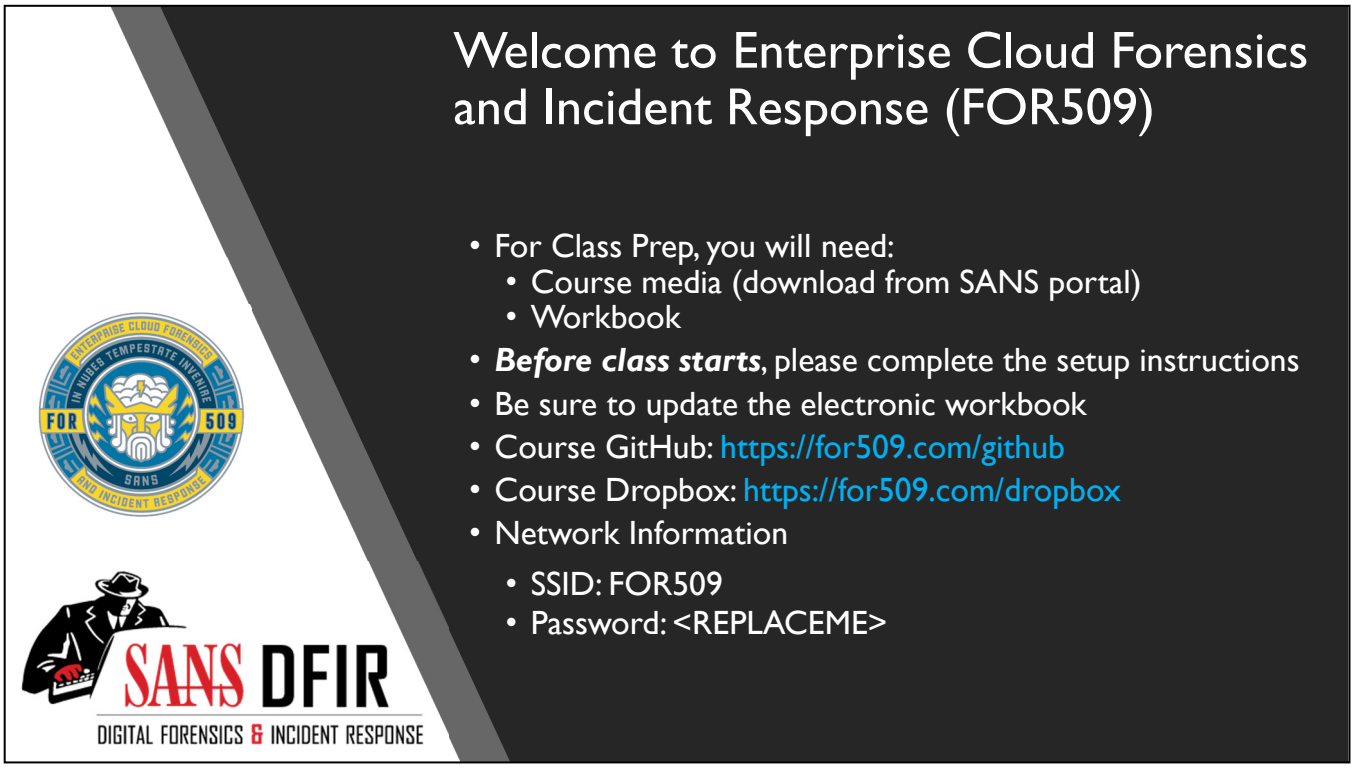
509.1

Microsoft 365 and Graph API

SANS | GIAC CERTIFICATIONS

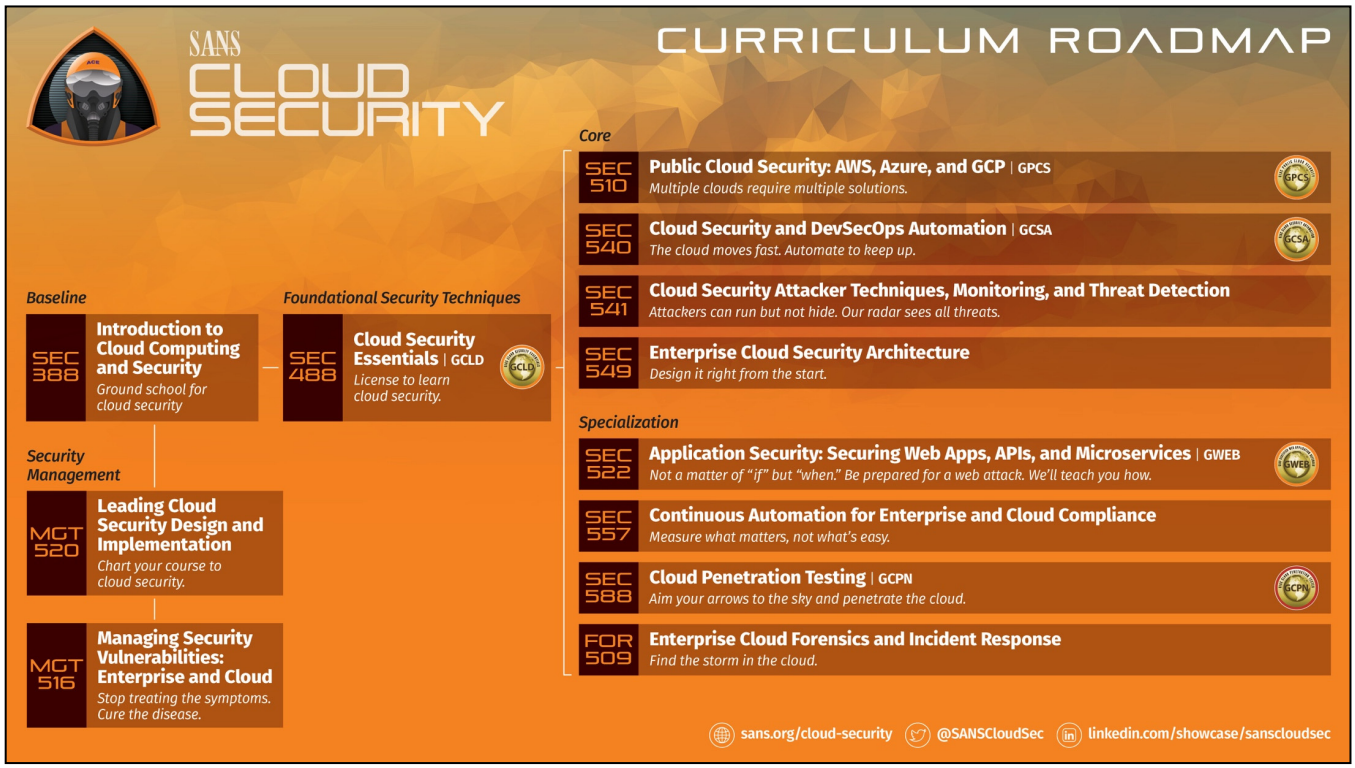
THE MOST TRUSTED SOURCE FOR INFORMATION SECURITY TRAINING, CERTIFICATION, AND RESEARCH | sans.org

All reference links are operational in the browser-based delivery of the electronic workbook.



This page intentionally left blank.

This page intentionally left blank.



The SANS Institute, established in 1989 as a cooperative research and education organization, is the most trusted and by far the largest source for information security training and security certification in the world. It also develops, maintains, and makes available at no cost the largest collection of research documents about various aspects of information security, and it operates the internet's early warning system—the Internet Storm Center. Its programs now reach more than 165,000 security professionals around the world.

SANS offers a number of courses that teach developers, architects, testers, security professionals, and managers how to build more secure applications. Anyone involved in developing, securing, and defending applications can benefit from the following courses in the SANS Cloud Security Curriculum:

SEC388: Introduction to Cloud Computing and Security | 3 Days
Advise and speak about a wide range of cloud security topics and help your organization successfully navigate both the security challenges as well as the opportunities presented by cloud services.

SEC488: Cloud Security Essentials | GCLD | 6 Days
Advise and speak about a wide range of cloud security topics and help your organization successfully navigate both the security challenges as well as the opportunities presented by cloud services.

SEC510: Public Cloud Security: AWS, Azure, and GCP | GPCS | 5 Days + Extended Lab Hours
Perform multicloud security assessments across AWS, Azure, and GCP clouds identifying key weaknesses and hardened configurations in core cloud services.

SEC522: Application Security: Securing Web Apps, APIs, and Microservices | GWEB | 6 Days
For anyone who wants to get up to speed on web application security issues and the best ways to prevent common web application vulnerabilities.

SEC540: Cloud Security & DevSecOps Automation | GCSA | 5 Days + Extended Lab Hours
Provides development, operations, and security professionals with a methodology to build and deliver secure infrastructure and software using cloud services and DevSecOps workflows.

SEC541: Cloud Security, Attacker Techniques, Monitoring, and Threat Detection | 5 Days
Leverage cloud security tools and services to monitor your environment and look for adversaries.

SEC549: Enterprise Cloud Security Architecture | 2 days
Design cloud-first architecture and scale cloud security best practices across the enterprise.

SEC557: Continuous Automation for Enterprise and Cloud Compliance | 5 Days
Teaching professionals tasked with ensuring security and compliance how to stop being a roadblock and work at the speed of the modern enterprise.

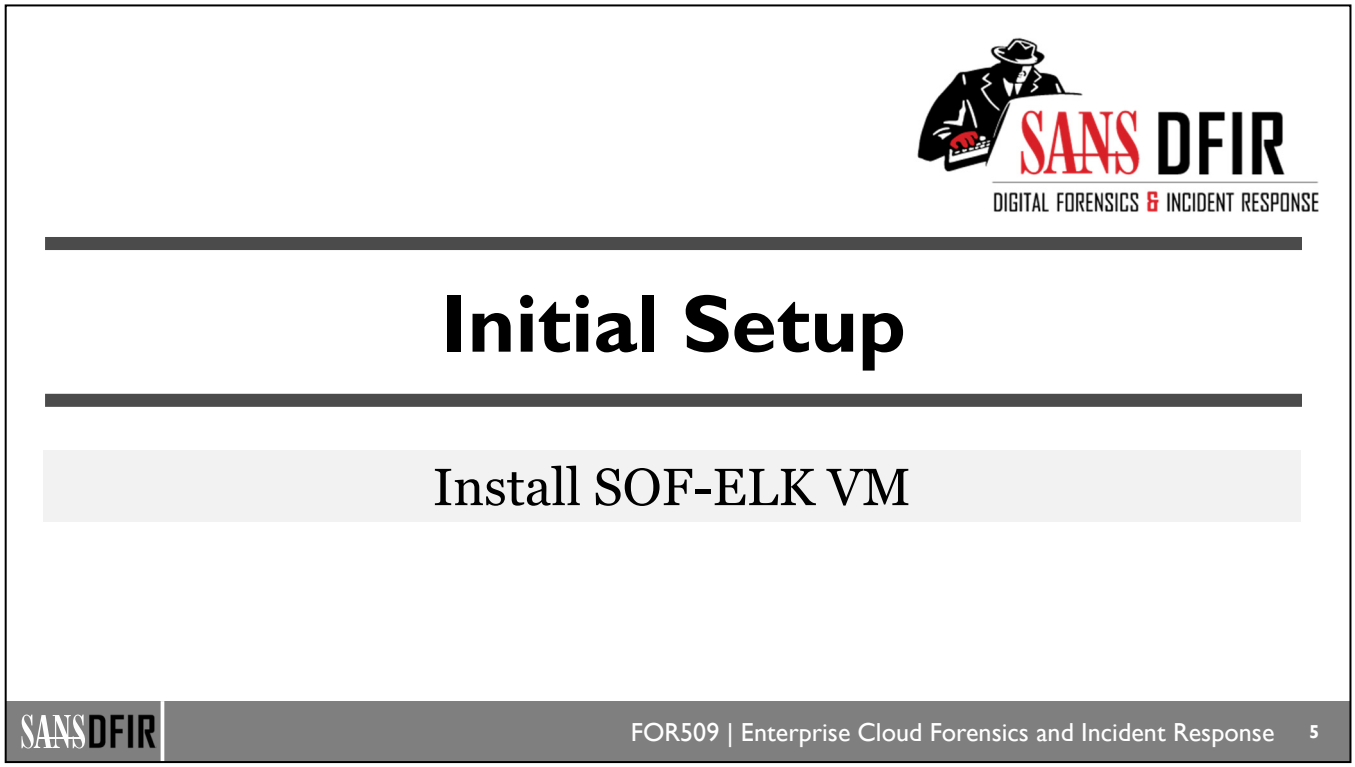
SEC588: Cloud Penetration Testing | GCPN | 6 Days
Prepares penetration testers to assess infrastructure and applications hosted in the public using platforms such as AWS, Azure, and Kubernetes.

FOR509: Enterprise Cloud Forensics and Incident Response | 4 Days
Designed to address today's need to bring examiners up to speed with the rapidly changing world of enterprise cloud environments.

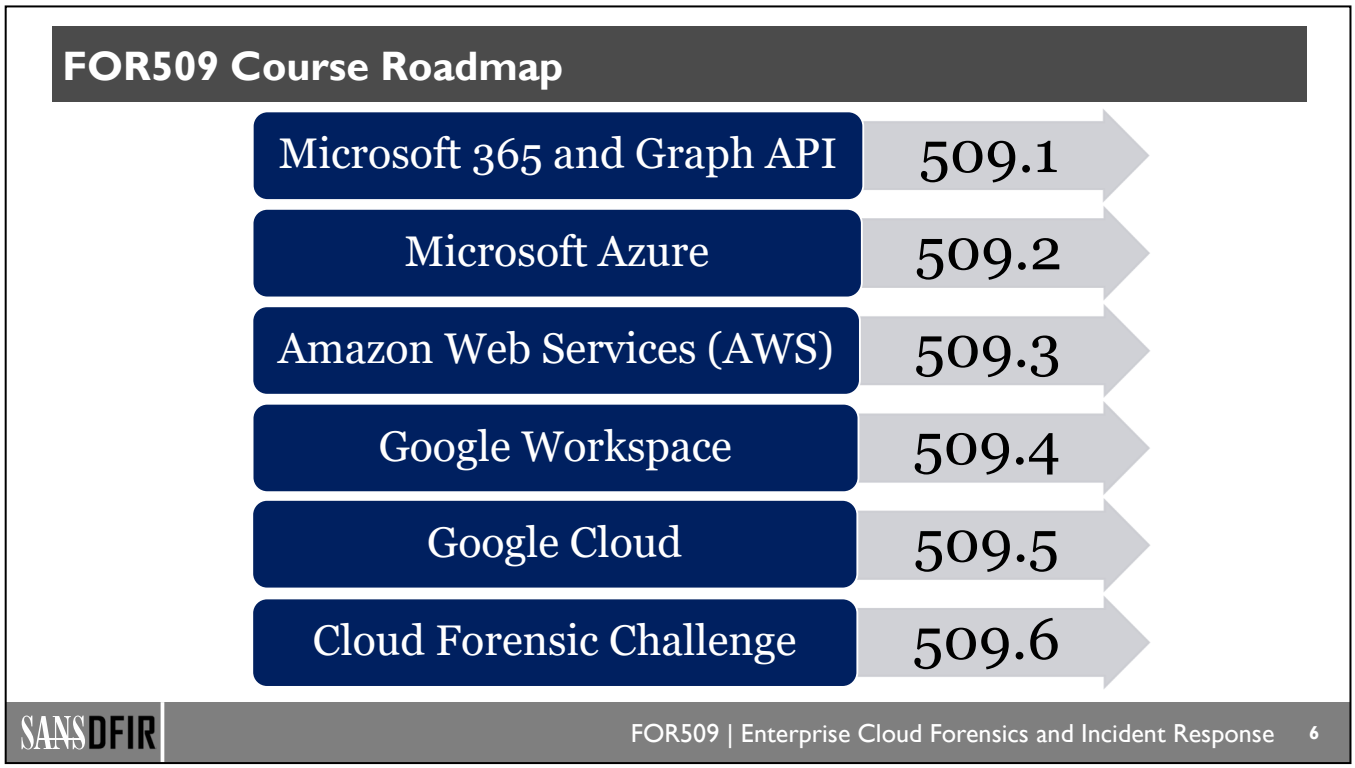
MGT516: Managing Security Vulnerabilities: Enterprise and Cloud | 5 Days
Highlights why organizations struggle with enterprise and cloud vulnerability management and shows how to solve these challenges.

MGT520: Leading Cloud Security Design & Implementation | 3 Days
Learn to build your cloud security program and roadmap.

Review our Job Role Flight Plan **sans.org/cloud-security**



This page intentionally left blank.



This course is made up of six parts:

1. In 509.1, we discuss Microsoft 365 as it's the prevalent office productivity suite in the corporate world. We then discuss the very powerful Microsoft Graph API and show how it can be used to access all the Microsoft cloud resources.
2. In 509.2, we discuss Azure, log sources, NSG flow logs, VM logs, and in-cloud IR.
3. In 509.3, we discuss AWS, log sources, CloudTrail, event-driven response, and in-cloud IR.
4. In 509.4, we discuss Google Workspace.
5. In 509.5, we discuss Google Cloud, log sources, data collection agent, and network forensics.
6. We wrap up the course with 509.6, which is our capstone challenge where we work a multi-cloud intrusion.

FOR509.1 | Enterprise Cloud Forensics and Incident Response

Microsoft 365 and Graph API

This page intentionally left blank.

FOR509.1: Microsoft 365 and Graph API

Section 1.1: Introducing SOF-ELK®

Section 1.2: Key Elements of Cloud for DFIR

Section 1.3: Microsoft 365 Unified Audit Log

Section 1.4: Microsoft Graph API

This page intentionally left blank.

Purpose of This Course

FOR509 focuses on cloud forensics and incident response for the enterprise so that you:

- Gain a basic understanding of key cloud resources and logs to facilitate incident response and forensics
- Become familiar with logs for virtual machines, networking, and storage, as well as the clouds themselves (management/control plane logs)
- Review the different methods available to access cloud logs (AWS, Azure, and Google Cloud)

Cloud security, architecture, and endpoint telemetry may be mentioned when addressing logs; however, other SANS courses are better suited if you would like to go in depth about these topics.

AWS, Azure, and Google Cloud are each an immense ecosystem made up of hundreds of services. However, when it comes to most investigations, there are a number of common elements.

In this course, we will examine the logs available for virtual machines, networks, and storage. In addition, we will review higher-level logs generated by the cloud themselves. These logs are referred by different names such as management logs, control plane logs, or hypervisor logs.
All combined, these logs will help us understand who is creating and accessing the various cloud resources.

Identity and Access Management (IAM) is key to accessing any cloud resource. Each cloud offers its own version of IAM. You will find that many large enterprises operate in a hybrid environment: on-premises systems plus a mixture of various clouds. On-premises (frequently abbreviated on-prem) refers to computing systems located within the physical confines of an enterprise.

One popular model for these companies is to leverage Azure Active Directory as their identity management system and control authorization with each cloud's own IAM system. The benefit is that employees of these companies only need to remember one set of login credentials no matter which cloud they access. From a digital forensics and incident response (DFIR) point of view, this will represent some interesting challenges, as you may be investigating an incident in Google Cloud, but the authentication took place in Azure Active Directory.

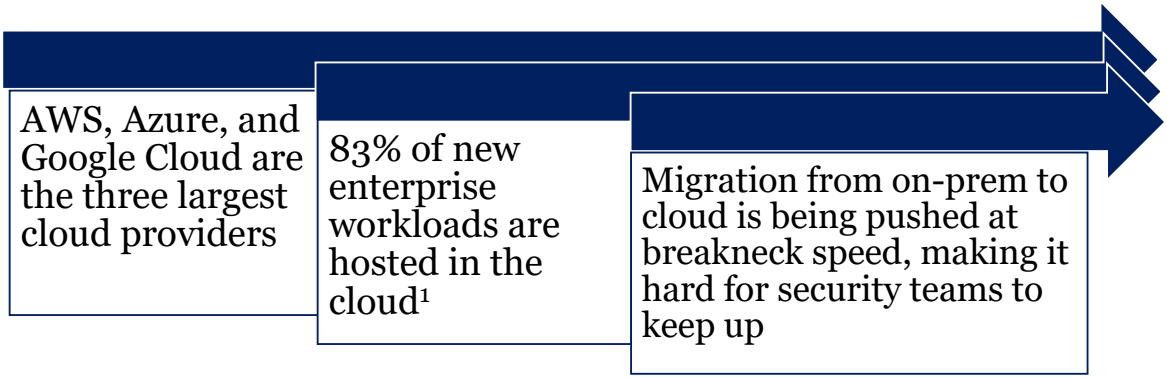
To optimize the learning experience, logs have already been downloaded from the cloud onto the SOF-ELK VM. The scripts we used for acquiring these logs will be provided in our GitHub repository.

To get the most out of the labs, focus on analyzing the data and imagine how you would work a similar incident in your environment.

By the end of the course, you will have a strong understanding of the logging capabilities of the main three cloud providers. This will provide you a solid foundation to conduct incident response and forensics in your environment.

While we will touch on endpoints, their logs are not our primary focus. Similarly, we will use SOF-ELK as our SIEM; however, architecting a SIEM solution is best left to other SANS courses.

Why We're Covering What We're Covering



- Attackers are ambivalent to on-prem systems versus cloud-based systems. They are all fair game to them.
- Cloud is the next frontier for attackers to monetize their crimeware by deploying crypto miners, phishing campaigns, and ransomware.

The elements we are covering (virtual machines, network, and storage) are the ones the authors have found to be present in nearly all investigations. The course will discuss many scenarios based on real attacks, including:

- A successful password spray attack
- Creation of new virtual machines for the purpose of crypto mining
- Exfiltration of data

While you may have conducted similar investigations within your on-prem environment, you will find that the cloud has a few unique challenges. One of the biggest challenges is that many logs are turned off by default, making investigations extremely difficult.

By knowing which logs are key to your investigations, you will be able to go back to your environment and make sure that all the necessary resources are configured and available to you.

1. LogicMonitor, "Cloud Vision 2020: The Future of the Cloud," www.scribd.com/document/403188911/LogicMonitor-Cloud-2020-The-Future-of-the-Cloud-pdf, page 3 [subscription required]

About the Labs

Incident response and forensics are primarily about following breadcrumbs left behind by attackers. The labs will focus on the logs needed to solidify your knowledge of the investigation process.

Logs have been downloaded and pre-staged in the VM to create predictable results and mitigate issues such as:

- Expiring logs
- Delays in log posting
- Frequent vendor changes
- Internet connectivity limitations

Scripts to retrieve logs from the cloud will be provided and where possible optional labs will give you the opportunity to retrieve data directly from the cloud.

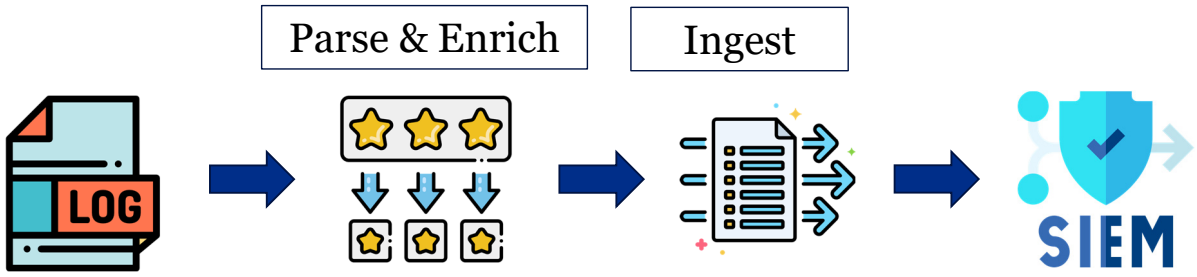
The purpose of labs is to demonstrate and solidify the class material. This requires labs with predictable results. To achieve that goal, the logs have been downloaded and pre-staged in the class VM.

Retrieving logs directly from the cloud would not achieve a predictable lab as the logs are constantly changing. We would also encounter substantial delays since logs aren't posted in real time by the cloud providers.

That being said, an optional lab in Google Cloud will be available to you so you can experience downloading log data directly from the cloud. The process and commands required to download logs will also be documented in the class. A number of scripts are also available in the class GitHub.

You will find that the mechanics of downloading data from the cloud is less important than the thought process we use during our investigations and the interpretation of these logs.

SIEM Intro



Wide variety of log types makes processing logs challenging. Some companies have created their own data models, for example:

- Elastic Common Schema
- Google Unified Data Model
- Splunk Common Information Model

SIEM = Security Information and Event Management

Each and every log has a different schema. A schema refers to the fields that are part of each log entry. Some logs have a fixed number of fields that you find in each entry; however, some may have a varying number of fields based on various conditions. As you would expect, log entries with a variable number of fields can be difficult to process.

In the cloud, you will encounter many different kind of logs. As a matter of fact, each Azure resource has a different log with its own schema. That's hundreds of potential logs.

Ingesting these logs, each with their own unique schema, is a difficult challenge. When using a commercial solution, like Splunk, the mechanics of ingesting all these log types are the vendor's responsibility.

Vendors like Google and Splunk have created data models to bring uniformity to their SIEM platform. Google calls their data model the Unified Data Model[1] (part of their Chronicle offering) and Splunk calls theirs the Common Information Model.[2] Elastic has also defined their version called the Elastic Common Schema.[3]

However, when building your own solution, it becomes your responsibility. The first step is to parse the log entry and decide which fields you wish to retain. Some of these fields may be enriched with additional information. For instance, an IP address can be enriched with geolocation information. This enrichment will create additional fields that were not part of the original log entry. Seldom used fields are usually discarded.

Once the log entry has been processed it can be ingested in the SIEM of your choice. For this class, we will use an open-source solution called ELK which we will discuss in detail in the next section.

1. https://for509.com/chronicle-udm
2. https://for509.com/splunk-cim
3. https://for509.com/elastic-schema

Microsoft 365 and Graph API Roadmap

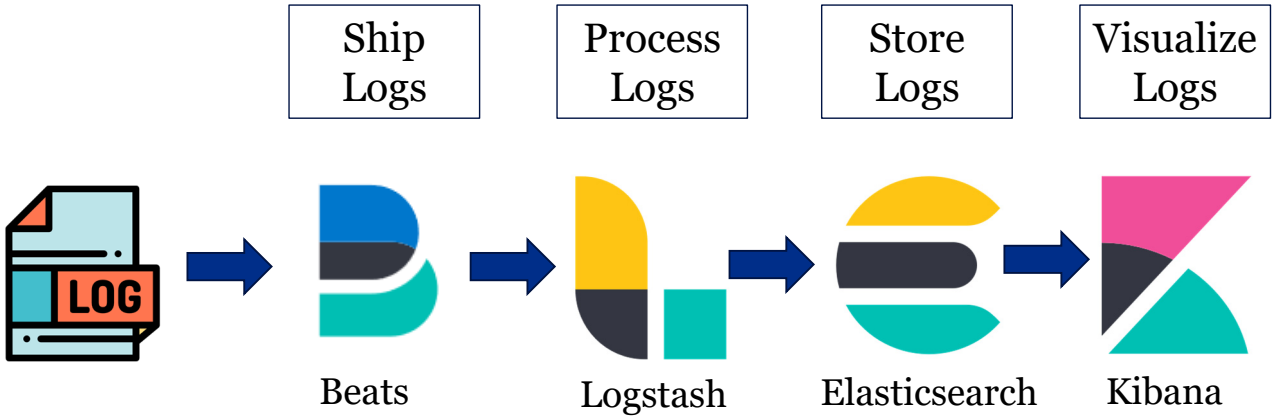
- 1.1: Introducing SOF-ELK®
- 1.2: Key Elements of Cloud for DFIR
- 1.3: Microsoft 365 Unified Audit Log
- 1.4: Microsoft Graph API

- Elastic Stack and SOF-ELK
- Logstash
- Search Process
- Discover
- Visualize
- Dashboard
- **Lab 1.1**: Visualize Data in SOF-ELK

This page intentionally left blank.

Elastic Stack

Free and scalable software with paid support model



The Elastic Stack (formerly known as ELK Stack and simply abbreviated ELK) is made up of four open-source projects: Elasticsearch, Logstash, Kibana, and Beats:

- **Beats** is a log shipper that sends data from hundreds or thousands of machines to Logstash.
- **Logstash** is a data collection and log-parsing engine. It reads input data and then transforms and enriches it. The enriched data is then transported to one or more destinations (such as Elasticsearch).
- **Elasticsearch** is a document-centric storage and analytic engine where the data is actually stored. It features fast and scalable search functionality.
- **Kibana** is the web-based frontend that allows users to explore data through dashboards and visualizations.

