm. For encrypting, the proposed system uses linuxGnuPG tool. GnuPG is a open source tool available in all linux operating systems. A key pair is initially generated for the client machines using the GnuPGtool, a key pair contains a public key and a private key. Figure 2 shows the key pair generation using GnuPG.

Figure 3: Encryption using public key

All the public keys generated for the client machines are exported in to a key repository. Figure 3 shows encryption of a package using the public key named sample. The server can get the public key of the client from the repository and use that key for encrypting the package. When the client system downloads the new package it uses the private key

pair to decrypt the encrypted package and the same hash function is used to calculate the hash value. If both the hash values are same the package integrity is preserved.

The diagrammatic representation of the entire process is explained in the figure 4. That is in the server side the new package is initially hashed to get the hash valued of the package. After getting the hash value both the package and hash value are together encrypted using the public key of the client utm. The encrypted package is then stored in the package repository. When the utm update the packages i will initially decrypt the encrypted package using the private key of the client utm. Then the hash value for the decrypted package is calculated and then it is compared with the stored hash value. If both the hash value matches the system will install the package and if it doesn't match then the package is dropped. Thus the whole package management system is made secure using the encryption technique and hashing.

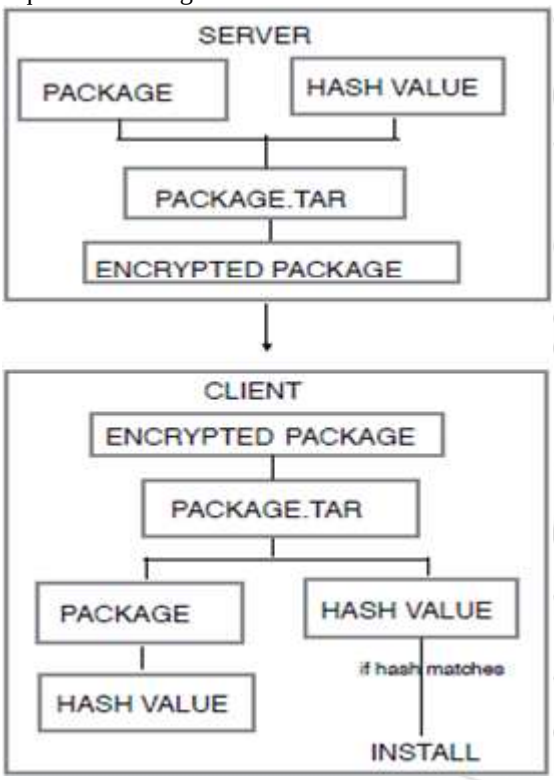

Figure 4: Diagrammatic Representation

4. Conclusion

The above paper proposes a way for implementing a package management system for a unified threat management system. The proposed system uses the open source advanced packaging tool as the package manager for the unified threat management system. Thus it helps to manage the functions like installing, updating, upgrading, removing packages in the unified threat management system. For maintaining the security of the packages MD5 hash function are used.

References

[1] UTM and Next generation firewall www.sophos.com/en-us/security-news/magic-quadrant-utm.aspx
[2] Package :en.wikipedia.org/wiki/package
[3] Debian Package http://www.debian.org/doc/manuals/debian-faq/ch-pkg_basics.en.html
[4] Package management en.wikipedia.org/wiki/package_manager
[5] APT https://wiki.debian.org/apt
[6] Apt packaging https://www.debian.org/doc/manuals/apt-howto
[7] Hashing Esiefarienrhe Michael Bukohwo, Fa'iz Ibrahim Jibia, "Design and implementation of data integrity system using MD5 cryptographic hash function"*The International Journal Of Engineering and Science(IJES) 2014*