12-15-2019

Honeykeys: deception mechanisms in single packet authorization

Sergey Butakov

Pavol Zavarsky

Seyedmohammad Mirheydari

Honeykeys: deception mechanisms in single packet authorization
(research in progress)

Sergey Butakov[1], Pavol Zavarsky, Seyedmohammad Mirheydari
IS Security and Assurance, Concordia University of Edmonton,
Edmonon, Alberta, Canada

ABSTRACT

Single packet authorization is a technique that allows shielding a protected network service from an outside world. The protection is achieved by hiding the respective transport layer port until cryptographically protected packet received by another service authorizes port opening. The technique has a known weakness related to the key leakage. If secret key is known to the attacker, the shield can be removed by one message. The paper proposes to use a novel Honeykeys authorization scheme that is aimed at deceiving the attacker by storing decoy cryptographic keys on both server and client sides along with the actual keys. In such scheme, if keys are compromised it will not lead to the full-scale system compromise. In addition to that, Honeykeys scheme allows establishing segregation of duties in the authorization process and enables early detection of compromised keys. Apart from presenting theoretical concept of Honeykeys the paper shows preliminary implementation results from the pilot project. These results show acceptable authorization delay times imposed by additional security mechanism.

Keywords: Network Protection, Single Packet Authorization, Deception, Network services protection, Authorization, Honeypots

INTRODUCTION

Protection of computer networks and services has become a vital function in the modern information technology (IT). Elements of the critical infrastructure require reliable protective mechanisms to reduce the surface of the potential cyberattacks. One of the known approaches for such a protection is to put an additional shield around the administrative entry to the hardened

[1] Corresponding author. sergey.butakov@concordia.ab.ca | +1 780 413 7821

system and to remove this shield temporarily by a secret non-repeatable command when needed.

One of the examples of such protection for networking services is port knocking technique, which was proposed in early 2000s (Barham, et al. 2002), (Krzywinski 2003). The technique later evolved into single packet authorization and related approaches discussed in the second section of the paper. Port knocking and similar technologies heavily rely on the security of the authorization keys. If attacker get access to a compromised port knocking key, then the concealed ports cannot be considered protected any more. It is especially true in case if a malicious insider is involved in the. This position paper explores the possibility to use deception mechanism to detect malicious insider and avoid system compromise. The novelty of the proposed approach includes ability to segregate user access rights for three types of users: ones who issue commands on the protected system, ones who authorize the commands and ones who have access to the cryptographic keys.

RELATED WORKS I: SINGLE PACKET AUTHORIZATION

The idea of additional layer for network service protection by hiding the respective port was suggested in 2002 in the form of port knocking (Barham, et al. 2002). Firewall closes protected ports and there is a daemon on the server which intercepts the incoming packets and waits for a set of predefined sequence of port knocks without providing any receipt to sender (Barham, et al. 2002). After receiving the predefined sequence, the daemon opens the desired port(s) on the firewall so that the connection can be established during a relatively short time slot. Although this approach has numerous benefits and had been commonly used by system administrators for a variety of security applications (Krzywinski 2003), simple implementations of port knocking itself have known issues such as replay attack, out-of-order packet delivery, and susceptibility to scanners (Manzanares, et al. 2005).

One of the techniques that evolved from port knocking was Single Packet Authorization (SPA) (Rash 2007). In SPA model, the server is also on the default dropstance and monitors the incoming packets but SPA has solved many of port knocking issues by using a single encrypted authorization UDP packet over the application layer (Rash 2007). SPA has been improved by the number of techniques, including, for example, an approach that suggested to use out of band communication over mobile network to deliver secondary authorization (Liew, et al. 2010). Such approach allows to avoid brute-force attacks on cryptographic keys. Port knocking - based techniques become popular protection tool for sensitive networking services. A number of SPA-based commercial solutions are being offered on the market by such vendors as CryptZone or Vidder. Another example of SPA evolution is WebSPA project supported by OWASP (OWASP 2017) that uses web application to send the authorization payload through properly secured web server communication.

