VECTRA

THREAT HUNTING GUIDE

Threat Hunting With Vectra Recall

By Gearóid Ó Fearghaíl and Niall Errity

TABLE OF CONTENTS

Vectra AI protects business by detecting and stopping cyberattacks.

As a leader in network detection and response (NDR), Vectra AI protects your data, systems and infrastructure. Vectra AI enables your SOC team to quickly discover and respond to would-be attackers —before they act.

Vectra AI rapidly identifies suspicious behavior and activity on your extended network, whether on-premises or in the cloud. Vectra AI will find it, flag it, and alert security personnel so they can respond immediately.

Vectra AI uses artificial intelligence to improve detection and response over time, eliminating false positives so you can focus on real threats.

Introduction

Threat hunting is an important part of any security program. Regardless of how well-designed a security tool is, we must assume these tools and defenses are imperfect.

Threat Hunting is about setting aside time to do in depth research on the idiosyncrasies of your own network.

Business environments are constantly changing, new tools are introduced, old tools are removed, and these configuration changes are made to support the changes, which can introduce new vulnerabilities to the environment. Recent examples include the F5 vulnerability CVE-2020-5902 which impacted the Traffic Management User Interface (TMUI) of F5's BIG-IP; this port should never be publicly accessible and should require users securely authenticating and connecting to the LAN first before being able to access. At Vectra AI, we've seen instances where this was not the case and the TMUI has been accessed and exploited.

2020 brought a huge shift of remote work due to COVID-19 and had operations teams scrambling to 1) support this new work environment and 2) secure users as they made the shift from office to home. Supporting this type of shift, especially for a business not ready to support it, introduces a multitude of security headaches. In a seismic shift like this, the primary focus for the business is ensuring operations are not interrupted, which leaves security teams with less influence over implementation and stuck supporting a solution not designed with security in mind. Without proper oversight, vulnerabilities can be exposed and attackers will take advantage.

New tools are introduced, old tools are removed, and these configuration changes are made to support the changes, which can introduce new vulnerabilities to the environment.

There are many examples of why hunting is important, and the two we discuss below underline the need for hunting programs. Let's explore how security teams can leverage Vectra Detect and your Network Metadata to hunt for malicious behavior. In addition, while we reference Vectra Recall in this document, the techniques described for Vectra Recall can easily be implemented leveraging your data from Vectra Stream.

Why Threat Hunt?

Threat Hunting is about setting aside time to do in depth research on the idiosyncrasies of your own network.

The aim of a threat hunt is not just to find malicious actors within your network which Vectra's Behavioral Driven Detections have not necessarily spotted or to find precursor activity. It's also to find network activity that is not necessarily malicious, but might be in breach of your security posture, or needlessly insecure. Primarily, threat hunting is a learning experience that helps you understand what is happening on your network. This should simplify future investigations, as you already have an understanding of what is going on in the network.

As an organization, you might want to document your findings for knowledge sharing within the company. You might want to set aside a set amount of time each week or month for threat hunting as a team, with a discussion at the end when the team discusses what they spotted and what you now know about your organization that you didn't know before. It might be that there's a server which backs up a swathe of files over SMB at 1am every day, or it might be that some servers in a Data Center send a lot of data externally on port 46780 for a legitimate business use. These findings will save time in the future, as you can quickly discount and exclude known, legitimate use cases to focus in on anything novel and concerning.

A threat hunt is not just to find malicious actors within your network...it's also to find network activity that might be in breach of your security posture, or needlessly insecure.

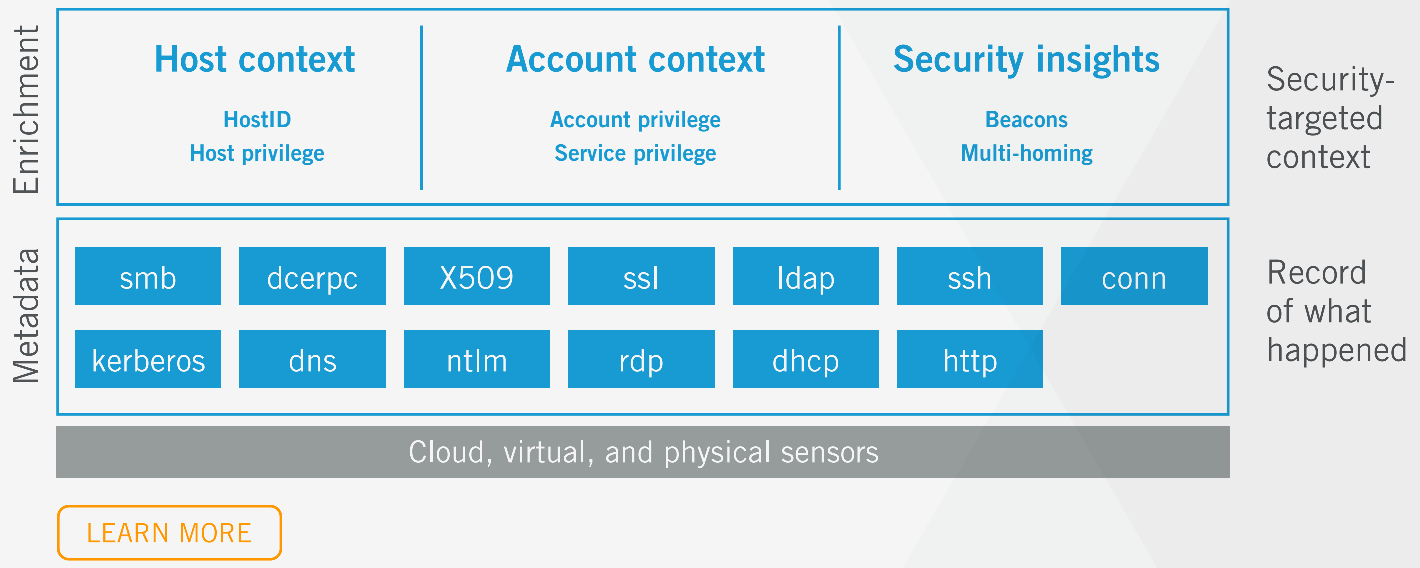
Why Network Data?

From an investigators point of view there are two main sources of evidence during an investigation: endpoint evidence and network evidence. The best way to describe the difference between these two sources is the analogy of grand theft auto. There are multiple stages from the carjacking, the joy ride and eventually the conclusion, which could be a car crash. Being at the scene of the crime is great, but it won't build the full picture. How did the thief find the car? Where did they come from? What route did the car take? The only way to see the full picture is to combine all elements.

While endpoint evidence is best to see the initial site of the breach, network data is best for seeing the full picture and connecting the dots. Imagine being in a helicopter observing the carjacking and watching as the car weaves in and out of traffic, down streets and across town. We'll see it all, and we'll see exactly where it ends.

Overview of What Metadata We Have

The following is a quick reference to the available metadata and the common attributes for each metadata stream.



The Value of Threat Hunting

Hunting is time consuming, there is a reason most organizations shy away from hunting; from a manager's perspective it's difficult to approve analyst time when you're not guaranteed an output. In our view, there are two things that usually result from a successful hunt.

Knowledge

Any analyst spending time on a hunt will inevitably learn from the experience, and they'll need to research and test their theory. This means a new topic is being explored as they become more comfortable with using the Vectra AI platform, which can be translated into time spent during investigations. They'll know Lucene syntax, how to stack data with visualize and the available metadata fields.

Environment Understanding

Along with this research they'll also understand their own environment better since every corporate network has a specific set of policies and tools they use. Getting to understand what's normal will help to identify what's abnormal. As an analyst spends time hunting, they'll increase their comprehension which will translate to efficiency.

A tangible outcome will be a custom model, so the knowledge and environment understanding can be applied to create a tailored custom model that will work for your organization. Doing this will allow for custom model enablement in Vectra Detect and fall into daily analyst workflow – both increasing attack coverage and efficiency.

Hunting Off Indicators

It's important to keep an ear to the ground and make sure you are aware of any new compromises which are announced. But it's just as important to be able to action new indicators of compromise when you hear of them.

In this section, we will describe common IOCs, what they can indicate, why you should care, and how you can search for these IOCs in your network metadata.

Domains

Any external actor needs to be able to manage breaches from outside of the network, and domains are a key tool for this to be managed. As we saw recently in the SUNBURST SolarWinds exploit, Command & Control operations were performed through domains along the lines of appsync-api.eu-west-1[.]avsvmcloud[.]com.

It is also a common tool in phishing attempts to use domain names which mimic popular sites in order to avoid suspicion in their target. For example, if a phishing attempt is made to steal credentials for someone's outlook account, a domain such as outlook.com.enteryourpassword.tk might be used to avoid suspicion.

Why Should I Care?

Domain Indicators of Compromise (IOCs) are a strong signal of compromise as these domains have been registered by a malicious actor and traffic for this express purpose. If any communication is seen to a domain IOC, then this strongly warrants investigation.

How to Search for Them

Known Bad Domains

You can easily convert a list of known bad domains from any source you have into a Recall query. We have created an excel file to do this for you, which you can from a Security Engineer, and given a list like this:

- Baddomain1.com
- Baddomain2.com

First convert this query to a Lucene query:

Resp_domain:(baddomain1.com OR baddomain2.com)

Then run this query on your iSession Metadata Stream

Suspiciously Familiar Domains

1. Create a list of common domains which users on your network access.
   E.g.: facebook, gmail, outlook, your company intranet.