SOF-ELK

Security **O**perations and **F**orensics **ELK**

- Self-contained VM
  - Preconfigured with ELK
  - Preloaded with custom parsers for SANS classes
- Created and maintained for various SANS courses
- Provided free for DFIR and information security communities

Building onto the existing SOF-ELK platform, numerous additional Logstash parsers were integrated to support AWS, Azure, Google Cloud, and Microsoft 365 data. You'll be using a custom version of SOF-ELK that has been specifically built for FOR509, including class lab data and the Electronic Workbook. You must use the version distributed with your courseware for the labs, but the overall cloud functionality is also included in the public SOF-ELK release.[1]

If you would like to learn more about SOF-ELK, we highly recommend listening to Phil Hagen's talk on that subject.[2]

Additionally, SANS instructor John Hubbard gave a talk at the Philadelphia Security Shell meetup called "How to Use The Elastic Stack as a SIEM."[3]

1. https://for509.com/sof-elk
2. https://for509.com/sof-elk-talk
3. https://for509.com/elk-siem

Logstash

- Logstash is the key to SOF-ELK's ability to parse and ingest many log sources
- Logstash parsers written for FOR509: AWS, Azure, Google Cloud, GWS, Microsoft 365
- Copy log to appropriate directory, and logs are "magically" imported into SOF-ELK

Log	Directory
AWS	/logstash/aws
Azure	/logstash/azure
Google Cloud	/logstash/gcp
GWS	/logstash/gws
Microsoft 365	/logstash/office365
Flow Logs (any clouds)	/logstash/nfarch

Flow logs require an additional step (see notes)

To support FOR509, a number of Logstash parsers were written:

- **AWS**: CloudTrail logs (includes a preprocessing ingest script).
- **Azure**: Tenant, subscription and resource logs—exported from storage account in JSON format.
- **Google Cloud**: Google Logging exports or using Pub/Sub, which will be discussed in the Google Cloud section.
- **GWS**: Google Workspace logs.
- **Microsoft 365**: Unified audit log—either exported from the portal or PowerShell, must be CSV formatted.
- **Flow logs**: VPC flow logs from AWS or Google Cloud. NSG flow logs from Azure. Requires additional step, described below.

SOF-ELK also has Logstash parsers for many other log formats that we don't use in FOR509.

SOF-ELK runs a filebeat process (part of the Beats log shippers mentioned earlier) that's continuously looking for changes in the directories mentioned in the slide. As soon as a file is copied in one of these directories, filebeat will grab it and send it to the appropriate Logstash parser.

Flow logs require a bit of additional massaging. The raw flow logs exported from AWS or Azure are in JSON format and need to be converted to a format that SOF-ELK can read with its NetFlow ingest feature. Run the appropriate ingest script, as follows:

AWS: `$ aws-vpcflow2sof-elk.sh -r /path/to/aws/flow/log -w /logstash/nfarch/aws_flow_log.txt`

Azure: `$ azure-vpcflow2sof-elk.py -r /path/to/azure/flow/log -w /logstash/nfarch/azure_flow_log.txt`

Google Cloud: (Because logs are natively parsed, no ingest script is needed.)

JSON to ELK

Example JSON-to-ELK mapping from the AWS Logstash parser

- The ELK indices use a flat schema, while JSON events can be deeply nested
- The Logstash parsers select the most important fields for DFIR and map them as shown above

JSON objects can be deeply nested. Since ELK uses a flat schema, it's necessary to create a map in the Logstash parsers.

There can be a large number of fields in each JSON object. As DFIR investigators, we always want to keep the maximum amount of information. Unfortunately, this is not always practical. The more information we process through the Logstash scripts, the more resources are consumed. To keep our VM optimized, we only mapped the key fields we use in DFIR.

The Logstash parsers are quite simple, so feel free to modify them to suit your needs.

Logstash Mapping

Log entries contain many fields we may not need. The Logstash parser must select and map the most important fields. Partial example from the AWS parser:

```
rename => {
       "[raw][eventName]" => "event_name"
       "[raw][eventSource]" => "event_source"
       "[raw][awsRegion]" => "aws_region"
       "[raw][sourceIPAddress]" => "source_host"
       "[raw][requestID]" => "request_guid"
       "[raw][eventID]" => "event_guid"
       "[raw][eventType]" => "event_type"
```

Unneeded fields are dropped:

```
mutate {
     remove_field => [ "raw" ] }
```


If you want to know what the Logstash parsers do, look in the directory `/usr/local/sof-elk/configfiles`.

You will see different "levels" of files: input, preprocess, postprocess, output, and the main files. The key files for FOR509 are:

- 6701-office365.conf
- 6801-azure.conf
- 6901-aws.conf
- 6950-gcp.conf

As an example, if you look at 6901-aws.conf, you will see that the parser's main function is to map the fields from the raw AWS log to SOF-ELK fields:

```
rename => {
       "[raw][eventName]" => "event_name"
       "[raw][eventSource]" => "event_source"
       "[raw][awsRegion]" => "aws_region"
       "[raw][sourceIPAddress]" => "source_host"
       "[raw][requestID]" => "request_guid"
       "[raw][eventID]" => "event_guid"
       "[raw][eventType]" => "event_type"
       "[raw][additionalEventData][bytesTransferredIn]" => "bytes_in"
       "[raw][additionalEventData][bytesTransferredOut]" => "bytes_out"
```

```
      "[raw][userIdentity][accessKeyId]" => "access_key_id"
      "[raw][requestParameters][bucketName]" => "bucket_name"
      "[raw][requestParameters][Host]" => "hostname"
      "[raw][resources][0][ARN]" => "aws_resource_name"
      "[raw][resources][0][type]" => "aws_resource_type"
      "[raw][userAgent]" => "useragent"
    }
```

For maximum efficiency, this mapping is limited to the most important fields and everything else is ignored:

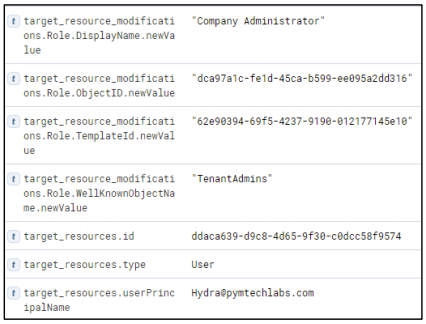
```
# remove remaining fields
   mutate {
     remove_field => [ "raw" ]
   }
```

This is an important nuance, as any other commercial SIEM will do some type of mapping but may not expose their decision process. You will want to check what fields your SIEM may have decided to drop for their schema.

Azure Logstash Split Entry


```
"time": "2022-02-13T22:25:07.6574495Z",
  "resourceId": "/tenants/7e325eda-7945-46d3-ac99-
f0dcfeb4628e/providers/Microsoft.aadiam",
  "operationName": "Add member to role",
     "modifiedProperties":

         "displayName": "Role.ObjectID",
         "oldValue": null,
         "newValue": "\"dca97a1c-fe1d-45ca-b599-
ee095a2dd316\""

          "displayName": "Role.DisplayName",
          "oldValue": null,
          "newValue": "\"Company Administrator\""

          "displayName":
"Role.WellKnownObjectName",
          "oldValue": null,
          "newValue": "\"TenantAdmins\""

          "displayName": "SPN",
          "oldValue": null,
          "newValue": "\"efd53fc0-2438-43f5-851a-
d88e46323305\""
```

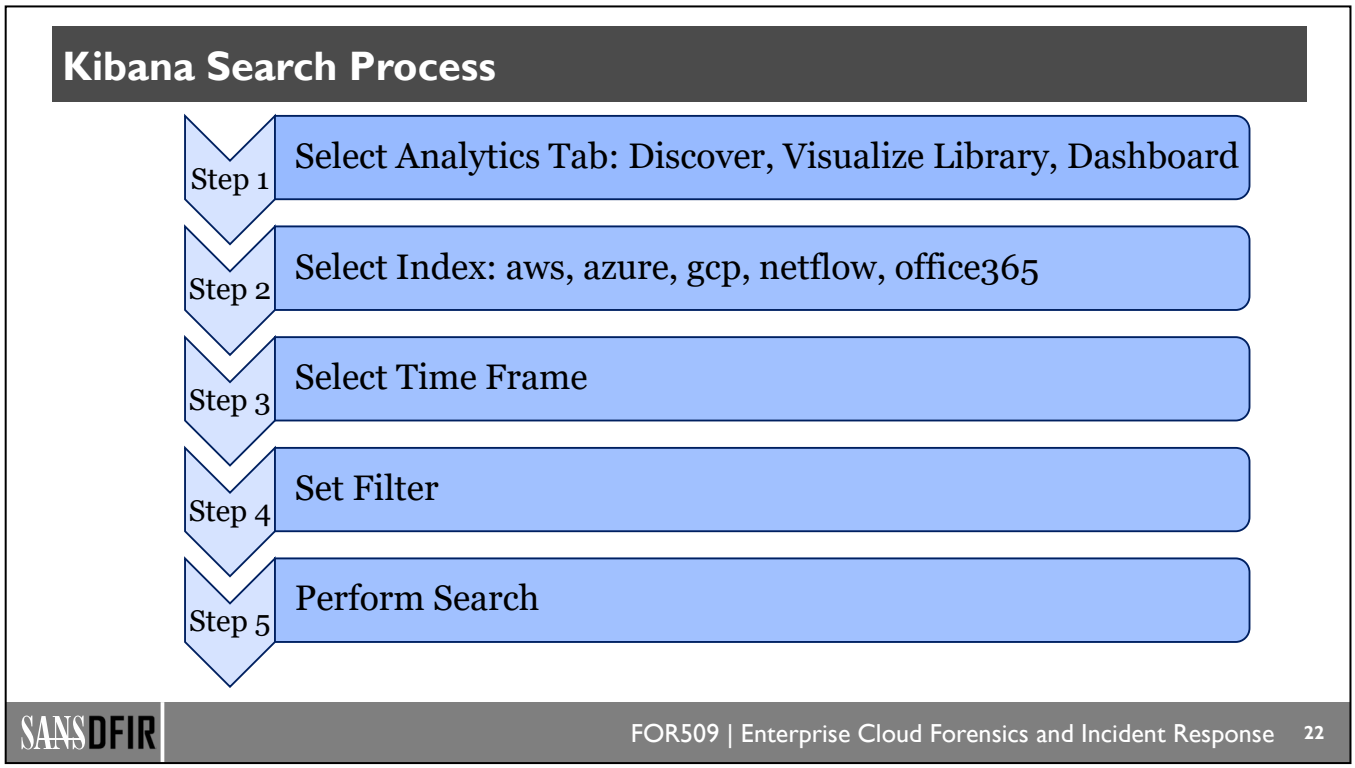

- Mapped to two entries in SOF-ELK
- Second entry shows select fields with changed information



Certain Azure log entries include repeating fields that indicate values being changed. The pattern consists of the field name, the old value, and the new value. This repeating pattern is difficult to handle in Logstash.

To avoid losing valuable information, the Azure Logstash parser was coded to create two entries in ELK when this condition is found. One of the entries will contain some of these changed attributes.

While far from a perfect solution, it provides enough information to lead the analyst to go explore the raw logs when appropriate.



We are now faced with a large amount of data in our SOF-ELK instance. Where do we start?

Step 1: Decide which SOF-ELK analytics tab is best for the query you would like to make.

- Discover: Filter and search raw events.
- Visualize Library: Create various types of charts (bar, percentage bar, area, pie, etc.) or tables.
- Dashboard: Collection of saved visualizations and/or searches in one location.

Step 2: Select your index.

- Each log must be imported into a specific index. Unlike certain commercial solutions, you can't search across all indices.
- Field names may vary between indices.
- If you don't get the results you expected, make sure you have the correct index and the correct field name.

Step 3: Select your time frame.

- This is a key step, as your results will only be valid for the time frame selected. If you don't get the expected results, double-check your time frame.

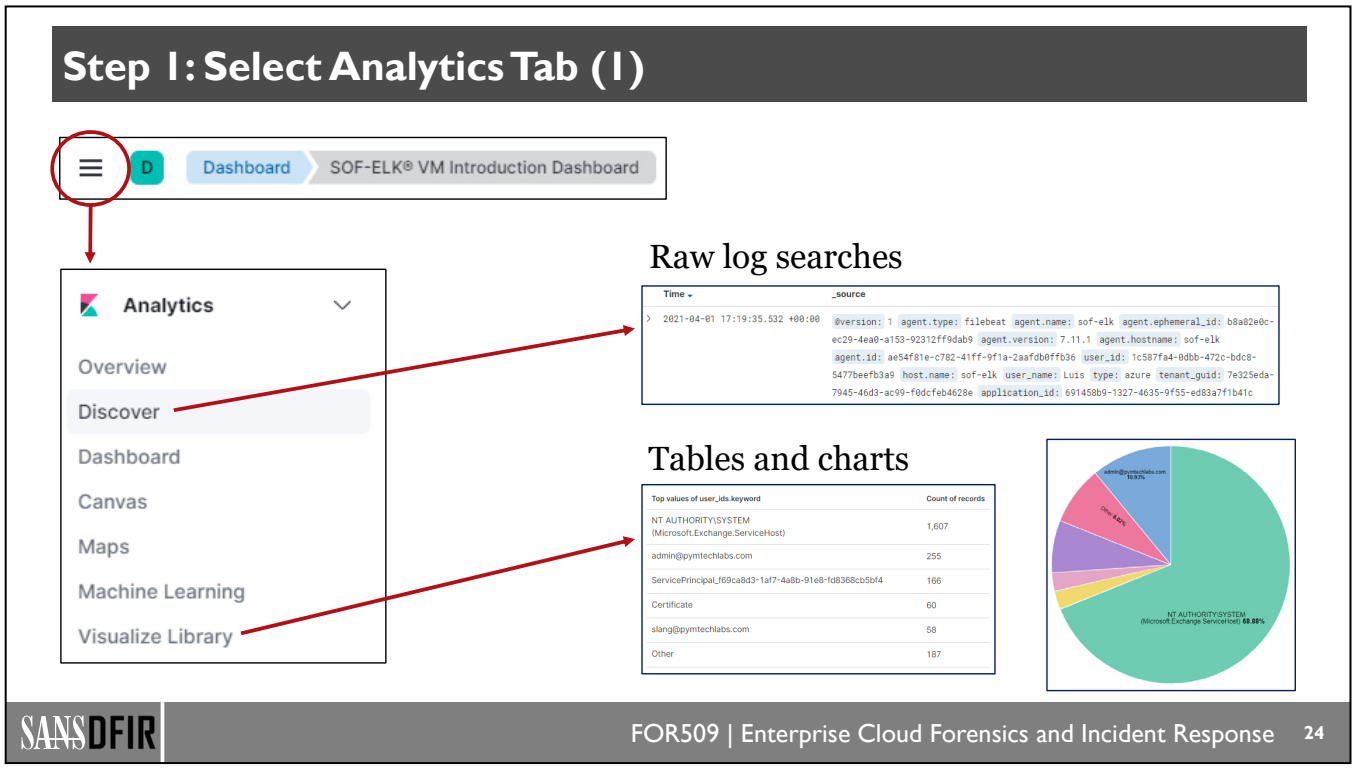
Step 4: Set a filter (optional).

- Data reduction can be done either with a filter or a search. If you know a specific piece of information, it's advisable to filter your data for that information and then perform your search in the next step. This will keep your search bar less cluttered. For example, you could set your filter for a specific userid and then use the search bar to find out what actions this person performed.

Step 5: Perform your search.

- You are now ready to search on any field that may be relevant to your investigation.
- You can save your searches and use them later or add them to panels in a dashboard.

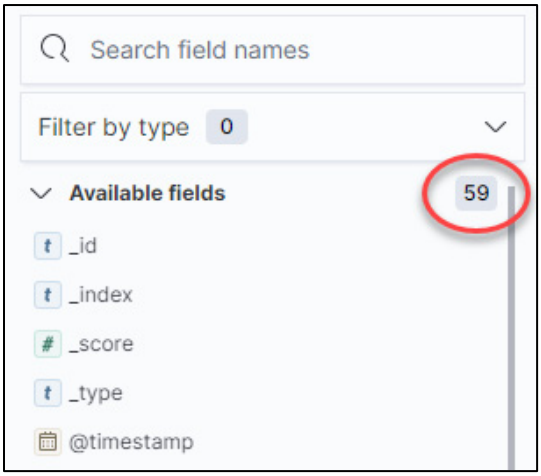
SOF-ELK has many other features, but the ones listed above are the ones we will use in FOR509.



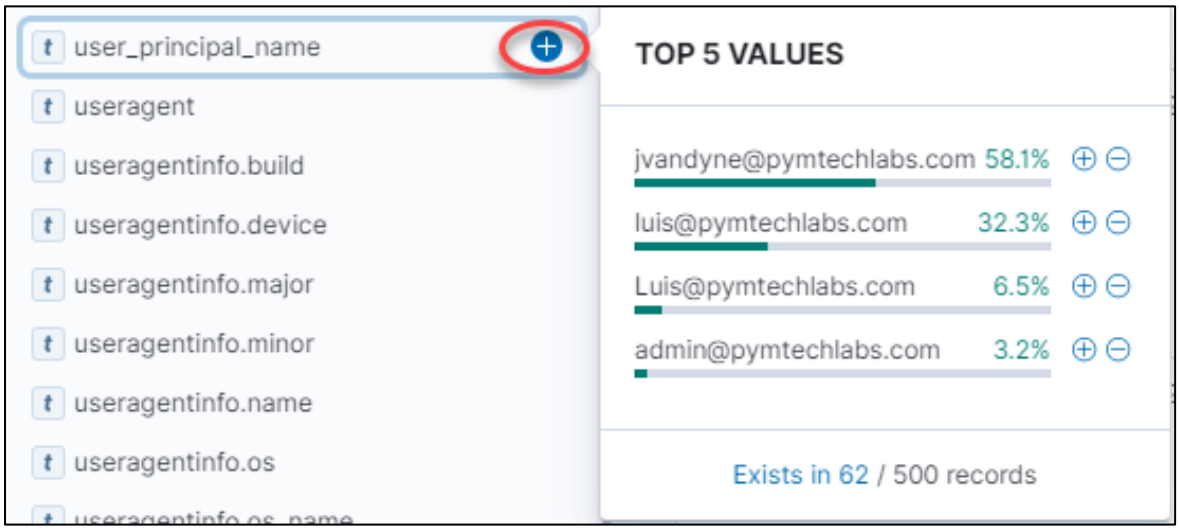
Under Analytics, you have a number of options. The ones we will use in FOR509 are:

- Discover
- Dashboard
- Visualize Library

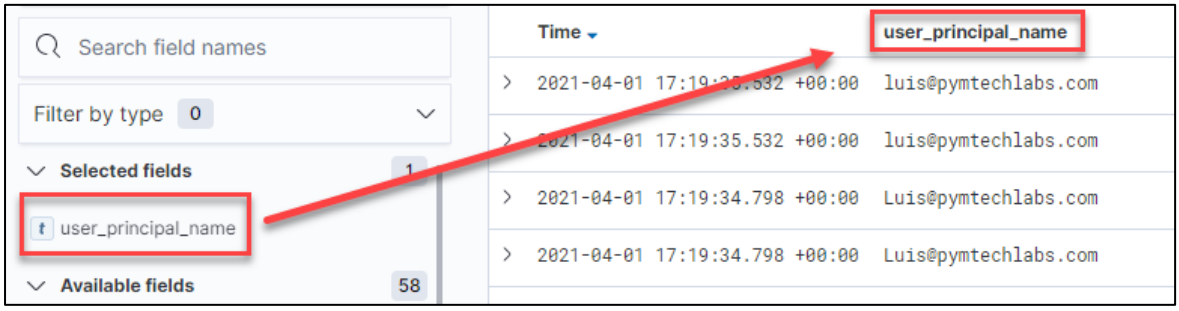
You should start with the Discover tab, as it gives you direct access to raw events. On the left side of the Discover tab, you will get a list of available fields. This is extremely useful if you are not familiar with the log source. In the example below, you can see that this log has 59 different fields. The number of fields will automatically reflect your choice of index and time frame. Only fields that contain data will be shown. You can also start typing the name of a field in "Search field names" to narrow down possible fields.



You can select any field and get a short summary of the top five values. Most importantly, you can click on the + sign and add this field to your list of selected fields.



Once the field is selected, your main window will no longer show the entire raw event. Rather, it will start building a table with the fields you have selected.

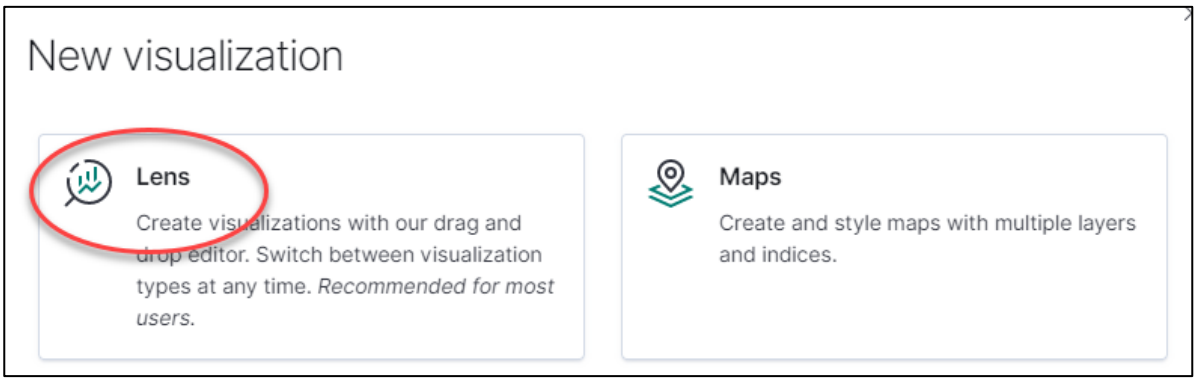


You can select as many fields as you wish, but anything above four or five will quickly exceed the real estate on your screen.

At any point in time, you can click on the chevron next to a raw event to expand it and see every field it contains:

The Visualize tab allows you to create charts and graphs. When you select that tab, you will get a screen showing all your saved visualizations. Select **Create visualization** to start a new one.

When you select **Create visualization**, you will be presented with a few options. The easiest one to use is **Lens**:

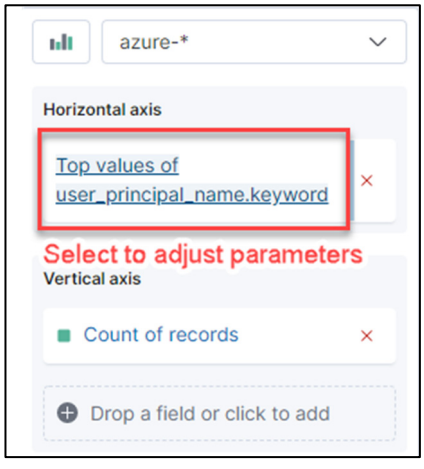


You can now select from a number of chart types.

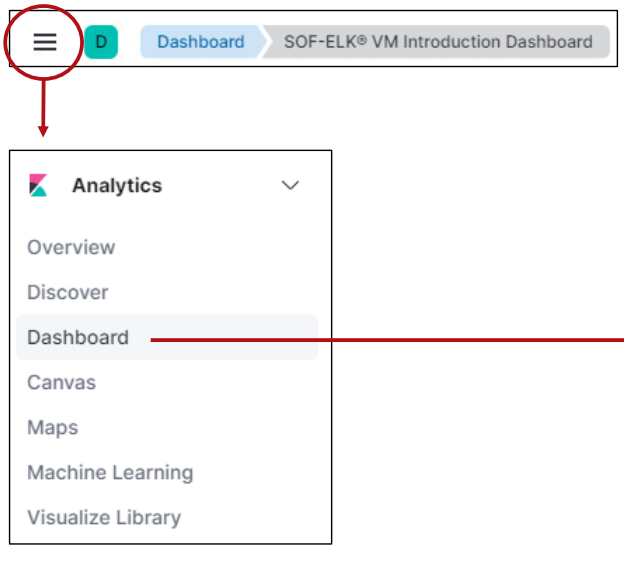
The best way to learn about all these chart types is to try them out.

Once you have selected a chart type, you can drag and drop any of the fields.

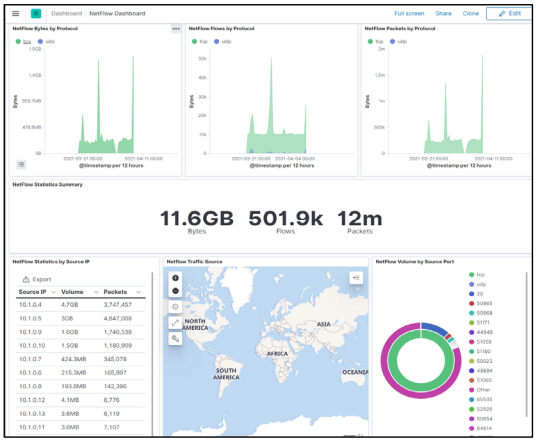
The menu below will let you adjust the parameters for the field you selected.



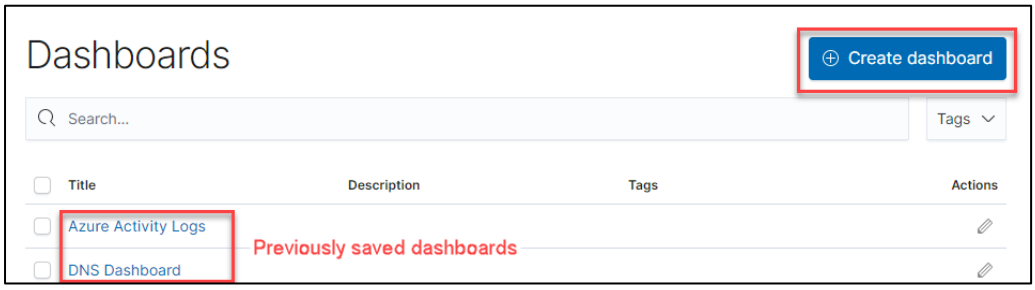
Step 1: Select Analytics Tab (2)



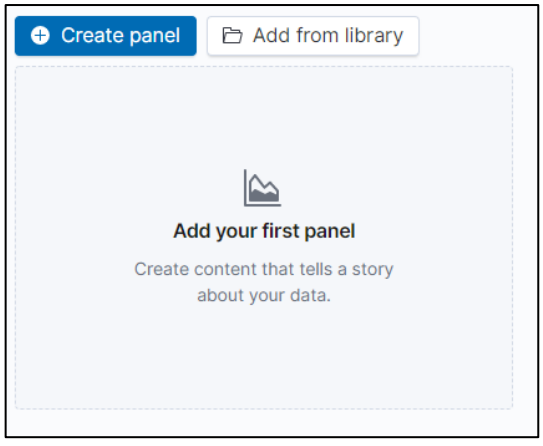
NetFlow dashboard: a collection of saved searches and visualizations



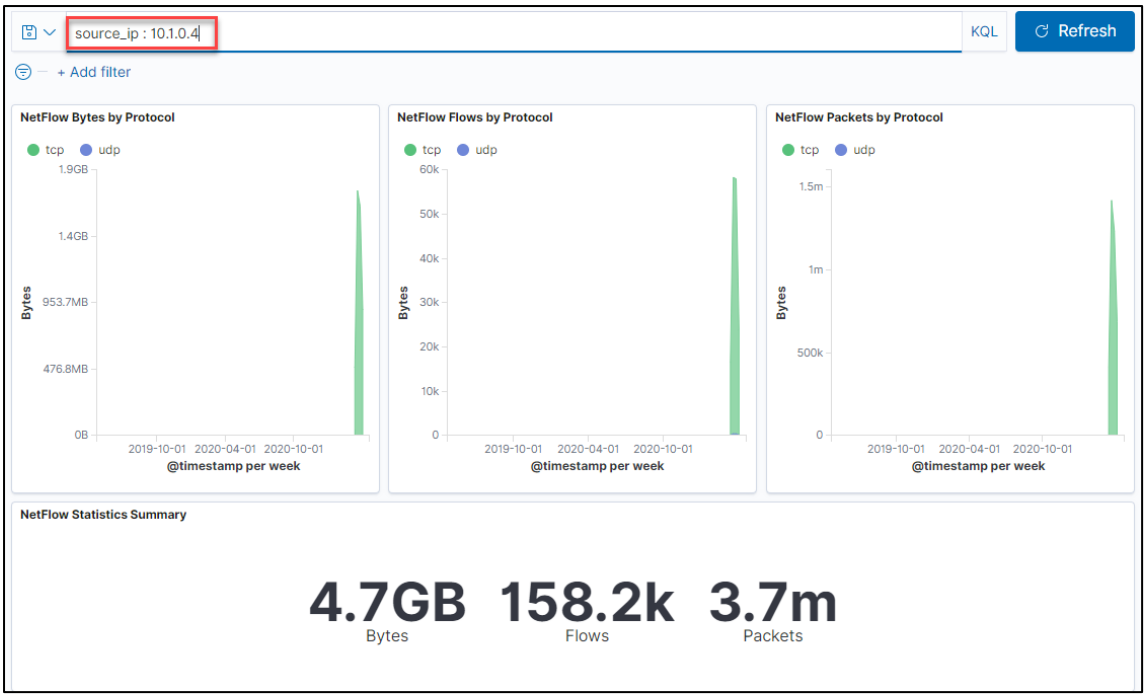
Now that we have created and saved various visualizations, we can combine them in a dashboard. Each visualization will be shown in a panel, and you can organize the panels any way you wish on the page. As an example, this slide shows the NetFlow dashboard.