One major problem remains open with SPA and related techniques: compromise of the keys by external attacker or malicious insider leads to the single point of failure. If authorization key(s) appear in the wrong hands, the SPA layer can be defeated by one single packet and this action might not be even noticed by the unsuspecting server that trusts client based on the key(s) provided. In such case security of the database with authorization key(s) becomes potentially the weakest link and a single point of failure for SPA-based layer of server protection. One potential way to address this issue is to use well-known key/password deception mechanisms that would help to confuse the potential attacker and also provide early detection mechanisms to alarm the server about potentially compromised keys on the client site.

RELATED WORKS II: DECEPTION MECHANISMS IN PASSWORD STORAGE

Deception mechanisms are being used in various security applications. Taxonomy of the such mechanisms, provided in (Almeshekah, Spafford and Atallah 2013), shows that they include honeypots, DNS re-directions, fake sites, fake keys and accounts, user jailing, anti-forensics tools and honeytokens in different forms. Honeytokens have been implemented in such projects as Kamouflage (Bojinov, et al. 2010) and Honeywords (Juels and Rivest 2013). In these two implementations and similar projects decoy passwords are being inserted in the password database to deceive potential thieve of credentials. Authentication system uses an additional hardened server called Honeychecker to verify if the matching password is real or decoy. In case if decoy password is discovered, system may silently switch to the fail-secure mode, trigger an alarm and direct potential attacker to the honeypot. Use of decoy passwords has been also proposed as part of SAuth authentication scheme (Kontaxis, et al. 2013).

Two important features are being added by honeywords to improve system security. First is the attacker confusion. Attacker needs to choose one of the passwords (hashes) that are stored for every user. If decoy passwords are “flat” – e.g. resemble the actual password then the attacker’s choice will not be obvious (Juels and Rivest 2013). Second feature is the ability of Honeychecker to detect the password database leaks: if improper index is being sent to the Honeychecker it is very likely that non-authorized user has access to decoy passwords and thus the database with the credentials had been compromised.

This paper proposes to use similar principle to store secret keys in the Single Packet Authorization (SPA) scheme: store decoy keys along with the proper one(-s) and implement Honeychecker to verify suitability of the candidate key. The following section outlines Honeykeys: a novel approach to SPA security that is using technique similar to honeywords in order to add the following features to SPA security: protection from key leakage and segregation of duties for sensitive authorization operations.

USING HONEYKEYS IN SINGLE PACKET AUTHORIZATION

Deception mechanisms are being used in various security applications. Taxonomy of the such mechanisms, provided in (Almeshekah and Spafford 2014), shows that they include honeypots, DNS re-directions, fake sites, fake keys and accounts, user jailing, anti-forensics tools and honeytokens in different forms. This research is proposing to use honeytokens to improve security of authentication mechanisms in SPA. Formal model for the proposed protocol includes 9 elements each of which knows only certain pieces of information as per the table below:

Table 1. Elements of the system and information available to them

Element of the system	Known information	
Key administrator (KA)	{PubK}, {PrivK}, {PIN}	- set of public keys used for authorization - set of private keys used for authorization - set of potential PINs
SPA Client (SC):	{PubK}	
SPA Server (SS):	{PrivK}	
Authorizing User (AuthU):	{PIN}, PIN_{secret}	
Honeychecker (HC):	PIN_{secret}	- secret PIN used for authorization
Application User (AppU):	PIN_{secret}	
Firewall (FW): --		--
Protected Service (PS):		--
Honeypot / Decoy Service (HD):		--

