

# KRYCY LAB 1B

## Autorzy:

Mateusz Borkowski, Michał Wawrzynczak, Paweł Gryka,  
Paweł Popiołek

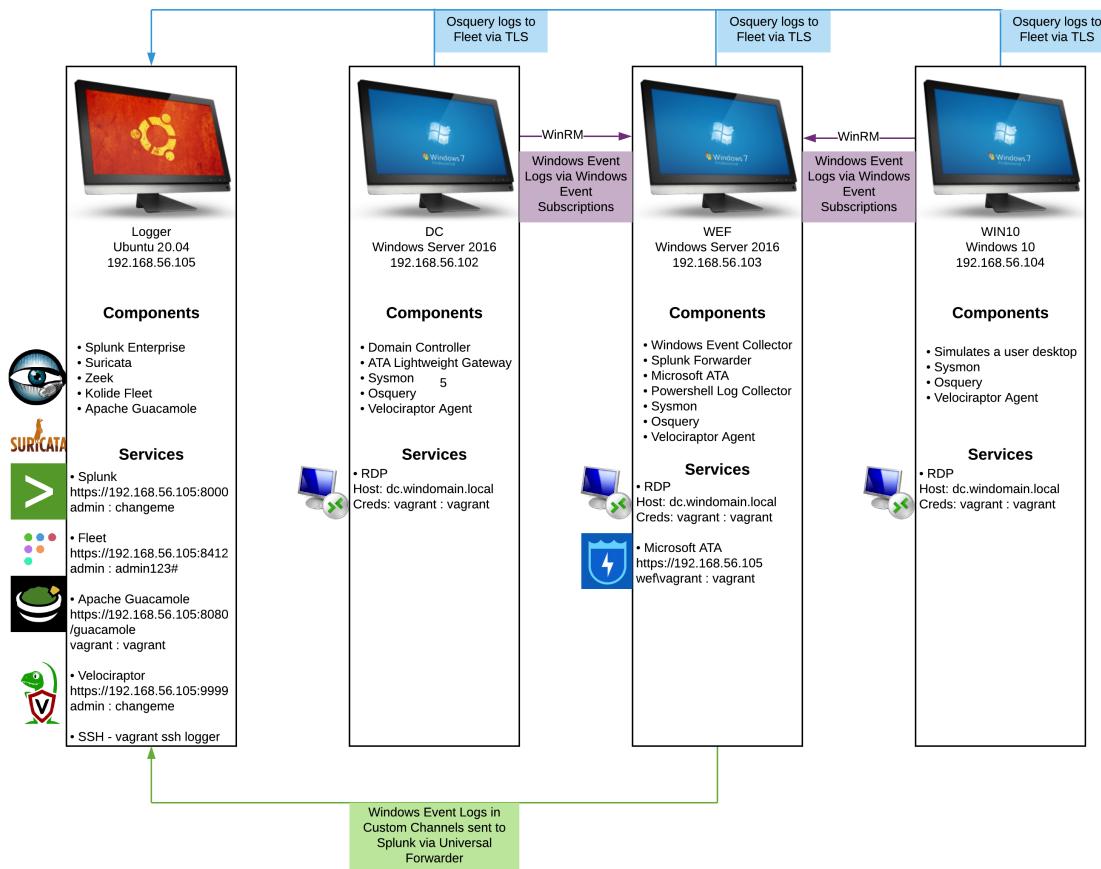
## Zadanie do wykonania

Naszym zadaniem było zapoznanie się z środowiskiem i dostępnymi narzędziami w DetectionLab oraz HELK

## DetectionLab

### Środowisko i narzędzia (Zapoznanie się z nim)

DetectionLab to zestaw potrzebnych skryptów i plików używanych do szybkiego uruchomienia wirtualnego środowiska laboratoryjnego wyposażonego w narzędzia do cyberbezpieczeństwa obronnego. Środowisko składa się z czterech hostów w następującej architekturze:



W dużej ogólności, hosty **DC** i **WIN10** przekazują logi z akcji na nich wykonanych do **Loggera** (albo bezpośrednio albo przez **WEF**)

Na hostach windows'owych znajdują się narzędzia do generowania podejrzanych akcji a następnie zbierania z nich logów:

- Atomic Red Team

- Microsoft ATA
- Sysmon
- Osquery
- Velociraptor

Na **Logger**'rze znajduje się wiele narzędzi do oglądania i analizy logów:

- Splunk
  - Suricata
  - Zeek
- Fleet
- Apache Guacamole
- Velociraptor

## Wykonane działania na maszynach

### Początkowe problemy

Do ćwiczenia podchodziliśmy w sumie czterokrotnie. Pierwszy raz (na samym początku udostępnienia ćwiczenia) jedynie zorientowaliśmy się jak wygląda środowisko. Niestety drugie podejście do lab zakończyło się niepowodzeniem przez wygaśnięcie licencji splunka. Trzecie podobnie, przez skończenie się miejsca na hoście (vagrant nie potrafił wystartować). W czwartym podejściu znów napotkaliśmy problem - host logger nie otrzymywał adresu ip:

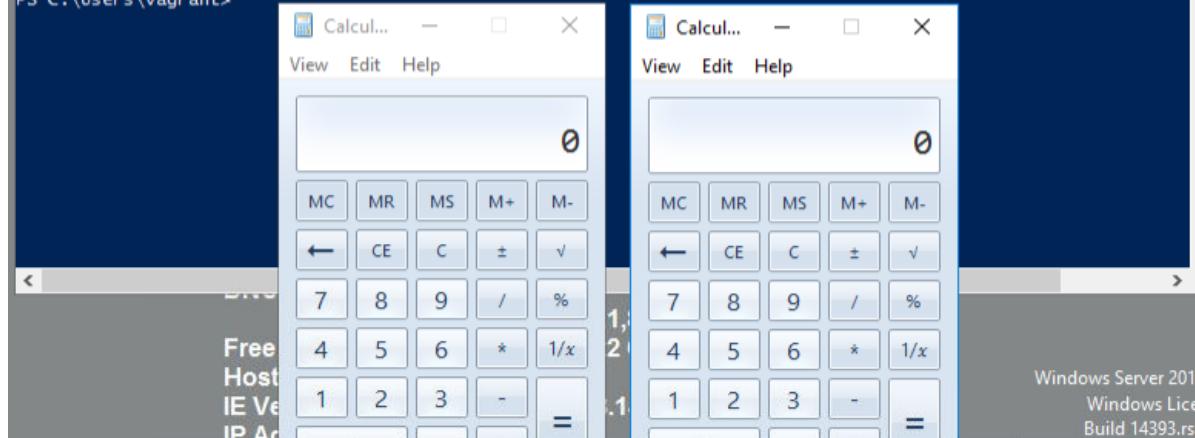
```
vagrant@logger:~$ ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
        valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
        valid_lft forever preferred_lft forever
2: eth0: <BROADCAST,MULTICAST> mtu 1500 qdisc noop state DOWN group default qlen 1000
    link/ether 08:00:27:73:60:cf brd ff:ff:ff:ff:ff:ff
3: eth1: <BROADCAST,MULTICAST> mtu 1500 qdisc noop state DOWN group default qlen 1000
    link/ether 08:00:27:9e:67:98 brd ff:ff:ff:ff:ff:ff
```

Mimo restartów vagranta nietety problem nie ustępował. Po dłuższej sesji debuggingu okazało się, że w konfiguracji netplan zostały trzykrotnie powtórzone komendy, po usunięciu nadmiaru sieć zaczęła działać.

### Generowanie logów

Żeby wygenerować dane do analizy użyliśmy (zgodnie z sugestią) narzędzia **Atomic Red Team**, na hostach **WIN10** i **DC**:

```
PS C:\Users\vagrant> Import-Module "C:\Tools\AtomicRedTeam\Invoke-AtomicRedTeam\Invoke-AtomicRedTeam.ps1" -P  
PS C:\Users\vagrant> $PSDefaultParameterValues = @{"Invoke-AtomicTest:PathToAtomsicsFolder"="C:\Tools\AtomicRe  
PS C:\Users\vagrant> Invoke-AtomicTest T1218.010 -TestNumbers 1,2  
PathToAtomsicsFolder = C:\Tools\AtomicRedTeam\atomsics  
  
Executing test: T1218.010-1 Regsvr32 local COM scriptlet execution  
Done executing test: T1218.010-1 Regsvr32 local COM scriptlet execution  
Executing test: T1218.010-2 Regsvr32 remote COM scriptlet execution  
Done executing test: T1218.010-2 Regsvr32 remote COM scriptlet execution  
PS C:\Users\vagrant>
```



Powyżej wykonaliśmy jedynie dwa proponowane testy z dokumentacji ale potem (żeby wygenerować więcej logów), włączliśmy także narzędzie z opcją *All*.