2. Search for these items in your Vectra Recall iSession activity.
   *E.g. Resp_domain(*corpnet* OR *facebook* OR *gmail* OR *outlook* OR *office*)*

3. Save this search

4. Create a new "Data Table" Visualization with this Search as the source and split rows by "Term" and aggregate by "resp_domain"
   N.B. You could also split row by "Significant Terms", which will show interesting and unusual terms, to reduce noise. Read more here: [*https://www.elastic.co/guide/en/elasticsearch/reference/current/search-aggregations-bucket-significantterms-aggregation.html*](https://www.elastic.co/guide/en/elasticsearch/reference/current/search-aggregations-bucket-significantterms-aggregation.html)

5. A list of the most frequently accessed sites which match your search will appear. Legitimate domains should appear first. Hover over legitimate items and click 🔍 to exclude these items.

6. Save this visualization.

Any items left in this data table warrant further investigation, and if they are benign they should be excluded. At the start, you may have a lot of internal variations on your company's domain name. For example, at Vectra AI we have a huge number of Vectra AI internal domains such as dev.vectrai.ai, HR assets in hr.vectra.ai etc. You will need to effectively triage out these false positives to get good, actionable data.

After a few days of manually checking this visualization, if you are happy with what results are remaining, you should turn this search into a Custom model by saving the underlying search, and then navigating to the "Manage" section of the Detect UI. In the "Custom Models" tab, find your new saved search and activate it within its edit modal.

IP addresses

Known Bad IP addresses

You can easily convert a list of known bad IP addresses from any source you have into a Recall query. We have created an excel file to do this for you, which you can get off any Security Engineer, but given a list like this:

- 192.0.2.1
- 192.0.2.2

First convert this query to a Lucene query:

Id.orig_h:(192.02.1 OR 192.02.2) OR Id.resp_h:(192.02.1 OR 192.02.2)

Then run this query on your iSession Metadata Stream.

Ports

Novel Port Usage

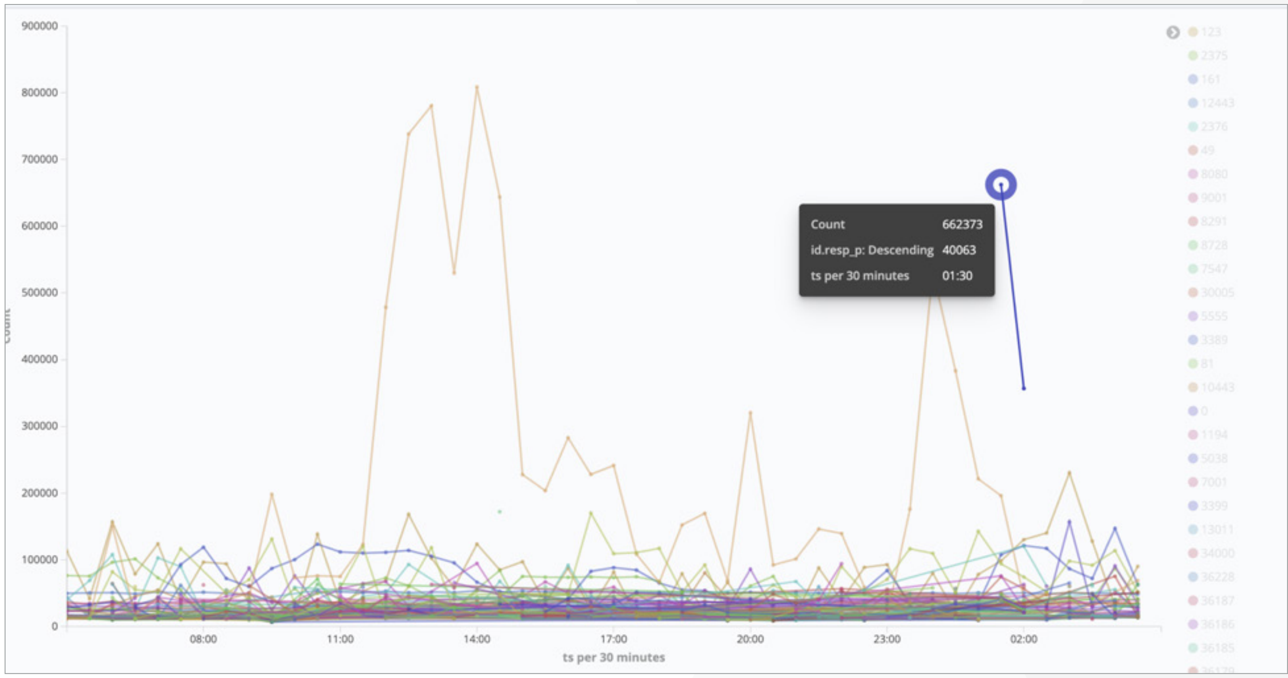
Most software uses a standard set of external ports to communicate. When new ports are seen on the network, this can indicate the installation of new software in the environment; or, in some cases, communication from a compromised host. Vectra AI creates info level detections which report when novel external connections are observed in the environment. You may want to aggregate these events using stream to leverage monitor if new software is being used on the network or potentially if malicious C2 channels are being set up. These events are mostly caused by benign user activity, but if your organization has a policy of limiting authorized applications, then spotting novel ports from systems outside of your IT admin team may be a sign of malicious activity, or a policy breach at the very least.

Spikes in Uncommon Port Usage

Spikes in network activity involving uncommon ports can be significant and warrant further investigation, as this port may be in use by malware to communicate with. A surge in activity necessitates further investigation.

The best way to review this activity like this is with a timeseries visualization. We have created one for you already in Vectra Recall called "*Hunting off Indicators: Spikes in Uncommon Port Usage - data" ? Y axis to count.*

An example of activity is below. The most common ports used have been excluded, and you can see two clearly spiking ports in use. Port 3283 is used for iChat, which is benign, and so could be excluded, but port 40063 is novel, and might warrant further investigation. You should also focus on standalone dots, which indicate spikes in traffic which were seen at no other time over your search period.



The steps involved above were:

- Create a date histogram with 24 hours of iSession data, with the count of connections as the y Axis
- Split the data into a separate series per responding port (id.resp_p)
- Filter out very common ports, e.g. 80/443 etc.
- Set a minimum threshold of activity to reduce the noise from minor ports by expanding "advanced" and setting {"min_doc_count":X}, where X would depend on the size of activity on your network, we have set this as 5,000 connections by default.

You should look to duplicate this visualization and to update the search query with ports which you know to be safe within your organisation. (N.B. If you try to make changes to the default Recall visualization, your changes will be overwritten).

Similarly, a spike of data transfer from an uncommon port is also something that is worth looking into and ensuring you have validated it is safe.

This works in the same way as with the count of traffic, but you should set the y-axis to show the total data sent. We have created a visualization called "*Hunting off Indicators: Spikes in Uncommon Port Usage – data*" which you can use as a starting point for this.

See Also Link to Protocols over non-standard ports section

File Names

Malicious files can be transported on the network over SMB or other protocols, and then could get executed on target hosts either through remote processes or social engineering. Monitoring specific known bad file extensions which can be used malicious actors can find instances where files are being transferred for nefarious purposes.

The SMB File Metadata stream in Vectra Recall shows each filename interacted with, and these data can be mined to find data which may be suspicious.

There are 2 specific examples we will describe here, but your mileage may vary.

- Suspicious file names
- Suspicious paths

Suspicious File Names

File names which are written by users tend to have real English words in them, whereas from experience, file names can be weak indicators of compromise.

Examples of these are:

- Very long file names, e.g.. TotallyNotMalwareactuallyThisVeryMuchIsMalware.jpg
- Files without vowels, e.g. dwtdfh.doc

But searches like these can be very noisy without organizational context.

You can search for file names like these using regular expression (regex) searches. These searches can be quite slow, so we would recommend running a search over 15 minutes initially, and then expanding the time range once you reduce noise.

To search for file names of 50 characters of longer, you would run the following search. **name:/.{50,}/**

You could use similar logic for files without vowels, searching with:

name:/.*\\[bcdfghjklmnpqrstvwxyz]{4,}\..*/

This search is a regular expression, with the regular expression logic contained in the forward slashes /

- .* = match any path
- \\ = backslash to denote start of a filename
- [bcdfghjklmnpqrstvwxyz]{{4,} = 4 or more consonants in a row
- **\. = full stop (denoting the start of the extension**
- **.* = match any extension**

You can also perform similar searches in metadata_httpsessioninfo to monitor unencrypted http traffic.

Using the above search, you should look to remove any common file names which you know are not malicious. For instance, if a Microsoft update server adds files to a specific folder with a long file name, you can exclude those files from the search with an exclude filter. Once you have removed these false positives from the network if they are appearing, you should expand the search time range to your full retention period. If there are very few files involved and you think that they are of a security significance, you should look to turn your search into a custom model and start firing detections for these filenames.

Suspicious File Paths

Some paths might be suspicious on your network, and warrant investigation, and a similar search should be used for this as for the suspicious file names example above.

You should collect a list of file paths you know are concerning in your org and create a search to monitor accesses against them. For this example below, let's focus on any accesses made of files in /App/Data/Roaming/.

The search you would perform is:

name:/ VAppVDataVRoamingV.*/

This search will match any file accesses in that directory. In our system, we had a lot of legitimate accesses to them from 2 update servers. To reduce the noise from this search, find legitimate servers that you would expect to be accessing the folder you are concerned about, and click 🔍 the beside its IP to exclude requests by that server.

Ja3 & Hassh

Ja3 & Hassh are fingerprinting methods which can recognize the source of SSL or SSH activity respectively based off available information from the cleartext packets sent before the encryption handshake is completed.