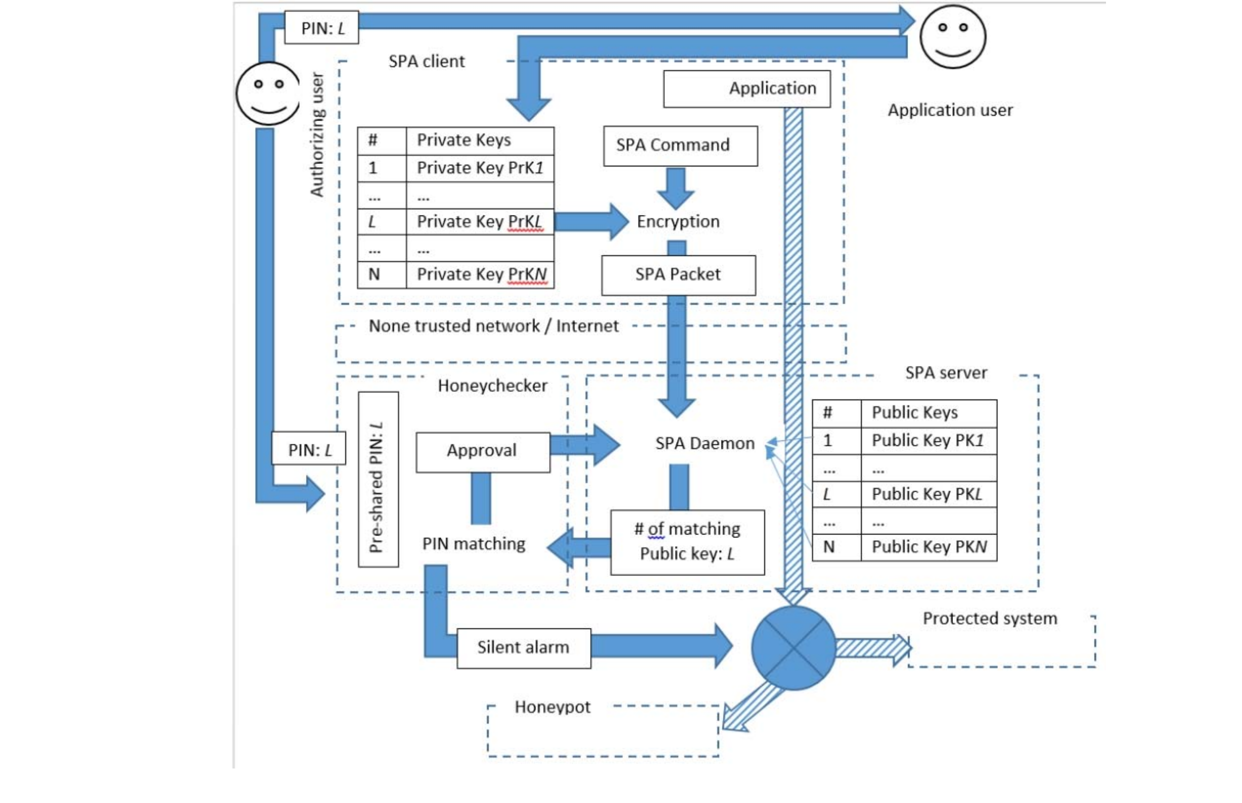
The information flow in the proposed approach is represented in figure 1. The diagram includes the following main elements:

- Users. There are three groups of users: 1) Authorizing user (AuthU). Role of the Authorizing user is to insert PIN_{secret} into Honeychecker database and share same PIN_{secret} with the Application User (AppU). PIN_{secret} represents the index of the actual key in the database. All other keys considered decoy keys. In the simplest case, PIN could be a sequential index of the key in the key storage. 2) Key administrator (KA). Role of this user is to generate cryptographic key pairs {PubK} and {PrivK} and subsequently transfer {PubK} to SC and {PrivK} keys to the SPA Server (SS). To simplify the picture the role of KA is not represented on the Figure 1. 3) Application user. Role of the user is to input $\mathrm{PIN}_{\mathrm{secret}}$ in the system and send authorization package to allow application to connect to protected system. As it can be seen from the description of the roles, proposed authorization scheme allows segregating duties of KA and AuthU. In case if KA is compromised, the system still stays in much more secure mode than a typical SPA-based system as KA role does not know the $\mathrm{PIN}_{\mathrm{secret}}$ thus imposter will face the risk of detection.

- SPA Server (SS) receives SPA packet, finds if the packet is decryptable with one of the {PrivK} keys and if yes verify index of the key with the HC. In case of positive verification from the HC SPA Server opens access to the Router.
- SPA client (SC) allows AU to select one of the private keys by PINsecret and send authorization packet to the SS to enable Application access to the Protected system.
- Honeycheker (HC) keeps the PINsecret provided by an AuthU. It verifies if index submitted by SS matches the PINsecret and in case of match sends authorization back to SS. If index does not match, it may generate positive or negative answer to the SS depending on the selected mode of operations and generates an alarm to the Router.

