Active Directory Kerberoasting Attack: Monitoring and Detection Techniques

Lukáš Kotlaba, Simona Buchovecká and Róbert Lórencz
Department of Information Security, Faculty of Information Technology, Czech Technical University in Prague, Czech Republic

Keywords: MS Active Directory, Kerberos Security, Kerberoasting, Cyber Security, Cyber Attacks.

Abstract: The paper focus is the detection of Kerberoasting attack in Active Directory environment. The purpose of the attack is to extract service accounts' passwords without need for any special user access rights or privilege escalation, which makes it suitable for initial phases of network compromise and further pivot for more interesting accounts. The main goal of the paper is to discuss the monitoring possibilities, setting up detection rules built on top of native Active Directory auditing capabilities, including possible ways to minimize false positive alerts.

1 INTRODUCTION

Active Directory (AD) is a proprietary implementation of a directory service for Microsoft's Network Operating System (NOS). NOS is the term used to describe a networked environment in which various types of resources, such as user, group, and computer accounts, are stored in a central repository. This repository, called Active Directory, contains network, application, or NOS information that is controlled by administrators and accessible to end users. The directory service that provides access to this repository is called Active Directory Domain Services (AD DS) (Desmond et. al. 2013).

The AD is widely used as the core part of the whole network infrastructure; as a central repository for information about objects that reside on a company network, such as users, groups, computers, printers, applications, and files. The objects have numerous attributes, specific permissions, and relations. AD stores all this data in a hierarchical organizational structure and provides access to it for users.

Microsoft Active Directory is based on the LDAPv3 protocol, which is an updated version of LDAP, introduced in 1997. The first version of Microsoft AD was released with Windows 2000 and has been a part of Windows Server operating systems (OSs) ever since. (Desmond, 2013, Francis, 2017)

As such, AD is a very attractive target for attackers and cybercriminals. It is crucial to understand how important role Active Directory plays in an enterprise domain, and what kind of data it stores. Thus, it is not surprising that AD is often a target of attacks. Indeed, AD does not even have to be the target itself, as it may only serve as a bare tool providing a path for compromising more interesting systems in the domain, as discussed in (Kotlaba, 2019).

The paper is focused on one such attack – Kerberoasting – the purpose of which is to extract service accounts' passwords without need for any special user access rights or privilege escalation. Discussion on the techniques for attack detection in (almost) real time is presented, including the monitoring scenarios and tuning options for minimizing potential false positive alerts.

The paper is structured as follows - Section 2 contains background information on authentication process in Active Directory environment, with focus on Kerberos protocol. Further, details of the Kerberoasting attack itself are discussed. Section 3 presents results of our work – design and implementation of monitoring scenarios for detecting the Kerberoasting attack, including discussion on efficiency and minimization of false positive alerts. Section 4 concludes the paper.

Kotlaba, L., Buchovecká, S. and Lórencz, R.
Active Directory Kerberoasting Attack: Monitoring and Detection Techniques.
DOI: 10.5220/0008955004320439
In *Proceedings of the 6th International Conference on Information Systems Security and Privacy (ICISSP 2020)*, pages 432-439
ISBN: 978-989-758-399-5; ISSN: 2184-4356

2 BACKGROUND

The Windows OSs require all users to log on to the computer with a valid account to access local and network resources. Authentication is a process of verifying the claimed identity of an object; authorization is a process of verifying that the object has rights to access particular resources. AD is the default technology for storing identity information on domain-joined systems, and therefore it is tied closely to authentication and authorization processes. Microsoft documentation provides details on key concepts (Microsoft, 2016) as follows.

2.1 Windows Authentication Overview

Users are authenticated to Windows-based computers by a logon process. Depending on how the logon process occurs, there are several scenarios defined:

- Interactive logon
  - Local logon
  - Remote logon
- Network logon
- Smart card logon
- Biometric logon

During an interactive logon, a user typically enters credentials in the credentials' entry dialog box. Alternatives for presenting credentials in the form of username and password are smart card logon and biometric logon.

Users can perform an interactive logon by using a local account or a domain account. Depending on the account type, the logon process confirms the user's identification to the security database on the user's local computer or to the AD database. A local logon grants a user permission to access resources on the local computer or resources on networked computers. A domain logon grants a user permission to access local and domain resources. Domain logon requires that both the user and computer have their accounts in AD and the computer is physically connected to the network.