When you create a dashboard, you are presented with a blank page and the options to either **Create panel** or **Add from library**. The library refers to previously saved visualizations or searches.

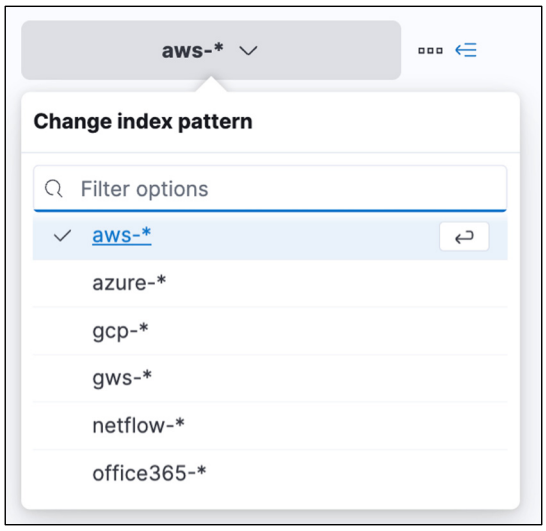


The Dashboard tab offers both the search and timeframe options. One very powerful aspect of these dashboards is that all panels will be dynamically updated based on any search or time parameters you enter.



There are many more features, and the best way to discover them is simply to create your own dashboard.

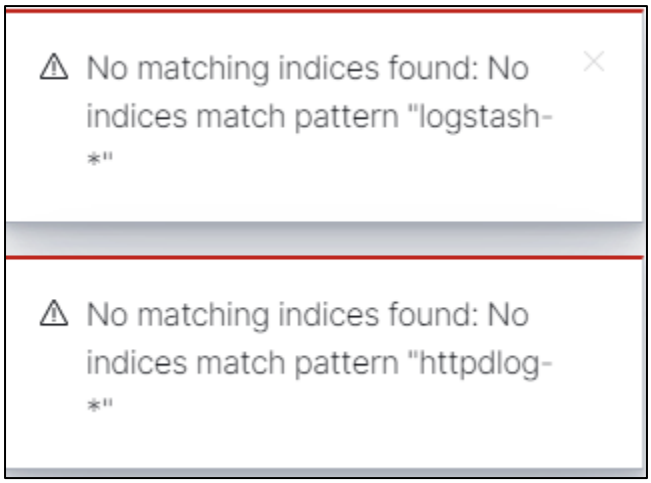
Step 2: Select Index



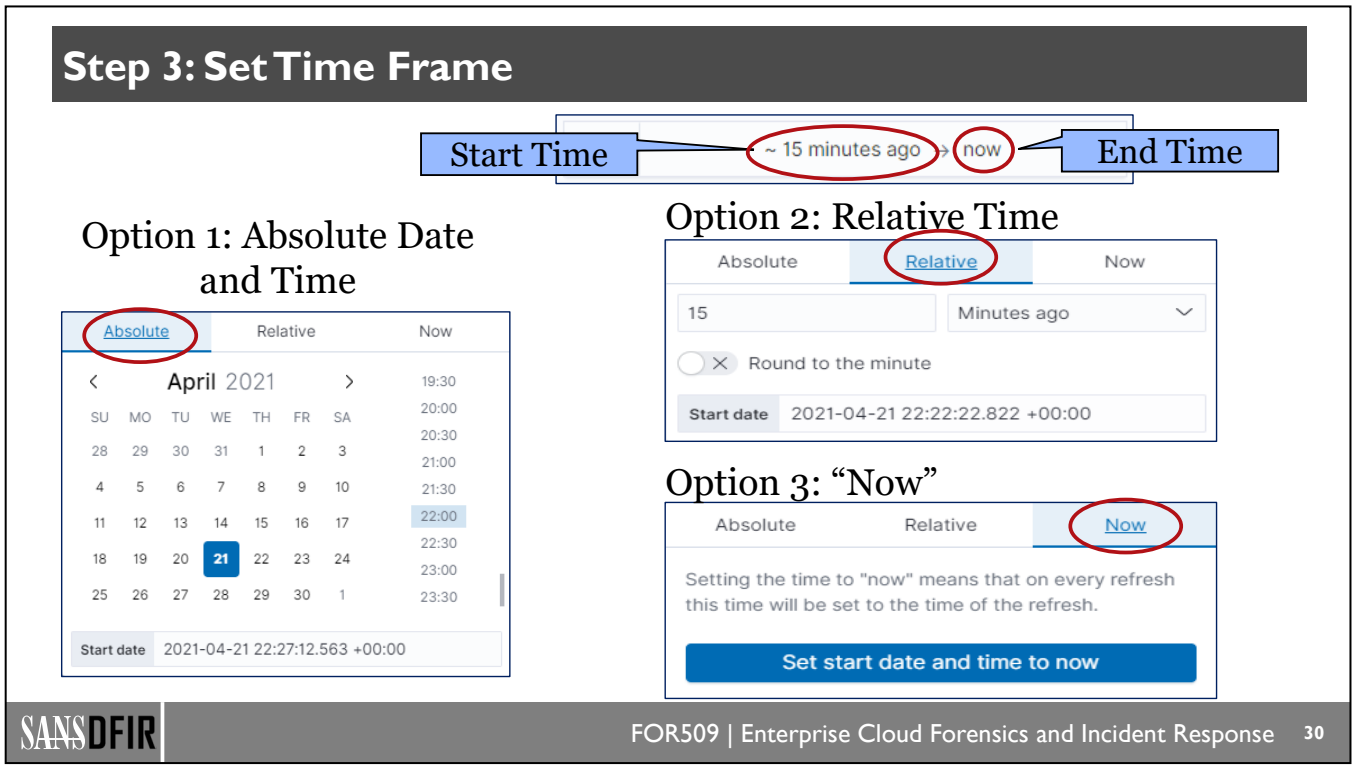
Index	Cloud
aws-*	Amazon Web Services
azure-*	Microsoft Azure
gcp-*	Google Cloud
gws-*	Google Workspace
netflow-*	Flow Logs (any clouds)
office365-*	Microsoft 365 UAL

When importing logs into SOF-ELK, we must specify an index. This slide shows the indices we will be using for FOR509.

As previously mentioned, the public release of SOF-ELK supports many more indices. Since we don't have data in these other indices, you will see these error messages:



These messages simply indicate that no data has been loaded under that specific index. All you need to do is close the pop-up windows.

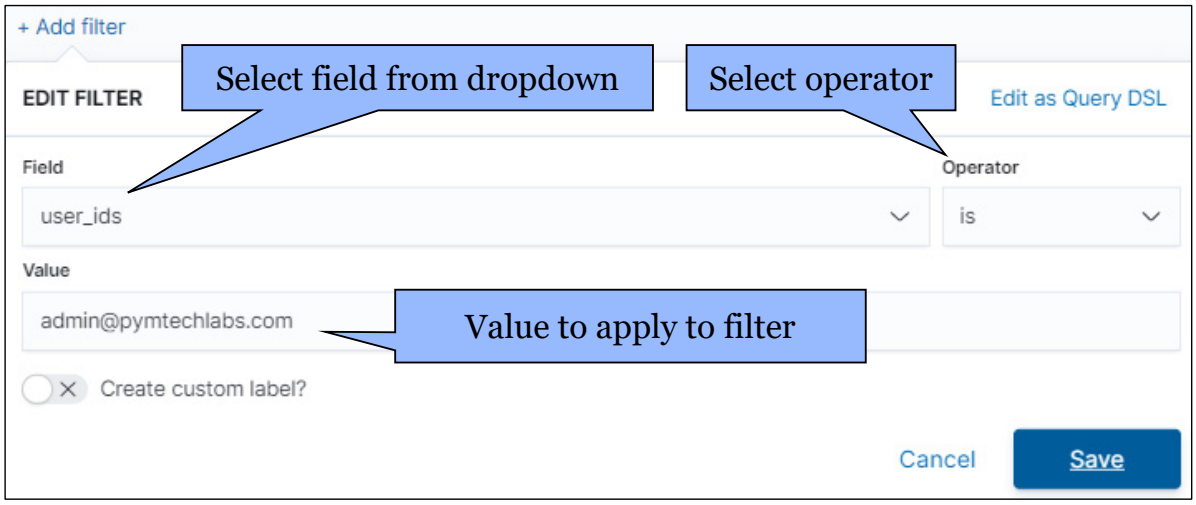


The time frame bar allows you to select a bracket of time to narrow down your data. There are three options to specify the date and time:

- Option 1: Specify an absolute date and time.
- Option 2: Use relative time.
- Option 3: Use "Now."

While these options are pretty self-explanatory, we would like to emphasize to always pay attention to the time frame bar, as it's easy to inadvertently change it and get incorrect results.

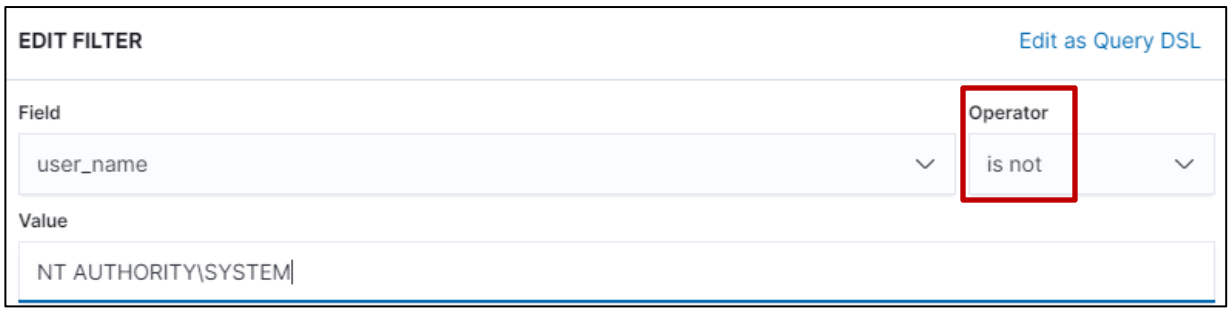
Step 4: Set Filter



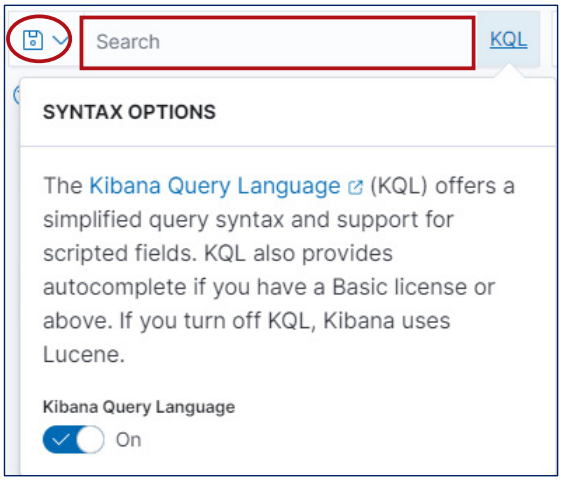
Use filters for general data reduction to save space on the search bar

The filter section is great for initial data reduction. It's very user friendly, as it presents all available fields in a dropdown menu. Possible operators are presented in the next menu. Finally, you only need to specify the value you wish to look for.

One great way to use filters is to exclude information by using the operator "is not":



Step 5: Perform Search



- Kibana uses the Kibana Query Language (KQL) by default
- KQL uses simple query syntax
- Alternatively, Kibana can use the Lucene language
- Searches can be saved (floppy disk icon on the left of the search bar)

The final step is to search for relevant information. Kibana supports two query languages: KQL (Kibana Query Language) and Lucene. KQL is the default and very easy to use. There are many good tutorials on the internet in addition to the official documentation.[1]

1. https://for509.com/kql

Search Examples

1	Search for a term across all fields	`pymtechlabs`
2	Search for a term within a specific field	`organization_name : pymtechlabs` `organization_name.keyword : pymtechlabs.com`
3	Boolean query	`user_ids : (luis or slang)`
4	Boolean query with multiple fields	`user_ids : luis AND operation : UserLoginFailed`
5	Exist query (very useful to eliminate records that don't have data in a specific field)	`useragent : *`
6	Negate a value	`not workload : Exchange`

This slide shows a few examples you will find useful for the FOR509 labs. As you can see, the syntax is quite simple.

If you want to search across all fields, you can simply enter your search term as shown in the first example.

The second example requires a bit more explanation. Elasticsearch will break up text strings on certain delimiters, including a dot (.), slash (/), dash (-), and whitespace, among many others. Elasticsearch refers to these "tokenized" fields as "analyzed." The benefit is that if you search "`organization_name : pymtechlabs`", you will match "`pymtechlabs`", "`pymtechlabs.com`", "`pymtechlabs.onmicrosoft.com`". However, Elasticsearch also leaves a version of that field intact and adds the term "`.keyword`" to the field to indicate the non-tokenized version. This version is useful if you want to do an exact search.

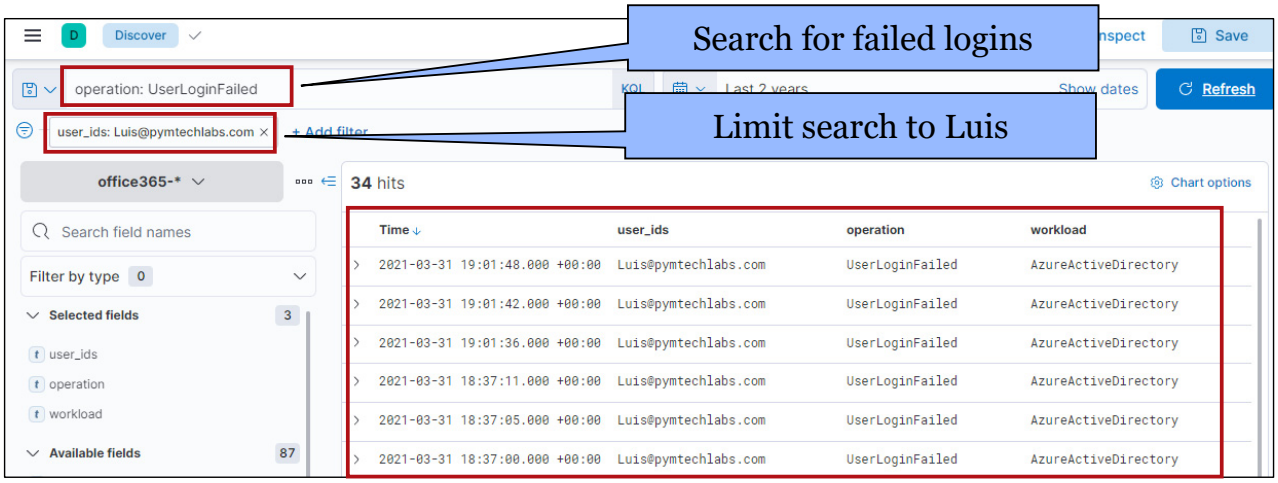
The third and fourth examples demonstrate various Boolean queries.

The fifth example is very useful and frequently used. Not all log entries contain data in every single field. You are frequently faced with empty fields, and this is the query you will use to limit your search to fields that contain data.

The sixth example comes into play when you have too much data. You may want to eliminate a certain type of data, and the NOT operator will do that for you.

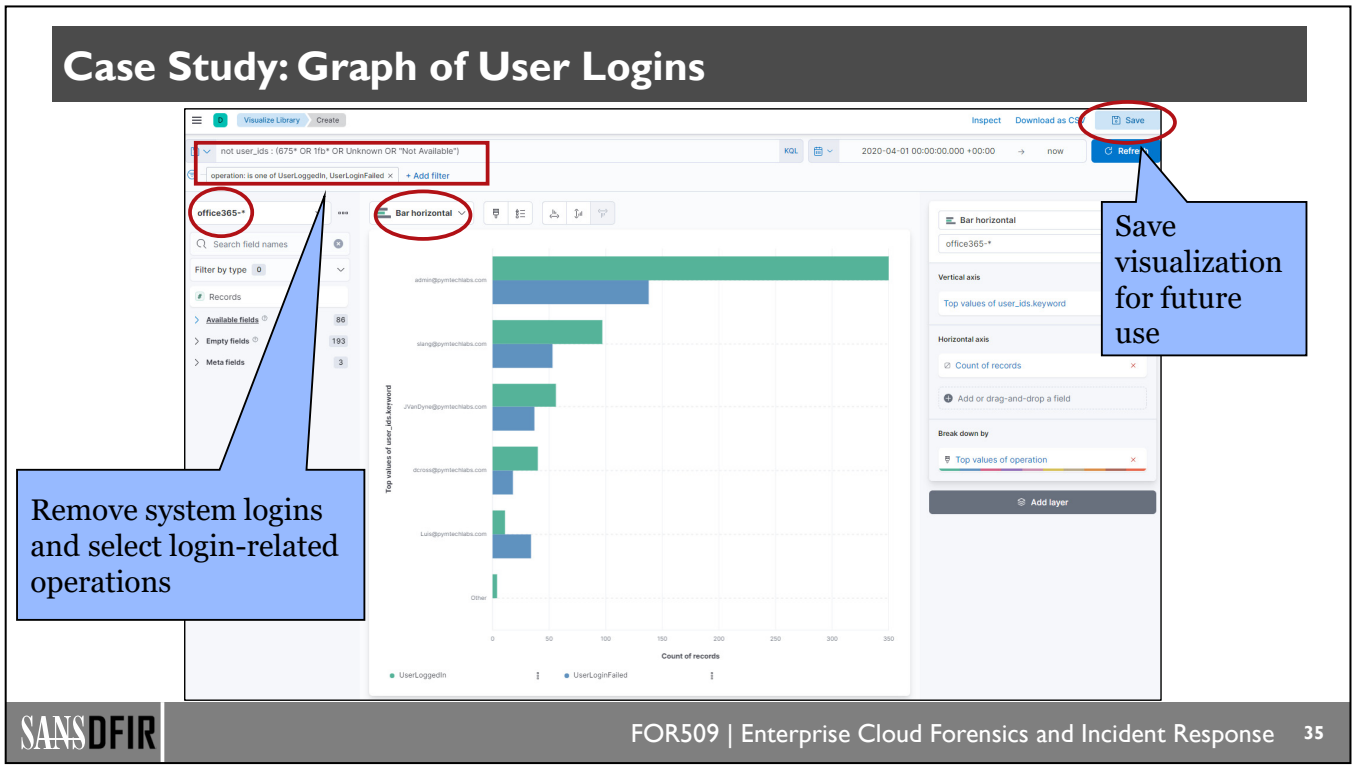
Note that you don't have to leave a space before and after the colon (:).

Case Study: Specific User Failed Login



As an example, we are searching for instances where Luis@pymtechlabs.com failed to successfully log in. As you build more and more complex queries, you have to go through a trial-and-error phase to find out which fields are meaningful to your query and which ones are not.

Be sure to verify your results and make sure you didn't accidently eliminate data that was relevant. It's a good idea to try multiple scenarios and test your assumptions.



We previously searched for failed logins for user Luis@pymtechlabs.com. Luis had 34 failed logins. Is this normal in our environment? Is this too many, indicating a possible attack? We can put this information into context by creating a chart of other users' successful and failed logins.

In this example, we use the following filter:
`operation : is one of UserLoggedIn, UserLoginFailed`

The operator "`is one of`" is very useful to provide a list of items. Since "operation" has numerous possible entries, we need to select the ones related to user logins.

Our initial results show a number of system accounts as well as entries called "Unknown" and "Not Available." For the purposes of this graph, we aren't interested in these entries, hence the filter:
`not user_ids : (675* OR 1fb* OR Unknown OR "Not Available")`

After double-checking the index and the time frame, you can select any chart type you would like.

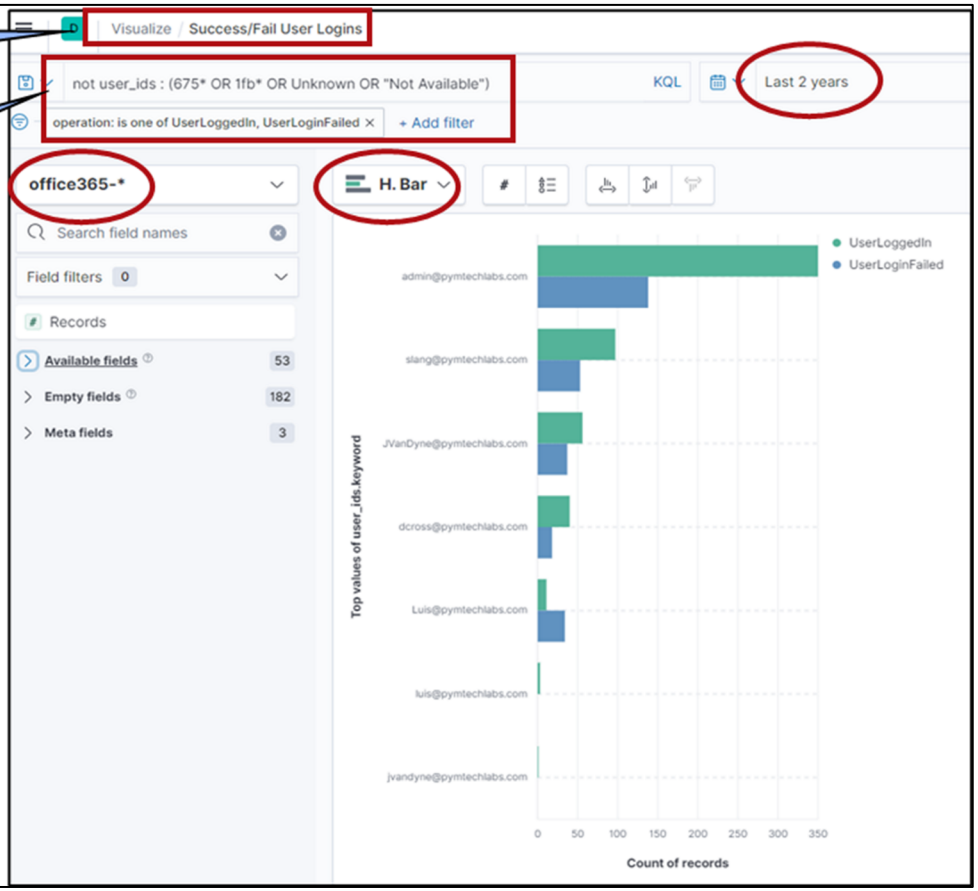
You will notice that we saved this visualization with the name "Success/Fail User Logins" so we can add it to our dashboard.

Save visualization for future use

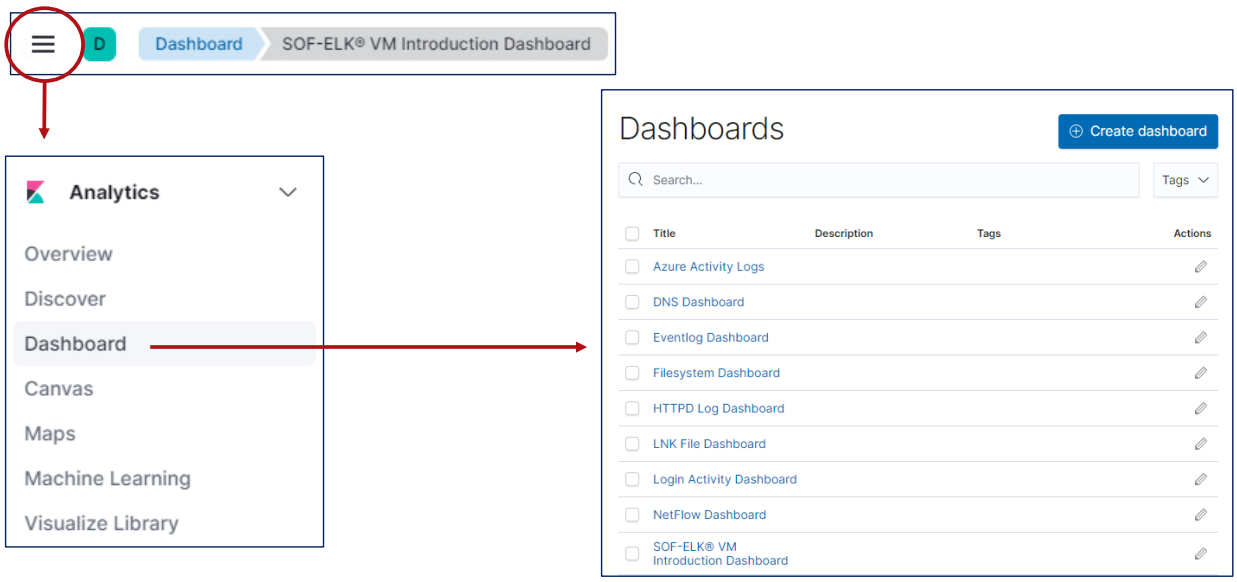
Remove system logins and select login-related operations

Other selections:

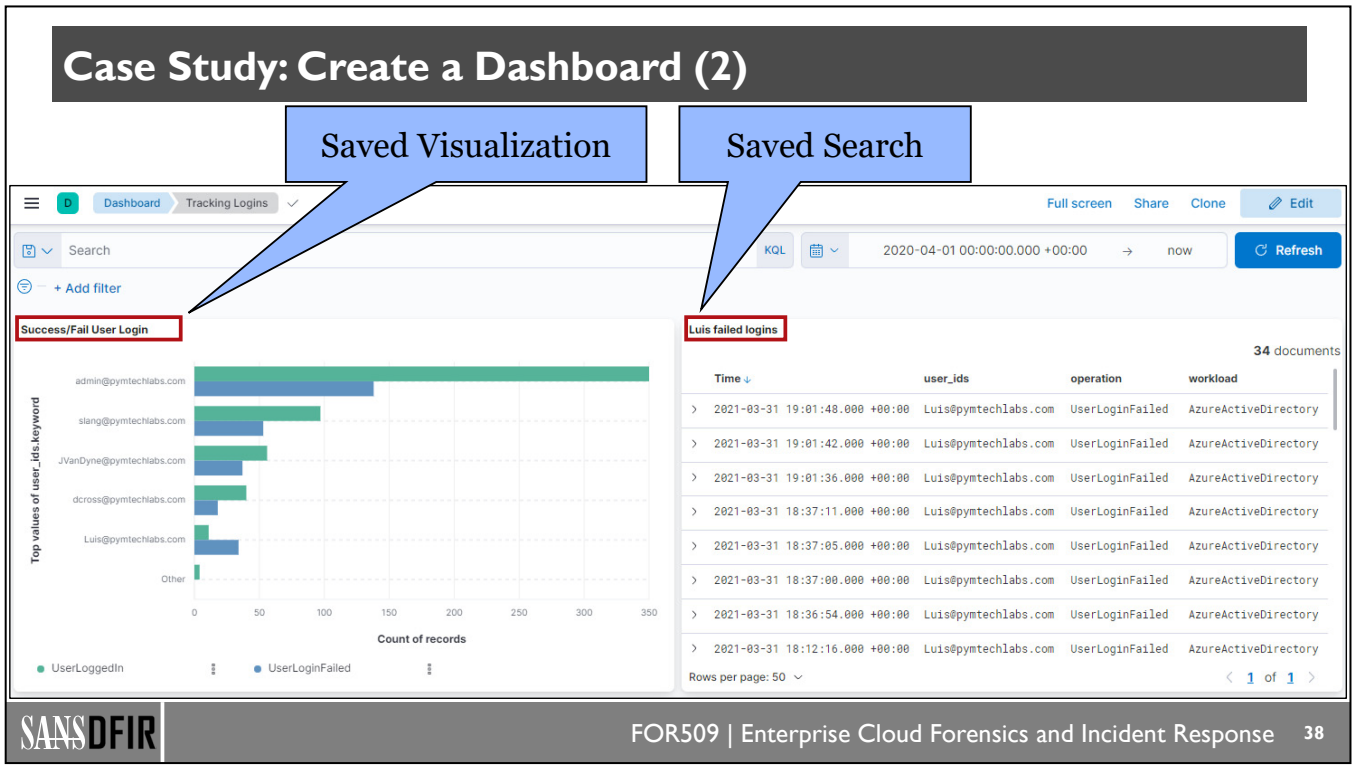
- Correct index
- Relevant timeframe
- Type of charts



Case Study: Create a Dashboard (1)



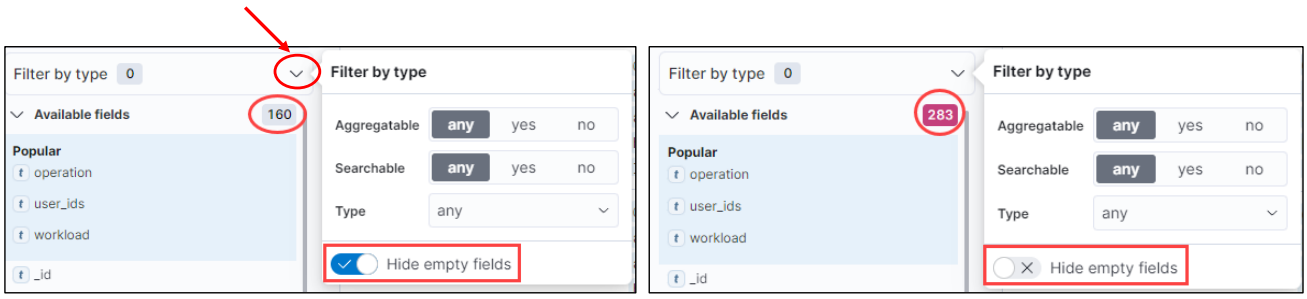
The "SOF-ELK VM Introduction Dashboard" is the default dashboard when you start SOF-ELK; however, there are many pre-defined dashboards available. You can select any of them and customize them to your liking. Alternatively, you can create your own.