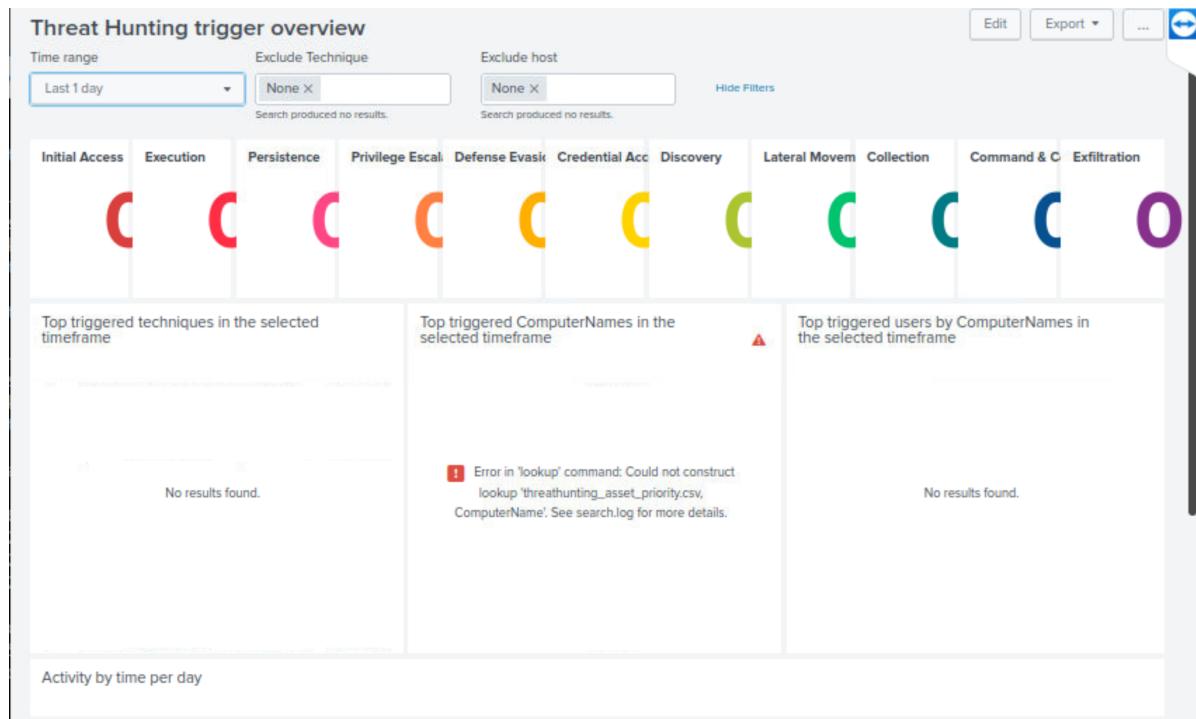
## Przeglądanie logów

Następnie na **logger'ze** włączliśmy *Splunk'a* i sprawdziliśmy, czy pojawiły się jakieś alerty/logi. Większość z logów pochodziła z *zeek*, dlatego je wyświetliśmy:

Time	Event
12/10/21 6:04:51.458 PM	{ [-] active_dns_requests: 0 active_files: 0 active_icmp_conn: 0 active_tcp_conn: 0 active_timers: 0 active_udp_conn: 0 bytes_recv: 0 dns_requests: 0 events_proc: 516 events_queued: 31 files: 0 icmp_conn: 0 mem: 93 peer: manager pkts_proc: 0 reassem_file_size: 0 reassem_frag_size: 0 reassem_tcp_size: 0 ... }

Należy jeszcze dodać, że przedtem także sprawdziliśmy logi (przed wykonaniem Atomic Red Team-testów) i żadne się nie pojawiły - dlatego mamy prawie pewność, że logi pochodzą właśnie z tych wydarzeń. Zgadzało się także pochodzenie logów - najpierw pojawiły się logi z hosta **WIN10** a dopiero później z **DC** - czyli w takiej samej kolejności w jakiej włączaliśmy testy.

Dodatkowo znaleźliśmy opcję mapowania otrzymanych alertów na matrycę **MITRE**, gdy podchodziliśmy do ćwiczenia za pierwszym razem, udało się nam zobaczyć prawidłowo załadowane logi zmapowane do matrycy **MITRE**, jednakże w ostatecznym podejściu nie udało się już załadować danych do tej opcji. Niestety nie mamy screenshota potwierdzającego z pierwszego podejścia.



## Stawianie środowiska na świeżo

Ze względu na istniejące problemy postanowiliśmy całe środowisko przywrócić do stanu początkowego. Tu także wystąpiły problemy ze względu na modyfikację url do pobrania Velociraptora, aktualizacja stanu do aktuanego z tym w repozytorium

<https://github.com/clong/DetectionLab> także nie pomagała. Po zamianie url w skrypcie

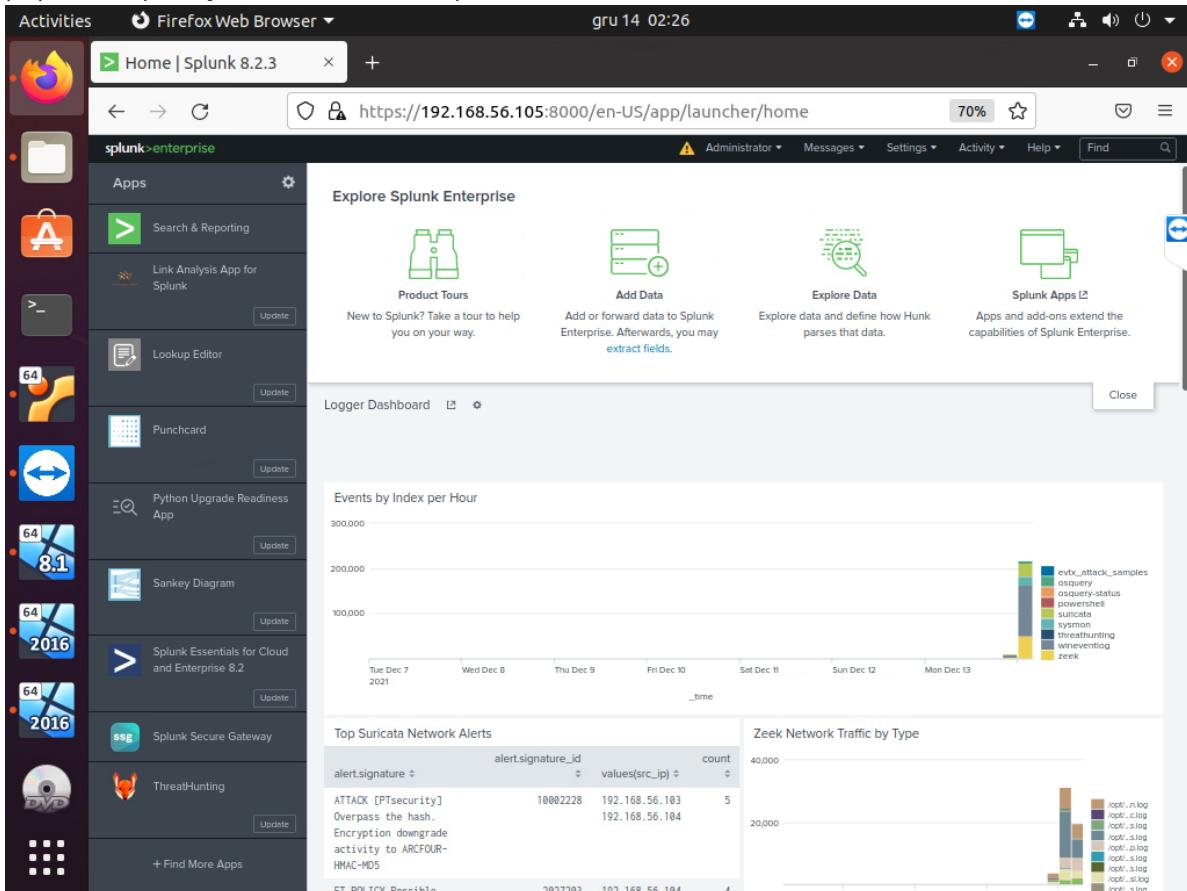
`install-velociraptor.ps1` udało się postawić poprawnie całe środowisko

```
krycy-lab1@krycy-lab1-detlab:~/Documents/DetectionLab/Vagrant$ git status
Refresh index: 100% (408/408), done.
On branch master
Your branch is up to date with 'origin/master'.

Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
    modified:   scripts/install-velociraptor.ps1

no changes added to commit (use "git add" and/or "git commit -a")
krycy-lab1@krycy-lab1-detlab:~/Documents/DetectionLab/Vagrant$ git diff
diff --git a/Vagrant/scripts/install-velociraptor.ps1 b/Vagrant/scripts/install-velociraptor.ps1
index 759335e..dc60539 100644
--- a/Vagrant/scripts/install-velociraptor.ps1
+++ b/Vagrant/scripts/install-velociraptor.ps1
@@ -15,7 +15,8 @@ Write-Host "$([{0:HH:mm}]`f (Get-Date)) Determining latest release of Velociraptor"
# Disabling the progress bar speeds up IWR https://github.com/PowerShell/PowerShell/issues/2138
$ProgressPreference = 'SilentlyContinue'
$tag = (Invoke-WebRequest "https://api.github.com/repos/Velocidex/velociraptor/releases" -UseBasicParsing | ConvertFrom-Json)[0].tag_name
-$VelociraptorDownloadUrl = "https://github.com" + ((Invoke-WebRequest "https://github.com/Velocidex/velociraptor/releases/latest" -UseBasicParsing).links | Select-Object -ExpandProperty href | Select-String "windows-amd64.msi$")
+$VelociraptorDownloadUrl = "https://github.com/Velocidex/velociraptor/releases/download/v0.6.2/velociraptor-v0.6.2-windows-amd64.msi"
+echo $VelociraptorDownloadUrl
$VelociraptorMSIPath = 'C:\Users\vagrant\AppData\Local\Temp\velociraptor.msi'
$VelociraptorLogFile = 'c:\Users\vagrant\AppData\Local\Temp\velociraptor_install.log'
If (-not (Test-Path $VelociraptorLogFile)) {
krycy-lab1@krycy-lab1-detlab:~/Documents/DetectionLab/Vagrant$
```