Proposed Honeykeys authorization scheme provides the following advantages:

- Segregation of duties allows protection from the malicious insider. Unlike in the original SPA scheme, case of compromised key administrator does not lead to the compromise of the system. Compromised key administrator may insert non-matching private/public key pairs thus leading to the denial of service but the Protected system still stays shielded from the outside world. In the same manner, compromised authorizing user (ZU) and application user do not have direct access to the keys stored in the application.



#	Private Keys
1	Private Key PrK*1*
…	…
L	Private Key PrK*L*
…	…
N	Private Key PrK*N*

#	Public Keys
1	Public Key PK*1*
…	…
L	Public Key PK*L*
…	…
N	Public Key PK*N*

Figure 1. Information flow in the proposed protocol

- Since private and public keys on both SPA client and SPA server are essentially random sequences there is no way for the attacker to guess which key "looks" like the proper one. Based on that it can be stated that the problem of creating "flat" decoy keys \cite{juels2013honeywords} is not relevant to the proposed scheme.

- All the advantages of the original SPA, such as confidentiality of the message or protection against replay attack are inherited in the proposed Honeykeys scheme.

CONCLUSION

The paper proposed Honeykeys - a novel authentication scheme for Single Packet Authorization (SPA) that uses decoy passwords to add additional layer of protection against the potential loss of the cryptographic keys. Such addition allows establishing additional layer of protection against malicious insiders by enabling early warnings on the keys leakage and ability to silently re-route an attacker to the honeypot. In addition, Honeykeys authorization allows separation of secret key administrator duties from authorizing person and application user. As a step to continue the research authors are working on the experimental software for proposed changes in the SPA authorization scheme. Another direction for further research will be to extend the protocol to multi-user and multi-command arrangements as well as adding key rotation mechanisms to deal with potential replay attacks.

REFERENCES

Almeshekah, Mohammed H., and Eugene H. Spafford. "Planning and integrating deception into computer security defenses." *Proceedings of the 2014 New Security Paradigms Workshop.* Victoria, BC, Canada: ACM 978-1-4503-3062-6/14/09 ...$15.00., 2014.

Almeshekah, Mohammed, Eugene H. Spafford, and Mikhail J. Atallah. *Improving security using deception.* Technical Report, Center for Education and Research Information Assurance and Security, Purdue University, 2013.

Barham, Paul, Steven Hand, Rebecca Isaacs, Paul Jardetzky, Richard Mortier, and Timothy Roscoe. *Techniques for lightweight concealment and authentication in IP networks.* Intel Research Berkeley, 2002.

Bojinov, Hristo, Elie Bursztein, Xavier Boyen, and Dan . Boneh. "Kamouflage: Loss-resistant password management." *European Symposium on Research in Computer Security.* Springer Berlin Heidelberg, 2010. 286-302.

Juels, Ari, and Ronald L. Rivest. "Honeywords: Making password-cracking detectable." 2013.

Kontaxis, Georgios, Elias Athanasopoulos, Georgios Portokalidis, and Angelos D. Keromytis. "SAuth: protecting user accounts from password database leaks." *2013 ACM SIGSAC conference on Computer & communications security.* ACM, 2013. 187-198.

Krzywinski, Martin. "Port knocking from the inside out." *SysAdmin Magazine* 12, no. 6 (2003): 12-17.

Liew, Jiun-Hau, Shirly Lee, Ivy Ong, Hoon-Jae Lee, and Hyotaek Lim. "One-time knocking framework using SPA and IPsec." *2010 2nd International Conference on Education Technology and Computers (ICETC).* Busan: IEEE, 2010.

Manzanares, Antonio Izquierdo, Joaquín Torres Márquez, Juan M. Estevez-Tapiador, and Julio César Hernández Castro. "Attacks on port knocking authentication mechanism." *International Conference on Computational Science and Its Applications.* Springer Berlin Heidelberg, 2005. 1292-1300.

OWASP. *OWASP WebSpa Project.* 2017. https://www.owasp.org/index.php/OWASP_WebSpa_Project (accessed June 07, 2019).

Rash, Michael. "Single Packet Authorization2007." *Linux Journal*, no. 156 (2007).