We can combine the previously saved visualization and search into a single dashboard. You can create very elaborate dashboards with many panels. A great feature is that any additional search, filter, or time change you enter on the dashboard will automatically update every panel.

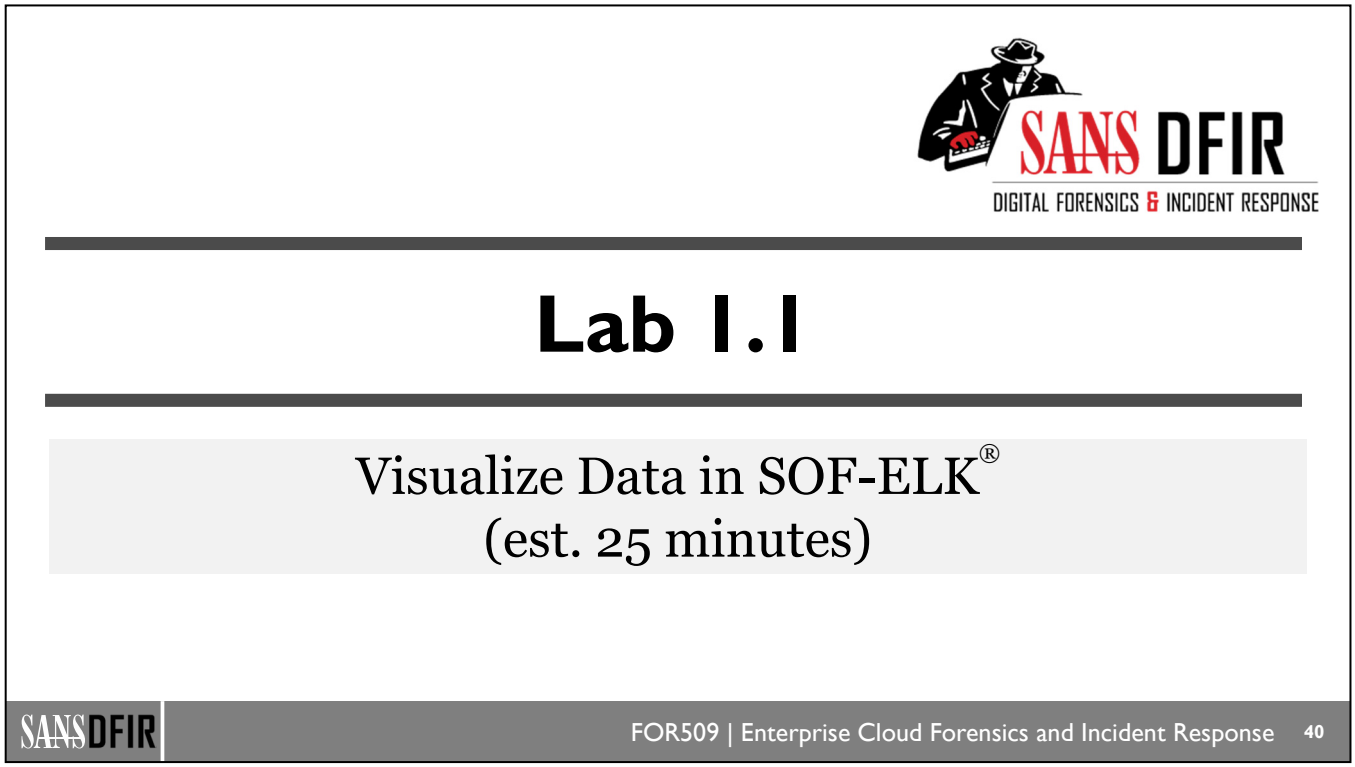
Show Empty Fields

- Kibana only pre-loads a fixed number of records
- By default, Kibana hides fields that are empty in the **current** view
- As a result, some fields may not be available for certain queries in the labs
- Recommendation to turn off "Hide empty fields"

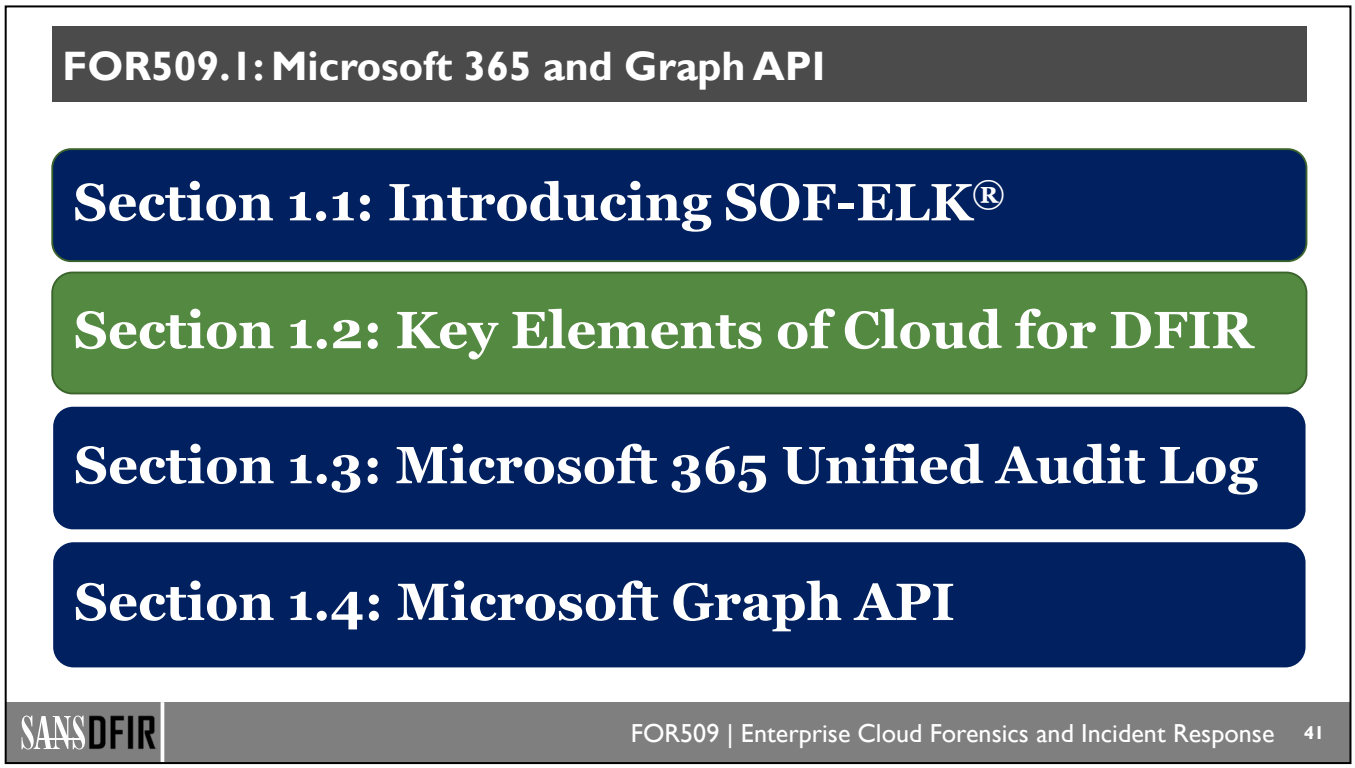


By default, Kibana only pre-loads 500 records (which we will change to 1000 in the first lab) and hides empty fields. However, this is not enough for later labs which load a large number of records. As a result, certain fields many not be available for you to use in your query until additional records are loaded. One solution is to turn off the "Hide empty fields" option.

The only way to really appreciate the power of Kibana is to try it for yourself, so let's go to Lab 1.1.



This page intentionally left blank.



This page intentionally left blank.

Microsoft 365 and Graph API Roadmap

1.1: Introducing SOF-ELK®

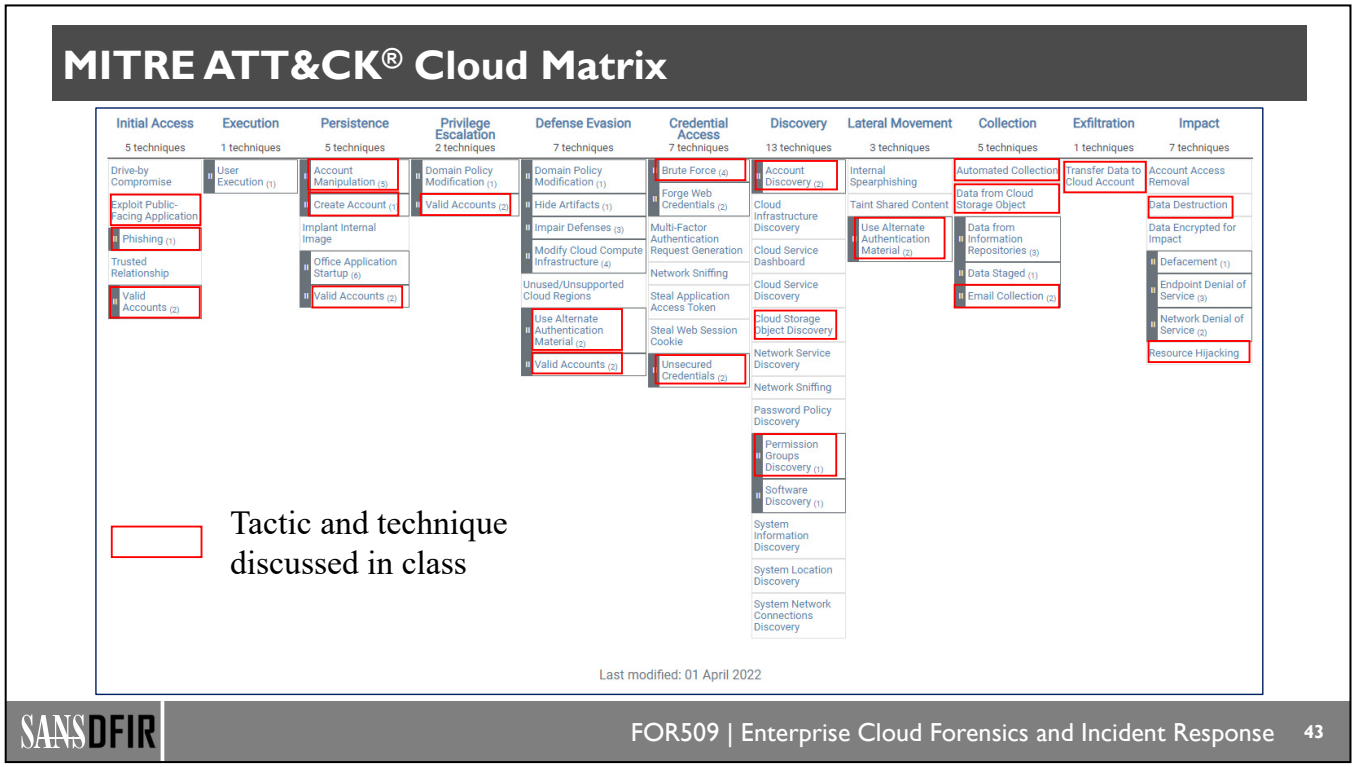
1.2: Key Elements of Cloud for DFIR

1.3: Microsoft 365 Unified Audit Log

1.4: Microsoft Graph API

- MITRE ATT&CK Matrix
- Common Attacks
- DFIR in the Cloud
- Common Cloud Concepts
  - Shared Responsibility Model
  - Log Hierarchy
  - Cloud Access
  - Snapshots
  - NetFlow
  - Pricing
  - Terminology Across Clouds

This page intentionally left blank.



The MITRE ATT&CK® framework[1] is widely adopted by enterprises to understand threat actors' techniques and assess the gaps in their currently deployed security tools. The cloud matrix contains common tactics and techniques aimed at AWS, Azure, and Google Cloud. The following categories are covered:

Initial Access: Techniques used to gain an initial foothold within a network. Examples include phishing, exploit of public-facing applications, and supply chain compromises.

Execution: Techniques used by adversaries to run code on compromised systems and establish long-term access. A current and very popular technique is to use LOLBins, which stands for Living Off the Land Binaries. These are pre-installed system tools that are used for malicious purposes.

Persistence: Techniques used by adversaries to keep access to systems through restarts or other interruptions. Examples include creating additional cloud credentials, creating their own virtual machines, and creating additional keys for existing applications.

Privilege Escalation: Techniques used by adversaries to gain higher-level permissions. The primary cloud technique used for privilege escalation is excessive permission policies on resources and access token manipulation.

Defense Evasion: Techniques used by adversaries to avoid detection. Examples include disabling threat detection services and modifying policies.

Credential Access: Techniques used by adversaries for stealing credentials to the cloud. Examples include leaked hard-coded credentials in public repositories. Brute force and password spraying are also frequently used techniques.

Discovery: Techniques used by adversaries to gain knowledge about the cloud resources used by the targeted enterprise. Examples include enumerating cloud services.

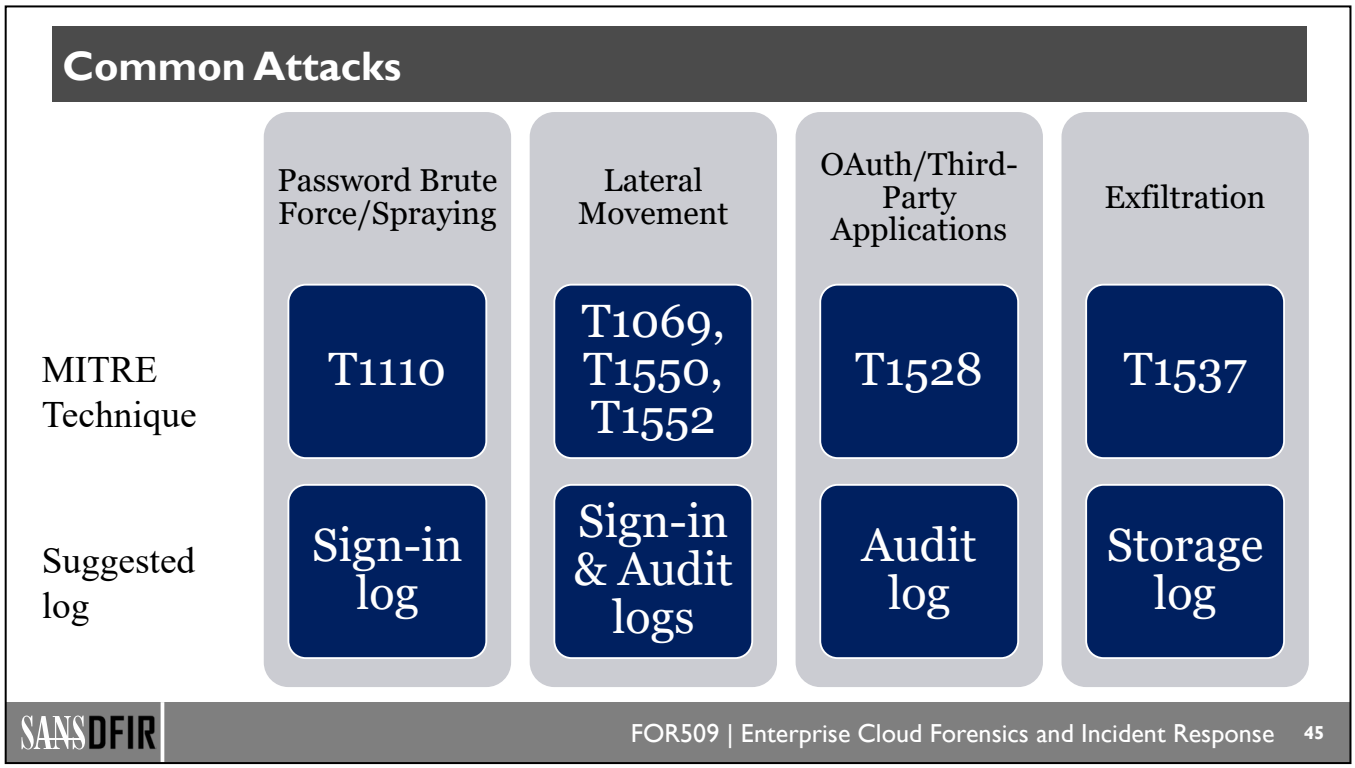
Lateral Movement: Techniques used by adversaries to obtain additional access within the cloud. Examples include the use of forged SAML tokens.

Collection: Techniques used by adversaries to gather information that is relevant to their objectives. Examples include collecting email and data from storage accounts.

Exfiltration: Techniques used by adversaries to steal data from the cloud. Examples include transferring data via APIs or storage buckets.

Impact: Techniques used by adversaries to disrupt availability or compromise data integrity. Examples include deleting resources (virtual machines, storage accounts, and so on) and deleting audit logs.

1. https://for509.com/mitre-cloud



Common cloud attacks you will see are:

- **Password brute force and password spraying**: There are two steps to mitigate this attack: implement multifactor authentication (MFA) and disable legacy protocols like IMAP. To detect this attack, you will want to monitor sign-in logs.
- **Lateral movement**: During this attack, threat actors will attempt to steal password hashes, Kerberos tickets, or application access tokens. In a cloud environment, there are two types of lateral movements to consider:
    - Access across resources
    - Access to platform-level accounts
- **OAuth and third-party applications**: "In 2020, Proofpoint detected more than 180 different malicious applications, attacking over 55% of customers with a success rate of 22%."[1] OAuth tokens were a contributing factor in the Solarwinds breach[2] which we will discuss in the Graph API section later in this class. Proper governance and monitoring audit logs are key to preventing this attack.
- **Exfiltration**: Nearly unlimited storage is one of the key benefits of the cloud. Unfortunately, it's also a great way for threat actors to discreetly exfiltrate data. Storage log monitoring is essential but not trivial given the volume of data in an enterprise.

While phishing is extremely prevalent in targeting user credential in cloud environments, we are not including it because it's not an attack against cloud-specific platforms themselves.

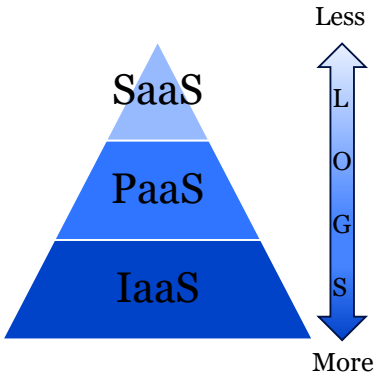
1. https://for509.com/proofpoint-oauth1
2. https://for509.com/proofpoint-oauth2

Infrastructure as a Service (IaaS)

Cloud provider hosts physical hardware (building, power, cooling) and makes virtualized resources available to customers.

WHAT DOES IT MEAN FOR DFIR?

- Logs available all the way to the operating system level
- Anything running within the infrastructure is the customer's forensic responsibility
- Customer is responsible for enabling and storing the logs
- Resources can come and go very quickly (scale up and down)
- Everything has a unit cost



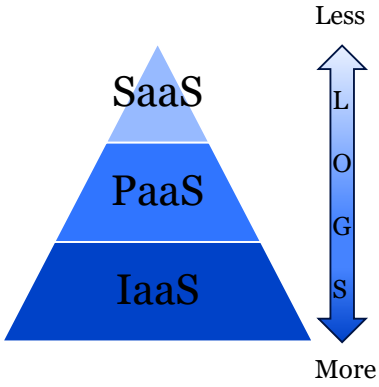
When it comes to having control over your environment, IaaS is the best solution for investigators. As long as logs have been enabled, you will have the most amount of data under this cloud model. The biggest challenge in this model is that since everything has a unit cost, the business may make the financial decision to not enable logs. Logs in and of themselves don't cost anything but storing these logs can quickly get very expensive in large environments. It's critical that you work with senior management to implement tenant-wide policies of mandatory logs.

Platform as a Service (PaaS)

Cloud provider includes everything from IaaS plus application development tools. Customer manages the applications.

WHAT DOES IT MEAN FOR DFIR?

- Platform logs will be available at the discretion of the cloud provider
- Request app developer to implement logs for their application
- Additional logs may be available based on where the authentication and authorization are performed



Under the PaaS model, you are very much at the mercy of the application developers. Providing security input early in the application development lifecycle will be key.

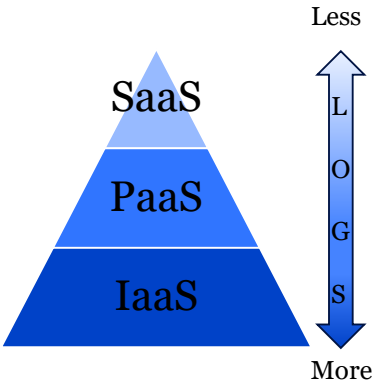
When dealing with an existing application, in the absence of application logs, you will want to look for authentication logs as well as network logs.

Software as a Service (SaaS)

Cloud provider provides a fully managed application. You pay an access fee to use but have no responsibility for the operation and maintenance.

WHAT DOES IT MEAN FOR DFIR?

- Logs are entirely at the discretion of the provider
- Access to logs should be part of the contract negotiation before using the service
- Service tier level may determine extent of logging



Under the SaaS model, logs will be at the discretion of the provider. That being said, it would be very surprising for a commercial application to not provide some type of logs. Both Microsoft 365 and Google Workspace provide extensive logs.

One caveat is that depending on the service level purchased, the extent of the logs may vary. For instance, the Microsoft E5 license provides significantly more logs than the E3 license. Also, the retention of these logs may vary based on the service level.

The other challenge with logs under the SaaS model is the ability to consume them. You will want to either import these logs to your SIEM platform and/or access them via API. Be sure to obtain the log schema from the SaaS provider so you can effectively use the information provided in these logs.

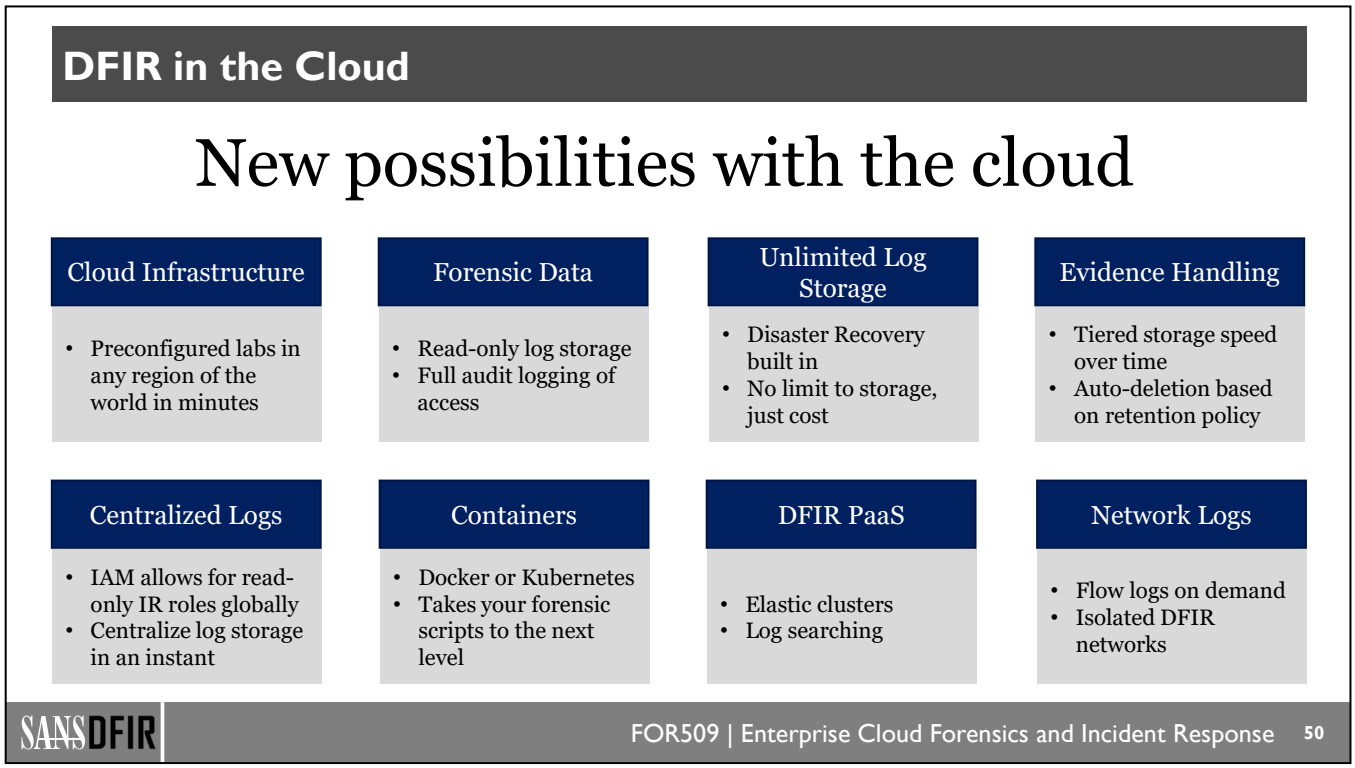
Serverless and Containers

Not technically a cloud type, but extremely popular. You just provide code to execute, and the cloud provider takes care of everything else. Serverless examples: Azure Functions, AWS Lambda, Google Cloud Functions. Container examples: Kubernetes, Docker.

WHAT DOES IT MEAN FOR DFIR?

- Systems may exist for minutes, hours, or days
- All log data may be purged on container exit
- Specialized tooling and configurations required for full visibility

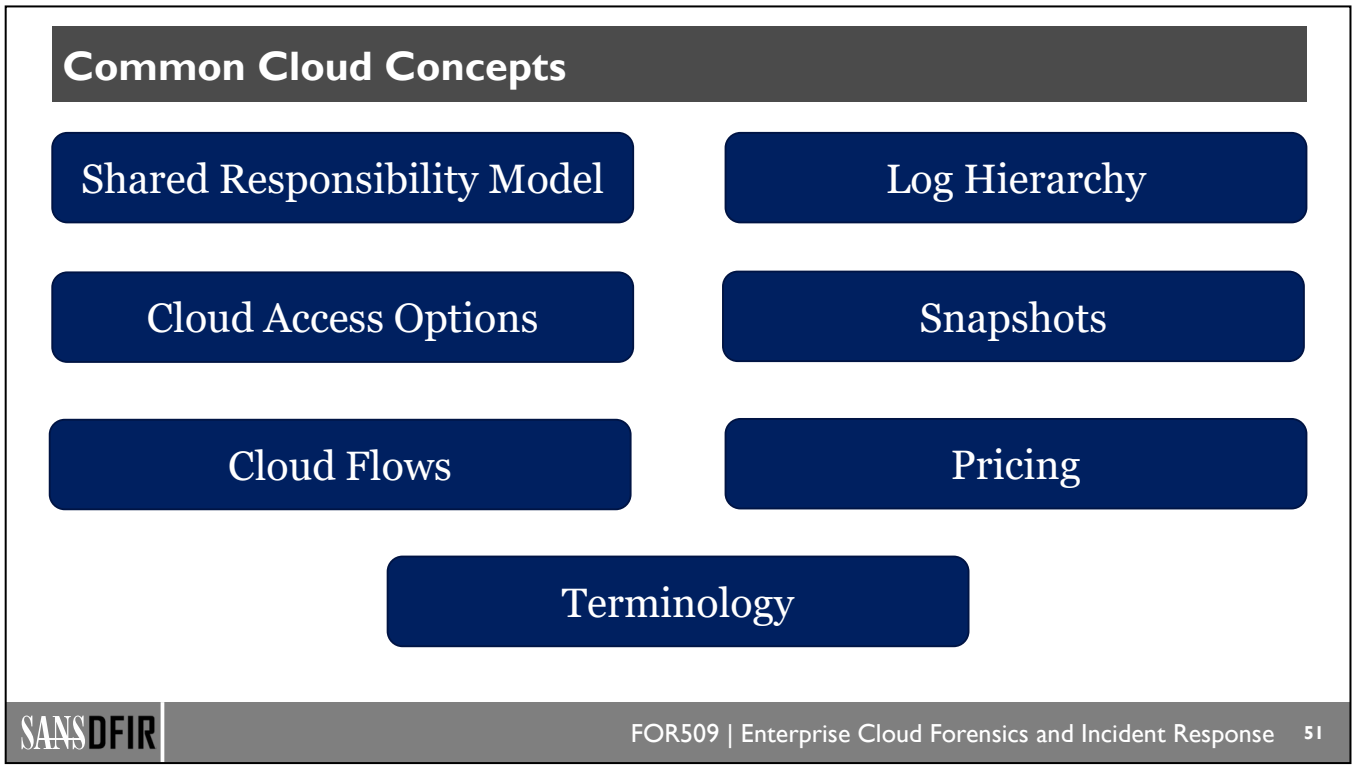
Serverless and containers are hot topics. Containers are a great way to sandbox an application, while serverless is an efficient way to run a small amount of code. From a DFIR perspective, they both represent challenges. Containers are likely to purge all log data on exit, while serverless typically runs for a very short amount of time, leaving very little in terms of logs.