Wykonaliśmy podstawowy `Invoke-AtomicTest T1218.010 -TestNumbers 1,2`, logi były poprawnie przesyłane i widoczne w Splunku.



Zakończyliśmy pracę i chcieliśmy wykonać więcej testów, następnego dnia. Po powrocie do środowiska zaczęły ponownie występować różne problemy.

# AtomicTests

Uruchomiliśmy kilka AtomicTestów i obserwowałyśmy winiki w Spunku.

## T1218 - Signed Binary Proxy Execution

The screenshot shows a Linux desktop environment with two windows open:

- Terminal Window:** A terminal window titled "wef.windomain.local [Running] - Oracle VM VirtualBox". It displays the output of running the command `Invoke-AtomicTest T1218.010 -TestNumbers 1,2`. The output shows various COM scriptlet execution tests being performed, such as `T1218.010-1 Regsvr32 local COM scriptlet execution` and `T1218.010-2 Regsvr32 remote COM scriptlet execution`. It also shows attempts to register evil DLLs and execute them, with one attempt failing due to a namespace error.
- Splunk Web Interface:** A Firefox browser window titled "Threat Hunting trigger" and "MITRE ATT&CK | Splunk". The URL is `https://192.168.56.105:8000/en-US/app/ThreatHunting/mitre_att`. The interface shows a "Threat Hunting" dashboard with a table of detected processes. One row in the table is highlighted with a yellow background, corresponding to the "Signed Binary Proxy Execution" test from the terminal. The table columns include: \_time, ID, Technique, Category, Trigger, ComputerName, user\_name, process\_parent\_path, and process\_path. The highlighted row shows the following values:

_time	ID	Technique	Category	Trigger	ComputerName	user_name	process_parent_path	process_path
2021-12-15 13:58:37	T1179	Hooking	Persistence,Privilege_Escalation,Credential_Access	dc.windomain.local	vagrant	C:\Windows\System32\cmd.exe	C:\Windows\System32\mavinject.exe	
2021-12-15 13:58:37	T1179	Hooking	Persistence,Privilege_Escalation,Credential_Access	dc.windomain.local	vagrant	C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe	C:\Windows\System32\cmd.exe	
2021-12-15 13:58:37	T1218	Signed Binary Proxy Execution	Defense_Evasion,Execution	dc.windomain.local	vagrant	C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe	C:\Windows\System32\cmd.exe	
2021-12-15 13:58:37	T1218	Signed Binary Proxy Execution	Defense_Evasion,Execution	dc.windomain.local	vagrant	C:\Windows\System32\cmd.exe	C:\Windows\System32\mavinject.exe	
2021-12-15 13:58:36	T1033	System Owner/User	Discovery	dc.windomain.local	vagrant	C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe	C:\Windows\System32	C:\Windows\System32

Po wykonaniu testu możemy zaobserwować w splunku, że IDS wykrył to zagrożenie.

## **T1078 - Default Accounts**

T1078	Valid Accounts	Adversaries may obtain and abuse credentials of existing accounts as a means of gaining Initial Access, Persistence, Privilege Escalation, or Defense Evasion. Compromised credentials may be used to bypass access controls placed on various resources on systems within the network and may even be used for persistent access to remote systems and externally available services, such as VPNs, Outlook Web Access and remote desktop. Compromised credentials may also grant an adversary increased privilege to specific systems or access to restricted areas of the network. Adversaries may choose not to use malware or tools in conjunction with the legitimate access those credentials provide to make it harder to detect their presence.
.001	Default Accounts	Adversaries may obtain and abuse credentials of a default account as a means of gaining Initial Access, Persistence, Privilege Escalation, or Defense Evasion. Default accounts are those that are built-into an OS, such as the Guest or Administrator accounts on Windows systems. Default accounts also include default factory/provider set accounts on other types of systems, software, or devices, including the root user account in AWS and the default service account in Kubernetes.
.002	Domain Accounts	Adversaries may obtain and abuse credentials of a domain account as a means of gaining Initial Access, Persistence, Privilege Escalation, or Defense Evasion. Domain accounts are those managed by Active Directory Domain Services where access and permissions are configured across systems and services that are part of that domain. Domain accounts can cover users, administrators, and services.
.003	Local Accounts	Adversaries may obtain and abuse credentials of a local account as a means of gaining Initial Access, Persistence, Privilege Escalation, or Defense Evasion. Local accounts are those configured by an organization for use by users, remote support, services, or for administration on a single system or service.
.004	Cloud Accounts	Adversaries may obtain and abuse credentials of a cloud account as a means of gaining Initial Access, Persistence, Privilege Escalation, or Defense Evasion. Cloud accounts are those created and configured by an organization for use by users, remote support, services, or for administration of resources within a cloud service provider or SaaS application. In some cases, cloud accounts may be federated with traditional identity management system, such as Window Active Directory.

Activities VirtualBox Machine gru 15 15:51

dc.windomain.local [Running] - Oracle VM VirtualBox

File Machine View Input Devices Help

Windows PowerShell

```
Executing cleanup for test: T1566.001-2 Word spawned a command shell and used an IP address in the command line
Done executing cleanup for test: T1566.001-2 Word spawned a command shell and used an IP address in the command line
PS C:\Users\vagrant> Invoke-AtomicTest T1566.002
PathToAtomsFolder = C:\Tools\AtomicRedTeam\atoms

ERROR: C:\Tools\AtomicRedTeam\atoms\T1566.002\T1566.002.yaml does not exist
Check your Atomic Number and your PathToAtomsFolder parameter
PS C:\Users\vagrant> Invoke-AtomicTest T1566.003
PathToAtomsFolder = C:\Tools\AtomicRedTeam\atoms

ERROR: C:\Tools\AtomicRedTeam\atoms\T1566.003\T1566.003.yaml does not exist
Check your Atomic Number and your PathToAtomsFolder parameter
PS C:\Users\vagrant> Invoke-AtomicTest T1078.001
PathToAtomsFolder = C:\Tools\AtomicRedTeam\atoms

Executing test: T1078.001-1 Enable Guest account with RDP capability and admin privileges
Done executing test: T1078.001-1 Enable Guest account with RDP capability and admin privileges
System error 5 has occurred.

Access is denied.

ERROR: Access is denied.

ERROR: Access is denied.

Executing test: T1078.001-2 Activate Guest Account
Done executing test: T1078.001-2 Activate Guest Account
System error 5 has occurred.

Access is denied.

PS C:\Users\vagrant> Invoke-AtomicTest T1078.001 -Cleanup
PathToAtomsFolder = C:\Tools\AtomicRedTeam\atoms

Executing cleanup for test: T1078.001-1 Enable Guest account with RDP capability and admin privileges
Done executing cleanup for test: T1078.001-1 Enable Guest account with RDP capability and admin privileges
Note: set remove_rdp_access_during_cleanup input argument to disable RDP access during cleanup
Executing cleanup for test: T1078.001-2 Activate Guest / Event Viewer
```