Ja3

Ja3 is a method to create SSL/TLS fingerprints which can be used to recognize the client or server in a given session. For example, a standard Tor client will have a Ja3 fingerprint of e7d705a3286e19ea42f587b344ee6865.

Ja3 fingerprints can indicate activity performed by the same application on multiple clients, and can also be used to check IOCs. This is not a foolproof solution, as it is possible for advanced attackers to alter their underlying fingerprints.

For example, to check if TOR activity has been seen on your network, go to the metadata_ssl* stream, and search for:

Ja3:e7d705a3286e19ea42f587b344ee6865

You can read more on Ja3, on the github profile https://github.com/salesforce/ja3

And a community driven repository of Ja3 fingerprints can be found here: https://ja3er.com/

A list of malicious JA3 fingerprints can be found here: https://sslbl.abuse.ch/ja3-fingerprints/.

Hassh & HasshServer

Hassh uses similar logic to Ja3 in SSH connections, creating fingerprints of SSH connections, this can be used to spot where a specific client has been communicating over SSH with multiple different servers, and to see if any activity which appears is novel.

Hassh is particularly useful in tightly controlled environments. If there are any important sections in a network, then you could look to search specifically in that subnet's SSH activity to see what Hassh are used there. Due diligence should be performed on these connections to make sure that none of this activity is malicious, and then each safe Hassh should be excluded from the search by clicking on the beside the field. Eventually, you should see no new Hassh in this subnet, and so you could save this search and activate it as a custom model (From manage -> custom models in the detect UI).

The Github community profile for Hassh lists many other uses from this data.

Read more on Hassh here: https://github.com/salesforce/hassh

Categories of Attack Steps

In the pages below, we will describe attack techniques you can search for in your Network Metadata. We have split these techniques down into what step described in the MITRE ATT&CK framework they relate to.

Reconnaissance

Inbound Scanning

Metastream: metadata_httpsessioninfo*, metadata_rdp*, metadata_smbfiles*, metadata_ssh*

ATT&CK: T1595

Description: Before compromising a victim, adversaries may execute active reconnaissance scans to gather information that can be used during targeting. Active scans are those where the adversary probes victim infrastructure via network traffic, as opposed to other forms of reconnaissance that do not involve direct interaction.

Example: Typically, if an attacker wants to fingerprint services running within your environment, they'll send HTTP requests inbound. Usually, Vectra AI sensors are placed behind the NAT point at the firewall making it difficult to determine the true source IP address. This is the behavior for outbound web traffic with a proxy described here. When inbound web traffic, the source IP address would not be NATed (only destination most likely) and XFF would not be present. The web server (and our sensor) would see real IP address. If LB between web server, it may be Source NAT-ed but we can capture the traffic before. As a work-around to this we can leverage HTTP X-Forward-For headers to ensure we're seeing the true source and not the NATTED IP behind the gateway/proxy/load balancer. Your ingress point needs to be configured to include this header information, by default most devices don't enable this header.

Once enabled you can identify the external requests by leveraging the local_orig field and setting to false

Initial Access

SQL Injection Activity

Metastream: metadata_httpsessioninfo*

ATT&CK: T1190

Description: Adversaries may attempt to take advantage of a weakness in an Internet-facing computer or program using software, data, or commands in order to cause unintended or unanticipated behavior. The weakness in the system can be a bug, a glitch, or a design vulnerability. These applications are often websites, but can include databases (like SQL), standard services (like SMB or SSH), network device administration and management protocols (like SNMP and Smart Install), and any other applications with Internet accessible open sockets, such as web servers and related services. Depending on the flaw being exploited this may include Exploitation for Defense Evasion.

If an application is hosted on cloud-based infrastructure, then exploiting it may lead to compromise of the underlying instance. This can allow an adversary a path to access the cloud APIs or to take advantage of weak identity and access management policies.

For websites and databases, the OWASP top 10 and CWE top 25 highlight the most common web-based vulnerabilities.

Example: For this example, we're going to pivot from a detection, if you come across an SQL Injection Activity detection you can follow these steps to investigate further.

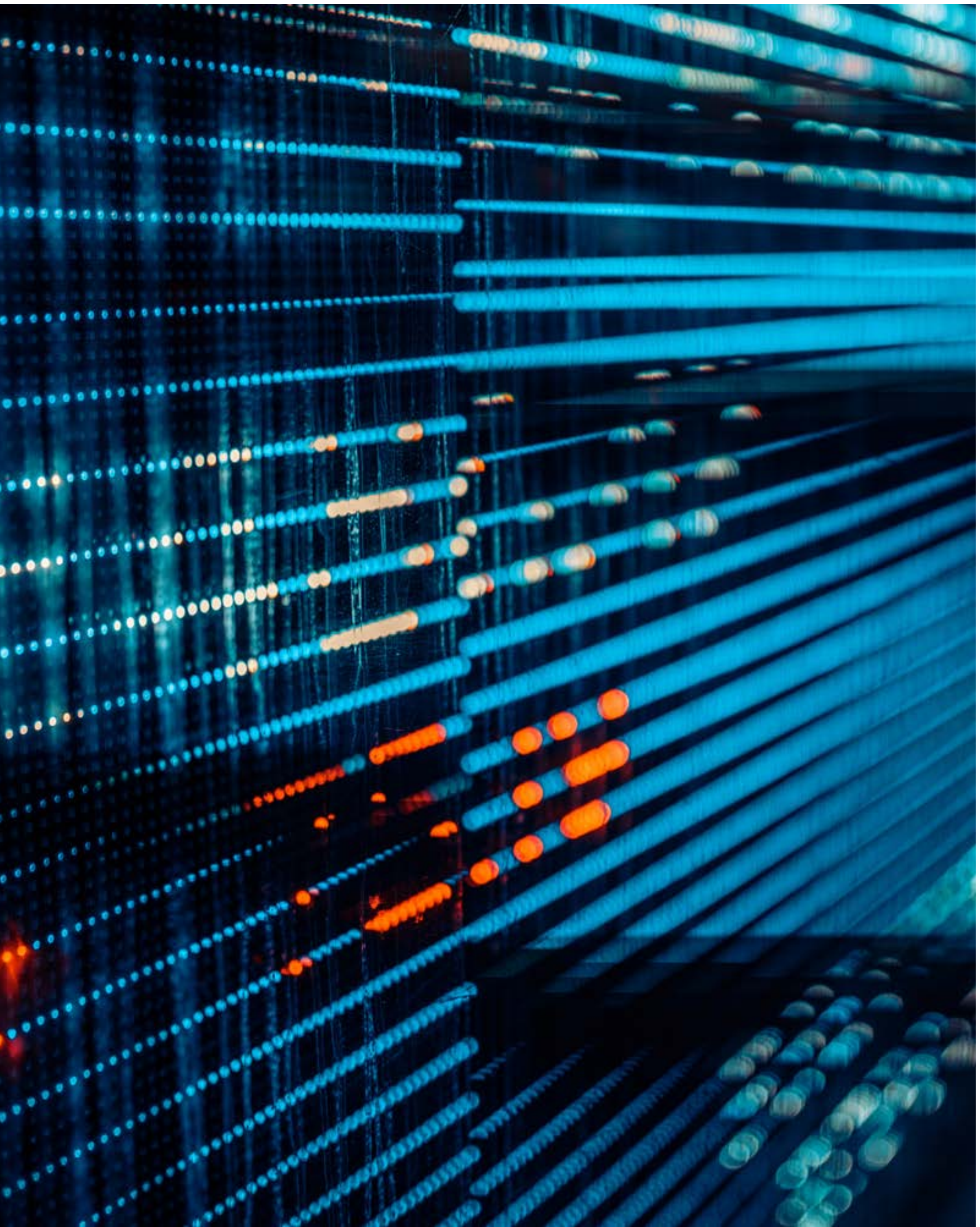
First identify active SQL Injection Activity detections within the detect platform that are not triaged, navigate to Detections > Advanced Search and type the following query into the search field:

detection.detection_type:"SQL Injection Activity" AND detection.is_triaged:false AND NOT detection.state:inactive

Pick one of the detections and review the details, note the source, target, the X-Forwarded-For header, port used and bytes received. If the bytes received is large it could indicate a successful SQL injection and database information was returned to the adversary, review the pcap to see what response data the requests were being met with. The response codes are not reliable here for SQL Injection, a server could respond to the clients request with an error code and yet still return valid data as the SQL statement was successful.

Next review the X-Forwarded For header IP, it's likely your web server will be behind a proxy or load balancer and the source IP will not be the true source. With the X-Forwarded For header configured it will append the true public source as a header to the HTTP request allowing the Vectra AI-driven platform to parse this information and present it within the UI.

Pivot into Vectra Recall and under Discover > metadata_httpsessioninfo* search for the X-forwarded-For IP using the proxied field and review all requests from this source. Start with a small window of maybe 10 to 20 minutes and gradually expand out your search time. Review the user-agent, targeted uri and bytes sent/received. These fields can help identify if this activity was an automated tool and if any of the requests were successful.



Execution

Possible Powershell User-Agent

Metastream: metadata_httpsessioninfo*

ATT&CK: T1059.001

Description: Adversaries may abuse PowerShell commands and scripts for execution. PowerShell is a powerful interactive command-line interface and scripting environment included in the Windows operating system. Adversaries can use PowerShell to perform a number of actions, including discovery of information and execution of code. Examples include the Start-Process cmdlet which can be used to run an executable and the Invoke-Command cmdlet which runs a command locally or on a remote computer (though administrator permissions are required to use PowerShell to connect to remote systems).