A network logon can only be used after user, service, or computer authentication has taken place. The network logon process does not use the credentials entry dialog boxes; the authentication is typically invisible to the user unless alternative credentials have to be provided. Previously established credentials are used to confirm identity to any network service that the user is attempting to access.

Various authentication protocols are used to provide network logon functionality, Kerberos protocol being the preferred authentication method in AD environment.

Windows OSs implement Kerberos version 5 authentication protocol, which is specified in RFC 4120 (Neumann, 2005). Microsoft's proprietary implementation of this protocol adds some functionality beyond the RFC specification, such as authorization or optional Privilege Account Certificate (PAC) validation (Microsoft, 2019).

Kerberos is the default protocol used within an Active Directory domain. With Kerberos, passwords never traverse the network in plaintext or encrypted formats. Instead, session-specific keys are generated for use over a short period of time through the use of tickets. The tickets are issued by Kerberos Key Distribution Center (KDC), which is integrated into a domain controller (DC) in the Microsoft's Kerberos implementation. The KDC uses the AD as its security account database.

Figure 1 illustrates Kerberos authentication steps (Desmond, 2013 and Metcalf, 2014), which occur when a user attempts to access a service:

1. To begin the authentication process, an AS_REQ message is sent from client to KDC. This message proves the user's identity and is partially encrypted with a hash of the user's password computed by the client computer.

2. The DC validates the request and produces a Ticket Granting Ticket (TGT). The TGT is sent back to the client as AS_REP message. The TGT contains PAC with information about all the security groups in which the user is a member. It is encrypted and signed by the KDC service account (*krbtgt*). The client caches the TGT in memory.

3. The client sends a TGS_REQ message to the DC to request a service ticket for a specific service. Rather than providing credentials again, the message contains the cached TGT obtained in the previous step.

4. The DC validates the TGS_REQ and constructs a Ticket Granting Service (TGS) ticket for the requested service. The TGS ticket, partially encrypted with a hash of the service's password, is sent back to the client in a TGS_REP message. The client caches this ticket in memory for subsequent use when authenticating directly to the service.

5. The client presents the TGS ticket to the service in an AP_REQ message. The service uses it to authenticate the user. The service might also use the user's access token (contained in the ticket) to perform authorization before allowing access.

6. Optionally, the service can respond with an AP_REQ message for mutual authentication of the service.

7. Optionally, the service may also send the TGS ticket to a KDC to validate the PAC to ensure the user's group membership presented in the ticket is accurate.

8. If the PAC validation occurs, the KDC informs the server hosting the specific service about the validation result.

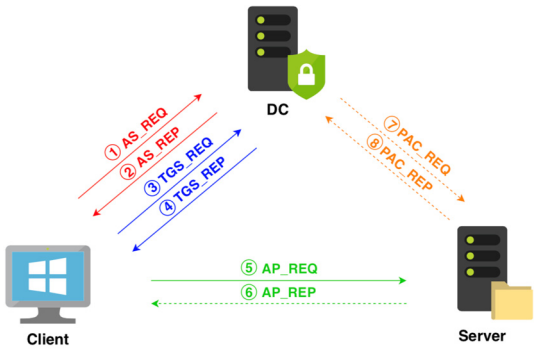

Figure 1: Kerberos authentication.

Kerberos allows users to access services on the network transparently by simply requesting a service ticket. When clients request service tickets for given services from a DC, they use identifiers called Service Principal Names (SPNs). An SPN is stored in AD, in the *servicePrincipalName* multivalued attribute. It is constructed in the form of a service identifier, followed by the hostname, and optionally, a port number. The service identifier is a predefined string that the client and server agree on. To enable authentication, Kerberos requires that SPN be associated with at least one service logon account (Desmond et. al. 2013).

All the authentication attempts, successful and not successful, are being audited. The event logging service records events from various sources and stores them in a single collection called Windows Event Log.

Several categories provided by the security audit policies represent an essential source of information for hunting attacks towards Active Directory. For instance, the categories Account Logon and Logon/Logoff track authentication and use of credentials, which is the core element of the attacks. Categories Account Management and DS Access record changes and replication of the AD schema. Other categories, such as Detailed Tracking, Object Tracking, and Privilege Use provide useful information that may be related to attack preparation, use of hacking tools, or resource access after the successful attack execution.

We will utilize the native auditing capabilities to build our detections later on.

