Data Provenance Verification for Secure Hosts using Advance Cryptography Algorithm

Anirudha Vikhe
PG Student,Computer Dept.,
SKNCOE, Pune,
University of Pune

Prema Desai
Prof., Computer Dept.,
SKNCOE, Pune,
University of Pune

ABSTRACT
Malware is the intrusive program that affects computer operation and sensitive information of the host system. The objective is to protect such data and prevent malware from injecting fake keystroke into host network stack. The new technique cryptographic provenance verification [CPV] uses a property known as data provenance integrity which improves the trustiness of the system and its data. The system security is enhanced at kernel level. CPV makes use of trusted platform module for detection of fake key stroke. With TPM operating system can identify malware initiated network calls. The propose system consist of two modules sign and verify which prevent tampering of data. Sign module generates signature for outgoing packets from application layer. The packets are encrypted with advanced cryptography algorithm at transport layer and send to verify module along with communication key. Verify module decrypts the received packets and verify them for being malicious. TMP is used for secure key storage which prevents malware from injecting fake keystrokes.

General Terms
Data integrity, security, networking

Keywords
Data provenance, keystroke integrity, message authentication, malware attacks, advanced cryptography, universal hashing, trust platform computing.

1. INTRODUCTION
There is large number of computer those affected by malware software or programs. The different type of intrusive program such as spyware, virus, worm, bots, Trojan causes damage to system. By creating variety of problems as identity theft, DOS attacks, fake keystroke injection, disabling the firewall, spam, backdoor entries, increased computing cycles. Number of malware attacks has grown significantly which is threat to OS integrity. Rootkit affects the system data at kernel level. It is stealthy software that used for hiding the some processes and programs. Rootkits are difficult to detect.

The goal of this paper is to detect the intrusive attack on the system which mostly took place at kernel level and prevents them efficiently. OS integrity maintained with help of cryptographic provenance verification technique and trust platform module.

Basic idea is to identify whether the network call is initiated by user or malicious bot. The attacker mostly target the network layer with intend to modify system resources. Operating system is unable to identify original user call and malware initiated call.As result malware could easily bypass the firewall of system. But with cryptographic provenance verification technique avoids such attacks and improves the trustiness of data.

The TPM is basically micro controller with cryptographic capabilities and is designed to improve platform security with protected storage for keys and other important tasks. It is hardware but along with supporting software it provides functionality for root of trust. TPM hardware allows execution of some cryptographic functions. It protects the cryptographic function from outside agents by restricting access to such agents. With both hardware and software TPM helps to protect encryption as well as signature keys. TMP is designed for protecting keys when they not encrypted.

TMP provide space for two keys endorsement key and attestation key. Endorsement key is public or private key pair of size 2048 bits. It is a unique key and private key is generated within the TMP which confidential. Service provider is authenticated with attestation key.

2. RELATED WORK

2.1 Survey
Deian Stefan, *et al*.[3] proposed a robust bot detection approach know as Telling hUman and Bot Apart (TUBA) with TPM. This approach deals with the way human and bots interact with computer. It uses human characteristic behavior of user to differentiate between human and bots. There is difference in the human and bots interaction with system which is captured by TUBA. This difference is used to detect weather the user is human or not. For example human have unique rhythms while key stroking, unique surf patterns, clicking patterns.

Deian Stefan, *et al*. [5] used keystroke patterns of user for detection of system affected by malware and find out the malware attacks.The proposed model is cost effective biometric technique used for distinguishing human users. User`s key event pattern was monitor with framework called as TUBA. Attacker was considered to be automated rather than human user which is program that responsible for manipulations by producing fake key events. Keystroke patterns of the user were considered as effective tool for identifying malicious activities on system. By implementing two automatic attacks Gaussian bot and noise bot were used to inject key events by copied the user patterns. With these attacks TUBA framework gave accurate result using human key stroke as feature. TUBA proved to effective and advanced tool for malware detection.TPM was used for detecting fake key events and identifying intruders.

AratiBaliga*et al*. [6]proposed tool Gibraltar used maintaining integrity of kernel level data structure against the rootkit attack. It detects the rootkit at control as well as non-control

data. The method for detection of rootkit comprises of two phases, training phase which enforces some invariants over the kernel level data structures. At enforcement phase the invariants are checked for violation. Any violation of the invariants indicates the presence of rootkits.

AratiBaliga*et al.*[7] discovered new class malware attack on kernel which not dose to use any hiding methodthat was used by rootkits. The attacks are worst because damage done to computer not known to user and intrusion detection system. Without prior knowledge such type of attacks are difficult detect.