PowerShell may also be used to download and run executables from the Internet, which can be executed from disk or in memory without touching disk.

A number of PowerShell-based offensive testing tools are available, including Empire, PowerSploit, PoshC2, and PSAttack.

PowerShell commands/scripts can also be executed without directly invoking the powershell.exe binary through interfaces to PowerShell's underlying System.Management.Automation assembly DLL exposed through the .NET framework and Windows Common Language Interface (CLI).

Example:

```
user_agent:*Powershell* OR user_agent:*powershell* OR user_agent:*PSVersionV*
```

Defense Evasion

Legitimate Binary – Certutil

Metastream: metadata_httpsessioninfo*

ATT&CK: T1140, T1105

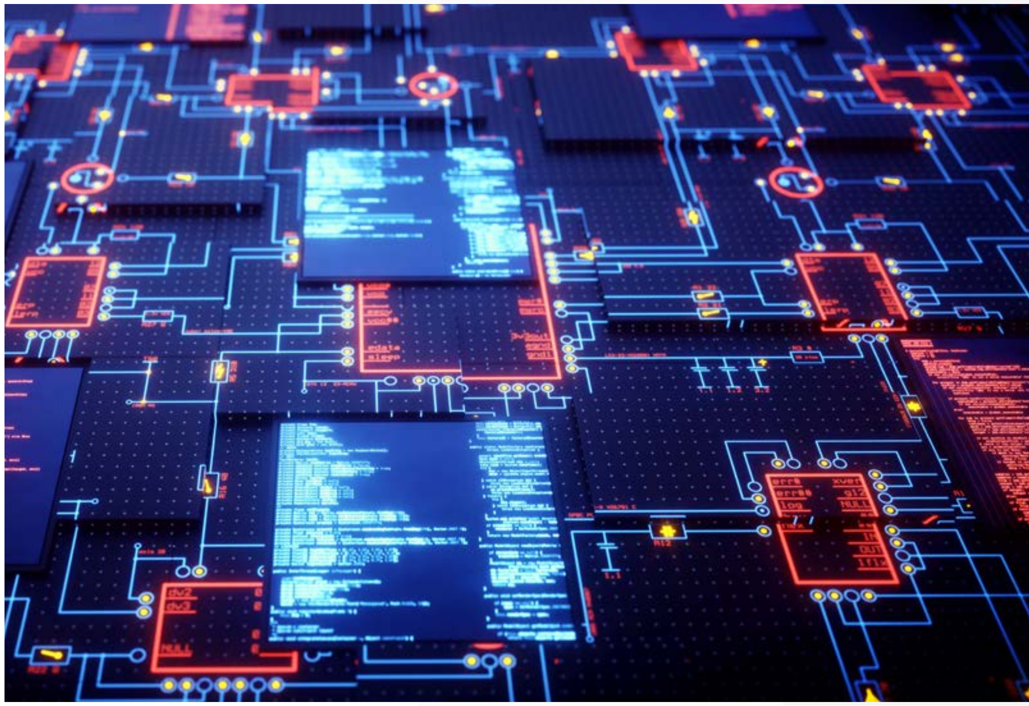
Description: Adversaries may use Obfuscated Files or Information to hide artifacts of an intrusion from analysis. They may require separate mechanisms to decode or deobfuscate that information depending on how they intend to use it. Methods for doing that include built-in functionality of malware or by using utilities present on the system.

One such example is use of certutil to decode a remote access tool portable executable file that has been hidden inside a certificate file. Another example is using the Windows copy /b command to reassemble binary fragments into a malicious payload.

Sometimes a user's action may be required to open it for deobfuscation or decryption as part of User Execution. The user may also be required to input a password to open a password protected compressed/encrypted file that was provided by the adversary.

Example: A common obfuscation technique is to use legitimate tools or binaries already on the target system to download your payload and execute it. There are many examples of this including bitsadmin.exe, esentutl.exe, ftp.exe, excel.exe and certutil.exe. Each one of these binaries has a legitimate use and are used legitimately within most organizations, we can however observe their execution and determine which usage is expected and which usage is unexpected. This is known as 'Living off the land'.

For this example, we'll take the tool certutil.exe, this binary is part of Windows operating system and used to manage certificates on the system. For any new tool or binary, we're attempting to hunt for we need to determine how we can identify its execution and if network visibility is the right choice. In our case of Certutil we can leverage its user agent to identify the execution.



Microsoft-CryptoAPI/10.0

To identify your target binary's user-agent string or other unique identifiers we could leverage to pivot on consider executing the binary and capturing the traffic. Other good pivoting points could be the certificate information if it's encrypted or the JA3/JA3s to fingerprint the header information.

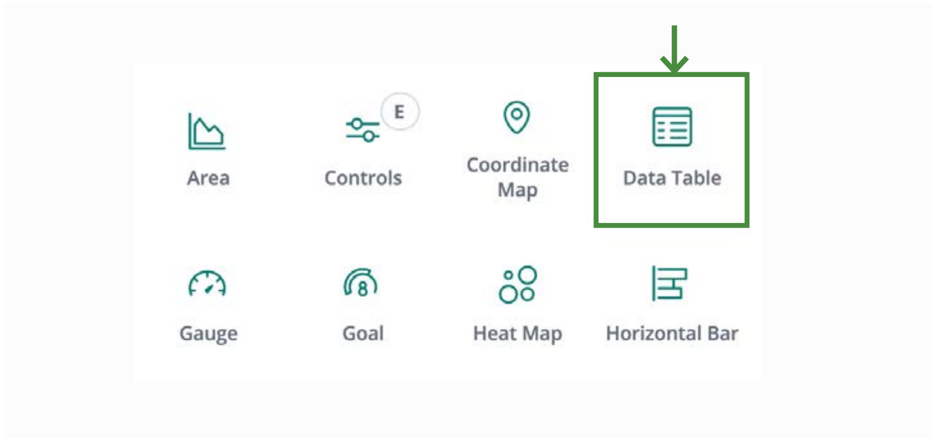
Next as we know this traffic is HTTP we can leverage the metadata_httpsessioninfo* and the field user_agent to search for its traffic. Run the following query:

user_agent:Microsoft-CryptoAPIV10.0

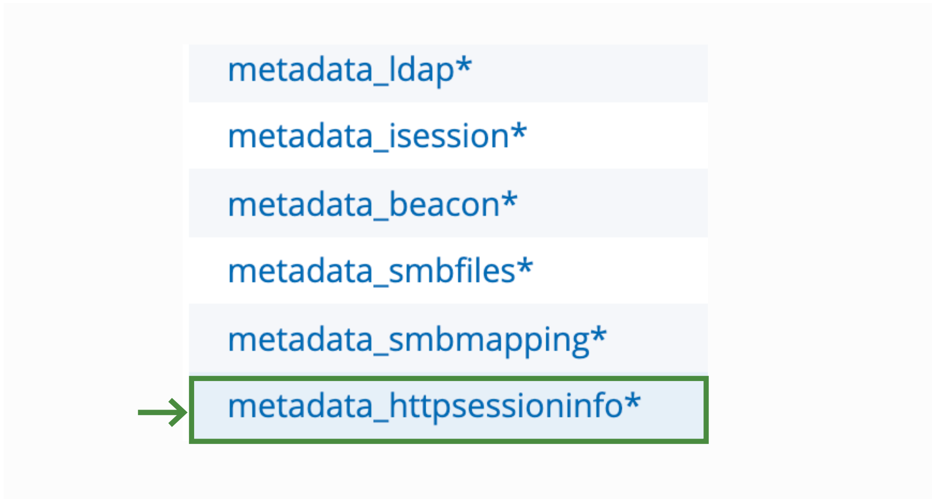
You're likely going to see a lot of results, for my test case over just a 24hr period I had 393,954 results, where do you even start?

Vectra Recall has the option to stack this data from a specific field and see how many results exist for each unique value, let's navigate over to Visualize to do this.

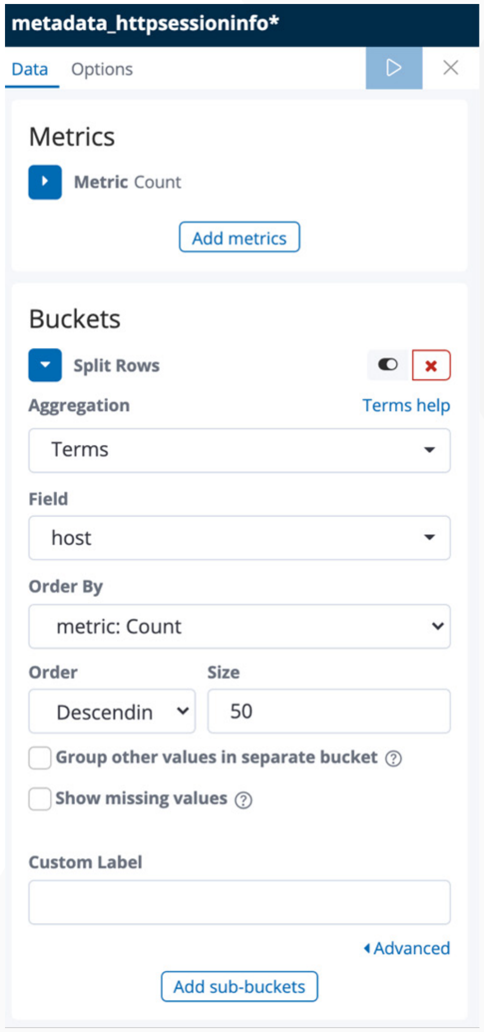
Select the + symbol in the top right then select 'Data Table'



and last select metadata_httpsession*.