2.2 Kerberoasting

The Kerberoasting attack was first introduced by Tim Medin (Medin, 2014), with the goal to crack passwords for remote service accounts completely offline, without sending a single packet to the service, and without requiring special or escalated privileges.

Since any authenticated user possessing a valid TGT may request one or more TGS tickets for any SPN from a domain controller, this process can be abused by adversaries in the Kerberoasting technique. An attacker that controls a user account can request a service ticket. The ticket may be encrypted with a weak cipher suite, such as RC4-HMAC-MD5, which means the service account's NT password hash is used to encrypt the service ticket. The attacker then exports the ticket from memory and attempts to crack it offline by trying different NT hashes. When the ticket is successfully opened, the correct service account password is discovered in plaintext. Cracking of hashes is usually done on adversary-controlled systems with high computational power, outside of the target network (MITRE, 2018, Metcalf, 2017).

Table 1: Encryption types implemented in Windows.

Type	Cipher suite name
0x1	DES-CBC-CRC
0x3	DES-CBC-MD5
0x11	AES128-CTS-HMAC-SHA1-96
0x12	AES256-CTS-HMAC-SHA1-96
0x17	RC4-HMAC-MD5
0x18	RC4-HMAC-EXP

Table 1 shows implemented encryption types used by Kerberos in Windows OSs. Starting from Windows Server 2008 and Windows Vista, the suites containing AES cipher have been set as default, replacing previous default RC4 cipher suites. Also, cipher suites involving DES cipher have been disabled starting from Windows 7 and Windows Server 2008 R2 (Microsoft, 2017).

These updates comply with security issues arising from RC4 and DES ciphers, as these ciphers are considered obsolete nowadays. However, Windows allows enabling these suites via policy setting for backward compatibility (Microsoft, 2017).

The main reason why Kerberoasting is successful is underrated administration of service accounts in organizations. Many service account passwords are often weak, and of the same length as the configured domain password minimum. Another problem is that service accounts often don't have passwords set to expire. Furthermore, most service accounts are over-permissioned; they contain rights to access certain objects or rights equivalent to Administrator (Metcalf

2017).

The first step of the Kerberoasting attack is usually SPN scanning. Querying for registered SPNs enables an attacker to identify all service accounts supporting Kerberos authentication together with their role. Checking whether the service accounts have the attribute *AdminCount* equal to "1" identifies accounts which are members of highly privileged groups. Attackers use these methods to identify interesting service accounts to focus on (Metcalf, 2017).

Kerberoasting and SPN scanning can be performed directly from PowerShell (Metcalf, 2017), or by using various tools. Such tools include PowerShell script *Invoke-Kerberoast*, which is also part of the offensive framework *Empire* (Schroeder, 2016), or *GetUserSPNs* module of *Impacket*, which is a collection of Python classes for working with network protocols (SECURAUTH, 2019).

From the nature of Kerberos authentication and the fact that usage of services is standard behavior in an AD domain, there is no mechanism of how Kerberoasting can be prevented by firewalls or IDS/IPS devices. Furthermore, the obsolete cipher suites are commonly enabled in the environment due to backward compatibility. This implies the need for monitoring and detection of Kerberoasting attack in the domain.

3 PROPOSED MONITORING APPROACH

The process of designing the detection rules starts with defining all related log sources that may contain relevant data. For events, it is crucial to identify what information they carry, and under which circumstances they are logged, or whether they are generated at all. In many cases, also a trade-off between the added value and the volume of generated events has to be taken into consideration. Microsoft's documentation of Advanced security audit policy settings (Microsoft, 2017) and Randy Franklin Smith's Log Encyclopedia (Smith, 2006) are the ultimate reference sources of event descriptions, logging settings, event occurrences, and other information related to Windows Event Log.

After a scenario is designed, it is necessary to test its detection capabilities and evaluate the relevancy of the returned results. We have tested all proposed detection rules and evaluated their efficiency from the perspective of True Positive/False Positive ratio. As it shows, the naive approach produces a high number of False Positive alerts, and thus, we focused on the detections tuning in the end.

For practical testing we used a virtual lab environment to simulate an example of a small domain, consisting of one physical machine and five virtual machines (VMs). The host computer ran Linux OS and was network-connected with the VMs to receive logs. The VMs include two servers, one DC (DC01) and one member server (SERVER2008), and two users' workstations (WINDOWS7 and WINDOWS10). The last VM (kali) runs Kali Linux distribution and serves as a simulation of an external attacker having network connectivity to the domain. Logs from all monitored assets are sent to the physical machine where they are indexed by a Splunk instance. The described environment is illustrated in Figure 2.