Table 1.Evaluation of Related Work

Referred topic	Description	Evaluation
Human characteristic behavior for malware detection (Deian Stefan)	Malware detection is based on characteristic behavior of user and which was used to avoid intrusive attack.	Telling hUman and Bot Apart (TUBA) technique is used bots detection.
Key stroke Integrity (Deian Stefan**)**	Key stroke event from user are used as behavior characteristic.	A cost effective biometric technique(TUBA) used for distinguishing human users.
Rootkit detection at kernel level (AratiBaliga)	Security to kernel data structure is provided against rootkits which modify the non-control data.	A technique is provided to detect kernel level attacks with tool Gibraltar that maintains kernel integrity.
Data tampering at kernel(Arati Baliga)	New class of attacks on kernel is reveled which needs the signature for detection.	Prototypes designed for such attack along with classification of data tampering methods.
Malware detection with BINDER(W eidong Cui)	Solution for automatic detection of break ins is provided which based on user intent and outgoing connections.	BINDER is host based system to detect break ins without need for signature new malware attack.
Data integrity at network (Jan Goebel**)**	Integrity of kernel is affected by intrusion through network.	N-gram analysis and scoring function is used to protect system in network against the malwares.

Weidong Cui *et al.* [8] proposed malware detection approach BINDER for detecting break-ins by capturing malicious extrusions. It works effectively and in efficient manner without requiring signature of various types of malware attacks. BINDER architecture detects new types of malicious attacks by establishing relationship with user intent and user key stroke or mouse clicks.

Jan Goebel *et al*. [9] provided approach for detecting affected system by intrusion on network. By using n-gram analysis with scoring function is used to serve the purpose. In network system is affected mostly by channels such as email or intrusive websites. Communication channel between bots and attacker provide a ways for detection.

All the above techniques are useful for increasing kernel level security.

2.2 Motivation

The modern day malware attacks are more sophisticated and complex which area of major concern. There are number of malware detection techniques used to protect kernel integrity and system data. The detection techniques are insufficient due stealthy and pervasive nature of attacks. Cryptographic provenance verification is efficient technique to protect sensitive information for system. This technique along with TPM provides the efficient way to prevent malware from fake key stroke injection with improved kernel level security.

3. PROPOSED MODEL

3.1 Model

The system will make use of trusted platform module and advanced cryptographic algorithm for security enhancements. The integrity of system is achieved by storing keys within TMP chip. The proposed system provides protection for both application data and Kernel data. It also protects the outgoing packets from application layer to the layers beneath by maintain the integrity of such packets. Original data is protected with provenance verification by providing resistant against malware attacks.Integrity application data and kernel data is maintained where application data is result of user actions and kernel data is system generated data.

For secure and effective malware detection two special modules are used, sign and verify. Both of these modules verify the key stroke entered by user with provenance approach. Sign module is placed at transport layer and verify module at network layer so that packets from network would be authenticated for signature at the verify module.

3.2 Key generation and exchange

Sign module establishes connection with verify module initial for key exchange process.Each of sign module and verify module have pair of public//private key. Initially Public key is shared between two modules.

Algorithm:

1) Connection establishment between sign and verify module.
2) Public key sharing with between sign and verify module.
3) Random key generation at sign module.
4) Random key generation at verify module.
5) Generation symmetric key and signing key using EOR operation with randomly generated numbers.
6) Sign module generates signature for packet with signing key and UMAC in encrypted format.

Sending the packet in encrypted with advanced cryptographic algorithm to verify module[2].

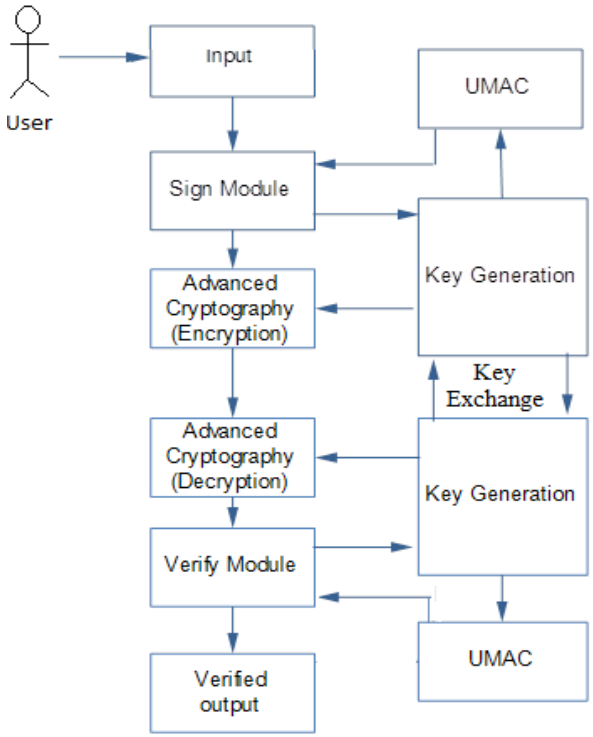

Fig 1: System overview

Use symmetric for at verify module for decryption of received packet and verifying it for being suspicious based on signature stored at hash table.