Once here we need to specify the filter properties, first drop your quey in the top search bar. Next on the left had side fill the options with the same values in the screenshot, this will give you the same table view I have to the right.


metadata_httpsessioninfo*

Data Options

Metrics

Metric Count

Add metrics

Buckets

Split Rows

Aggregation: Terms

Field: host

Order By: metric: Count

Order: Descendin

Size: 50

Group other values in separate bucket

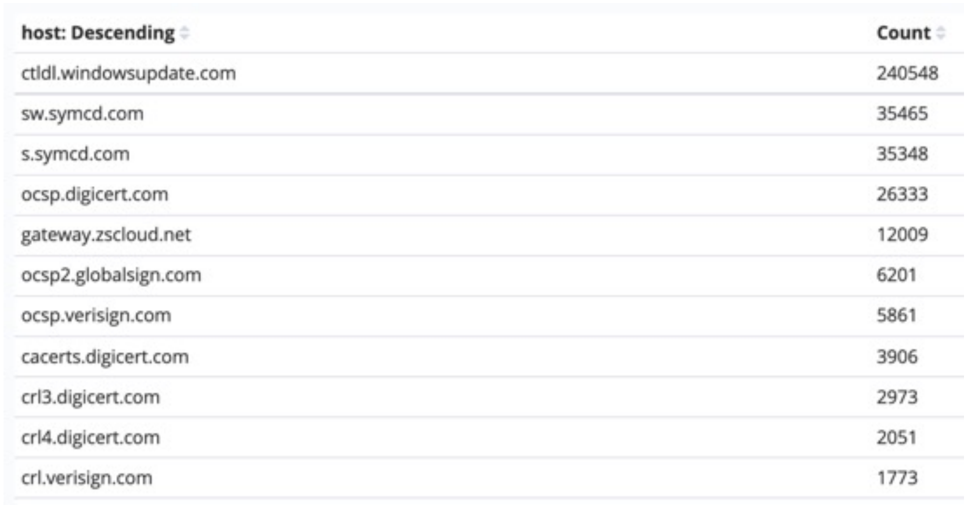
Show missing values

Custom Label

Advanced

Add sub-buckets

And this is the result we should see:



host: Descending	Count
ctldl.windowsupdate.com	240548
sw.symcd.com	35465
s.symcd.com	35348
ocsp.digicert.com	26333
gateway.zscloud.net	12009
ocsp2.globalsign.com	6201
ocsp.verisign.com	5861
cacerts.digicert.com	3906
crl3.digicert.com	2973
crl4.digicert.com	2051
crl.verisign.com	1773

Certutil is only used with a hand full of domains, from my example although we observed a total of nearly 400,000 unique requests the host count was less than 100! Focus on the bottom and work your way up, why are you only seeing a handful of requests to these hosts?

Credential Access

DCSync Execution

Metastream: metadata_dcerpc-*, metadata_ldap*

ATT&CK: T1003.006

Description: Adversaries may attempt to access credentials and other sensitive information by abusing a Windows Domain Controller's application programming interface (API) to simulate the replication process from a remote domain controller using a technique called DCSync.

Members of the Administrators, Domain Admins, and Enterprise Admin groups or computer accounts on the domain controller are able to run DCSync to pull password data from Active Directory, which may include current and historical hashes of potentially useful accounts such as KRBTGT and Administrators. The hashes can then in turn be used to create a Golden Ticket for use in Pass the Ticket or change an account's password as noted in Account Manipulation.

DCSync functionality has been included in the "lsadump" module in Mimikatz. Lsadump also includes NetSync, which performs DCSync over a legacy replication protocol.

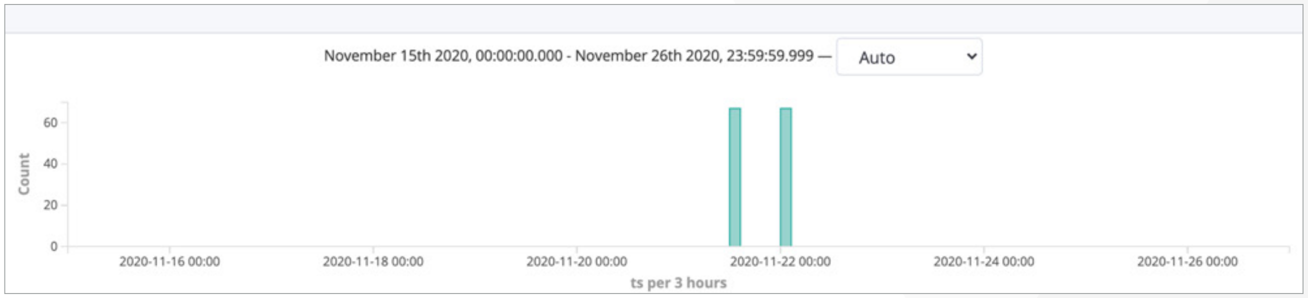
Example: Vectra AI tracks RPC usage across all operations, specifically for this attack we're interested in the *DRSGetNCChanges* operation. It's a normal operation for Domain Controllers to execute when syncing passwords, however, we shouldn't see this operation occurring from non-domain controller endpoints.

```
operation:DRSGetNCChanges
```

Where possible, narrow the source condition down to exclude Domain Controllers and focus on any systems executing this operation against a domain controller. With a good naming convention, we can quickly achieve this by leveraging the **orig_hostname** field as seen below.

```
operation:DRSGetNCChanges AND NOT orig_hostname:/.*[dD][cC].*/
```

Look for multiple requests in rapid succession.



endpoint	drsuapi
id.ip_ver	ipv4
id.orig_h	10.234.50.200
id.orig_p	58058
id.resp_h	10.234.100.32
id.resp_p	49666
local_orig	true
local_resp	true
operation	DRSGetNCChanges
rtt	596
sensor_uid	ousp605q
uid	gKA-N8TTousp605q
username	DC01$

From here we can identify hosts of interest and review the username and other operations conducted around the same timeframe.

Furthermore, we can pivot into **metadata_ldap*** and review queries of interest, in this example we identified a number of ***samaccountname*** and ***computer*** queries for objects the attacker was enumerating.

```
(samaccountname=admin1)
(samaccountname=admin2)
(&(objectClass=user)(samaccountname=*[account1]))
(&(objectClass=user)(samaccountname=*[account2]))
(&(objectClass=user)(samaccountname=*[account3]))
(&(objectClass=user)(samaccountname=*[account4]))
(&(objectClass=user)(samaccountname=*[account5]))
(&(objectClass=user)(samaccountname=*[account6]))
(&(objectClass=user)(samaccountname=*[account7]))
(&(objectClass=computer)(name=[hostname1]))
(&(objectClass=computer)(name=[hostname2]))
(&(objectClass=computer)(name=[hostname3]))
(&(objectClass=computer)(name=[hostname4]))
(&(objectClass=computer)(Name=[hostname5]))
(servicePrincipalName=*/)
```

This type of enumeration behavior is covered as a detection under **Targeted RPC Recon** detection. If you are investigating one of these detections, you can leverage the above steps to investigate the activity further.

Kerberoasting

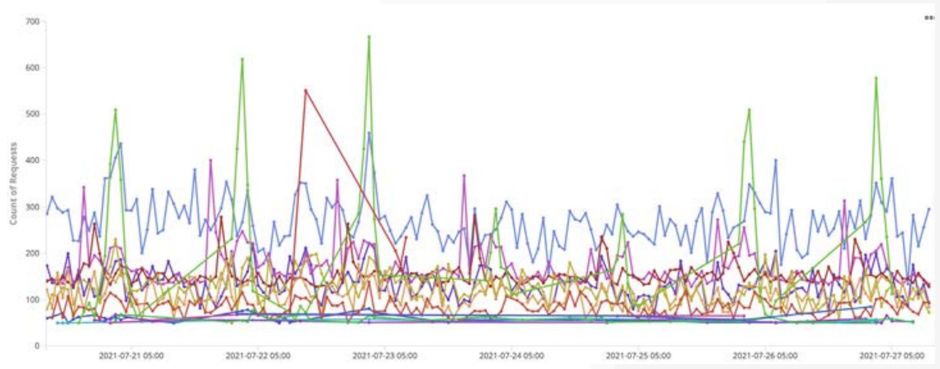
Metastream: metadata_Kerberostxn*

ATT&CK: T1558.003

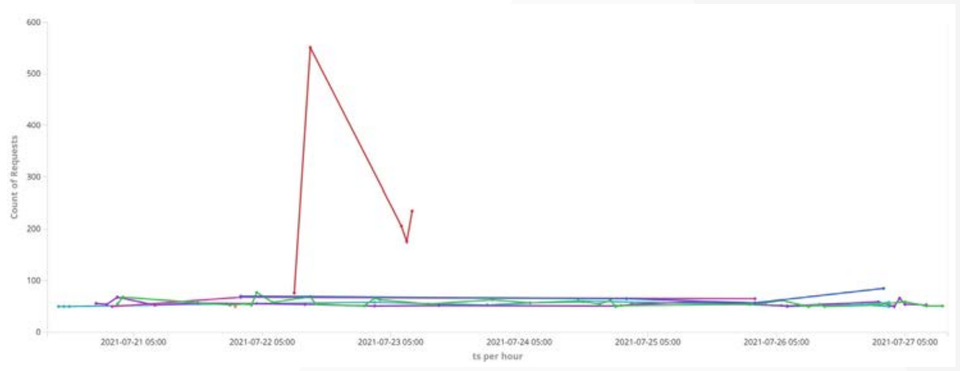
Using either a Forged Ticket Granting Ticket (TGT / Golden ticket) or a compromised account, the attacker can request access to a service (SPN) on the network. This service I associated with a high privilege service account, for example a SQL service account. The Key Distribution Centre (KDC) will issue a service ticket, which is encrypted with the public key of the Service Accounts password. The attacker can then convert this service ticket to a hash which can be exported to Hashcat or John The Ripper and then proceed to crack the password offline. This attack is reliant on poor password hygiene for service accounts, reuse of passwords across service accounts, non expiry of passwords for service accounts, and even non removal of old SPN entries in Active Directory.