When it comes to DFIR, there are so many new possibilities with the cloud. Here are just a few examples:

- **Cloud Infrastructure**: You can build a DFIR lab in any region offered by the cloud provider without leaving your office. Without the cloud, the mere logistics of acquiring physical hardware in certain countries was an insurmountable obstacle.
- **Forensic Data**: You are no longer limited to network and endpoint logs. These logs can also be stored in read-only storage to meet specific regulatory requirements.
- **Unlimited Log Storage**: No need to ask for capital dollars to purchase more disk drives. You can store as much as you want in the cloud and only pay for what you use.
- **Evidence Handling**: As your investigation progresses, you can move your evidence to slower and cheaper storage. You can also implement a retention policy to clean up older data.
- **Centralized Logs:** Make it so much easier to conduct a global investigation.
- **Containers:** Allow you to automate forensic investigations by scripting repetitive tasks.
- **DFIR PaaS:** Offers the opportunity to use a fully hosted elastic instance, where you can upload your data and immediately start your investigation.
- **Network Logs:** No longer require hardware taps and sniffers. Virtual networks can easily be configured to provide flow logs.

These are just a few of the benefits, and you will surely discover many more as you conduct your investigations in the cloud.



There are many features that are common to all cloud providers. Each provider may have a slightly different implementation, but the concepts remain the same. In order to save time during the rest of the class, we are going to briefly introduce these concepts in this section. Then, for each provider we will highlight the specific features in their corresponding days.

For the purposes of DFIR, the key common concepts are:
- Shared Responsibility Model
- Hierarchy of logs as we conduct our investigations
- Options to access the cloud
- Snapshots for virtual machine disks
- NetFlow to capture network traffic information
- Pricing concept since all clouds use consumption-based models

Finally, we will have a quick chart highlighting the names of key services for each cloud provider.

Shared Responsibility Model

Similar model for each cloud vendor. May change over time.

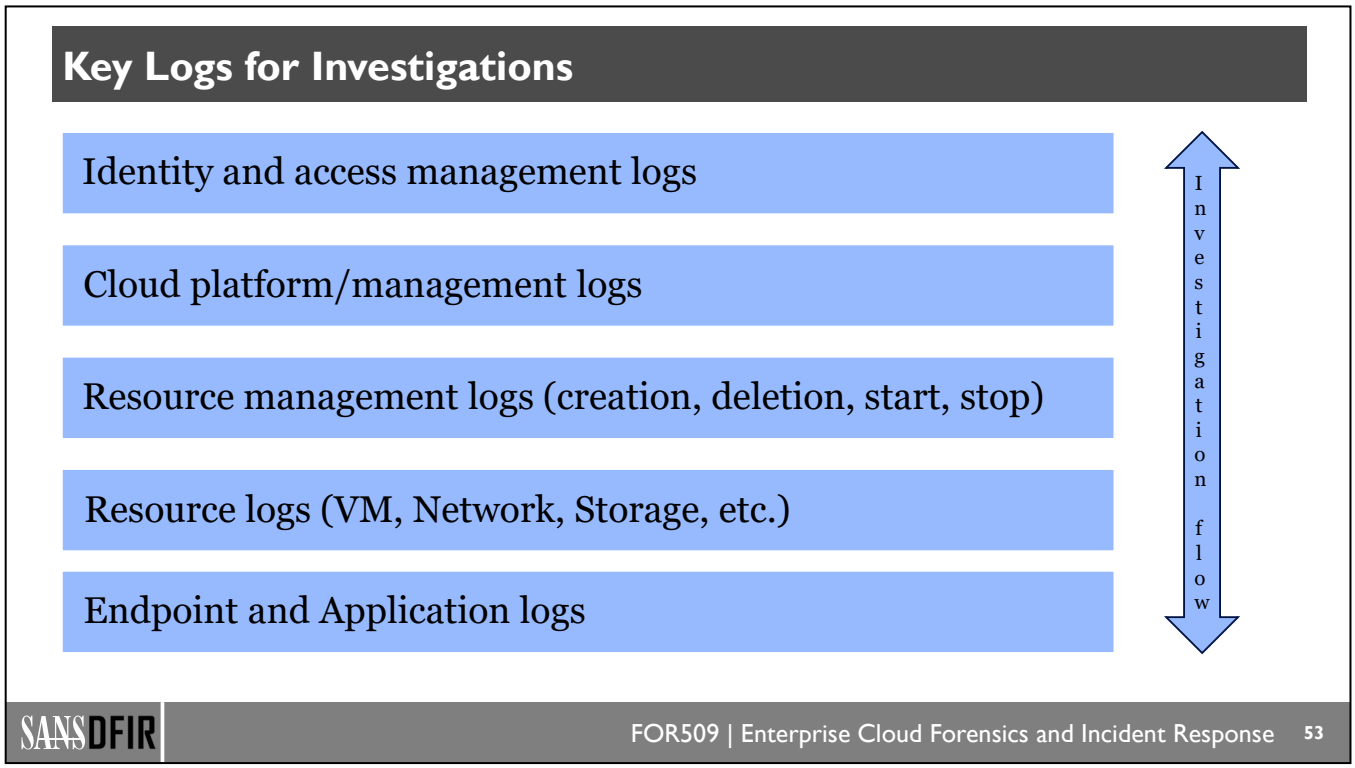
Image courtesy of Microsoft

Each cloud vendor has a shared responsibility model explaining what they will take care of versus what the customer is expected to handle. This slide shows the Microsoft shared responsibility model[1] as an example. The shared responsibility model may change at any time, so it's imperative to keep up with the latest version from your cloud provider.

As discussed in the prior slides, from a DFIR perspective, the question will always be what logs are available for us to conduct our investigation. Should we expect the cloud vendor to provide these logs? Is it our company's responsibility to have enabled these logs?

The more you can address these questions up front, the better prepared you will be when faced with an investigation.

1. https://for509.com/model



There are a huge number of log sources contained in the cloud. Which log you start with will depend on where your investigation originates.

While each cloud provider uses slightly different nomenclature, you will have the following log categories to investigate:

- Identity and access management log(s), usually referred to as IAM log(s). This log will help you determine if credentials were possibly compromised. As you develop indicators of compromise, you will likely refer back to this log over and over to investigate which credentials the threat actor may have used.
- The cloud platform log is also called cloud management log, or even audit log. This log is at the top of the cloud organization: organization root in AWS, tenant in Azure, and organization in Google Cloud. A compromise at this level is usually very bad news. That being said, in a well-configured cloud, there should be less than a handful of accounts with permissions to make changes at this level.
- The resource management log will tell you which resources were created, deleted, started, and stopped. This is an extremely important log to examine if you believe that a cloud account has been compromised. A cloud account could be either a user account, a service account, or an API key. This log is at the tenant level in AWS, subscription level in Azure, and project level in Google Cloud.
- The resource log will provide information about a resource's activity. In many cases, the attackers will directly compromise a resource such as a virtual machine or storage account. All cloud providers have logs that will help you track down such compromises. Don't forget that most clouds consider networking a resource, and this is where you will find network logs such as cloud flow logs and firewall logs.
- Finally, the operating system of each virtual machine will have the traditional logs that you have seen in other SANS classes, such as FOR500.

Cloud Access Options

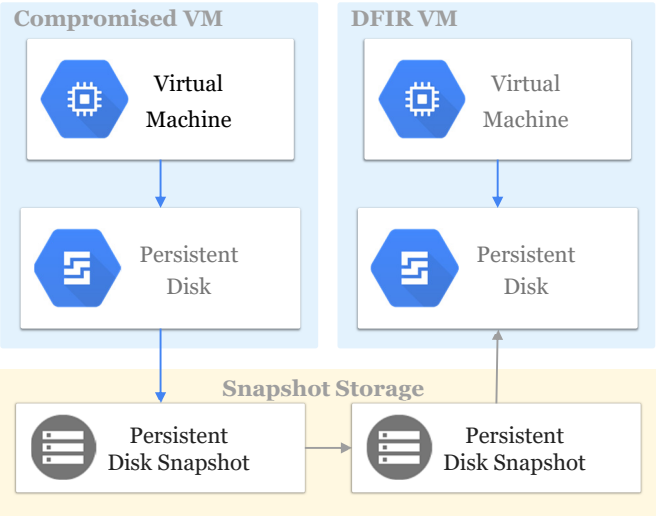
There are many ways to access the various clouds. The three ways we will be using during this class are:

- Web console
  - AWS: https://console.aws.amazon.com
  - Azure: https://portal.azure.com
  - Google Cloud: https://console.cloud.google.com
  - Microsoft has a number of other portals for specific services, such as the Microsoft Purview compliance portal
- Command line (CLI)
  - AWS: aws
  - Azure: az, PowerShell
  - Google Cloud: gcloud
- Application Programming Interface (API)
  - AWS: boto (SDK for Python)
  - Azure: Graph API
  - Google Cloud: gRPC
  - All providers offer libraries in many languages

There are many ways to access the AWS, Azure, and Google clouds. It's important that we use the same terminology, and for this class we will refer to three access methods:

1. **Web console:** This access is through a web browser. It's the easiest method to access the cloud but doesn't include all possible features.
2. **Command line (CLI):** This access leverages the operating system on your own machine to run cloud commands. Cloud-specific software must be installed on your machine to use these commands: aws, az, and gcloud, respectively, from AWS, Azure, and Google Cloud. This is a very powerful way to access these cloud environments, as these commands can be integrated within scripts. For the Azure cloud, Microsoft has also released PowerShell cmdlets. In many cases, command line access is also accessible in container-type environments. For Azure this environment is called CloudShell and is accessed from within the web console.
3. **Application Programming Interface (API):** This is the most powerful way to access cloud resources. Cloud vendors provide libraries that support many programming languages so you can directly control your cloud resources. This is achieved via a REST API.

Snapshots



VM Disk Image Snapshot

- A snapshot is a full, read-only copy of a virtual hard drive
- Snapshots can be taken while the VM is running or shut down
- Snapshots will be in the same region as the VM
- Snapshots are created at the hypervisor level and have no impact on the VM

One of the greatest features of the cloud, is the ability to snapshot a virtual disk. A snapshot is a point-in-time copy of a virtual disk. This feature is available on all three cloud providers (AWS, Azure, and Google Cloud).

Snapshots are done by the hypervisor and have no impact to the VM. They can be taken at any time whether the VM is running or shut down. Note that this last statement is made within the context of DFIR. When using snapshots for backup purposes, cloud providers usually recommend that snapshots be made while the VM is shut down.

Snapshots are read-only which is ideal for DFIR. To leverage these snapshots in our investigation, we will typically attach them to a separate analysis machine. The mechanics of this process are slightly different for each cloud, and we will cover them in their respective sections.

Cloud Flows

- Cloud flows captures information about network data flows
- No content—limited to metadata
  - Source & Destination IPs
  - Source & Destination ports
  - Protocol
  - Start & Stop times
  - Data volume
  - Virtual network information
- Each cloud has slightly different formats
- SOF-ELK includes scripts to ingest cloud flow logs from AWS, Azure, and Google Cloud

Example from Google Cloud:

```
"jsonPayload": {
  "connection": {
    "dest_port": 2222,
    "src_port": 56422,
    "protocol": 6,
    "dest_ip": "98.143.240.212",
    "src_ip": "10.138.0.10"
  },
  "start_time": "2021-04-27T15:59:54.838917542Z",
  "end_time": "2021-04-27T15:59:54.838917542Z",

  "bytes_sent": "0",
  "src_vpc": {
    "subnetwork_name": "default",
    "project_id": "suit-ai",
    "vpc_name": "default"
  },
```

Example from AWS:

```
3 vpc-abcdefab012345678 subnet-aaaaaaaa012345678 i-
01234567890123456 eni-1235b8ca123456789 123456789010
IPv4 52.213.180.42 10.0.0.62 43416 5001 52.213.180.42
10.0.0.62 6 568 8 1566848875 1566848933 ACCEPT 2 OK
```


Cloud flows are a summarized version of network data that is passed across the wire. However, cloud flows do not contain the full context of the data that transited across the network; instead, it contains summarized information. Generally, cloud flows contain a wide range of fields; however, the most common fields that are of use to us as digital forensics Investigators are:

- Source IP address
- Destination IP address
- Source port
- Destination port
- Start time and end time for traffic that flowed in one direction

This means TCP traffic that produced send and receive traffic will produce two flow records.

The advantage of cloud flows for incident response is they are extremely quick to search and consumes far less space compared to PCAP data. This also means that we could store cloud flow logs for a much longer period of time than we could PCAP data.

The data you will see within the cloud infrastructure that we will show you during this course includes information that is very similar to NetFlow but is not technically NetFlow. The IETF produced a standard for NetFlow called IPFIX, which matches up to Cisco's NetFlow v9. The data you will see during this course is what we would typically call "flow data," as it contains information about network flows. Regardless of this data not being IETF compliant for IPFIX, it still provides significant digital forensics value when investigating compromises within cloud infrastructures.

Additional information often included in flow data for cloud environments is:
1. Virtual network information
2. Host information showing where traffic has originated or is destined

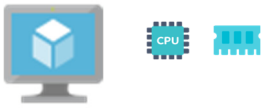
This will change in different cloud environments based on where the flow capture is taken.

Pricing

Clouds use a per-consumption model, which means everything you do has a cost.

Temporary Costs

- Virtual machines are priced based on number of CPUs and amount of memory
- They only accrue cost when running
- Option to shut them down when not in use



Virtual Machine

Persistent Costs

- Disks and snapshots are priced based on performance and quantity
- They accrue cost all the time until deleted

Persistent Disk Disk Snapshot

Some services can incur a combination of temporary and persistent costs, for example Google's BigQuery.

We need to talk about pricing, as it may impact your investigations. Cloud providers charge on a per-consumption model, which means you have to pay for everything as you use it. For our purposes, the charges we need to be concerned with are virtual machine costs, storage costs, and possibly data transfer costs. Data transfer costs can be avoided by conducting as much of the investigation in the cloud and, specifically, in the same region as your victim machines.

Virtual machine cost has two components:

1. Memory and CPU provisioned for the VM
2. Length of time you keep the machine running

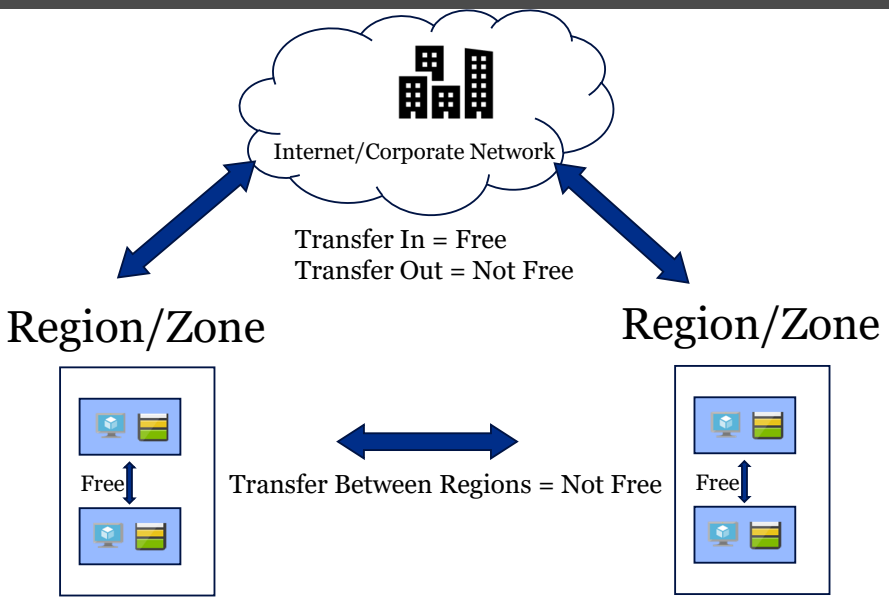
Storage is more complex. We will cover the different types of storage available in the next section. For the purpose of pricing, the main components are:

1. Performance
2. Quantity

Storing logs also requires storage, which is one of the main reasons you will find that logs are often not configured beyond the defaults set by the provider.

Data Transfer Cost

- Generalized view of data transfer in the cloud
- Each cloud will be slightly different
- Terminology (region/zone) will vary
- As a rule of thumb
  - In: free
  - Out: not free



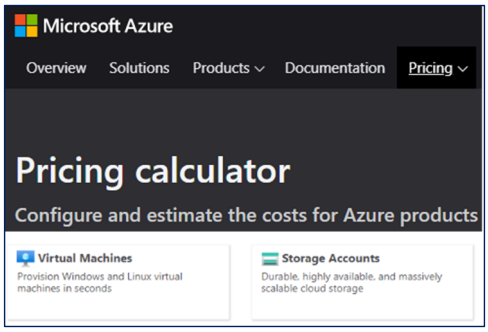
In addition to the costs for the virtual machines and the storage, you may incur cost if you move data from one region to another. The best way to avoid data transfer costs is to leave the data in the same region as your victim machine. If you must transfer data, it's helpful to know when you will be charged.

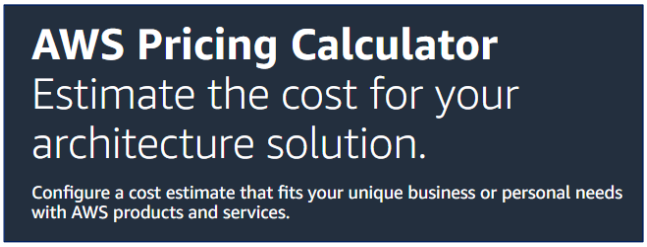
Each cloud provider will have slightly different rules, but a good rule of thumb is that if you are transferring data to the cloud, there is no charge. However, when you transfer data out of the cloud (or to a different region/zone), there is a charge. Cloud providers have a vested interest in getting more of your data in their cloud, as it consumes more and more storage.

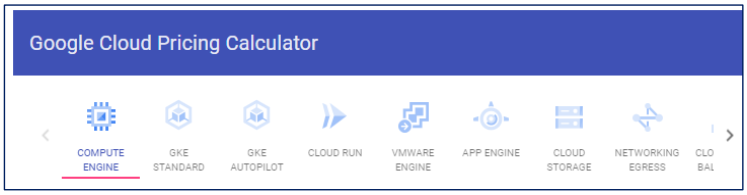
Before transferring large amounts of data, it's a good idea to refer to the cloud provider's documentation to know the rules and get an estimate of the cost.

Pricing Calculators

Estimate resource costs with the pricing calculators







It's important to estimate your costs before starting your investigation. Large and complex investigations may have significant long-term persistent costs.

Each cloud provider has a pricing calculator you can use:

- AWS: https://calculator.aws/
- Azure: https://azure.microsoft.com/en-us/pricing/calculator/
- Google Cloud: https://cloud.google.com/products/calculator/

Terminology Across Clouds

Service	AWS	Azure	Google Cloud
Virtual Machine	EC2 Instance	Virtual Machine	Compute Engine
Serverless	Lambda	Functions	Cloud Functions
VM Disk Storage	EBS (Elastic Block Store)	Managed Disks	Persistent Disks
Object Storage	S3 (Simple Storage Service)	Blob Storage	Cloud Storage
Network File Storage	EFS (Elastic File System)	File Storage	File Store
Virtual Networking	VPC	VNet	Cloud Virtual Network
Logging	CloudWatch & CloudTrail	Log Analytics	Log Explorer
Message Queuing	SQS (Simple Queue Service)	Event Hub	Cloud Pub/Sub

This is a small sample of the hundreds of cloud terms.

This table shows a very small sample of the hundreds of cloud terms. The terms in this table will be used throughout this class.

For a more complete set of cloud terminology, we recommend the website https://comparecloud.in.

FOR509.1: Microsoft 365 and Graph API

Section 1.1: Introducing SOF-ELK®

Section 1.2: Key Elements of Cloud for DFIR

Section 1.3: Microsoft 365 Unified Audit Log

Section 1.4: Microsoft Graph API

This page intentionally left blank.

Microsoft 365 and Graph API Roadmap

1.1: Introducing SOF-ELK®

1.2: Key Elements of Cloud for DFIR

1.3: Microsoft 365 Unified Audit Log

1.4: Microsoft Graph API

- Connecting to Microsoft 365
- Unified Audit Log (UAL)
- Searching the UAL
- UAL Workloads
- Case Study: Exchange Workload
- Mail Clients Logs
- Azure Active Directory
- **Lab 1.2**: Exploring the UAL

This page intentionally left blank.

PowerShell: Connecting to Microsoft 365

Many PowerShell commands will be shown in this class. Before these can be used, an authenticated session must be created:

```
PS> Install-Module -Name ExchangeOnlineManagement
PS> Import-Module ExchangeOnlineManagement; Get-Module ExchangeOnlineManagement

PS> Connect-ExchangeOnline -UserPrincipalName <UPN> -ShowProgress $true
```

Replace <UPN> with your userid. A separate window will open, and you will be prompted for your credentials, including your second factor.

All further slides in this section with PowerShell instructions will assume that you have successfully established a connection to Microsoft 365.

In this section, we will introduce a number of PowerShell commands. Before we can issue any Microsoft 365 PowerShell command, we must establish a session and import the appropriate PowerShell cmdlets in your terminal session. To support authentication with MFA, we must use the Exchange Online PowerShell V2 module (EXO V2 module).[1]

Step 1: Install the EXO V2 module (if not already installed)
`Install-Module -Name ExchangeOnlineManagement`

Step 2: Verify the module was installed and check the version
`Import-Module ExchangeOnlineManagement; Get-Module ExchangeOnlineManagement`

Step 3: Connect to Exchange Online using the EXO V2 module
`Connect-ExchangeOnline -UserPrincipalName <UPN> -ShowProgress $true`

Replace <UPN> with your userid (example: admin@pymtechlabs.com).

Steps 1 and 2 are only needed the one time. Once the Exchange Online PowerShell V2 module is installed on your computer, you will only need step 3.

Each PowerShell session (i.e., your current PowerShell window) is authenticated separately. If you open a new window, you will have to issue the command from step 3 to authenticate that session.

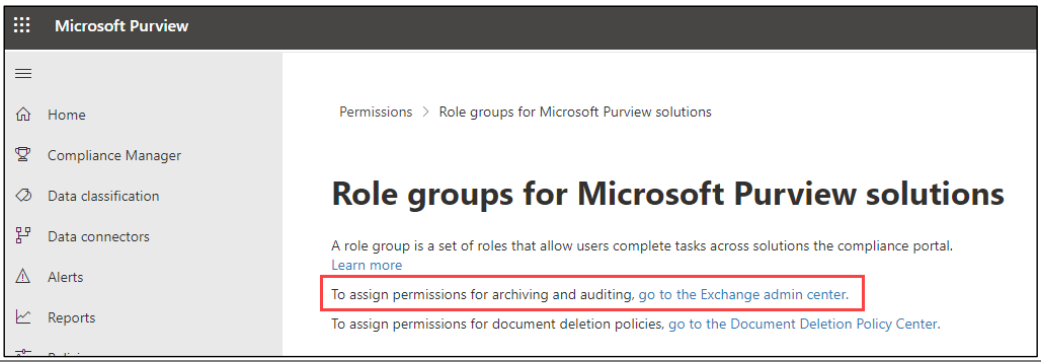
A session will be created based on the privileges of the account provided in step 1. Step 3 will import cmdlets based on the privileges of that account. If the account doesn't have the necessary administrative permissions, you will be missing key cmdlets such as Search-UnifiedAuditLog, Get-Mailbox, etc.

All future slides with PowerShell instructions will assume that you have successfully established a connection to Microsoft 365.

1. https://for509.com/exchangeonline

Minimum Permissions for PowerShell

- Principle of least privilege
- Create new role group. For example, call it “AuditOnly”
- Assign role: `View-Only Audit Logs`
- Assign to desired user
- Create in both: Microsoft Purview compliance portal & Exchange Admin center



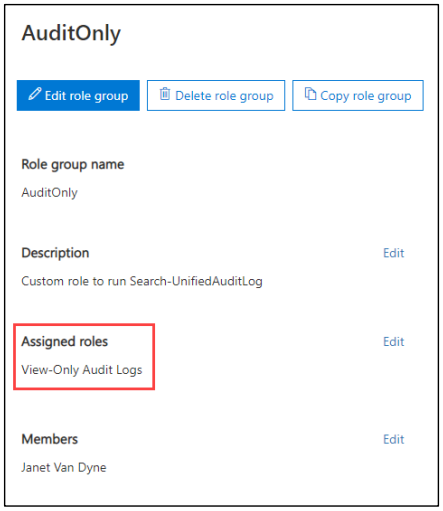
High privilege roles such as Global Admin and Security Admin should be used with extreme caution. Azure active directory has many pre-defined roles but none that follow the principle of least privilege for accessing with UAL via PowerShell.

We can create our own role group and assign it the “View-Only Audit Logs” role. This role will provide read-only capabilities to the UAL. Appropriate users can then be assigned to this new role group. In this slide, we named this new role group “AuditOnly”. You can give it whichever name you would like.

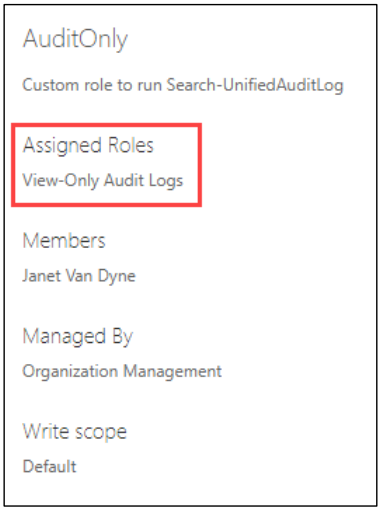
There is a very important step that must be completed: the role group must be created in both Microsoft Purview compliance portal and Exchange Admin center. If it’s only created in the Microsoft Purview compliance portal, the assigned users will not be able to use PowerShell.

Role "View-Only Audit Logs"

Role in Microsoft Purview compliance portal

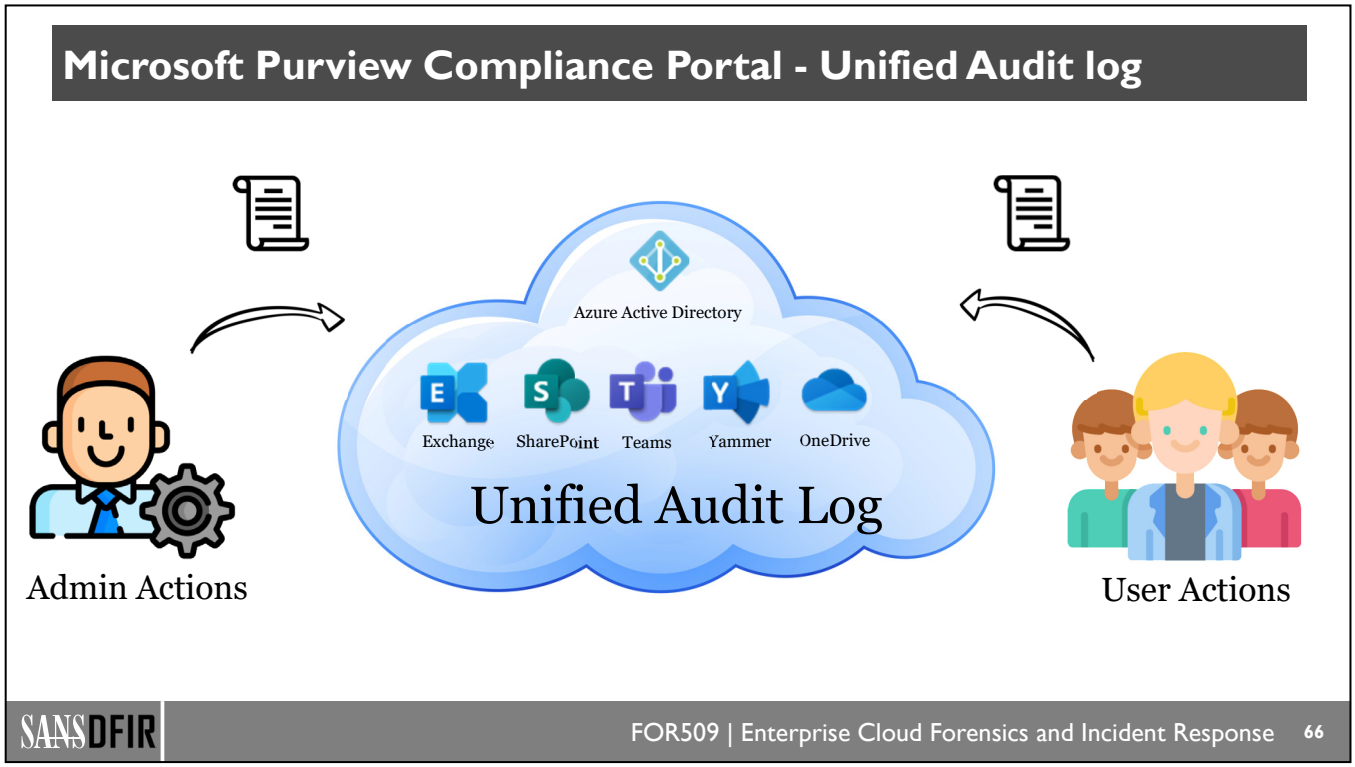


Role in Exchange Admin center



In this slide, you can see how we created a role group called "AuditOnly" in both Microsoft Purview compliance portal and Exchange Admin center. In both cases, we assigned the role "View-Only Audit Logs" to each group. Finally, we assigned "Janet Van Dyne" as a member of these groups.