Splunk is a software product that enables to search, analyze, and visualize the data gathered from the components of IT infrastructure or business, it takes in data from websites, applications, sensors, devices, etc. (Splunk 2019). We have used Splunk instance as a central collection point for Active Directory logs, as well as central monitoring point for our designed detections – all the presented scenarios were developed in Splunk Processing Language in form of detection searches from the collected audit data.

For testing the designed scenarios, we used three tools to request a service ticket:

- GetUserSPNs module of Impacket (SECUREAUTH, 2016);
- Invoke-Kerberoast module of Empire (Schroeder, 2016);
- PowerShell commands based on (Metcalf, 2017).

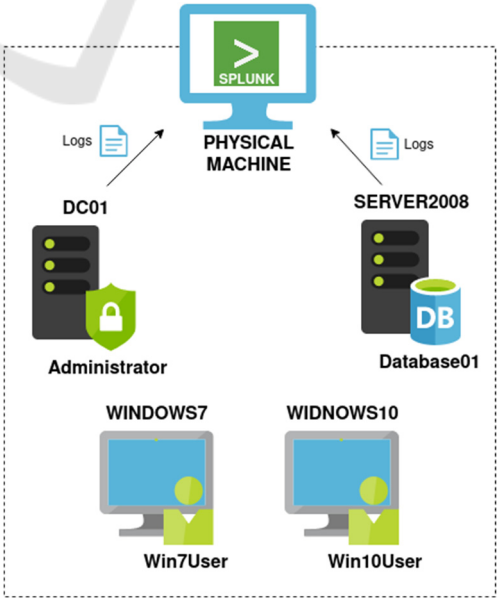

Figure 2: Lab environment.

3.1 Log Sources

The Kerberoasting technique is targeting Kerberos mechanism used to authenticate users who access protected network resources. The variety of events which contain useful information for this scenario narrows to a single subcategory of Advanced security audit policies: Account Logon\Kerberos Service Ticket Operations. This policy subcategory should generate three events:

- 4769(S, F) A Kerberos service ticket was requested;
- 4770(S) A Kerberos service ticket was renewed;
- 4773(F) A Kerberos service ticket request failed.

The Microsoft documentation narrows the choice of events even more. The event 4773 is defined but never invoked, and failure event 4769 is generated instead. Event 4770 logs every TGS ticket renewal. However, it has only informational character, and no security monitoring recommendations exist for it (Microsoft, 2017).

The event 4769 generates every time KDC gets a Kerberos TGS ticket request. The event generates only on DCs, however, it is one of the most numerous events logged (Metcalf, 2017). This event contains lots of valuable information, including account, service, or network information, encryption type used, and failure code. It is a key element for monitoring suspicious activities related to services.

Another type of logs that may be useful for this scenario, although not so directly, are PowerShell logs. PowerShell Script Block Logging records compiled blocks of scripts into event 4104; PowerShell Module Logging records module usage into event 4103.

3.2 Detection Scenarios

Kerberoasting technique, as described in the previous section, involves the use of a valid domain user's authentication ticket (TGT) to request one or several service tickets using their SPNs. Since the goal of an attacker is to crack the service ticket offline, tickets encrypted with weak cipher suites are preferred.

Sean Metcalf did some research and published several articles on this topic, which name elements suitable for detection of Kerberoasting. We were inspired by ideas published in these articles (Metcalf, 2017) while designing the detection scenarios.

3.2.1 Detecting Kerberoasting via Event 4769

Unless there are incompatible or legacy systems used in the environment, all Kerberos authentication should use AES cipher suites, and therefore, any requests for TGS tickets with lower encryption types can be considered suspicious. The detection rule *D01 - Possible Kerberoasting activity* looks for any ticket requests with encryption type constants equal to the values of these cipher suites (visible from Table 1). The snippet of the search is in Listing 1.