Example: To hunt for potential evidence of Kerberoasting on your network, a good starting point is Vectra Recall's Kerberoasting Dashboard. This dashboard monitors for tickets responses with weak ciphers (RC4) that can be potentially cracked offline. Typically, the usage of weak ciphers should be minimal within your enviornment, as with any example here it's possible your environment might have a large number of Kerberos RC4 requests rendering this dashboard less effective.

When you look at this dashboard, you'll see a top chart which shows all users of the weak RC4 cipher, this chart should hopefully be empty, as no one in your org is using this weak cipher, but it may also look like this. It's safe to say that these Kerberos transactions are all from legitimate business cases, so you should look to hide these instances from the chart by clicking on the "–" icon beside each IP in the legend.



After hiding the most commonly occurring servers, you should see a chart like the one below with a clear outlier that warrants investigation.



Click on this server IP and click on the "+" icon to focus only on this, and at the bottom of this dashboard, you'll be able to quickly see the clients making requests to this server, and if a single client has made a large number of requests against it, you should pivot into other metadata sources such as LDAP and RPC to determine if any other suspicious activity was occurring around the given timeframe.

Discovery

LDAP Queries

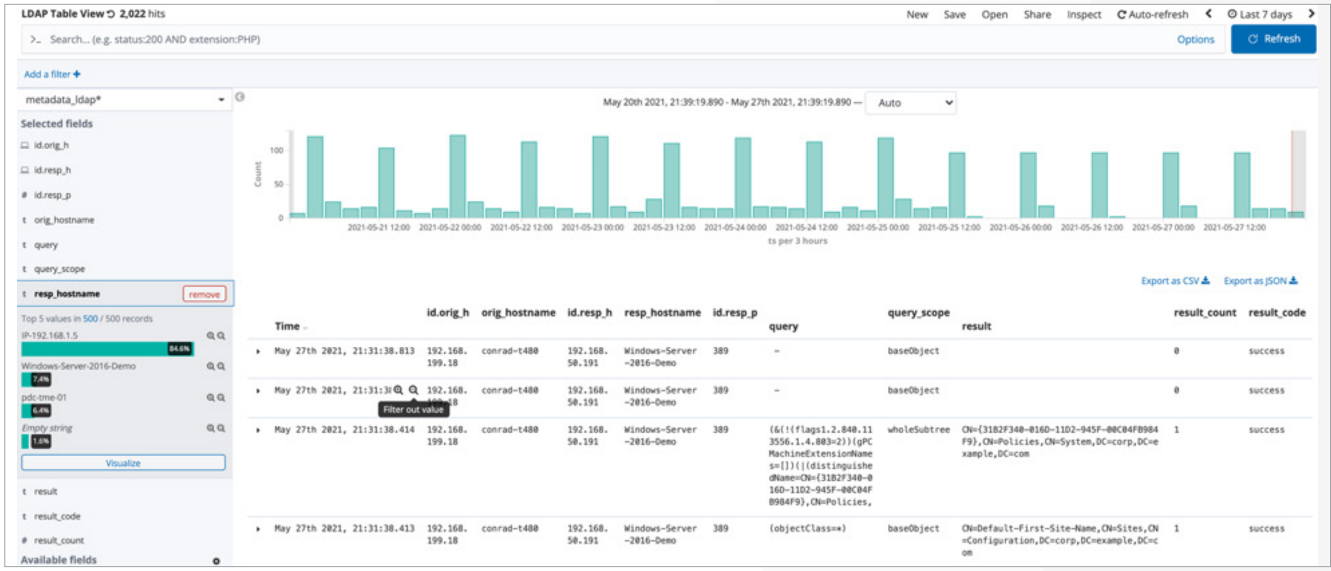
ATT&CK: T1087.002

Description: Once an adversary has access to the network, they're going to begin the internal reconnaissance or 'discovery' phase to identify what level of access they have. There are several ways to achieve this, which Vectra AI offers strong detection coverage for out of the box, but a prominent way is by querying active directory over LDAP (Lightweight Directory Access Protocol) which is an industry standard and a default protocol for Active Directory servers.

Using LDAP we can identify computers, accounts, groups and applications within the domain. Using this information, we can quickly identify targets of interest, why waste your time cracking passwords if the accounts won't give you the permissions you need to RDP from your current host to the domain controller? Common tools that leverage LDAP include Bloodhound, Sharphound and ADFind.

https://github.com/BloodHoundAD/**BloodHound**

http://www.joeware.net/freetools/tools/adfind/



Time	id.orig_h	orig_hostname	id.resp_h	resp_hostname	id.resp_p	query	query_scope	result	result_count	result_code
May 27th 2021, 21:31:38.813	192.168.199.18	conrad-t480	192.168.50.191	Windows-Server-2016-Demo	389	-	baseObject		0	success
May 27th 2021, 21:31:3[illegible]	192.168.[illegible]18	conrad-t480	192.168.50.191	Windows-Server-2016-Demo	389	-	baseObject		0	success
May 27th 2021, 21:31:38.414	192.168.199.18	conrad-t480	192.168.50.191	Windows-Server-2016-Demo	389	(&(!(flags1.2.840.113556.1.4.803=2))(gPCMachineExtensionNames=[])(|(distinguishedName=CN={31B2F340-016D-11D2-945F-00C04FB984F9},CN=Policies,	wholeSubtree	CN={31B2F340-016D-11D2-945F-00C04FB984F9},CN=Policies,CN=System,DC=corp,DC=example,DC=com	1	success
May 27th 2021, 21:31:38.413	192.168.199.18	conrad-t480	192.168.50.191	Windows-Server-2016-Demo	389	(objectClass=*)	baseObject	CN=Default-First-Site-Name,CN=Sites,CN=Configuration,DC=corp,DC=example,DC=com	1	success

Note from the example screenshot I've selected 'LDAP Table View', this is a quick way to create a table view of the most relevant data.

Example: Starting in Vectra Recall Discover select the metadata_ldap* metastream and select a time period to begin, for this technique I'd start with a minimum of 24hr to ensure we have a sufficient data set. Next let's get a picture of which servers are accepting LDAP requests by selecting the resp_hostname on the left side of the screen.

I can see most of my requests are hitting IP-192.168.1.5, a hostname beginning with IP- is an IP host container and occurs when the Vectra AI platform hasn't seen evidence on the network to identify the hostname associated with the IP address. This can sometimes occur for static IP addresses; we can improve hostname coverage by integrating RDNS and updating the setting page with static IP subnets such as ranges assigned to servers.

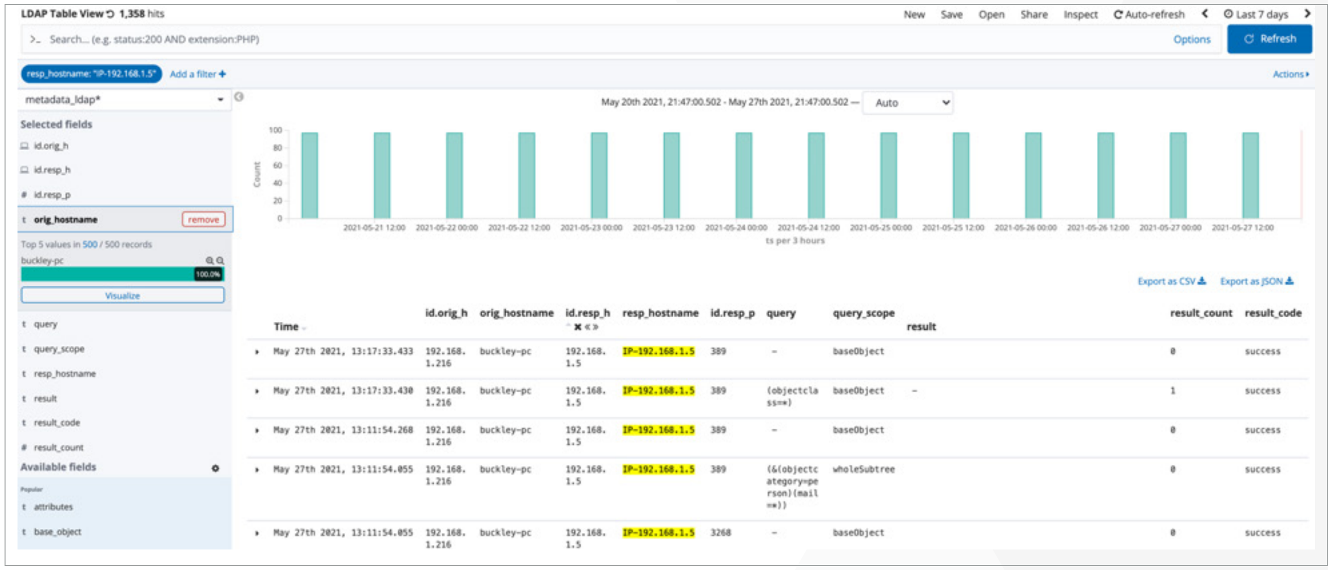
In my demo lab I know 192.168.1.5 is the primary domain controller and I'm going to filter for this specific IP by clicking on the magnifying glass with the + symbol. You'll now see a pill filter added below the search bar.