Janet will now be able to access the UAL both in Microsoft Purview compliance portal and via PowerShell.



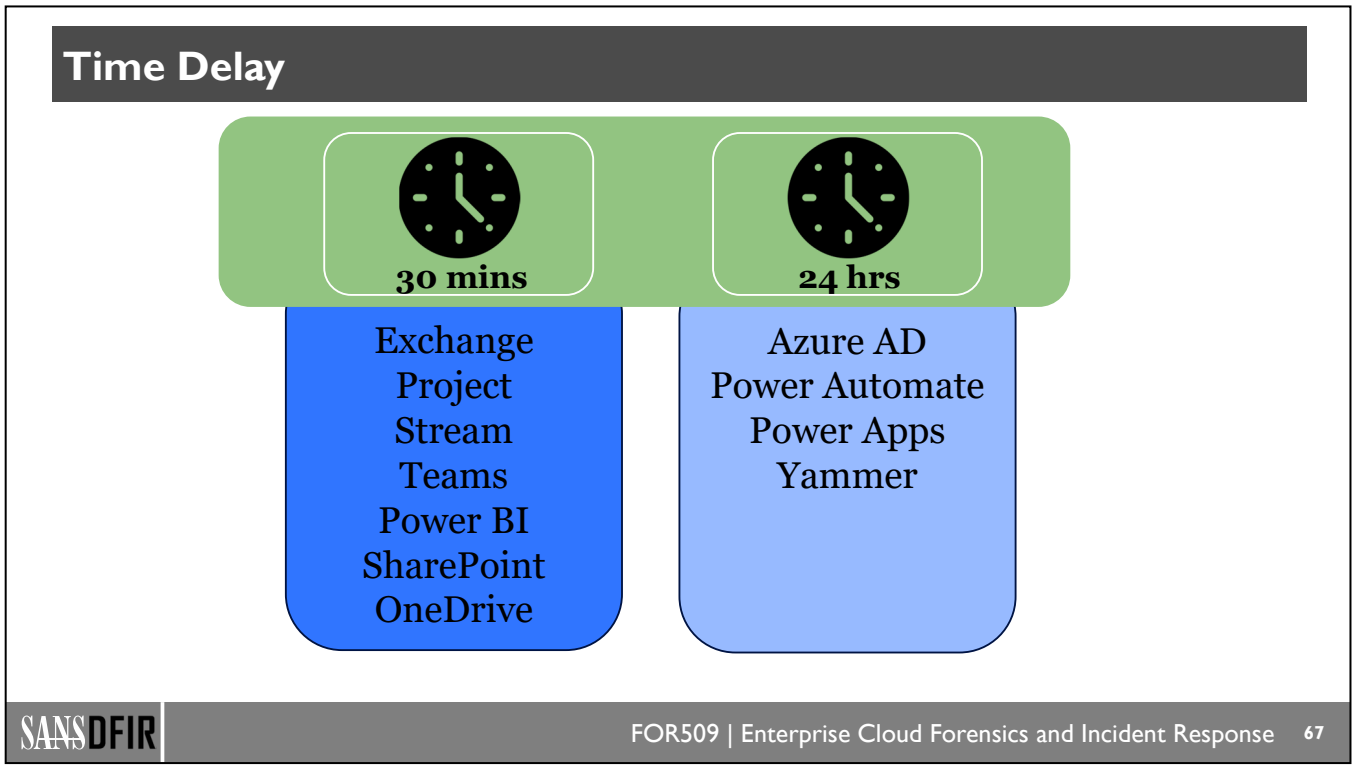
Microsoft 365 offers numerous applications. Fortunately, logs are aggregated in a single location called the Unified Audit Log (UAL).

The UAL will record both user activity and admin activity. The list of Microsoft 365 applications is constantly changing. These are some of the most common:

- Azure Active Directory
- SharePoint Online
- OneDrive for Business
- Exchange Online
- Power BI
- Teams
- Dynamics 365
- Yammer
- Flow
- Stream
- Workplace Analytics
- PowerApps
- Forms

As we will discuss shortly, the UAL can be queried in three different ways:

1. Microsoft Purview compliance portal
2. PowerShell
3. Microsoft 365 API



Audit log entries can take between 30 minutes and up to 24 hours before they are displayed in the search results.[1]

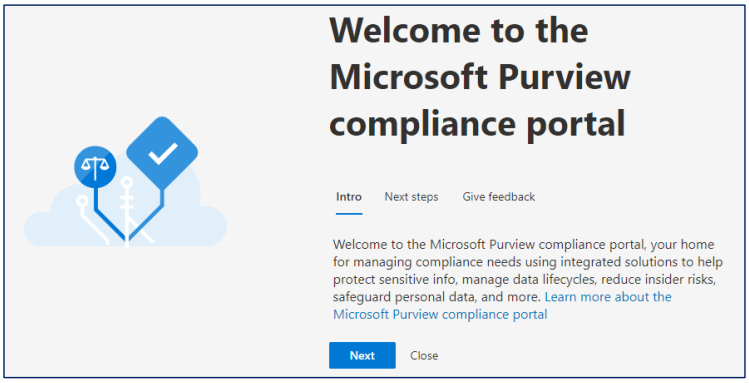
In practice, the time is far from exact. It also depends whether the data is being extracted via the Microsoft Purview compliance portal, PowerShell, or API.

1. https://for509.com/search-ual

Are Audit Logs Turned On?

Microsoft Purview compliance portal

- As of October 2021, audit logging is turned on by default for newly created tenants
- Existing tenants should verify that audit logging was previously enabled



As of October 2021, the Unified Audit Log (UAL) is turned on by default. Unfortunately, administrators have the option to turn it off using PowerShell as shown in the next slide.

Turn on Audit Log with PowerShell and Verify

```
PS> Set-AdminAuditLogConfig -UnifiedAuditLogIngestionEnabled $true

PS> Get-AdminAuditLogConfig

AdminAuditLogEnabled            : True

AdminAuditLogAgeLimit           : 90.00:00:00

UnifiedAuditLogFirstOptInDate : 12/23/2019 4:30:51 PM

WhenChangedUTC                  : 12/23/2019 10:31:00 PM

WhenCreatedUTC                  : 12/23/2019 10:30:07 PM
```

Great data point for DFIR

In this slide, we show the PowerShell cmdlet to turn on the audit log and the cmdlet to verify that it has been correctly turned on.

The cmdlet `Get-AdminAuditLogConfig` will confirm that audit logging has been enabled and provide some very interesting information. From a forensics point of view, you should record the date in the field "UnifiedAuditLogFirstOptInDate" and compare that date with the time frame of any incident you might be investigating.

The dates "WhenChanged" and "WhenCreated" may also be useful if a threat actor stopped and started the audit log in order to make sure that their misdeed wasn't recorded.

By default, advanced audit in Microsoft 365 will retain all Exchange, SharePoint, and Azure Active Directory audit records for 1 year. This default policy can't be modified. This default policy only applies to certain record types, as documented in reference.[1]

Further, the default policy retention of 1 year only applies to users who are assigned a Microsoft E5 license. For all other licenses, audit records are only retained for 90 days.

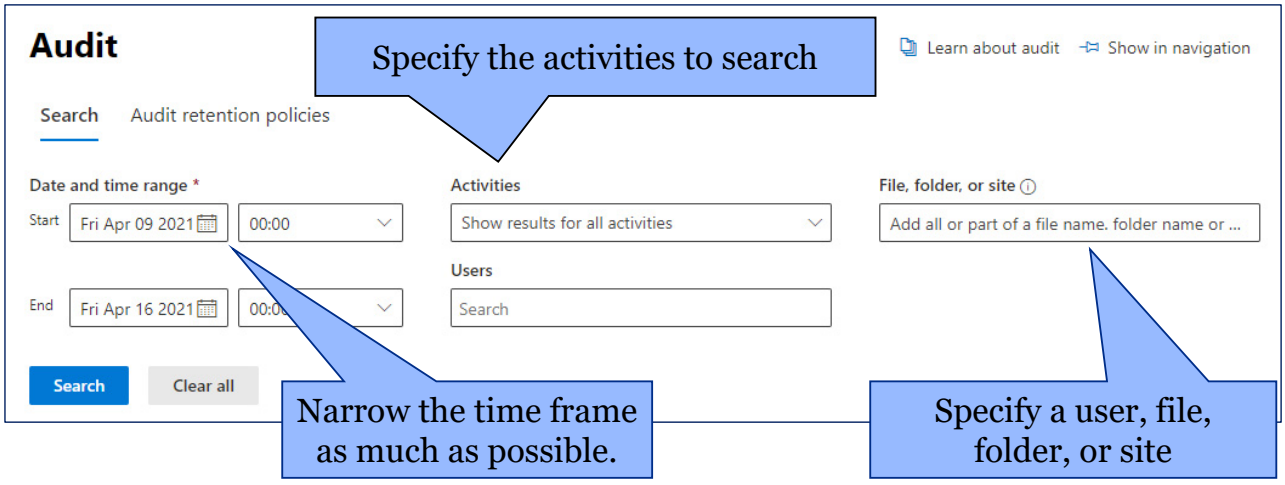
We expect most large organizations to export audit records to a third-party security information and event management (SIEM) application. In later slides, we will show you different methods to export this data. In this course, we will use SOF-ELK to analyze the UAL.

On July 7, 2021, o365reports.com published an article[2] stating that Microsoft was increasing retention to 1 year for all license levels. Until we get an official notice from Microsoft, we will consider this anecdotal evidence. However, it's welcomed news.

1. https://for509.com/logretention
2. https://for509.com/1yr-retention

Searching the UAL: Microsoft Purview Compliance Portal

Microsoft Purview compliance portal: https://compliance.microsoft.com



As previously mentioned, there are three methods to search the UAL: Microsoft Purview compliance portal, PowerShell, and API. The easiest is the Microsoft Purview compliance portal, which provides a graphical user interface.

To start a search, provide any of the four pieces of information:

1. Activities to search for
2. Timeframe
3. Specific users
4. Specific file, folder, or site

This search can be quite slow, and it's highly recommended to be as specific as possible. Microsoft may also rate-limit the data extraction, which can result in incomplete information. This method of searching the UAL is only recommended for small data sets.

Unfortunately, the output on the screen is not very useful, and therefore the information is best when downloaded as a CSV file. However, the CSV file is itself very difficult to use, as all the information is contained in a single field called AuditData. This is the reason why a tool like SOF-ELK is needed.

The UAL log contains a large number of user and admin activities for each application. An exhaustive list of these activities and their description in documented in the reference.[1]

1. https://for509.com/search-ual

Searching the UAL: PowerShell (1)

1. Basic search

```
PS> Search-UnifiedAuditLog -StartDate 2021-04-09 -EndDate 2021-04-16
```

2. Search for all login records

```
PS> Search-UnifiedAuditLog -StartDate 2021-04-09 -EndDate 2021-04-16 ⏎
    -Operations UserLoggedIn
```

3. Search for all login records for a specific user

```
PS> Search-UnifiedAuditLog -StartDate 2021-04-09 -EndDate 2021-04-16 ⏎
    -Operations UserLoggedIn -UserIds jvandyne@pymtechlabs.com
```


These commands require that you are connected to Microsoft 365 as shown in the slides at the beginning of this section.

The second method to search the UAL is to use the PowerShell cmdlet `Search-UnifiedAuditLog`.[1] This cmdlet has many options, and we will provide a few examples here.

Example 1:
In its simplest form, the UAL can be searched by simply providing a start date and an end date. As we have previously discussed, depending on your tenant's license, you may be able to search up to 1 year (E5 license) or 90 days (all other licenses).

Example 2:
The first example is likely to provide way too much data. Each audit log has a field called "Operations" that specifies the type of action being recorded. For example, the operation "UserLoggedIn" will record authentication attempts to Azure AD.

Example 3:
We may not be interested in every authentication attempt during the specified period. We may only be looking for a specific user. In that case, we would use the option "UserIds" and specify the name of the user we are looking for.

1. https://for509.com/search-unifiedauditlog

Searching the UAL: PowerShell (2)

4. Search for all failed logins and export to CSV (increase to maximum number of results)

```
PS> Search-UnifiedAuditLog -StartDate 2021-04-09 -EndDate 2021-04-16 ⏎
    -Operations UserLoginFailed -ResultSize 5000 -SessionCommand ⏎
    ReturnNextPreviewPage | Export-Csv -Path "c:\data\userloginfailed.csv"
```

5. Search for all events and return results as JSON

```
PS> Search-UnifiedAuditLog -StartDate 2021-04-09 -EndDate 2021-04-16 ⏎
    -ResultSize 5000 -SessionCommand ReturnNextPreviewPage | Select-Object ⏎
    -ExpandProperty AuditData | Out-File "c:\data\ual.json"
```


Example 4:
In this example, we changed the Operations parameter to "UserLoginFailed" in order to look for failed login attempts. By default, `Search-UnifiedAuditLog` will only return 100 results. The "-ResultSize 5000" parameter will increase the number of records to its maximum, and the option "-SessionCommand ReturnNextPreviewPage" will sort the output by date.

If an unsorted output is acceptable, the option "-SessionCommand ReturnLargeSet" will output up to 50,000 records.

Since we expect a large data set, we will store it in a CSV (comma-separated values) file by using the cmdlet `Export-Csv`.

Example 5:
One of the advantages of PowerShell is that results are returned as objects. As such, we can choose specific objects from the results and discard everything else. The UAL stores all the relevant information in a field called "AuditData." In this example, we are extracting that field as a JSON object.

These are just five examples to show the flexibility of using PowerShell to extract information from the UAL. You may find these types of searches appropriate when conducting a small, highly focused investigation. However, in the normal course of business, the UAL should be exported on a continuous basis to a SIEM via the Microsoft 365 API.

UAL: Import into SOF-ELK

1. Export data to CSV file (portal or PowerShell)
   - The Logstash parser was written to import data from a CSV file (rather than JSON) to be compatible with both the portal and PowerShell
   - The format is a bit different between the portal and PowerShell, but the Logstash parser will accommodate both versions
2. Copy file to the Logstash folder
   - The Logstash folder for the Microsoft 365 log is `/logstash/office365`
3. Wait a few minutes; CSV files take a bit longer to process than JSON files

The UAL can be exported either through the Microsoft Purview compliance portal or via PowerShell, as previously described. The CSV file format is a little bit different between the two, but the Logstash parser was written to accommodate both formats.

PowerShell can also export the UAL in JSON format, which is more efficient to process. However, we didn't write a Logstash parser for this format at this time.

The export file should then be copied to the /logstash/office365 folder for processing.

Before being known as Microsoft 365, the office suite was called Office 365, hence the name of the directory and index in Kibana.

You will get the opportunity to practice these steps in Lab 1.2.

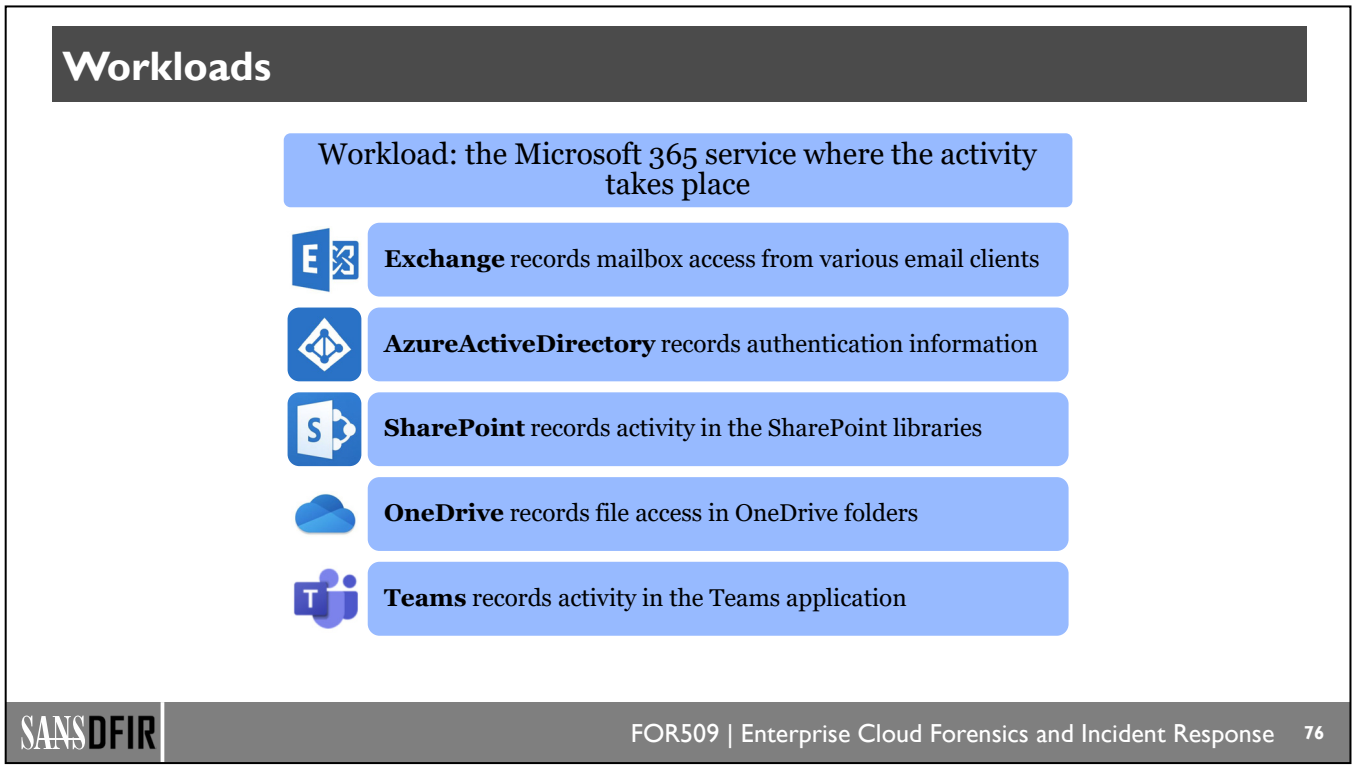
UAL Collection Tools

There are many limitations when exporting the UAL via PowerShell. A number of open-source tools have been written to facilitate this task.

- Tesorion CERT UAL Extractor
  - https://for509.com/tcert-ual
- Invictus Incident Response Microsoft 365 Extractor
  - https://for509.com/invictus-ual
- Blue team app for Splunk by Korstiaan Stam
  - https://for509.com/splunk-blueteamapp

The UAL contains valuable information for our investigations. However, when exporting the UAL via PowerShell, there is a limitation of 5,000 sorted records or 50,000 unsorted records (depending on the setting of the ResultSize parameter).

This complexity requires additional logic to extract UAL entries for a large organization. Fortunately, a number of open-source tools have been written to facilitate this task. Some of them will not only facilitate extracting the UAL, but also Azure logs. We recommend you experiment with them to see which one works best for you.



Each UAL entry contains a wealth of information. As a matter of fact, the quantity of information can be overwhelming.[1]

Each UAL entry contains a large number of fields. The most important field is called "workload." Microsoft uses the term "workload" to describe which Microsoft 365 service wrote the log entry. The primary workloads you will see are:

- AzureActiveDirectory
- Exchange
- SharePoint
- OneDrive
- MicrosoftTeams
- SecurityComplianceCenter

Some fields are found in most log entries, while others are unique to specific workloads. The thing to keep in mind during your investigation is that fields can be named differently from one workload to another.

For example, the Exchange workload records the IP address in a field called "ClientIPAddress," while the AzureActiveDirectory workload records the IP address twice in fields called "ActorIPAddress" and "ClientIP." As such, you must be very careful when you filter the data.

When looking at a log for the first time, it's best to focus on a few key fields to narrow down the search. Suggested key fields are Time, UserId, Workload, Operations, and ResultStatus.

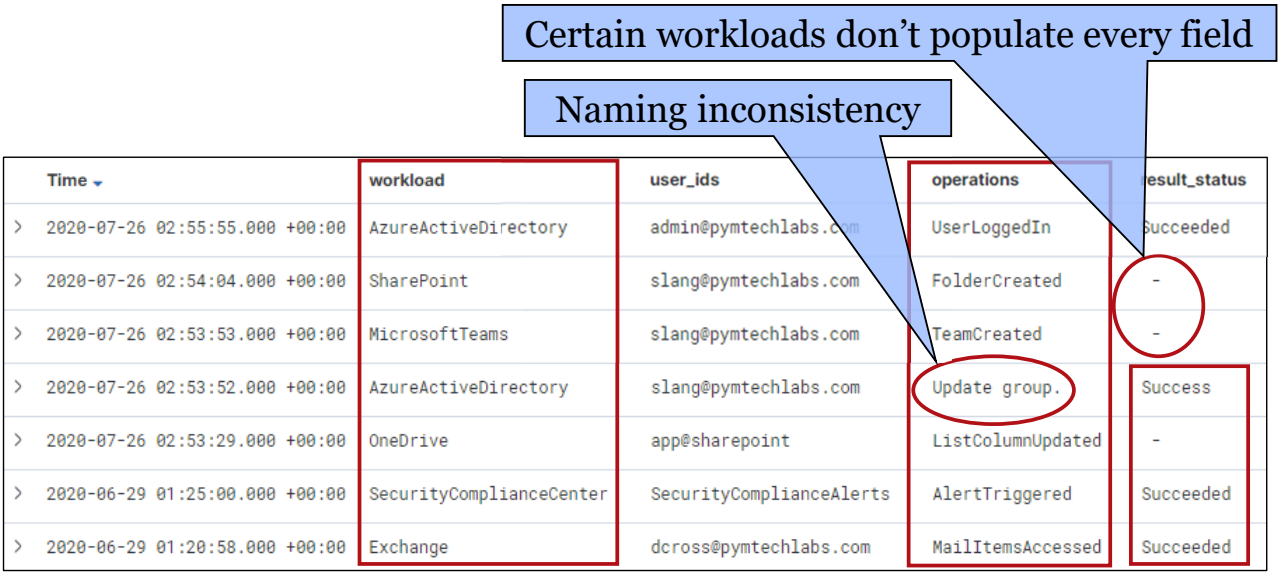
Time, UserId, and ResultStatus are self-explanatory.

Operations will show you what action was taken in that log entry. Some of the most common are:

- UserLoggedIn
- MailItemsAccessed
- FileAccessed
- FilePreviewed
- PageViewed
- MoveToDeletedItems
- SoftDelete

1. https://for509.com/ual-properties

Workload Examples

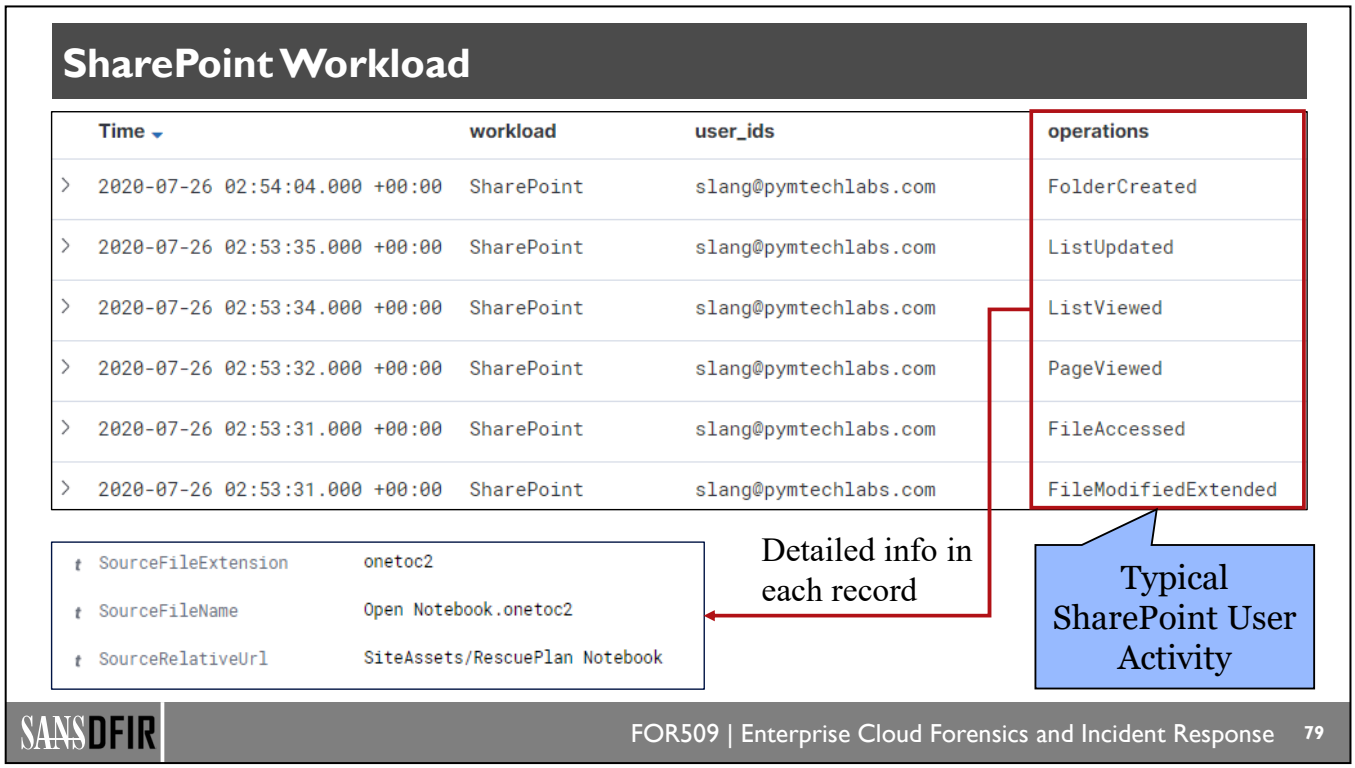


Time	workload	user_ids	operations	result_status
2020-07-26 02:55:55.000 +00:00	AzureActiveDirectory	admin@pymtechlabs.com	UserLoggedIn	Succeeded
2020-07-26 02:54:04.000 +00:00	SharePoint	slang@pymtechlabs.com	FolderCreated	-
2020-07-26 02:53:53.000 +00:00	MicrosoftTeams	slang@pymtechlabs.com	TeamCreated	-
2020-07-26 02:53:52.000 +00:00	AzureActiveDirectory	slang@pymtechlabs.com	Update group.	Success
2020-07-26 02:53:29.000 +00:00	OneDrive	app@sharepoint	ListColumnUpdated	-
2020-06-29 01:25:00.000 +00:00	SecurityComplianceCenter	SecurityComplianceAlerts	AlertTriggered	Succeeded
2020-06-29 01:20:58.000 +00:00	Exchange	dcross@pymtechlabs.com	MailItemsAccessed	Succeeded

In this slide, we used SOF-ELK to filter the UAL to show five key fields: time, workload, user_ids, operations, and result_status.

These are random events picked from the UAL to give you an idea of what you may see once you import your data into SOF-ELK. Notice that the different workloads don't always populate every field and may use different terminology (for example, success versus succeeded).

In the next few slides, we will show more detailed examples of the SharePoint, OneDrive, and Teams workloads. We will then explore the Exchange workload in more detail as it's key in your Business Email Compromise (BEC) investigations.