Activities Firefox Web Browser gru 15 15:56

MITRE ATT&CK | Splunk

splunk>enterprise Apps

Threat Hunting trigger overview Drilldowns Stacking Tools Hunting Tools Hunting Indicators Lists About Search THREAT HUNTING

MITRE ATT&CK

Timespan Last 4 hours MITRE Category All Mitre Technique All Mitre Technique ID All Exclude Technique None

Exclude host None win10.windomain.local wef@windomain.local

Submit Hide Filters

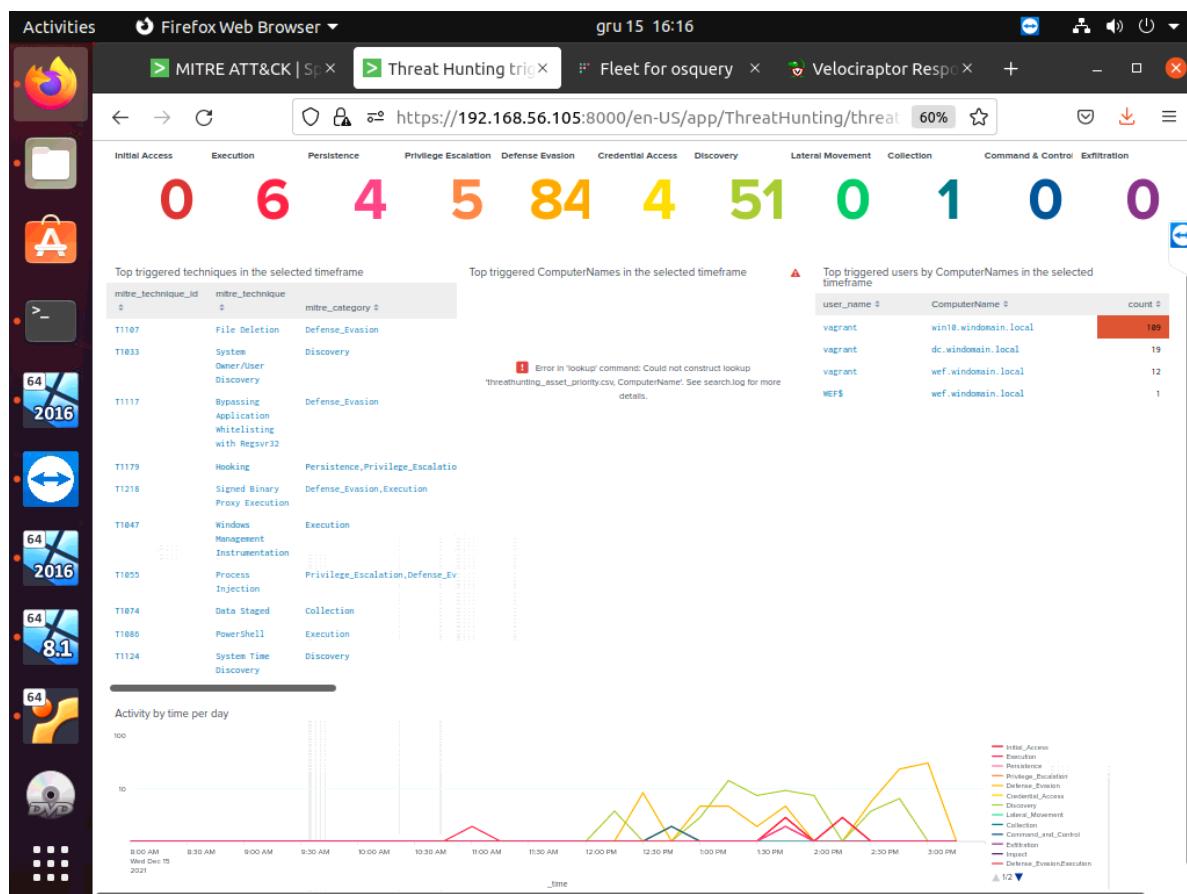
Process Create

_time	ID	Technique	Category	Trigger	ComputerName	user_name	process_parent_path	process_path
-------	----	-----------	----------	---------	--------------	-----------	---------------------	--------------

	2021-12-15 14:48:43	T1107	File Deletion	Defense_Evasion	dc.windomain.local	vagrant	C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe	C:\Windows\System32\WindowsPowerShell\v1.0\powershell
	2021-12-15 14:33:14	T1033	System Owner/User Discovery	Discovery	dc.windomain.local	vagrant	C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe	C:\Windows\System\whoami.exe
	2021-12-15 14:33:08	T1033	System Owner/User Discovery	Discovery	dc.windomain.local	vagrant	C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe	C:\Windows\System\whoami.exe
	2021-12-15 13:58:37	T1179	Hooking	Persistence,Privilege_Escalation,Credential_Access	dc.windomain.local	vagrant	C:\Windows\System32\cmd.exe	C:\Windows\System\mavinject.exe
	2021-12-15 13:58:37	T1179	Hooking	Persistence,Privilege_Escalation,Credential_Access	dc.windomain.local	vagrant	C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe	C:\Windows\System\cmd.exe

Nie udało nam się zaobserwować tej technik (uzyskania dostępu z wykorzystaniem istniejących kont domyślnych) w splanku. Możliwe, że IDS nie wykrył tej techniki. widzimy jednak logi które, mogą dotyczyć innych uruchamianych przez nas testów. Jak np T1107 - File Deletion.

## Podsumowanie wykrytych zagrożeń po uruchomionych przez nas testach.



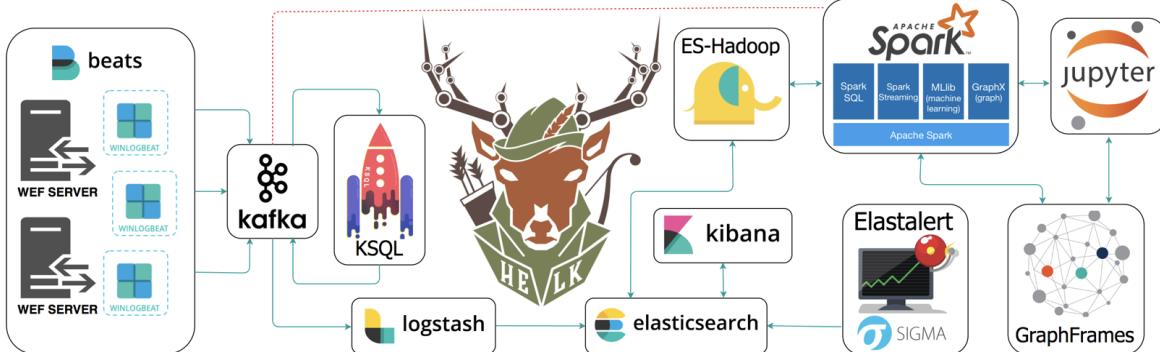
## HELK

### Środowisko i narzędzia

HELK jest open-source'ową platformą wykorzystywaną w cyberbezpieczeństwie defensym, składającą się z poniższych komponentów:

- *Kafka* - umożliwia obsługę danych (logów) pochodzących z wielu źródeł w czasie rzeczywistym,
- *Elasticsearch* - silnik do wyszukiwania i analizy danych,
- *Logstash* - przekazuje zagregowane przez *Kafk*e logi dalej do *Elasticsearch*'a,
- *Kibana* - platforma służąca do wizualizacji i analizy danych, zaprojektowana aby współpracować z *Elasticsearch*,

- *Apache Spark* - platforma programistyczna dla obliczeń rozproszonych,
- *ES-Hadoop* - pozwala na interakcję pomiędzy *Elasticsearch*'em, a *Spark*'iem,
- *GraphFrames* - package do *Spark*'a, rozbudowuje jego możliwości obsługi grafów,
- *JupyterNotebook (JupyterLab)* - pozwala na tworzenie i udostępnianie dokumentów zawierających wykonywalny kod,
- *Elastalert* - rozwiązanie służące do powiadomiania o anomaliiach, albo o dopasowaniach do określonych wzorców danych z *Elasticsearch*'a.