These queries appear to be consistent, maybe it's just some software running regular queries against my server? Using the same quick filters to the left of my screen I can quickly see the host 'buckley-pc' is responsible for all these requests. Typically, we wouldn't expect to see such a large volume of requests from desktop or laptop unless there is some specific software that requires LDAP queries to function. I've only seen one example of this in the wild and that was for a manufacturing plant where the front-line manages ran local software to manage employees scheduling, that software made regular LDAP queries to get user information.

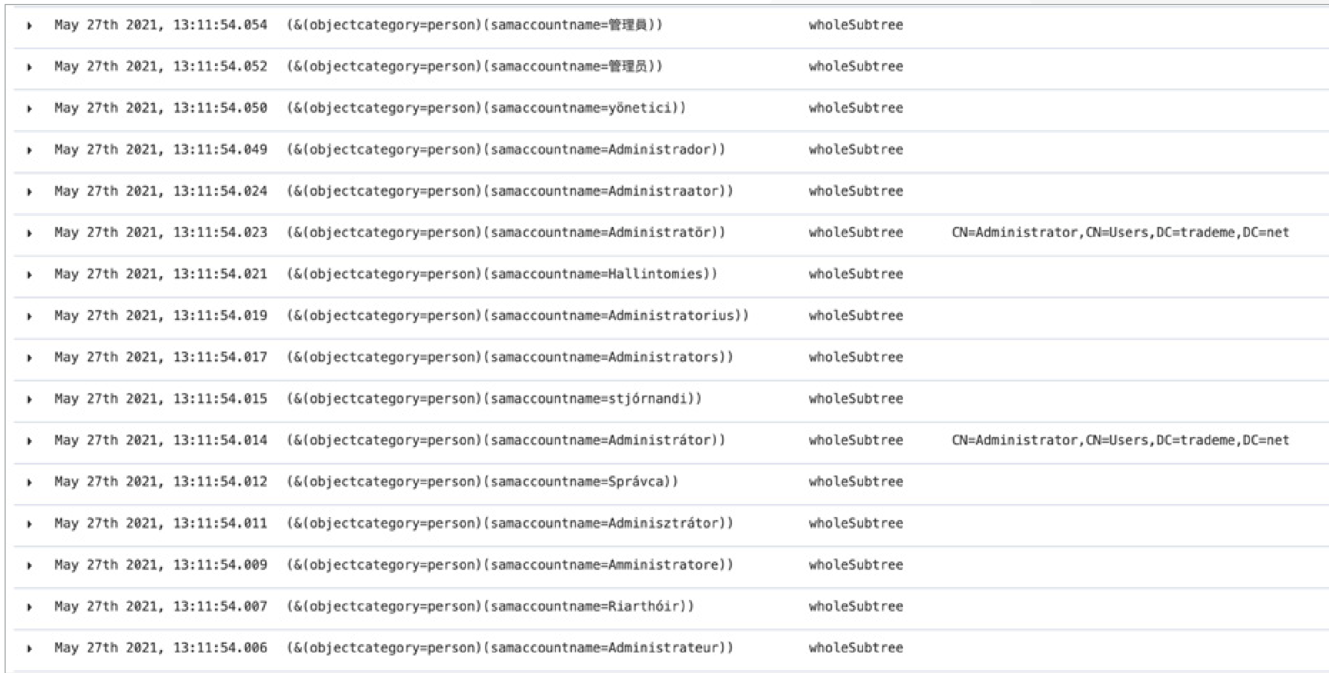
Next, I'm going to clean up my table view, I can do this by selecting the 'x' below each column header I want to remove, this will help me clean up the view and see the QUERY and RESULT fields better.

Once you have the right table view you can remove the query (objectClass=*), this is a common chatter query and although it returns all results I'm going to focus on targeted queries. Once this is complete, I can easily see what buckley-pc has been doing.

Straight away you should notice the times for these queries, these are far too fast for a user to manually query and must have been automated. The queries



Time	id.orig_h	orig_hostname	id.resp_h	resp_hostname	id.resp_p	query	query_scope	result	result_count	result_code
May 27th 2021, 13:17:33.433	192.168.1.216	buckley-pc	192.168.1.5	IP-192.168.1.5	389	-	baseObject		0	success
May 27th 2021, 13:17:33.430	192.168.1.216	buckley-pc	192.168.1.5	IP-192.168.1.5	389	(objectclass=*)	baseObject	-	1	success
May 27th 2021, 13:11:54.268	192.168.1.216	buckley-pc	192.168.1.5	IP-192.168.1.5	389	-	baseObject		0	success
May 27th 2021, 13:11:54.055	192.168.1.216	buckley-pc	192.168.1.5	IP-192.168.1.5	389	(&(objectcategory=person)(mail=*))	wholeSubtree		0	success
May 27th 2021, 13:11:54.055	192.168.1.216	buckley-pc	192.168.1.5	IP-192.168.1.5	3268	-	baseObject		0	success



May 27th 2021, 13:11:54.054	(&(objectcategory=person)(samaccountname=管理員))	wholeSubtree	
May 27th 2021, 13:11:54.052	(&(objectcategory=person)(samaccountname=管理员))	wholeSubtree	
May 27th 2021, 13:11:54.050	(&(objectcategory=person)(samaccountname=yönetici))	wholeSubtree	
May 27th 2021, 13:11:54.049	(&(objectcategory=person)(samaccountname=Administrador))	wholeSubtree	
May 27th 2021, 13:11:54.024	(&(objectcategory=person)(samaccountname=Administraator))	wholeSubtree	
May 27th 2021, 13:11:54.023	(&(objectcategory=person)(samaccountname=Administratör))	wholeSubtree	CN=Administrator,CN=Users,DC=trademe,DC=net
May 27th 2021, 13:11:54.021	(&(objectcategory=person)(samaccountname=Hallintomies))	wholeSubtree	
May 27th 2021, 13:11:54.019	(&(objectcategory=person)(samaccountname=Administratorius))	wholeSubtree	
May 27th 2021, 13:11:54.017	(&(objectcategory=person)(samaccountname=Administrators))	wholeSubtree	
May 27th 2021, 13:11:54.015	(&(objectcategory=person)(samaccountname=stjórnandi))	wholeSubtree	
May 27th 2021, 13:11:54.014	(&(objectcategory=person)(samaccountname=Administrátor))	wholeSubtree	CN=Administrator,CN=Users,DC=trademe,DC=net
May 27th 2021, 13:11:54.012	(&(objectcategory=person)(samaccountname=Správca))	wholeSubtree	
May 27th 2021, 13:11:54.011	(&(objectcategory=person)(samaccountname=Adminisztrátor))	wholeSubtree	
May 27th 2021, 13:11:54.009	(&(objectcategory=person)(samaccountname=Amministratore))	wholeSubtree	
May 27th 2021, 13:11:54.007	(&(objectcategory=person)(samaccountname=Riarthóir))	wholeSubtree	
May 27th 2021, 13:11:54.006	(&(objectcategory=person)(samaccountname=Administrateur))	wholeSubtree	

also appear to be very specific and iterating through different language variations of 'Administrator', lets break down one of these queries.

(&(objectcategory=person)(samaccountname=Administrator))

Very simply we can see two parameters, the first is the objectcategroy which is a single-valued property that contains the distinguished name of either the class of which the object is an instance or one of its superclasses. The second parameter is the logon name used to support clients and servers within Windows environments.

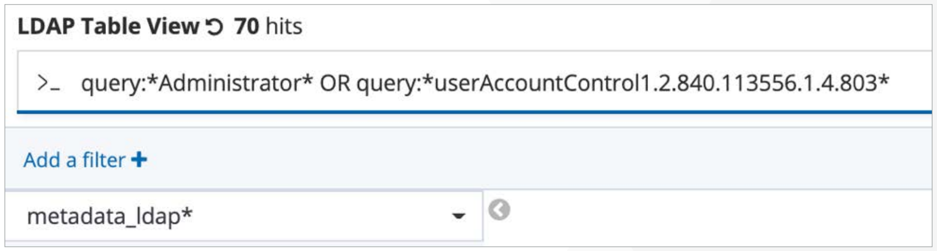
Taking another slightly more complicated example

(&(objectCategory=computer)
(userAccountControl1.2.840.113556.1.4.803=8192))

The new value here, userAccountControl, is the windows name for the User-Account-Control attribute and identifies the permission level of an account. Thankfully most of these user IDs are documented, the specific ID 1.2.840.113556.1.4.803 is used to identify objects that are domain controllers.

This activity is clearly reconnaissance type behavior and requires further analysis, but what have we learnt from this and how can we improve our hunting for next time? For a start we've identified a number of interesting queries the attacker was performing, how often do we see systems querying for Administrator account or looking for domain controllers? The following query will help us to determine this.

query:*Administrator* OR
query:*userAccountControl*1.2.840.113556.1.4.803**


LDAP Table View **70** hits

```
>_ query:*Administrator* OR query:*userAccountControl1.2.840.113556.1.4.803*
```

I've now only got 70 hits total and all of these I've linked back to my attacker's activity. If I've discovered an interesting account or query not common in my environment, I could consider creating a custom model and enabled it. The Vectra AI platform already has very good detection coverage for suspicious LDAP queries, but you might want to create some very specific models around unexpected queries for your environment.