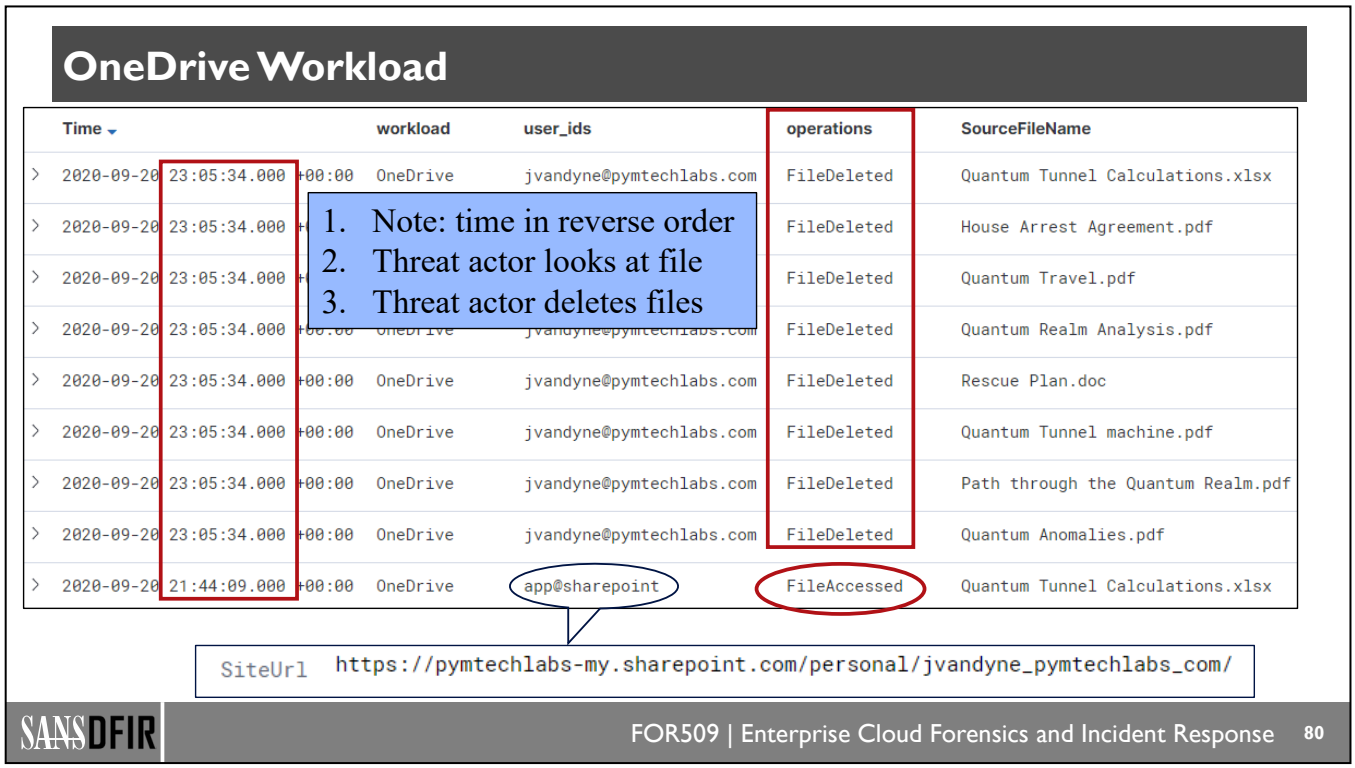


This slide is an example of typical SharePoint user activity. Some of the more frequent operations you will see related to SharePoint are:

Operation	**Description**
FolderCreated	User creates a folder on a site.
ListUpdated	User updates a SharePoint list by modifying one or more properties.
ListViewed	User views a list on a site.
PageViewed	User views a page on a site.
FileAccessed	User or system account accesses a file.
FileModifiedExtended	User continually modifies a file (up to 3 hours).

The name given to each operation is pretty vague, making it difficult to ascertain the actual activity of the user. It's important to review the entire log entry to get a full picture. In this particular example, both "FileAccessed" and "FileModifiedExtended" were generated when Scott Lang opened a OneNote notebook that's embedded in the SharePoint site and modified it.

While not depicted on this slide, you will also see a number of activities where the userid is set to app@sharepoint. This means the system performed the activity on behalf of the user who initiated the action. Unfortunately, such a generic userid makes our job more difficult.

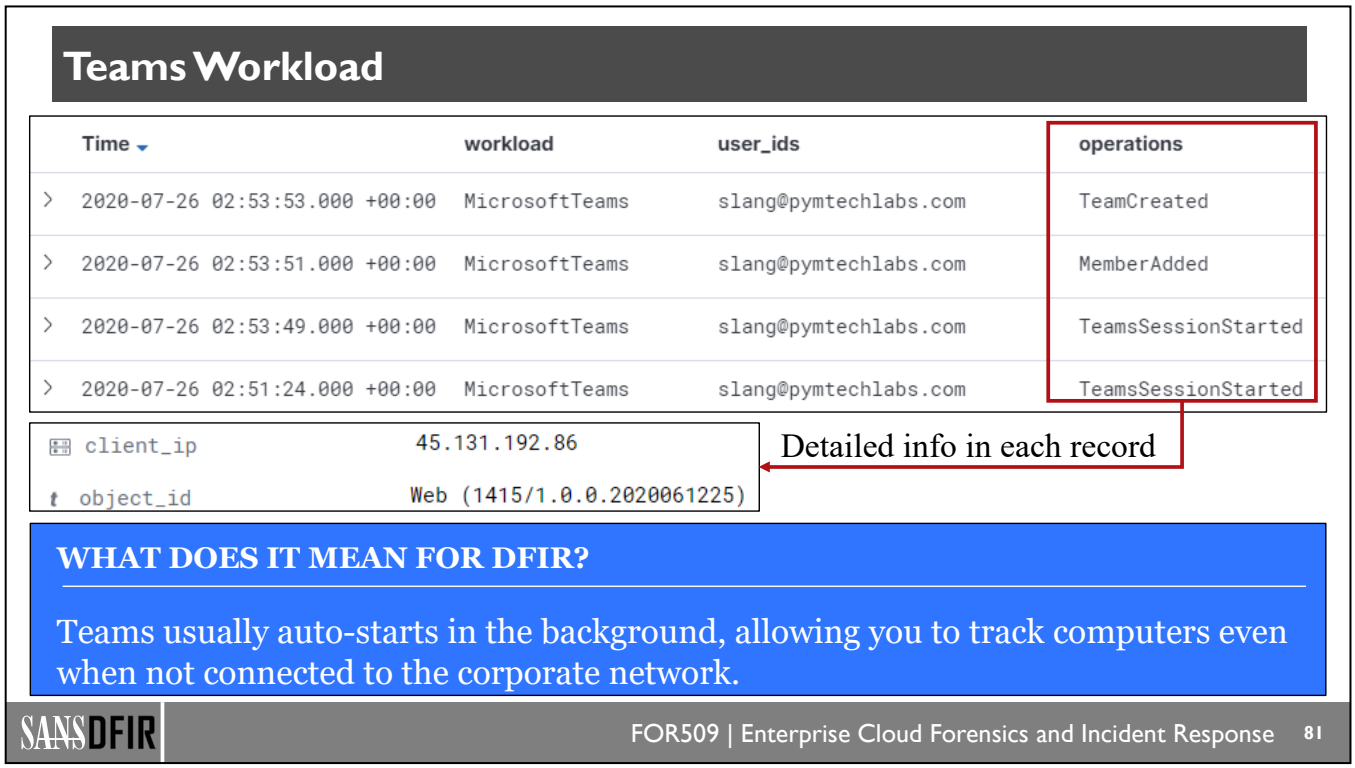


OneDrive for Business (simply called OneDrive) allows users to automatically sync their files to the Microsoft 365 cloud. This is a treasure trove of information for our investigation. While the user thinks of their file as being local on their computer, a copy is actually made to the cloud and every access is logged.

The situation shown above may happen when a user decides to leave the company. They may check the content of a file, as shown by the FileAccessed operation, followed by a mass deletion.

As mentioned in the SharePoint Workload example, you may sometimes see activities where the userid is set to app@sharepoint. This means the system performed the activity on behalf of the user who initiated the action. By looking at the name of the file and the siteurl, you may be able to deduce who performed the operation. Unfortunately, that's not always possible for large sites with a lot of activity.

You will also notice that OneDrive is using SharePoint in the background and that the OneDrive folder is represented by a URL.



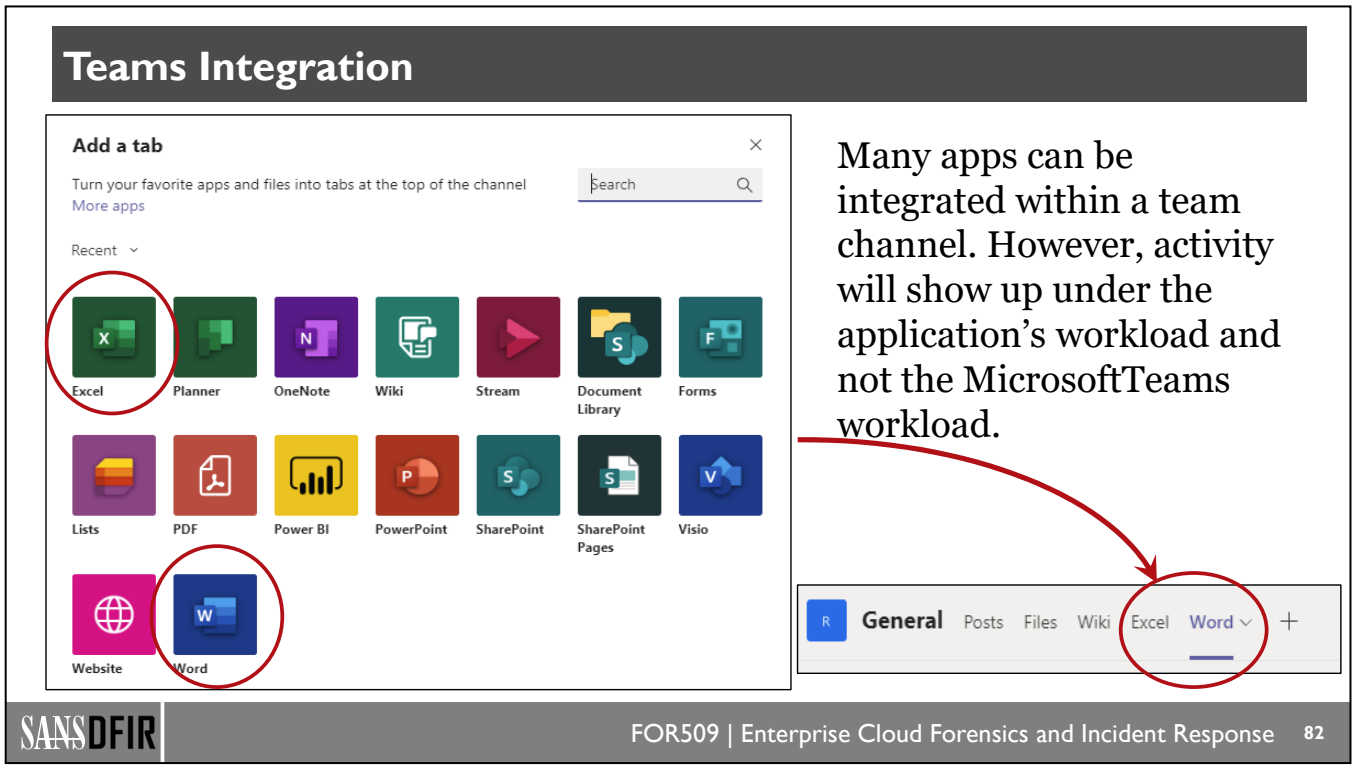
Microsoft Teams is constantly adding new features, and the latest list of operations logged is documented in the reference.[1]

In large environments, the most common operation is TeamsSessionStarted. This operation is interesting because it will provide the IP address and client string (called object_id) of the computer connected to the Teams session. Some of the client strings observed are:

Object_id	**Description**
Web (1415/1.0.0.2020061225)	Web access
Android (1416/1.0.0.2020091301)	Android mobile device
TeamsGraphService (Unknown)	Microsoft Graph API
Unknown (Unknown)	Possibly iPhone?

From a DFIR perspective, remember that Teams is frequently set to auto-start in the background when the computer boots up. As such, you may be able to get IP address information for computers even when they are not inside your corporate network. This could be very useful to track the whereabouts of a computer.

1. https://for509.com/ual-teams

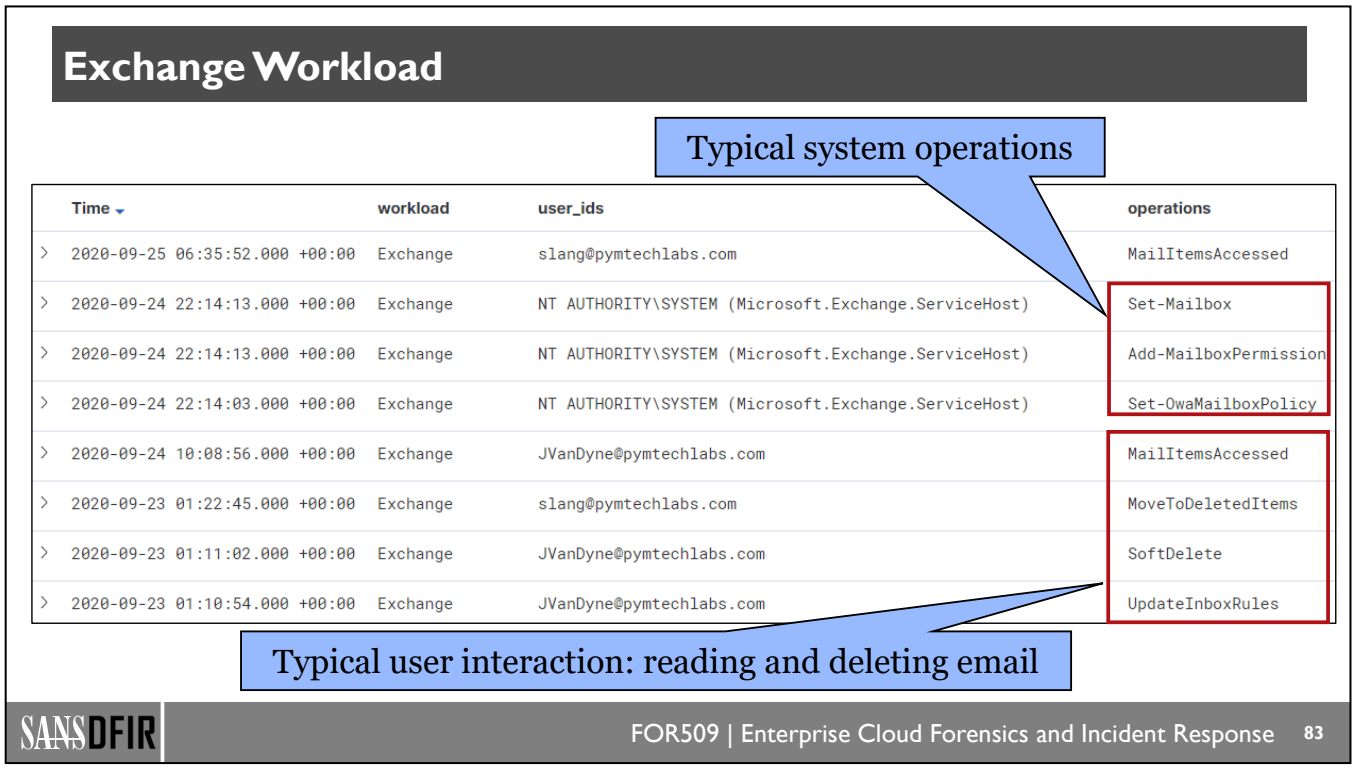


Many Microsoft 365 applications can be integrated within a team channel and will show up as a new tab in the Teams channel.[1]

One of the consequences of integration is that the log entries show up under that application's workload and not the MicrosoftTeams workload.

Don't be surprised if you interview a user and they don't realize that another Microsoft 365 application was invoked within their team's channel. From their point of view, they were "just using Teams."

1. https://for509.com/teamsintegration

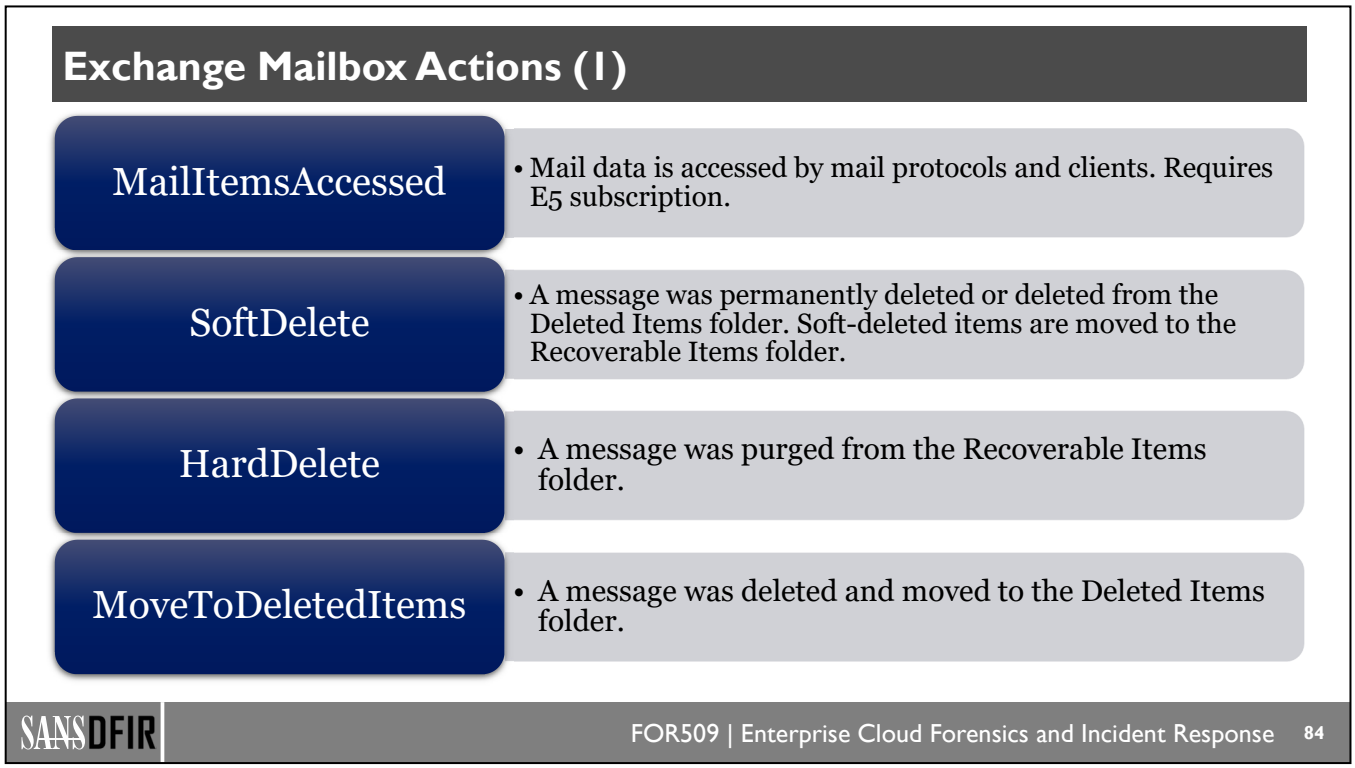


Some typical user and system Exchange operations are:

Operation	Description
MailItemsAccessed	Messages were read or accessed in mailbox. This activity is only logged for users with an E5 license.
Set-Mailbox	Change mailbox configuration parameters.
Add-MailboxPermission	Modify the permissions assigned to a mailbox.
Set-OwaMailboxPolicy	Configure OWA mailbox policies.
MoveToDeletedItems	A message was deleted and moved to the Deleted Items folder.
SoftDelete	A message was permanently deleted or deleted from the Deleted Items folder.
UpdateInboxRules	A mailbox owner modified an inbox rule in the Outlook client.

Under the SharePoint and OneDrive workloads, Microsoft 365 will log system events under the app@sharepoint userid. For the Exchange workload, the system will use the NT AUTHORITY\SYSTEM userid.

The most important operation is the MailItemsAccessed, as it will provide us detailed information about message activity. Unfortunately, that operation is only logged for mailboxes with an E5 license. In the next slides we will explore the information we can obtain from the MailItemsAccessed log entries.



There are a number of mailbox actions that can be logged. These actions may apply to user mailboxes, shared mailboxes, or group mailboxes. In addition, there are three logon types:

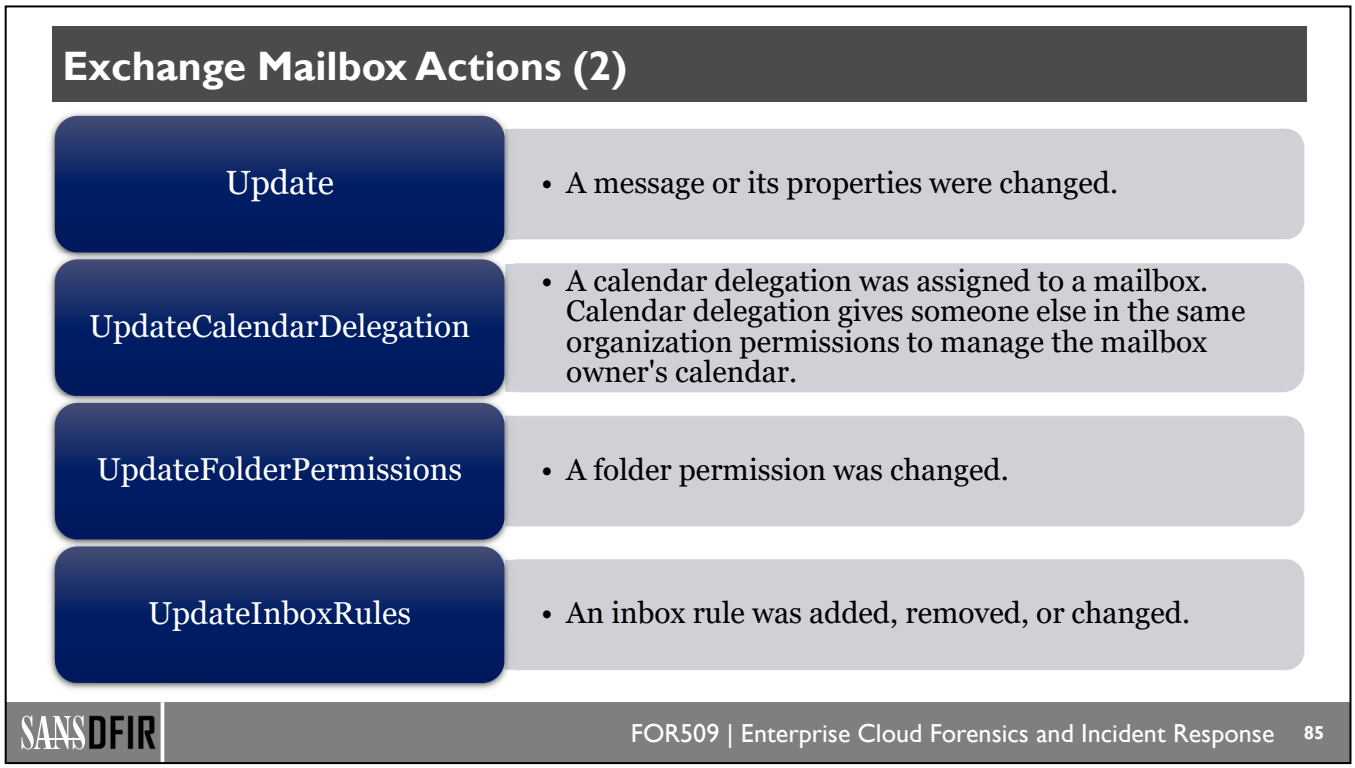
1. **Owner**: The mailbox owner.
2. **Delegate**: A user who has been assigned the SendAs, SendOnBehalf, or FullAccess permission to another mailbox.
3. **Admin**: The mailbox is accessed via the Microsoft Purview compliance portal.

There are nuances as to which mailbox action is logged for which logon type, but we will focus on the **owner** logon. These are the default actions logged for owner logon:

- **MailItemsAccessed**: Mail data is accessed by mail protocols and clients. Requires E5 subscription.
- **SoftDelete**: A message was permanently deleted or deleted from the Deleted Items folder. Soft-deleted items are moved to the Recoverable Items folder.
- **HardDelete**: A message was purged from the Recoverable Items folder.
- **MoveToDeletedItems**: A message was deleted and moved to the Deleted Items folder.

The Microsoft documentation contains a detailed table with the various possible scenarios.[1]

1. https://for509.com/mailboxauditing



Additional default actions logged for owner logon (continued from previous slide) are:

- **Update**: A message or its properties were changed.
- **UpdateCalendarDelegation**: A calendar delegation was assigned to a mailbox. Calendar delegation gives someone else in the same organization permissions to manage the mailbox owner's calendar.
- **UpdateFolderPermissions**: A folder permission was changed.
- **UpdateInboxRules**: An inbox rule was added, removed, or changed.

Exchange Mailbox Auditing

- Mailbox auditing is on by default
- To verify that mailbox auditing is on:

```
PS> Get-OrganizationConfig | Format-List AuditDisabled

AuditDisabled : False
```

- Key mailbox action to look for is "MailItemsAccessed"

```
PS> Get-Mailbox -Identity admin | Select-Object -ExpandProperty AuditOwner

Update
MoveToDeletedItems
SoftDelete
HardDelete
UpdateFolderPermissions
UpdateInboxRules
UpdateCalendarDelegation
MailItemsAccessed
```

- E5 license required to get MailItemsAccessed events

One of the most interesting workloads is Exchange. Many incidents will involve Exchange and as such it's critical to make sure the Exchange logs are enabled.

As of January 2019, the Exchange logs should be turned on by default for newly created tenants.[1] Be sure to confirm the correct configuration for tenants created prior to January 2019.

The PowerShell command `Get-OrganizationConfig | Format-List AuditDisabled` will help you confirm that audit logs are not disabled (hence, they are enabled). The double negative can be confusing: False means that auditing is turned on.

Mailbox auditing can be turned on either at the organization level or the individual mailbox level. If auditing is turned on at the organization level, then Microsoft 365 will ignore attempts to turn off auditing at the mailbox level. The only exception is if the mailbox is configured for auditing bypass. This subtlety is important to understand in case you are working a case involving an administrator abusing their privileges. Said administrator could attempt to hide their bad actions by enabling mailbox audit bypass. Hence, it's not sufficient to check mailbox auditing at the organization level.

To conduct an effective investigation involving Exchange, the key field is "MailItemsAccessed." Unfortunately, that field is only available if your tenant is licensed at the E5 level.[2]

To make sure that a mailbox is being audited for "MailItemsAccessed," use this PowerShell command:
`PS> Get-Mailbox -Identity <name of mailbox> | Select-Object -ExpandProperty AuditOwner`

1. https://for509.com/onbydefault
2. https://for509.com/advancedaudit

Mail Clients

Outlook

OWA

IMAP/POP3

Messages can be retrieved from mail servers in various ways. We have two elements to consider: the protocol and the mail client. The main email protocols are POP3 and IMAP. Examples of mail clients are Microsoft Outlook and Mozilla Thunderbird.

- Microsoft Outlook is primarily an email client that is part of the Microsoft Office suite. However, it also provides a calendar, task management, contact management, and many more features. Outlook is best known for connecting to Exchange servers. However, it also has the ability to connect to POP3 and IMAP servers.

- IMAP is the Internet Message Access Protocol, and the specifications are defined in RFC 3501.[1]
    - While POP was designed for a single user to manage a single mailbox, IMAP's design allows for the management of a mailbox by multiple email clients. To accomplish that goal, IMAP will leave email on the server unless explicitly configured otherwise.
    - The latest version is IMAP4.
    - IMAP servers listen on TCP port 143. When IMAP over SSL is used, it listens on TCP port 993.

- POP3 is Post Office Protocol version 3.
    - This protocol is rarely seen today, although it's still supported by most mail clients.
    - The Post Office Protocol was first defined in RFC 918 (POP1). In 1985, POP2 was defined in RFC 937. The most common version is POP3, which was initially defined in 1988 with RFC 1081. The current versions of POP3 is defined in RFC 1939.[2]
    - POP was originally designed to connect to a mail server, download all email, and delete them from the server. Many POP3 clients have implemented the option to leave the mail on the server.
    - POP3 servers listen on TCP port 110. When SSL is used, it listens on TCP port 995.

- OWA is an acronym for either Outlook Web Access or Outlook Web App. It permits access to your mailbox via the web and removes the need for the desktop client. With the introduction of Microsoft 365, OWA is now part of the entire suite of Microsoft applications and is accessed via the office365.com portal.