## Wykonanie działania na maszynach

### Ożywienie środowiska danymi

Aby poznać dostępne narzędzia, należało najpierw ożywić środowisko danymi. Przy próbie załadowania danych do środowiska napotkaliśmy na następujące problemy:

- dane ożywiające nie istniały w miejscu zalinkowanym w podanym w instrukcji poradniku,
- brak komunikacji z *Elasticsearch* po domyślnym porcie (porzebne w celu wrzucenia tam danych za pomocą skryptu w *Pythonie*),
- problem z wczytaniem danych w formacie *json* po ich rozpakowaniu.

Aby uporać się z powyższymi problemami odpowiednio:

- odnaleźliśmy odpowiednie dane ożywiające w sieci (repozytorium z tymi danymi zmieniło nazwę dlatego link ze strony nie zadziałał - nowa nazwa repozytorium to *OTRF/Security-Datasets*),
- w celu zapewnienia komunikacji z *Elasticsearch* należało zmodyfikować plik konfiguracyjny *helk-kibana-analysis-basic.yml*, tak aby *docker* wystawił odpowiedni port do komunikacji z bazą - 9200,
- pliki do *Elasticsearch*'a należało przekazywać nie w formacie *json*, a spakować je do *tar.gz*.

Dla ułatwienia pracy i rozwiązywania problemów całe środowisko postanowiliśmy postawić u siebie lokalnie.

8,465 hits	
<code>_source</code>	
>	<code>@timestamp: Sep 21, 2020 @ 18:59:53.446 @version: 1 log.file.name: mordor-data.tar.gz/empire_dcsync_dcerpc_drsuapi_DsGetNCChanges_2020-09-21185829.json winlog.event_id: 10 winlog.event_data.Keywords: -9,223,372,036,854,775,888 winlog.event_data.SeverityValue: 2 winlog.event_data.SourceImage: C:\windows\system32\svchost.exe winlog.event_data.ProviderGuid: {5770385F-C22A-43E0-BF4C-06F5698FFBD9} winlog.event_data.ExecutionProcessID: 3,172 winlog.event_data.AccountType: User winlog.event_data.UserID: S-1-5-18 winlog.event_data.SourceProcessGUID: {b34bc01c-226b-5f69-1000-000000000900} winlog.event_data.ThreadID: 4,048 winlog.event_data.TargetImage: C:\windows\System32\svchost.exe winlog.event_data.GrantedAccess: 0x1000 winlog.event_data.EventType: INFO winlog.event_data.Opcodes: Info</code>
>	<code>@timestamp: Sep 21, 2020 @ 18:59:53.445 @version: 1 log.file.name: mordor-data.tar.gz/empire_dcsync_dcerpc_drsuapi_DsGetNCChanges_2020-09-21185829.json winlog.event_id: 10 winlog.event_data.Keywords: -9,223,372,036,854,775,888 winlog.event_data.SeverityValue: 2 winlog.event_data.SourceImage: C:\windows\system32\svchost.exe winlog.event_data.ProviderGuid: {5770385F-C22A-43E0-BF4C-06F5698FFBD9} winlog.event_data.ExecutionProcessID: 3,172 winlog.event_data.AccountType: User winlog.event_data.UserID: S-1-5-18 winlog.event_data.SourceProcessGUID: {b34bc01c-226b-5f69-1000-000000000900} winlog.event_data.ThreadID: 4,048 winlog.event_data.TargetImage: C:\windows\System32\svchost.exe winlog.event_data.GrantedAccess: 0x1000 winlog.event_data.EventType: INFO winlog.event_data.Opcodes: Info</code>
>	<code>@timestamp: Sep 21, 2020 @ 18:59:53.445 @version: 1 log.file.name: mordor-data.tar.gz/empire_dcsync_dcerpc_drsuapi_DsGetNCChanges_2020-09-21185829.json winlog.event_id: 10 winlog.event_data.Keywords: -9,223,372,036,854,775,888 winlog.event_data.SeverityValue: 2 winlog.event_data.SourceImage: C:\windows\system32\svchost.exe winlog.event_data.ProviderGuid: {5770385F-C22A-43E0-BF4C-06F5698FFBD9} winlog.event_data.ExecutionProcessID: 3,172 winlog.event_data.AccountType: User winlog.event_data.UserID: S-1-5-18 winlog.event_data.SourceProcessGUID: {b34bc01c-226b-5f69-1000-000000000900} winlog.event_data.ThreadID: 4,048 winlog.event_data.TargetImage: C:\windows\System32\svchost.exe winlog.event_data.GrantedAccess: 0x1000 winlog.event_data.EventType: INFO winlog.event_data.Opcodes: Info</code>
>	<code>@timestamp: Sep 21, 2020 @ 18:59:53.445 @version: 1 log.file.name: mordor-data.tar.gz/empire_dcsync_dcerpc_drsuapi_DsGetNCChanges_2020-09-21185829.json winlog.event_id: 10 winlog.event_data.Keywords: -9,223,372,036,854,775,888 winlog.event_data.SeverityValue: 2 winlog.event_data.SourceImage: C:\windows\system32\svchost.exe winlog.event_data.ProviderGuid: {5770385F-C22A-43E0-BF4C-06F5698FFBD9} winlog.event_data.ExecutionProcessID: 3,172 winlog.event_data.AccountType: User winlog.event_data.UserID: S-1-5-18 winlog.event_data.SourceProcessGUID: {b34bc01c-226b-5f69-1000-000000000900} winlog.event_data.ThreadID: 4,048 winlog.event_data.TargetImage: C:\windows\System32\svchost.exe winlog.event_data.GrantedAccess: 0x1000 winlog.event_data.EventType: INFO winlog.event_data.Opcodes: Info</code>
>	<code>@timestamp: Sep 21, 2020 @ 18:59:53.444 @version: 1 log.file.name: mordor-data.tar.gz/empire_dcsync_dcerpc_drsuapi_DsGetNCChanges_2020-09-21185829.json winlog.event_id: 10</code>

Na powyższym screenie widać że udało nam się załadować odpowiednie logi i mamy na nie podgląd z *Kiban'y*.

## Kibana

Najbardziej interesującym z naszej perspektywy komponentem całego środowiska jest *Kibana*. Jest tak ponieważ oferuje ona wiele możliwości operowania na zebranych danych, a to wszystko za pomocą interfejsu użytkownika. Z powyższego powodu postanowiliśmy skupić nasze działania wokół tego programu.

### Discover

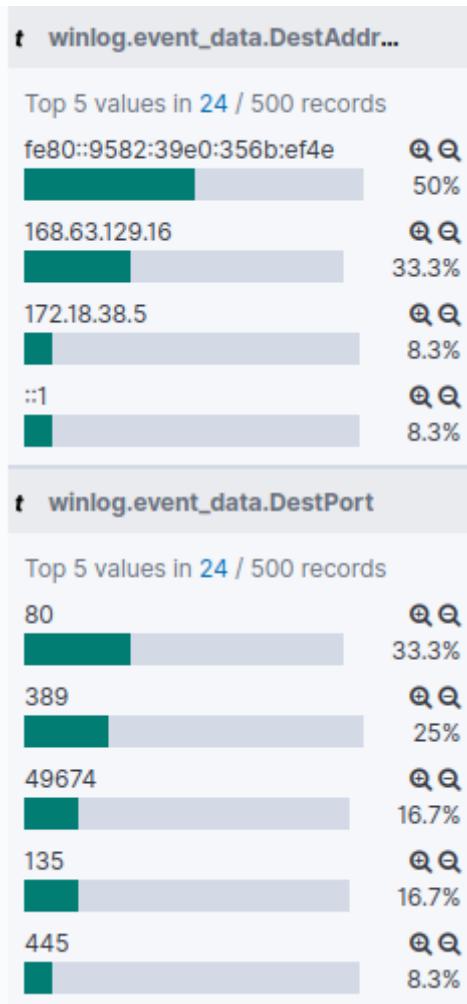
Zakładka *Discover* pozwala na podgląd wszystkich zebranych logów. Oferuje ona wiele możliwości ułatwienia sobie pracy z przekopywaniem się przez ogromne ilości mało czytelnych logów. W tym celu możemy np.:

- rozwinąć konkretny, interesujący nas rekord, otrzymując tym samym bardziej czytelny format:

>	<code>@timestamp: Sep 21, 2020 @ 18:59:53.446 @version: 1 log.file.name: mordor-data.tar.gz/empire_dcsync_dcerpc_drsuapi_DsGetNCChanges_2020-09-21185829.json winlog.event_id: 10 winlog.event_data.Keywords: -9,223,372,036,854,775,888 winlog.event_data.SeverityValue: 2 winlog.event_data.SourceImage: C:\windows\system32\svchost.exe winlog.event_data.ProviderGuid: {5770385F-C22A-43E0-BF4C-06F5698FFBD9} winlog.event_data.ExecutionProcessID: 3,172 winlog.event_data.AccountType: User winlog.event_data.UserID: S-1-5-18 winlog.event_data.SourceProcessGUID: {b34bc01c-226b-5f69-1000-000000000900} winlog.event_data.ThreadID: 4,048 winlog.event_data.TargetImage: C:\windows\System32\svchost.exe winlog.event_data.GrantedAccess: 0x1000 winlog.event_data.EventType: INFO winlog.event_data.Opcodes: Info</code>
---	--

Expanded document		View single document																												
Table	JSON																													
<table border="1"> <thead> <tr> <th></th><th></th></tr> </thead> <tbody> <tr> <td> </td><td>Sep 21, 2020 @ 18:59:53.446</td></tr> <tr> <td>t @version</td><td>1</td></tr> <tr> <td>t _id</td><td>qaN_q30BqVe54Fk-Lh1o</td></tr> <tr> <td>t _index</td><td>winlogbeat-mordor</td></tr> <tr> <td># _score</td><td>-</td></tr> <tr> <td>t _type</td><td>_doc</td></tr> <tr> <td># event.code</td><td>10</td></tr> <tr> <td>t log.file.name</td><td>mordor-data.tar.gz/empire_dcsync_dcerpc_drsuapi_DsGetNCChanges_2020-09-21185829.json</td></tr> <tr> <td>t winlog.channel</td><td>Microsoft-Windows-Sysmon/Operational</td></tr> <tr> <td>t winlog.computer_name</td><td>WORKSTATION5.theshire.local</td></tr> <tr> <td>t winlog.event_data.AccountName</td><td>SYSTEM</td></tr> <tr> <td>t winlog.event_data.AccountType</td><td>User</td></tr> <tr> <td>t winlog.event_data.CallTrace</td><td>&gt; C:\windows\SYSTEM32\ntdll.dll+9c534 C:\windows\SYSTEM32\psmserviceexhost.dll+222a3 C:\windows\SYSTEM32\psmserviceexhost.dll+1a172 C:\windows\SYSTEM32\psmserviceexhost.dll+19e3b C:\windows\SYSTEM32\psmserviceexhost.dll+19318 C:\windows\SYSTEM32\ntdll.dll+3081d C:\windows\SYSTEM32\ntdll.dll+345b4 C:\windows\System32\KERNEL32.DLL+17bd4 C:\windows\SYST</td></tr> </tbody> </table>				Sep 21, 2020 @ 18:59:53.446	t @version	1	t _id	qaN_q30BqVe54Fk-Lh1o	t _index	winlogbeat-mordor	# _score	-	t _type	_doc	# event.code	10	t log.file.name	mordor-data.tar.gz/empire_dcsync_dcerpc_drsuapi_DsGetNCChanges_2020-09-21185829.json	t winlog.channel	Microsoft-Windows-Sysmon/Operational	t winlog.computer_name	WORKSTATION5.theshire.local	t winlog.event_data.AccountName	SYSTEM	t winlog.event_data.AccountType	User	t winlog.event_data.CallTrace	> C:\windows\SYSTEM32\ntdll.dll+9c534 C:\windows\SYSTEM32\psmserviceexhost.dll+222a3 C:\windows\SYSTEM32\psmserviceexhost.dll+1a172 C:\windows\SYSTEM32\psmserviceexhost.dll+19e3b C:\windows\SYSTEM32\psmserviceexhost.dll+19318 C:\windows\SYSTEM32\ntdll.dll+3081d C:\windows\SYSTEM32\ntdll.dll+345b4 C:\windows\System32\KERNEL32.DLL+17bd4 C:\windows\SYST		
	Sep 21, 2020 @ 18:59:53.446																													
t @version	1																													
t _id	qaN_q30BqVe54Fk-Lh1o																													
t _index	winlogbeat-mordor																													
# _score	-																													
t _type	_doc																													
# event.code	10																													
t log.file.name	mordor-data.tar.gz/empire_dcsync_dcerpc_drsuapi_DsGetNCChanges_2020-09-21185829.json																													
t winlog.channel	Microsoft-Windows-Sysmon/Operational																													
t winlog.computer_name	WORKSTATION5.theshire.local																													
t winlog.event_data.AccountName	SYSTEM																													
t winlog.event_data.AccountType	User																													
t winlog.event_data.CallTrace	> C:\windows\SYSTEM32\ntdll.dll+9c534 C:\windows\SYSTEM32\psmserviceexhost.dll+222a3 C:\windows\SYSTEM32\psmserviceexhost.dll+1a172 C:\windows\SYSTEM32\psmserviceexhost.dll+19e3b C:\windows\SYSTEM32\psmserviceexhost.dll+19318 C:\windows\SYSTEM32\ntdll.dll+3081d C:\windows\SYSTEM32\ntdll.dll+345b4 C:\windows\System32\KERNEL32.DLL+17bd4 C:\windows\SYST																													

- podejrzeć ilość powtórzeń danej wartości konkretnego pola dla różnych logów:



- przeglądać wszystkie logi z podaniem tylko na interesujące nas pola:

The screenshot shows the Kibana Log View interface for the 'winlogbeat-mordor\*' index pattern. On the left, there's a sidebar with a search bar, a 'Filter by type' section (set to 0), and sections for 'Selected fields' (containing @timestamp, winlog.computer\_name, and winlog.event\_data.SourceName) and 'Available fields' (a long list including \_id, \_score, winlog.event\_data.DestAddress, winlog.event\_data.DestPort, winlog.event\_data.IpAddress, winlog.event\_data.ProcessId, winlog.event\_data.TargetObj, winlog.event\_data.UserId, @version, \_index, \_type, and event.code). The main area shows a table of log entries with the following columns: @timestamp, winlog.computer\_name, winlog.event\_data.SourceName, and winlog.event\_data.Message. There are 8,465 hits. The first few entries are as follows:

@timestamp	winlog.computer_name	winlog.event_data.SourceName	winlog.event_data.Message
Sep 21, 2020 @ 18:58:30.645	WORKSTATION5.theshi	PowerShell	Pipeline execution details for command line: Start-Sleep -Seconds \$sleepTime; .
Sep 21, 2020 @ 18:58:30.645	WORKSTATION5.theshi	PowerShell	Pipeline execution details for command line: \$IV=[BitConverter]::GetBytes(\$([Get-Random])); .
Sep 21, 2020 @ 18:58:30.647	WORKSTATION5.theshi	PowerShell	Pipeline execution details for command line: 0..255   ForEach-Object { .
Sep 21, 2020 @ 18:58:30.648	WORKSTATION5.theshi	Microsoft-Windows-PowerShe	Context Information: DetailSequence=1 DetailTotal=2
Sep 21, 2020 @ 18:58:30.648	WORKSTATION5.theshi	PowerShell	Pipeline execution details for command line: 0..255   ForEach-Object { .
Sep 21, 2020 @ 18:58:30.649	WORKSTATION5.theshi	Microsoft-Windows-PowerShe	Context Information: DetailSequence=2 DetailTotal=2