Listing 1: D01 – Possible Kerberoasting activity (snippet).

```
source=XmlWinEventLog:Security
EventCode=4769 (TicketEncryptionType=0x1 OR
TicketEncryptionType=0x3 OR
TicketEncryptionType=0x17 OR
TicketEncryptionType=0x18)
|eval Source=if(IpAddress=="::1", Computer,
IpAddress)
|table _time, host, Source, TargetUserName,
ServiceName, TicketEncryptionType
|sort - _time
| ...
```

3.2.2 Suspicious Service Ticket Requests

The next two detection searches focus on service ticket requests and aim to detect suspicious usage of services more generally. The rule *D02 - Excessive service ticket requests from one source* (Listing 2) triggers if there is a higher amount of different service requests observed in a short time from a single source. This kind of activity is even more suspicious if the service names are not related to each other, or if the type of requested services is unusual for that particular source.

The search uses events 4769. Service ticket requests for *krbtgt* service and computer account service names (those ending with $ character) are filtered out from the results, as the search focuses mostly on service accounts that were intentionally created for specific resources. Subsequent events are grouped on *IpAddress* field by the *transaction* command. The number of services in each transaction is calculated and filtered to display only results where the number is higher than the one specified in the condition. The number constant and time span used in the condition represent a variable and have to be adjusted to the needs of the particular environment. The values presented in the search snippet were used in the lab environment.

Listing 2: D02 – Excessive service ticket requests from one source (snippet).

```
source=XmlWinEventLog:Security
EventCode=4769 ServiceName != krbtgt
|regex ServiceName != "\$$"
|transaction IpAddress maxpause=5m
maxevents=-1
```

```
|eval services=mvcount(ServiceName)
|where services > 5
| ...
```

Listing 3: D03 – Suspicious external service ticket requests (snippet).

```
source=XmlWinEventLog:Security
EventCode=4769 IpPort > 0 (IpPort < 1024 OR
(NOT (IpAddress=10.0.0.0/8 OR
IpAddress=172.16.0.0/12 OR
IpAddress=192.168.0.0/16 OR
IpAddress=127.0.0.1 OR IpAddress=::1)))
| ...
```

Another search, *D03 - Suspicious external service ticket requests*, follows a security recommendation described by Microsoft in its documentation for the event 4769 (Microsoft, 2017). The search focuses on network information provided in the event. It monitors usage of well-known ports or any events where the IP address is not from the private IP ranges, which are signs of an outbound connection. The Listing 3. shows the detection logic used.

The range of IP addresses can be narrowed to only those used in the environment. If there is a scenario where the monitored ports or IP addresses are used by legitimate services, the values can be whitelisted by modifying the detection condition.

3.2.3 Detecting Kerberoasting with a Honeypot

In one of his articles, Sean Metcalf presents an effective method on how to detect Kerberoasting (Metcalf, 2017). He suggests creating a honeypot - a fake account, with a fake SPN associated, having some attributes (e.g. *AdminCount*) set, making it attractive for potential attackers. This account has no effective role and privileges in the environment; it is created merely to attract attackers. Monitoring service ticket requests for this account gives clear results of malicious activities with a low false positive ratio, since there is no legitimate reason to request tickets for this service.

We named the account *Honeypot01* for illustration, but the account should look as legitimate as possible in reality. Apart from the *AdminCount* attribute set, it could be a member of seemingly privileged groups to lower potential suspicions of an attacker. Listing 4 shows the detection rule *D04 - Detecting Kerberoasting with a honeypot*.

Listing 4: D04 – Detecting Kerberoasting with a honeypot (snippet).

```
source=XmlWinEventLog:Security
EventCode=4769 ServiceName=Honeypot01
|eval Source=if(IpAddress=="::1", Computer,
IpAddress)
|table _time, host, Source, TargetUserName,
ServiceName, TicketEncryptionType
|sort - _time
| ...
```

3.2.4 Detecting Kerberoasting via PowerShell

Kerberoasting activity can be carried through PowerShell on a workstation controlled by an attacker. The search *D05 – Detecting Kerberoasting via PowerShell* uses features of PowerShell logging and its goal is to catch SPN scanning activity or successful acquisition of the service ticket hash.

The search looks for PowerShell events 4103 and 4104 and performs a full-text search in them, looking for strings containing names of service accounts. Transactions are created for all subsequent PowerShell events coming from a single workstation. Results are produced if the number of events containing matching strings is higher than the specified threshold. The list of service accounts and SPNs must be prepared as an input. Listing 5 contains details of this rule.