4. CONTRIBUTION AND FUTURE WORK

In this paper cryptographic data provenance verification approach is used for integrity of kernel level data. It provide secure channel for key storage. This system supports the malware detection using advanced cryptography and UMAC signature generation concept. System proposes the use of advanced method against forgery by storing cryptographic keys in TPM, which acts as sealed storage and identifies malicious attacks.

In future, this trusted approach could be used for distributed networks which are more vulnerablecollusive attacks.

5. ACKNOWLEDGMENTS

For proposing this model referred the IEEE Transaction paper under the title “Data Provenance verification for secure hosts” published in IEEE TRANSACTIONS ON DEPENDABLE AND SECURE COMPUTING, VOL.9 NO.2 YEAR 2012.This paper contains the results as per given on single system, which can be implement in real time within network.

6. REFERENCES

[1] KuiXu, HuijunXiong, Chehai Wu, Deian Stefan, DanfengYao Member, Data-Provenance Veri_cation For Secure Hosts , IEEE Transactions On Dependable andsecure computing vol.9 no.2 year 2012.

[2] Vishwagupta, Gajendra Singh, Ravindra Gupta Advance cryptography algorithmfor improving data security , International Journal of Advanced Research in Computer Science and Software Engineering (IJARCSSE)Volume 2, Issue 1, January 2012.

[3] D. Stefan, C. Wu, D. Yao and G. XU. Ensuring host integrity with crypto-graphic provenance verification. In CCSS 09,poster, November 10-12, Chicago, IL,USA,2009

[4] D.Stefan, C. Wu, D. Yao and G. XU. A Cryptographic Provenance Verification Approach ForHost-Based Malware Detection.

[5] D.Stefan and D. Yao. Keystroke-dynamics authentication against synthetic forgeries. In Proceedings of the International Conference on Collaborative Computing:Networking, Applications and Worksharing (CollaborateCom), November 2010

[6] A. Baliga, V. Ganapathy, and L. Iftode. Automatic inferenceand enforcement of kernel data structure invariants. In 24thAnnual Computer Security Applications Conference (ACSAC) 2008.

[7] A. Baliga, P. Kamat, and L. Iftode. Lurking in the shadows: Identifying systemic threats to kernel data. In IEEE Symposiumon Security and Privacy, pages 246–251. IEEE ComputerSociety, 2007

[8] W. Cui, R. H. Katz, andW. tian Tan. Design and implementationof an extrusion-based break-in detector for personal computers.In ACSAC, pages 361–370. IEEE Computer Society, 2005

[9] S. Garriss, R. C´aceres, S. Berger, R. Sailer, L. van Doorn, and X. Zhang. Trustworthy and personalized computing on public kiosks. In MobiSys ’08: Proceeding of the 6th international conference on Mobile systems, applications, and services, pages 199–210, New York, NY, USA, 2008. ACM.

[10] J. Goebel and T. Holz. Rishi: Identify bot contaminated hosts byIRC nickname evaluation. In Proceedings of the First USENIX Workshop on Hot Topics in Understanding Botnets, April 2007.

[11] R. Gummadi, H. Balakrishnan, P. Maniatis, and S. Ratnasamy. Not-a-Bot:Improving service availability in the face of botnet attacks. In Proceedings of the 6th USENIX Symposium on Networked Systems Design and Implementation (NDSI), 2009.

[12] S. W. Smith. Trusted Computing Platforms: Design andApplications. New York: Springer, 2005.

[13] B. Schneier and N. Ferguson. Practical cryptography, 2003.

[14] J. M. McCune, A. Perrig, and M. K. Reiter. Safe passage for passwords and other sensitive data. In NDSS. The Internet Society, 2009.

[15] M. Rajab, J. Zarfoss, F. Monrose, and A. Terzis. My botnet isbigger than yours (maybe, better than yours). In Proceedingsof the First USENIX Workshop on Hot Topics in Understanding Botnets, April 2007.