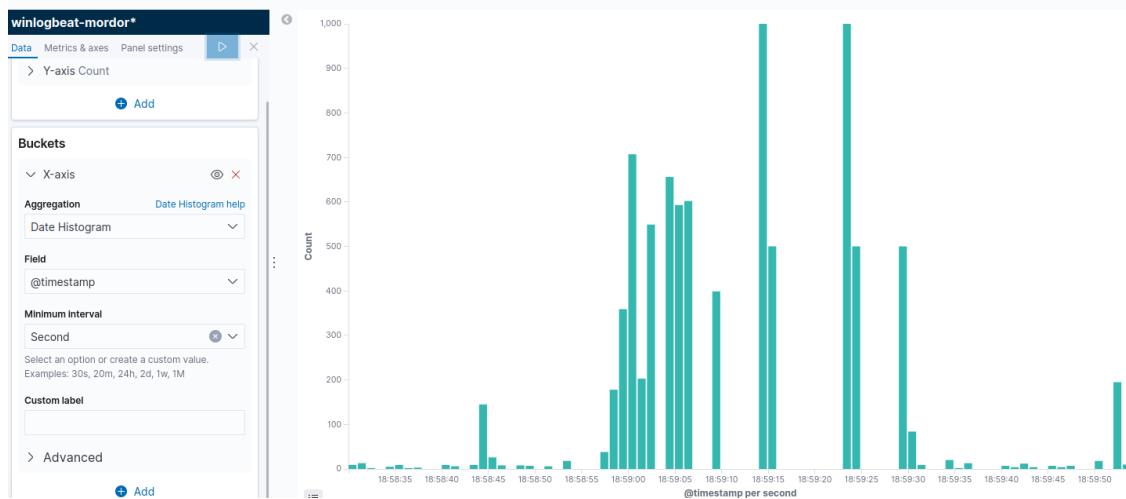
- tworzyć i wysyłać zapytania KQL co pozwala na przefiltrowanie danych (przykład uzycia w dalszej części dokumentu - punkt *Dashboards*)

## Visualization

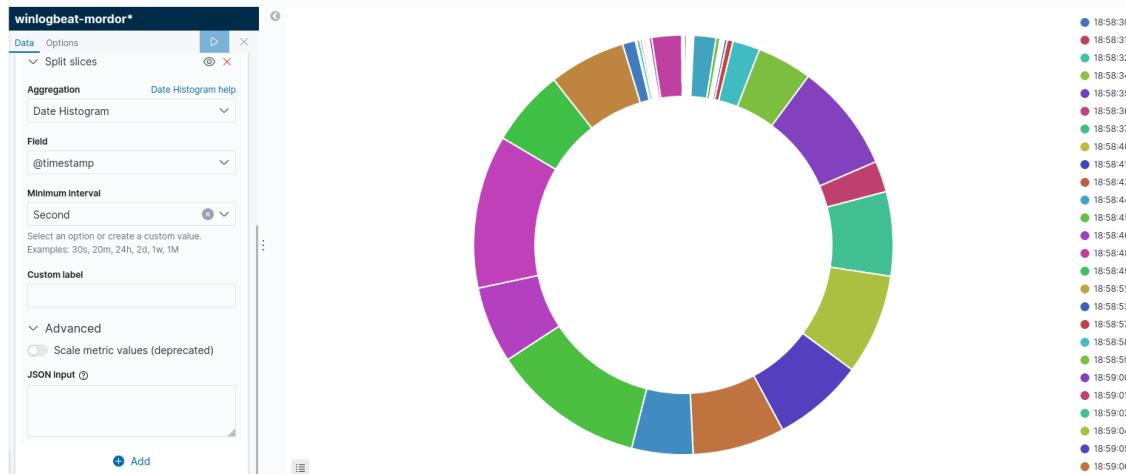
Zakładka *Visualization* pozwala nam na tworzenie różnego rodzaju wykresów. Stworzenie dobrej wizualizacji może w dużym stopniu ułatwić dla człowieka zauważenie różnych anomalii.

Zakładka *Visualization* pozwala między innymi na:

- tworzenie histogramów:



- tworzenie wykresów kołowych:



Jak widać dla powyższych duże znaczenie ma wybór odpowiedniej metody wizualizacji do analizowanych danych. W obu przykładach wyświetlane są czasy zachodzenia zdarzeń w systemie. Dla pierwszego przykładu (histogram) na pierwszy rzut oka widać jak wiele zdarzeń zaszło w jakim czasie i analityk bezpieczeństwa może z takiego widoku szybko wyciągnąć wnioski. W drugim przykładzie (wykres kołowy) widoczność jest zdecydowanie gorsza i wyciągnięcie wniosków z takiego widoku może być zdecydowanie bardziej czasochłonne, a przy odpowiednich warunkach - niemożliwe.

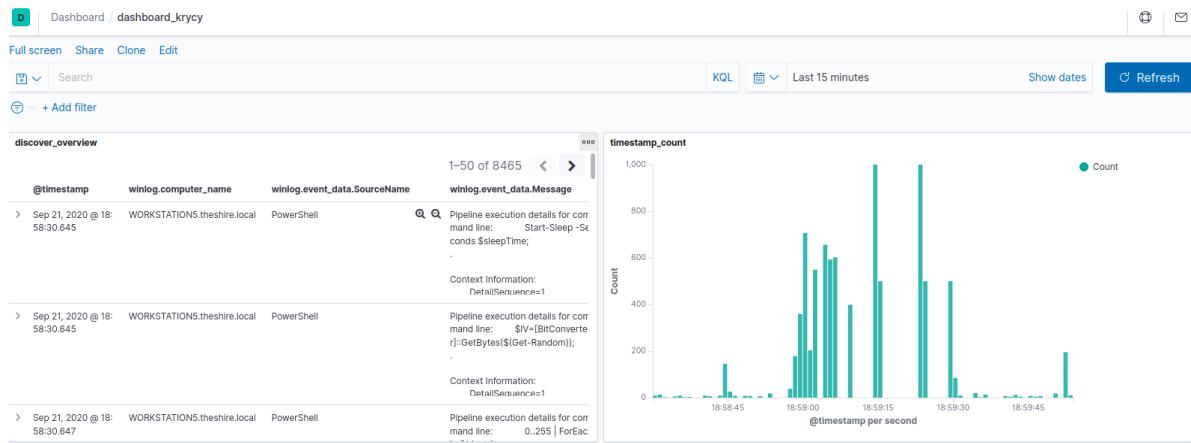
## Dashboards

Zakładka *Dashboards* oferuje rozwiązanie, które ostatecznie może być dla analityka bezpieczeństwa najczęściej odwiedzanym miejscem. Umożliwia ona tworzenie i zapisywanie wielu dashboard'ów. Pojedynczy dashboard może składać się z wielu wybranych komponentów. Komponentami możliwymi do dodania na taką talicę są np. różnego rodzaju wizualizacje, tablice danych czy inne informacje niepochodzące z zagregowanych logów, ale takie które chcielibyśmy mieć pod ręką.

Dla prezentowanego poniżej *dashboard'a* wykorzystaliśmy przedstawione w poprzednich punktach konstrukcje. Zostały one przez nas zapisane, co umożliwiło wykorzystanie ich w tym momencie. Tymi konstrukcjami są:

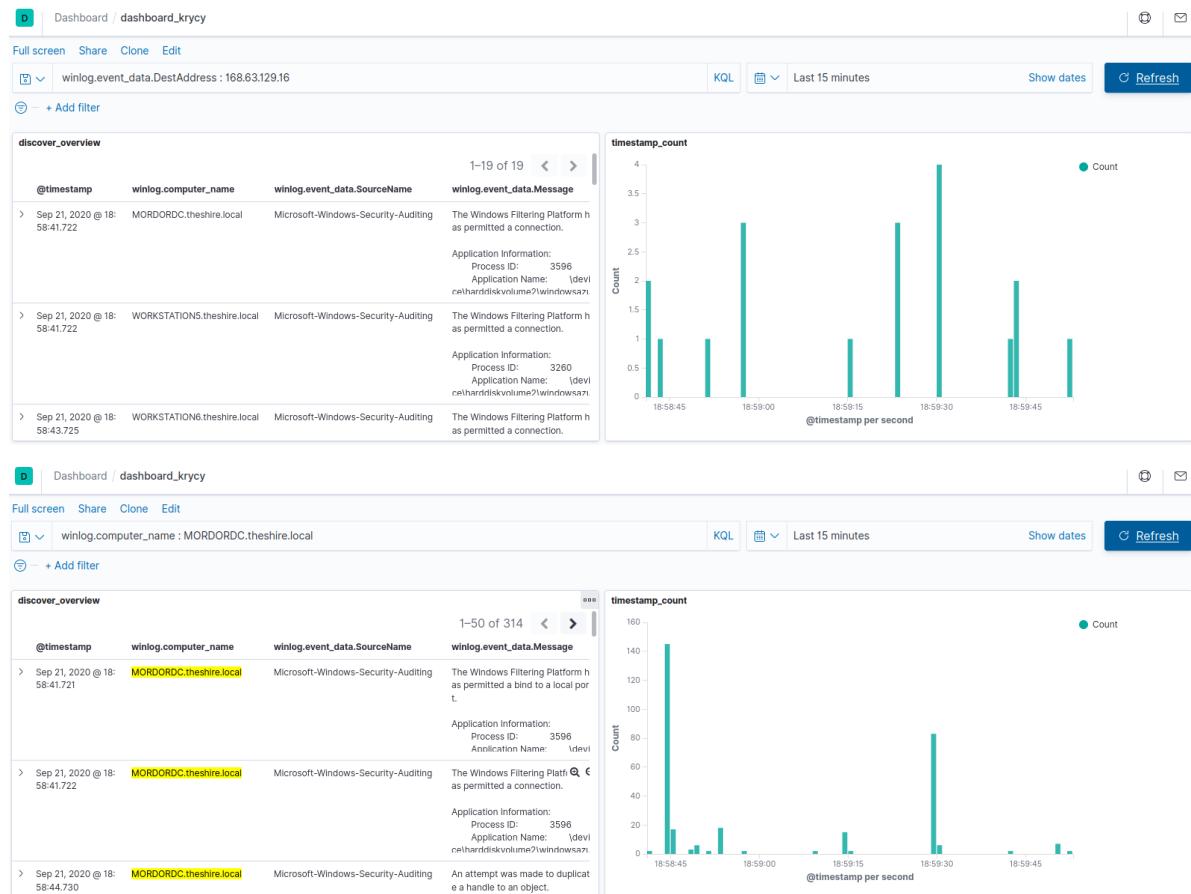
- lista wszystkich logów, ale z wyświetlonymi jedynie interesującymi nas polami,

- histogram czasu zachodzenia zdarzeń.



Na powyższym screenie mamy wiele spojrzeń na ten sam zbiór logów na jednej karcie, co już samo w sobie może stanowić duże ułatwienie przy analizowaniu logów.

Przy operowaniu na takich tablicach możemy dodatkowo korzystać z zapytań *KQL*:



Takie rozwiązanie daje analitykom bezpieczeństwa dużą dowolność i praktycznie nieograniczone możliwości w tworzeniu tablic zgodnie z preferencjami danego zespołu. Dobrze przygotowany *dashboard* może skutkować oszczędnością czasu zespołu w przyszłości.

## Elastalert

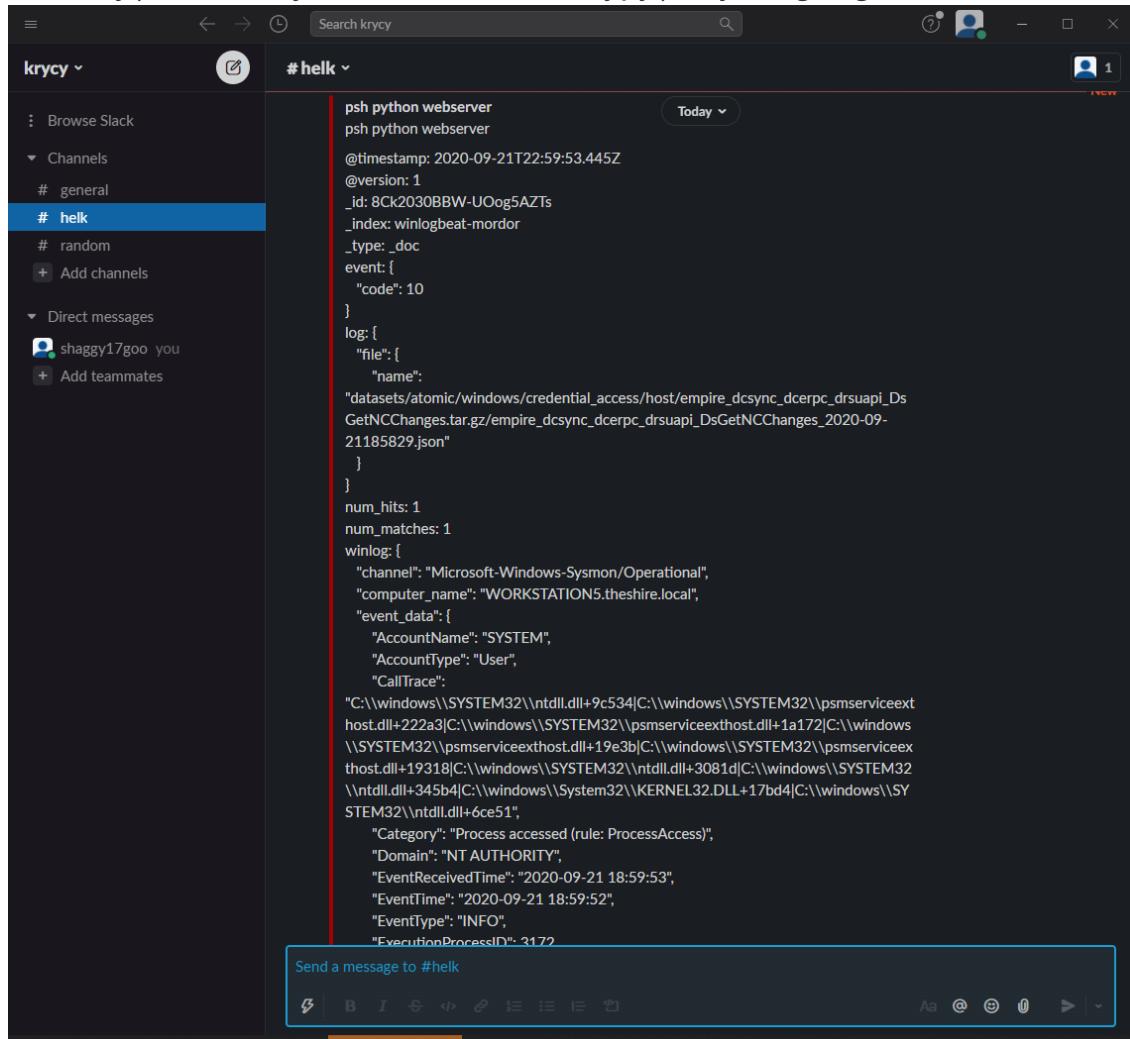
Elastalert jest kolejnym komponentem całego *HELK'a*, który pozwala nam na operowanie na logach z *Elasticsearch'a*. Tym razem mamy do czynienia z brakiem interfejsu użytkownika.

Możemy przygotować odpowiednie reguły np. w formacie zapytań *KQL* i w przypadku wykrycia dopasowania wysłać alert we wskazane miejsce.

- Przygotowana przez nas reguła wskazująca na *id* loga znajdującego się we wczytanym przez nas zestawie danych:

```
author: popolek
name: psh python webserver
description: krycyalert
date: 2021/11/25
index: winlogbeat-mordor
type: any
filter:
- query:
  query_string:
    query: (_id:8Ck2030BBW-UOog5AZTs)
alert:
- "slack"
slack_webhook_url: "https://hooks.slack.com/services/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"
```

- Odebrany pod wskazanym adresem alert zawierający podejrzanego loga:



Powyższe rozwiązanie pozwala na zmniejszenie czasu reakcji zespołu na podejrzane zdarzenie.

## **Reszta komponentów**

Pozostałe komponenty *HELK*'a jak np. *Apache Spark* w połączeniu z *Jupyter Notebook* i innymi pozwalały na dodatkowe zwiększenie możliwości analizy danych zebranych w *Elasticsearch*'u. Przykładem takiego działania może być tworzenie diagramów relacyjnych. Takie działania otwierają badaczom nowe perspektywy na analizowane zbiory danych.

## **HELK - Podsumowanie**

Wszelkie wykonane przez nas działania zostały wykonane na stosunkowo małych zbiorach danych. W normalnych warunkach takich logów może być znacznie więcej, dodatkowo mogą one pochodzić z zupełnie różnych źródeł. Sprawdzone przez nas w tym punkcie środowisko może okazać się bezcennym narzędziem w rękach doświadczonego analityka. Dzięki dołączonym do niego narzędziom, analiza zdarzeń dla większych zbiorów danych staje się zdecydowanie łatwiejsza.