Listing 5: D05 – Detecting Kerberoasting via PowerShell (snippet).

```
source="WinEventLog:Microsoft-Windows-
PowerShell/ Operational" (EventCode=4103 OR
EventCode=4104)
|transaction Computer maxpause=15m
maxevents=-1 | eval raw=_raw
|search
[| inputlookup service_accounts.csv | eval
raw="*" . account . "*"
|fields raw]
|where eventcount > 2 | ...
```

Time	Host	Source	Target Username	Service Name	Ticket Encryption
03/31/2019 20:03:15	DC01	192.168.56.104	Win7User@TEST.LOCAL	Honeypot01	0x17
03/31/2019 20:03:15	DC01	192.168.56.104	Win7User@TEST.LOCAL	Database01	0x17
03/31/2019 19:58:51	DC01	192.168.56.105	Win7User@TEST.LOCAL	Database01	0x17
03/31/2019 19:58:51	DC01	192.168.56.105	Win7User@TEST.LOCAL	Honeypot01	0x17

Figure 3: Kerberoasting detected in Splunk.

3.3 False Positives and Tuning

The number of false positive detections produced by the proposed detection rules depends on several factors. Firstly, the usage of obsolete cipher suites in the environment. In case these suites are not disabled, and whitelisting is not entirely implemented, false positive detections may appear in the search D01.

The second search, D02, contains numeric values that control thresholds for detection. These need to be adjusted, as the number of requests for different services in a small environment would not be on the same level as in large environments. Alternatively, the search D02 can be combined with D01 to see excessive service ticket requests with suspicious encryption types only. We tested multiple filtering options to minimize the false positive alerts – filtering out only *krbtgt* account, adding ticket encryption types and filtering dollar accounts, which increased accuracy of the detection scenario significantly.

Search D03 should not trigger at all unless there actually is a configuration that allows the use of well-known ports or external IP addresses. The same applies to detection using honeypot in D04. There is no legitimate reason to request a service ticket for the honeypot account. Detected activities are very likely to be malicious.

Table 2: Summary of detection scenarios efficiency.

Scenario	Total Detected Events	True positives		Fales positives	
		Count	%	Count	%
D01 - Possible Kerberoasting activity	13	7	58.85	6	46.15
D02 - Excessive service ticket requests from one source – filtering krbtgt account	326	7	2.15	319	97.85
D02 - Excessive service ticket requests from one source – add weak encryption types	10	7	70	3	30
D02 - Excessive service ticket requests from one source – filter $ accounts	5	5	100	0	0
D04 - Detecting Kerberoasting with a honeypot	7	7	100	0	0

If PowerShell is utilized for routine administration tasks for the specified service accounts, these activities will also be reported by the search D05. Reliable filtering is quite tricky due to the variety of commands that could be used by a potential attacker and nature of the PowerShell logs. The logs contain blocks of code, which limits parsing and also filtering options, thus the search quality will be dependent on the quality of the input list of the scenario.

Table 2 summarizes the True to False positive ratio for the discussed scenarios. Scenario D04 alerted on no False Positives, and it can be used as a reference search. The D02 scenario was tested with three different modifications. Note that filtering on both *krbtgt* and dollar accounts caused two attack attempts to be missed, while producing no false positives. Scenarios D03 and D05 are missing from the table, since given the fact that we are able to describe our environment so precisely, their False Positive rate would be always zero.

Even though there is implementation overhead and changes in the environment are required, we suggest honeypot and/or PowerShell script monitoring to be deployed.

4 CONCLUSIONS

In the paper, we proposed the design of detection scenarios usable for monitoring the network for a potential occurrence of a Kerberoasting attack. The purpose of this attack is to extract service accounts' passwords without the need for any special user access rights or privilege escalation.

The main goal was to develop a set of detection rules, which would be able to detect the Kerberoasting attack by using Windows Security auditing. We designed, implemented and tested multiple monitoring scenarios, that can be used as a baseline for organizations implementing detection mechanisms for their Active Directory environments. The detections were presented in Splunk SPL language, however, the detection principles used in the searches are not limited to the use of Splunk technology.

We have shown the detection capabilities of the designed rules and found out that the false-positive rate of the designed rules may vary. Non-standard approaches, that use honeypots or PowerShell monitoring for detection, offer strong detection capabilities with a low false-positive ratio, but carry on implementation overhead.

ACKNOWLEDGEMENTS

The authors acknowledge the support of the OP VVV

MEYS funded project CZ.02.1.01/0.0/0.0/16_019/0000765 "Research Center for Informatics".