For further information:

http://www.ldapexplorer.com/en/manual/109050000-famous-filters.htm

https://ldapwiki.com/

Lateral Movement

Suspicious Remote Desktop Protocol

Metastream: metadata_rdp*

ATT&CK: T1021.001

Description: Adversaries may use Valid Accounts to log into a computer using the Remote Desktop Protocol (RDP). The adversary may then perform actions as the logged-on user.

Remote desktop is a common feature in operating systems. It allows a user to log into an interactive session with a system desktop graphical user interface on a remote system. Microsoft refers to its implementation of the Remote Desktop Protocol (RDP) as Remote Desktop Services (RDS).

Adversaries may connect to a remote system over RDP/RDS to expand access if the service is enabled and allows access to accounts with known credentials. Adversaries will likely use Credential Access techniques to acquire credentials to use with RDP. Adversaries may also use RDP in conjunction with the Accessibility Features technique for Persistence.

Example: Vectra Recall tracks and parses all RDP sessions that are seen on the network. In Vectra Recall, open the saved search “RDP Table view”, you will see a list of recent RDP sessions with the most common fields of interest will be listed. You should look to exclude sessions which are expected for your organisation to find any suspicious instances, for example you might want to exclude jump servers or system admins who typically leverage RDP for their day-to-day operations.

A good place to start is to look at the keyboard_layout in use in RDP connections. Commonly, this information is encrypted. If you search by:

NOT keyboard_layout: “encrypted RDP keyboard”

You will see all unencrypted keyboard layouts, look through these, and see if any keyboard layouts don’t align with where your normal users are based. If you are an American organisation, and spot a French keyboard layout, that could be suspicious.

You could also look for abnormal desktop screen sizes, which can be symptomatic of malicious actors masquerading as normal users. But first you need to remove encrypted communications, where the screen size will be shown as 0 with this search:

NOT desktop_height: 0 AND NOT desktop_width: 0

You will see the request dimensions for any remaining values. Values which do not represent standard screen sizes (1280 x 720 etc.) are suspicious and may warrant further investigation.

Finally, you could look to see if any RDP requests were made from external locations. External sources should not be able to make requests against internal instances. To find these requests, search for:

NOT local_orig: “true”

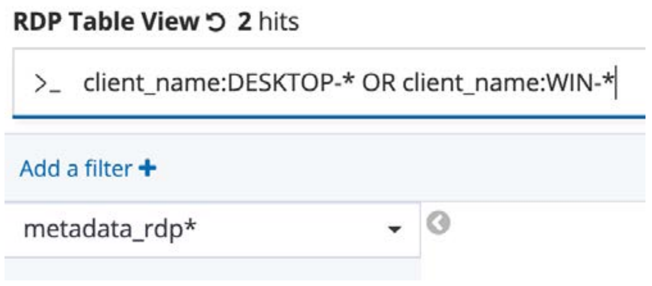
For each item listed here, check and see if the IP is from a known, safe, subnet, check what machine is being RDP’d to, and check the cookie that was used to log in, these will indicate if a connection warrants further investigation.

In a recent investigation RDP data came to be a very handy source of evidence, RDP sessions inherited whatever the source systems details are (hostname, keyboard layout, screensize etc). Leveraging this we can look for unusual hostnames leveraging RDP on our network, would you expect a guest device to RDP? If your organisation uses clear precise naming convention for hosts we can filter out the expected hosts and look for unusual hosts.

For example leveraging the following query we’ll find any clients with default hostnames initiating RDP sessions, can you account for these clients?

client_name:DESKTOP-* OR client_name:WIN-*

We've found this to be a very good indicator in several customer environments while hunting for suspicious sessions.



```
>_ client_name:DESKTOP-* OR client_name:WIN-*
```

Command and Control

Suspicious Beaconing Activity

Metastream:

ATT&CK: T1008

Description: Adversaries may use fallback or alternate communication channels if the primary channel is compromised or inaccessible in order to maintain reliable command and control and to avoid data transfer thresholds.

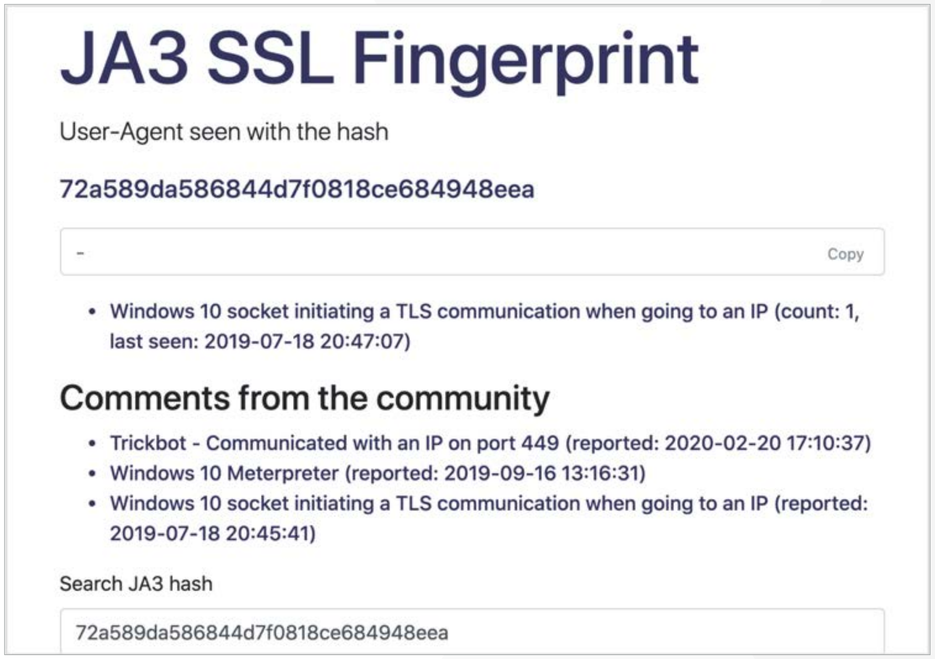
Example: Beaconing, the periodic outreach of an internal host to an external server for updates, may be used as the foundation for identifying tunnelling attack traffic. Vectra Detect currently leverages a machine learning algorithm to detect beacons as part of its Hidden HTTP/S Tunnel algorithms. However, beaconing is not an attacker behavior alone. It is also observed in traffic related to applications that require frequent updates, like stock tickers, sports score trackers, and advertising networks. Knowing what hosts on the network are exhibiting beaconing and which external destinations they are beaconing to can be valuable for the threat hunter or security analyst responding to an incident.

Therefore, while any strongly suspicious behaviour will cause a detection to fire in Vectra Detect, any beacons identified by the internal beacon detector algorithm regardless of whether it warrants a detection is published to Vectra Recall in the metadata_beacon* metastream.

Each single instance of beacon metadata will report a summary of multiple closed sessions that were identified as being part of the beaconing behavior. Since the duration of beacon behavior is variable, periodic updates are reported containing the cumulative beacon behavior and it is tracked over multiple beacon metadata instances with a beacon_uid. Additional information about a single beacon metadata instance can be gained by looking across the metadata streams with the same uid associated with the first session in the reported beacon behavior.

You should look to periodically track beaconing activity in your network, and to see what connections are happening, and to check commonly occurring items.

Open the Beacon Table View saved search to see recent beacons. The vast majority should be connecting with known, benign, domains, and a quick scan of the resp_domains field should let you validate this. If you spot any connections to unknown domains, you should check the JA3 hash that was spotted in this connection, and search for it against a Ja3 repo like ja3er.com, (See more on Ja3 hashes above).



JA3 SSL Fingerprint

User-Agent seen with the hash

72a589da586844d7f0818ce684948eea

- Windows 10 socket initiating a TLS communication when going to an IP (count: 1, last seen: 2019-07-18 20:47:07)

Comments from the community

- Trickbot - Communicated with an IP on port 449 (reported: 2020-02-20 17:10:37)
- Windows 10 Meterpreter (reported: 2019-09-16 13:16:31)
- Windows 10 socket initiating a TLS communication when going to an IP (reported: 2019-07-18 20:45:41)

Search JA3 hash

72a589da586844d7f0818ce684948eea

Conclusion

The techniques we've listed in this document are meant to be a representative sample of methods you can use to hunt for threats in your organization. These act as an extra layer of security beyond Vectra AI's advanced behavior-based detections, which utilize your expertise of the network to gain insights from the hugely valuable network metadata which Vectra AI monitors in your organization.

However, none of these methods are intended to be weekly recurring tasks. The intention should always be to begin a threat hunt with an open mind to what you might discover, to learn more about your network, find a bad actor if you're lucky (or unlucky, depending on your outlook), and, if possible, create an automated output to fire detections on any activity you'd aim to learn from it.

Improve your Posture with Automation of Hunt Results

Custom Models are designed to create this automated output. In each example above, we reference that once you have investigated an item in depth, removed any legitimate uses and false positives, then you should create a saved search and enable it as a custom model. Once this custom model is created, detections will be fired that seamlessly integrate into your existing workflows, and any future false positives can be triaged directly from within your Vectra Brain using our existing triage system, or by editing the underlying search itself.

If you get into the habit of repeated threat hunting with concrete custom model outputs, you will be able to reduce the risk to your company of a breach every single week, and with automated outputs from your previous hunts, you can move onto new and interesting techniques every time you perform a hunt.

For more information please contact us at info@vectra.ai.

Email info@vectra.ai vectra.ai