REFERENCES

Desmond, B; Richards, J; Allen, R; Lowe-Norris, A G. (2013) Active Directory: Designing, Deploying, and Running Active Directory. In: [on- line]. 5th ed. O'Reilly Media, 2013, chap. 1-2, 9-10 ISBN 978-1449320027.

Francis, D. (2017) Mastering Active Directory. In: Birmingham: Packt Publishing, 2017, chap. 1-2. ISBN 978-1787289352.

Kotlaba, L. (2019). Detection of Active Directory attacks (Bachelor Thesis). FIT CTU in Prague.

Microsoft Corporation. Microsoft Docs: Windows Authentication [online]. 2016 [visited on 2019-05-11]. Available from: https://docs.microsoft.com/en-us/windows-server/security/windows-authentication/windows-authentication- overview.

Microsoft Corporation. [MS-Kile]: Kerberos Protocol Extensions [on- line]. 2019 [visited on 2019-05-11]. Available from: https://docs.microsoft.com/ en - us / openspecs / windows _ protocols / ms - kile / 2a32282e - dd48 - 4ad9 - a542 - 609804b02cc9.

Microsoft Corporation (2017). Microsoft Docs: Security auditing: 4769(S, F): A Kerberos service ticket was requested. [online]. 2017 [visited on 2019-05-11]. Available from: https://docs.microsoft.com/en-us/windows/security/threat- protection/auditing/event-4769.

Microsoft Corporation (2017). Microsoft Docs: Network security: Configure encryption types allowed for Kerberos [online]. 2017 [visited on 2019-05-11]. Available from: https://docs.microsoft.com/en-us/windows / security / threat-protection/security-policy-settings/network-security-configure-encryption-types-allowed-for-kerberos.

Microsoft Corporation. Microsoft Docs: Advanced security audit policy settings [online]. 2017 [visited on 2019-05-11]. Available from: https://docs. microsoft.com/en-us/windows/security/threat - protection/auditing/advanced-security-audit-policy-settings.

Microsoft Corporation. Microsoft Docs: Security auditing: Audit Kerberos Service Ticket Operations [online]. 2017 [visited on 2019-05-11]. Available from: https://docs.microsoft.com/en-us/windows/security/threat-protection/ auditing/audit-kerberos-service-ticket-operations.

Medin, T, (2014). Attacking Microsoft Kerberos Kicking the Guard Dog of Hades. In *DerbyCon 4.0,* Louisville, USA.

Neuman, C.; Yu, T.; Hartman, S.; Raeburn, K. RFC 4120: The Kerberos network authentication service (V5) [online]. 2005 [visited on 2019-05-11]. Available from: https://tools.ietf.org/html/rfc4120. RFC. MIT.

Metcalf S. (2014). METCALF, Sean. Active Directory Security: MS14-068: Vulnerability in (Active Directory) Kerberos Could Allow Elevation of Privilege [online]. © 2011-2017 [visited on 2019-05-11]. Available from: https://adsecurity.org/?p=525

Metcalf, S. (2017). Active Directory Security: Detecting Kerberoasting Activity [on- line]. © 2011-2017 [visited on 2019-05-11]. Available from: https://adsecurity. org/?p=3458.

Metcalf, S. (2017). Active Directory Security: Detecting Kerberoasting Activity Part 2 – Creating a Kerberoast Service Account Honeypot [online]. © 2011-2017 [visited on 2019-05-11]. Available from: https://adsecurity.org/?p=3513.

Mitre Corporation. Techniques: Kerberoasting [online]. 2018 [visited on 2019-05-11]. Available from: https://attack.mitre.org/techniques/T1208/

Schroeder, W. Invoke-Kerberoast.ps1 [software]. GitHub, 2016 [visited on 2019-05-11]. Available from: https://github.com/EmpireProject/Empire/blob/master/data/module_source/credentials/Invoke-Kerberoast.ps1.

Secureauth Corporation. Impacket: GetUserSPNs.py [software]. GitHub, 2016 [visited on 2019-05-11]. Available from: https://github.com/SecureAuthCorp/impacket/blob/master/examples/GetUserSPNs.py.

Smith, R. F (2006). Security Log Encyclopedia: Windows Security Log Events [online] [visited on 2019-05-11]. Available from: https://www.ultimatewindowssecurity.com/securitylog/encyclopedia/default.aspx.

Splunk (2019). *Splunk Documentation,* available online at https://docs.splunk.com/Documentation.