Why the history lesson? Because the log entries are going to show different pieces of information depending on the mail client used to access the mailbox. In addition, threat actors love to exploit older protocols that are frequently less secure.

1. https://for509.com/rfc3501
2. https://for509.com/rfc1939

MailItemsAccess: Sync vs. Bind Access

- Sync: Outlook
  - Audit event only includes the **mail folder** being synced
- Bind: OWA, IMAP/POP3
  - Audit event includes each **individual email** message
  - All Bind operations within a 2-min interval are aggregated in a single audit record
- Throttling
  - Audit records will no longer be generated if more than 1,000 MailItemsAccessed in 24 hours
  - This is per mailbox
  - Only applies to Bind operations
  - OperationProperties will show a value of True under the key "IsThrottled"

There are two types of mailbox access: Sync and Bind.

Sync access is used by Outlook, where entire folders are synced between Exchange in the cloud and Outlook on the desktop. In this case, we will not get information about individual email messages.
Bind access is used by web clients such as OWA, IMAP, and POP3. Each email is recorded in the audit log. In rare cases, if more than 1,000 MailItemsAccessed audit records are generated in less than 24 hours, Exchange Online will stop generating auditing records for MailItemsAccessed activity.

Email messages that were accessed are identified by their **internet message Id**.

Each log entry will have a number of interesting fields (they are subfields of the Auditdata field):

Property	Description
MailAccessType	Whether the access is a Bind or a Sync operation.
ClientInfoString	Describes protocol, client (includes version).
ClientIPAddress	IP address of the client machine.
SessionId	Session ID helps to differentiate attacker actions vs. day-to-day user activities on the same account (in the case of a compromised account).
UserId	UPN of the user reading the message.
ParentFolder	The full folder path of the mail item that was accessed.
Logon_type	The logon type of the user who performed the action. The logon types (and their corresponding Enum value) are Owner (0), Admin (1), and Delegate (2).
MailboxUPN	The UPN of the mailbox where the message being read is located.

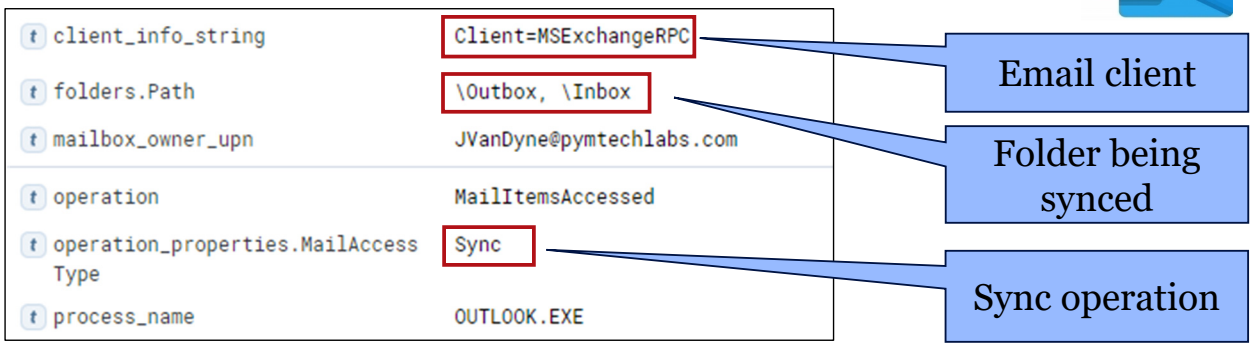
UPN stands for User Principal Name. For example, admin@pymtechlabs.com is a UPN where admin is the username and pymtechlabs.com is the domain.

ClientInfoString can provide some unique information about the type of device used to access the mail server. Rachel Moorehead has compiled an interesting list on GitHub,[1] as shown on the next page.

1. https://for509.com/clientinfostring

```
Outlook 2011  Client=WebServices;UserAgent=MacOutlook/14.3.2.130206 (Intel Mac OS X 10.8.3)
Outlook MAPI  Client=WebServices;UserAgent=Microsoft Office/14.0 (Windows NT 6.1; Microsoft Outlook 14.0.6129; Pro)
Thunderbird?  Client=WebServices;UserAgent=ExchangeWebServicesProxy/CrossSite/EXCH/14.16.0287.008/Mozilla/4.0
MacMail       Client=WebServices;UserAgent=ExchangeWebServicesProxy/CrossSite/EXCH/14.15.0129.007/Mac OS X/10.6.8
              Client=WebServices;UserAgent=Mac OS X/10.8.2 (12C2034); ExchangeWebServices/3.0 (157); Mail/6.2 (1499)
              Client=WebServices;UserAgent=Mac OS X/10.8.3 (12D78); ExchangeWebServices/3.0 (157); Mail/6.3 (1503)
iPhone 3      Client=ActiveSync;UserAgent=Apple-iPhone3C3/1002.329;Action=/Microsoft-Server-ActiveSync/default.eas
              Client=ActiveSync;UserAgent=Apple-iPhone3C1/1001.523;Action=/Microsoft-Server-ActiveSync/default.eas
iPhone 4      Client=ActiveSync;UserAgent=Apple-iPhone4C1/1002.142;Action=/Microsoft-Server-ActiveSync/default.eas
iPhone 5      Client=ActiveSync;UserAgent=Apple-iPhone5C1/1002.143;Action=/Microsoft-Server-ActiveSync/default.eas
              Client=ActiveSync;UserAgent=Apple-iPhone5C2/1002.329;Action=/Microsoft-Server-ActiveSync/default.eas
iPad 1        Client=ActiveSync;UserAgent=Apple-iPad1C1/902.206;Action=/Microsoft-Server-ActiveSync/default.eas
iPad 2        Client=ActiveSync;UserAgent=Apple-iPad2C1/1002.146;Action=/Microsoft-Server-ActiveSync/default.eas
              Client=ActiveSync;UserAgent=Apple-iPad2C2/1002.329;Action=/Microsoft-Server-ActiveSync/default.eas
              Client=ActiveSync;UserAgent=Apple-iPad2C5/1002.329;Action=/Microsoft-Server-ActiveSync/default.eas
Android       Client=ActiveSync;UserAgent=Android/4.2.1-EAS-1.3;Action=/Microsoft-Server-ActiveSync/default.eas
              Client=ActiveSync;UserAgent=Android/4.2.2-EAS-1.3;Action=/Microsoft-Server-ActiveSync/default.eas
              Client=ActiveSync;UserAgent=Android-EAS/0.1;Action=/Microsoft-Server-ActiveSync/default.eas
TouchDown     Client=ActiveSync;UserAgent=TouchDown(MSRPC)/8.1.00020/;Action=/Microsoft-Server-ActiveSync/default.eas
              Client=ActiveSync;UserAgent=TouchDown(MSRPC)/7.2.00016/;Action=/Microsoft-Server-ActiveSync/default.eas
DroidRazr     Client=ActiveSync;UserAgent=motorola-DROIDRAZR/1.0;Action=/Microsoft-Server-ActiveSync/default.eas
Samsung       Client=ActiveSync;UserAgent=SAMSUNGSCHS720C/2.3.6-EAS-1.2;Action=/Microsoft-Server-ActiveSync/default.eas
Outlook RPC   Client=MSExchangeRPC
Passive DAG   Client=CI (Content Indexing)
OWA           Client=OWA;Action=ViaProxy
              Client=WebServices;UserAgent=OwaProxy
IMAP          Client=IMAP4
```

Microsoft Outlook Sync Log Entry



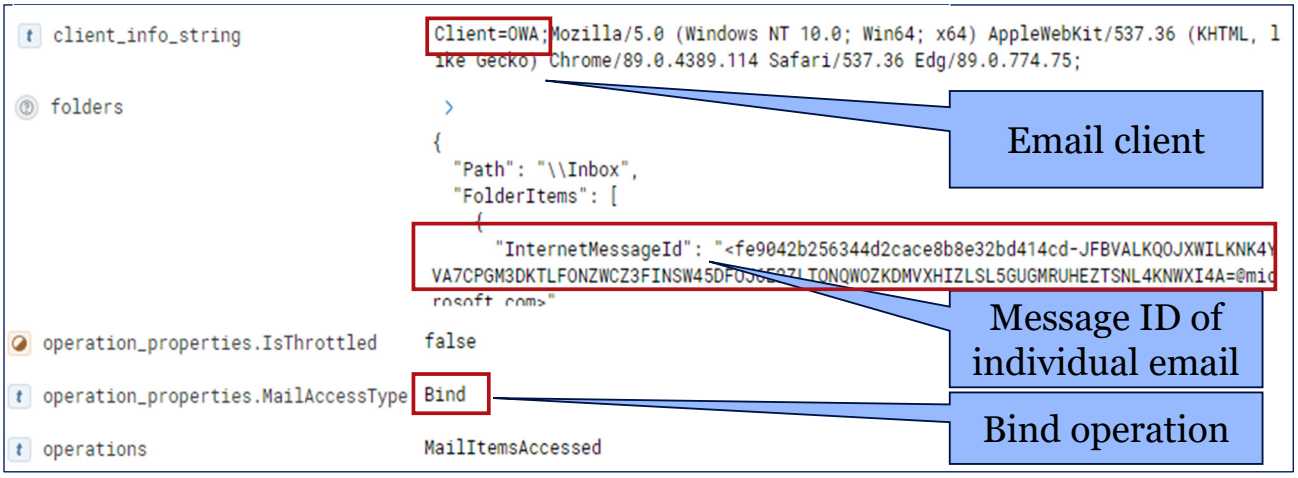
Accessing mailbox via Outlook will only show the folder being updated.

Users will normally have multiple folders in their mailbox. In this example, the Outbox and `Inbox` folders are being synced between the Exchange server and the Outlook client. Since the Sync operation is at the folder level, the log entry only contains the name of the folders being synced and not the individual email message information.

Our investigation is impacted by this sync behavior, as we won't be able to prove which individual messages were read solely based on the UAL. Other forensic techniques will need to be used.

OWA Bind Log Entry

Accessing mailbox via OWA will show the name of the folder and the message ID (InternetMessageID)



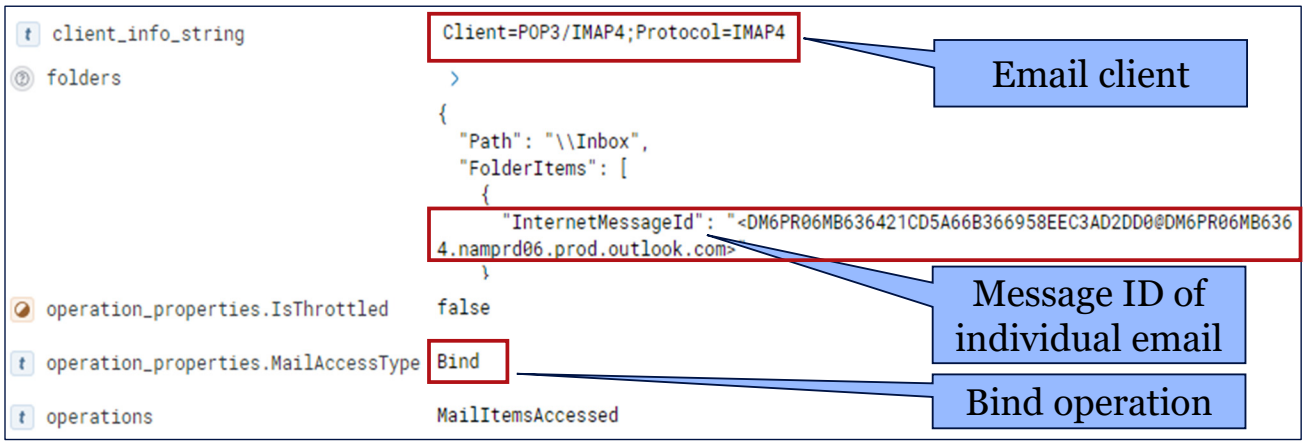
Many users will access their mailbox with Outlook Web Access (OWA). OWA records a Bind operation for each email viewed by the user.

This is very advantageous for our investigations, as the logs will contain a unique message ID (InternetMessageId[1]) for each email.

1. https://for509.com/internetmessageid

POP3/IMAP4 Bind Log Entry

Accessing mailbox via IMAP will show the name of the folder and the message ID (InternetMessageId)



This example shows the Thunderbird email client downloading a single email using the IMAP protocol. IMAP (and POP3) records a Bind operation for each email viewed by the user.

Since POP3/IMAP4 are becoming deprecated, you may want to inquire if the company authorizes the use of POP3/IMAP. Many don't, so this kind of access may be very suspicious.

ForwardingSMTPAddress

- Auto-forward email messages with `ForwardingSMTPAddress`
- Also watch for `DeliverToMailboxAndForward` or `ForwardingAddress` rules
- Solution: block at domain level:

```
PS> Set-AutoForwardEnabled $false

AutoForwardEnabled : False
```

- Forwarded email messages are dropped, even if forward is set by the user at the mailbox level

MESSAGE EVENTS

DATE (UTC)	EVENT	DETAIL
7/16/2020 3:08:04 PM	Receive	Message received by: HE1PR0402MB3354
7/16/2020 3:08:04 PM	Receive	Message received by: HE1PR04MB3113 using TLS1.2 with AES256
7/16/2020 3:08:05 PM	Submit	The message was submitted.
7/16/2020 3:08:05 PM	Drop	Reason: [{LED=250 2.1.5 RESOLVER.MSGTYPE.AF; handled AutoForward addressed to external recipient};{MSG=};{FQDN=};{IP=};{LRT=}]
Reason: [{LED=250 2.1.5 RESOLVER.MSGTYPE.AF; handled AutoForward addressed to external recipient};{MSG=};{FQDN=};{IP=};{LRT=}]		
7/16/2020 3:08:05 PM	Spam Diagnostics	

Once a mailbox has been compromised, bad actors will frequently set a mail-forwarding rule in order to get a copy of each email. This is a discreet way to obtain information that can be later used against the company. This is commonly used to commit financial fraud and is often referred to as business email compromise (BEC). ForwardingAddress and ForwardingSMTPAddress are two potential methods to forward email.[1]

The best solution is to disable auto-forwarding at the domain level with the PowerShell command: `Set-AutoForwardEnabled $false`

This will prevent email from being forwarded even if a user sets a forwarding rule at the mailbox level.

1. https://for509.com/forwardingaddress

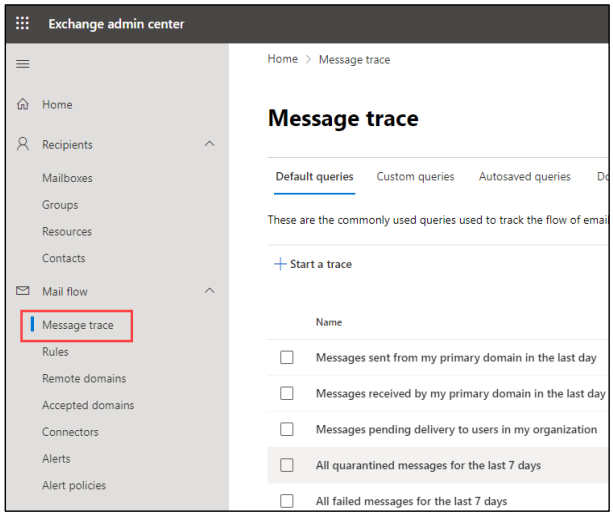
MESSAGE EVENTS

DATE (UTC)	EVENT	DETAIL
7/16/2020 3:08:04 PM	Receive	Message received by: HE1PR0402MB3354
7/16/2020 3:08:04 PM	Receive	Message received by: HE1PR04MB3113 using TLS1.2 with AES256
7/16/2020 3:08:05 PM	Submit	The message was submitted.
7/16/2020 3:08:05 PM	Drop	Reason: [{LED=250 2.1.5 RESOLVER.MSGTYPE.AF; handled AutoForward addressed to external recipient};{MSG=};{FQDN=};{IP=};{LRT=}]
Reason: [{LED=250 2.1.5 RESOLVER.MSGTYPE.AF; handled AutoForward addressed to external recipient};{MSG=};{FQDN=};{IP=};{LRT=}]		
7/16/2020 3:08:05 PM	Spam Diagnostics	

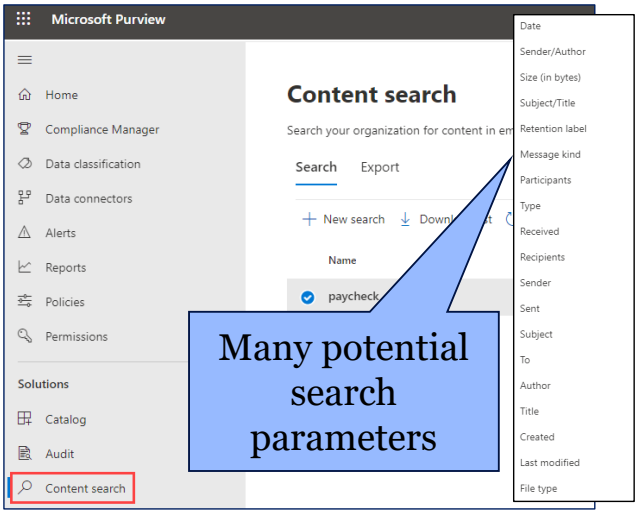
Message Tracing

The UAL isn't able to provide the necessary details for our investigation, we have two options:

Option 1: Exchange Admin center



Option 2: Microsoft Purview compliance portal



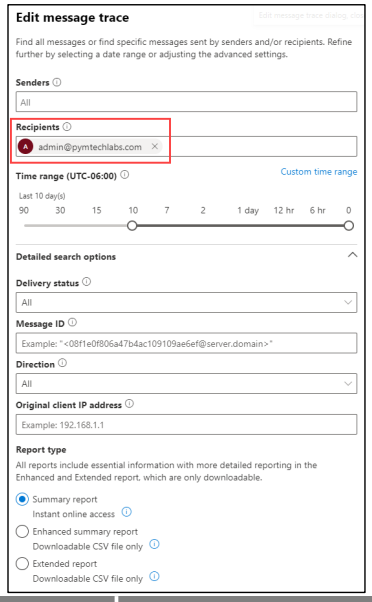
When you are investigating a suspicious email, you have two options to get more information about that email:

1. The Exchange admin center will allow you to search for email based on sender, recipient, or message ID.
2. The Microsoft Purview compliance portal will allow you to search for email based on keywords or many other parameters.

The advantage of the Exchange admin center is the ability to see details about the processing of the email as well as its final disposition (delivered, quarantine, etc.).

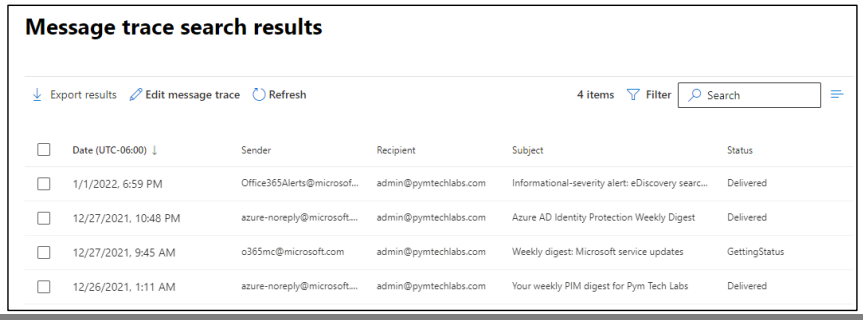
On the other hand, the Microsoft Purview compliance portal has a lot more search options. It can even display a preview of the message which is not available in the Exchange admin center.

Tracking Suspicious Email: Exchange Admin Center



Option 1: Exchange Admin Center

- Search for Senders or Recipients
- Receive list of email to then identify suspicious email
- Once identified, leverage "Message ID" to track other recipients and message status (delivered, quarantined, etc.)
- This option is limited to 10 days for immediate search and 90 days for historical search (slow)

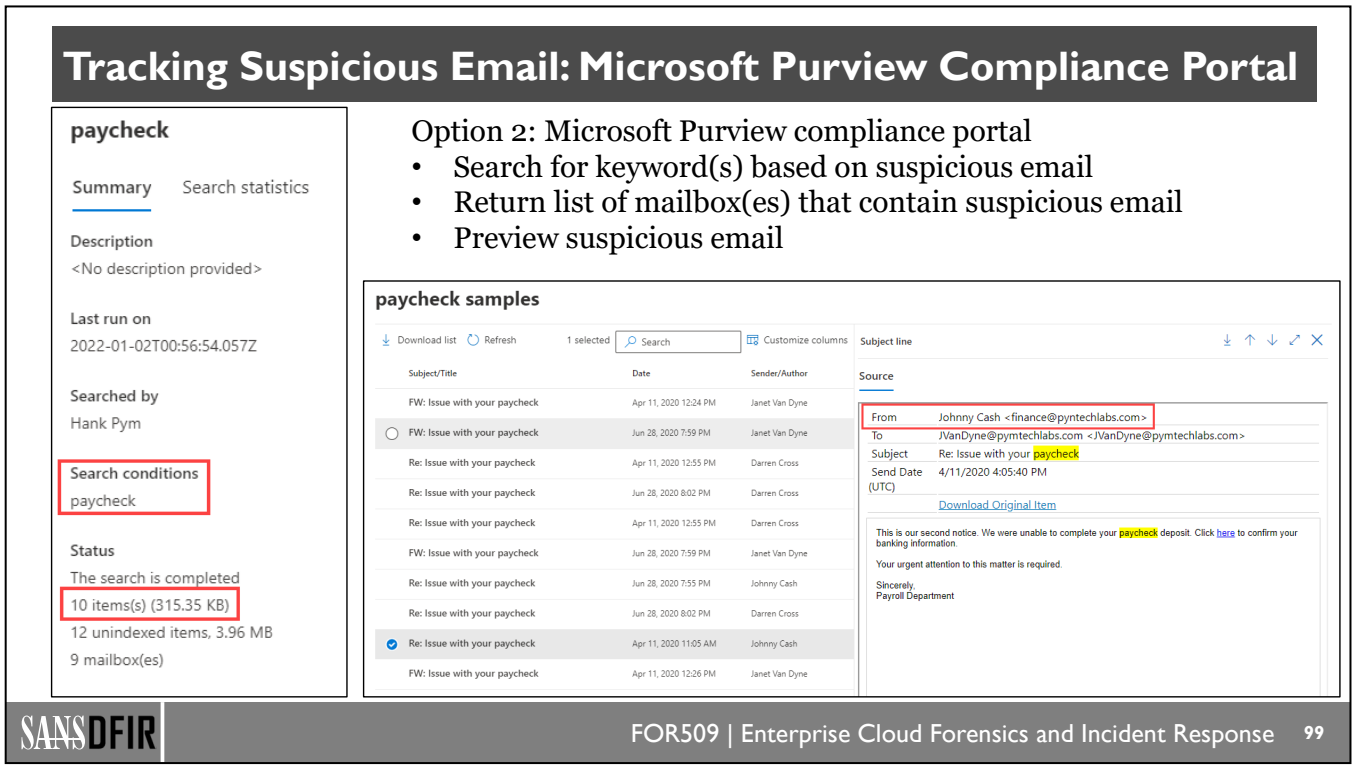


In the Exchange admin center, you can search for email based on sender, recipient, or message ID. If you limit your search to the last 10 days (or less), the results will be immediately available. For any time period greater than 10 days (with a maximum of 90 days), the search will be scheduled, and you will eventually be notified that a report is available for you to download.

The advantage of the Exchange admin center is that you will get a lot of details about the various events affecting the delivery of the email. For instance, you may see that an email was delivered, that it failed to be delivered, that it was quarantined, etc.

You can also use PowerShell to obtain this information with the cmdlet `Get-MessageTrace` for time frames of 10 days or less, or the cmdlet `Start-HistoricalSearch` for longer time frames. The cmdlet `Get-MessageTraceDetail` will provide the detailed information mentioned in the last paragraph.

A limitation of the Exchange admin center is that it won't show the contents of the email. For that, we need to use the Microsoft Purview compliance portal.



The Microsoft Purview compliance portal has numerous features. To search for email, we use the "Content Search" feature.

The first step is to create a new search. You will then specify the locations to search. You can search Exchange mailboxes as well as SharePoint sites. In a large organization, we highly recommend being as specific as possible. These searches can take a very long time to complete.

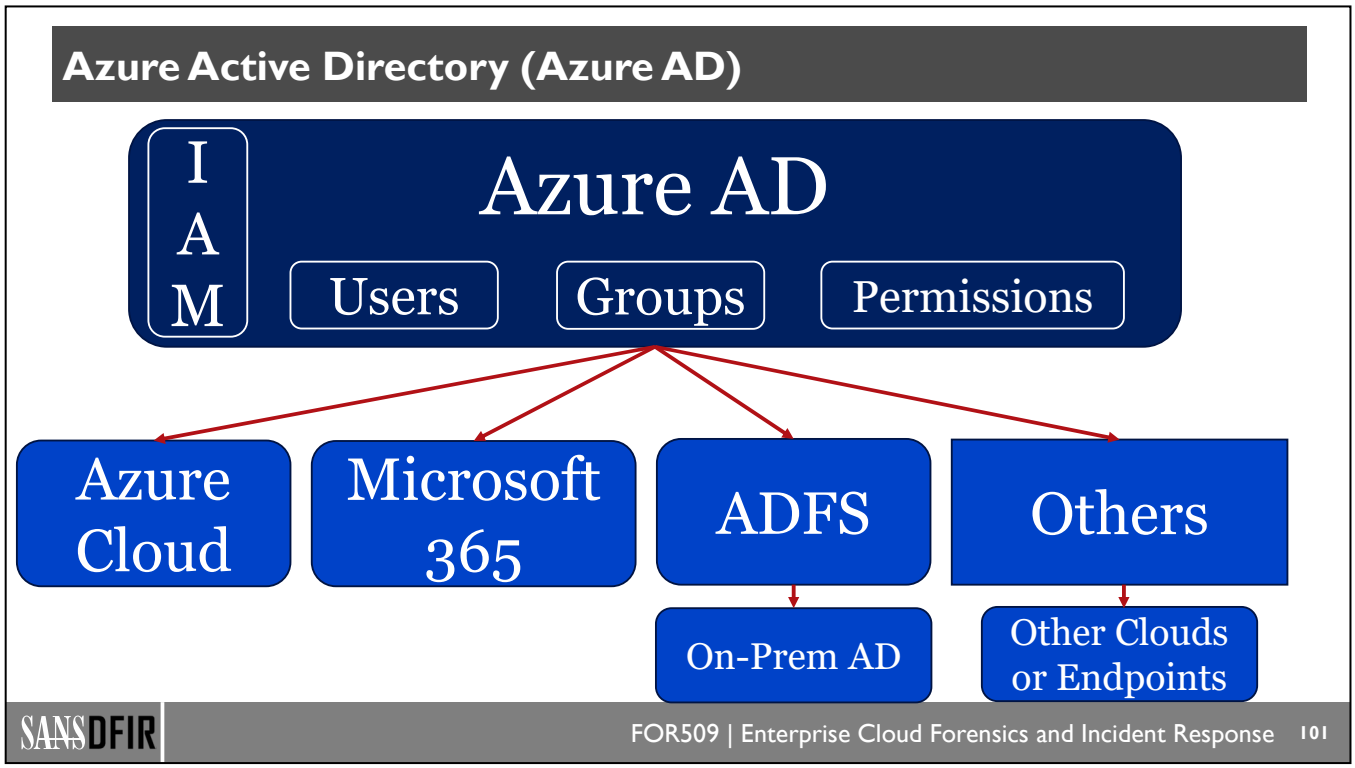
At this point you will get to define the search conditions. It can be as simple as searching for a keyword or as complex as you would like it to be. You can even use KQL (Keyword Query Language) to create your search. The example shown in the slide searches for the word "paycheck" based on the complaint received from the user. Any intelligence you can leverage from your investigation will make your searches more efficient.

If your search returns valid results, you will be able to review a sample and download these results. The more specific the search the better, or you may be looking at downloading gigabytes of data.

Business Email Compromise Resources

- BECs frequently start with a phishing email
- External references for more information
  - Microsoft incident response playbook for phishing
    https://for509.com/phishing-playbook
  - PwC's Business Email Compromise Guide
    https://for509.com/pwcir-bec
  - Phill Moore's Awesome-BEC collection of resources
    https://for509.com/awesome-bec

This page intentionally left blank.



Azure Active Directory[1] is Microsoft's Identity and Access Management (IAM) solution. For each tenant there is a dedicated Azure Active Directory (also called AAD or Azure AD) instance. Azure AD is critical, as it manages users, groups, and permissions for all applications. In this regard, Azure and Microsoft 365 are considered applications.

Azure AD tenants are globally unique and use the domain "onmicrosoft.com". So, for the Pymtechlabs tenant, the Azure AD tenant is pymtechlabs.onmicrosoft.com.

Users can only belong to a single tenant. They may be guests of other tenants.

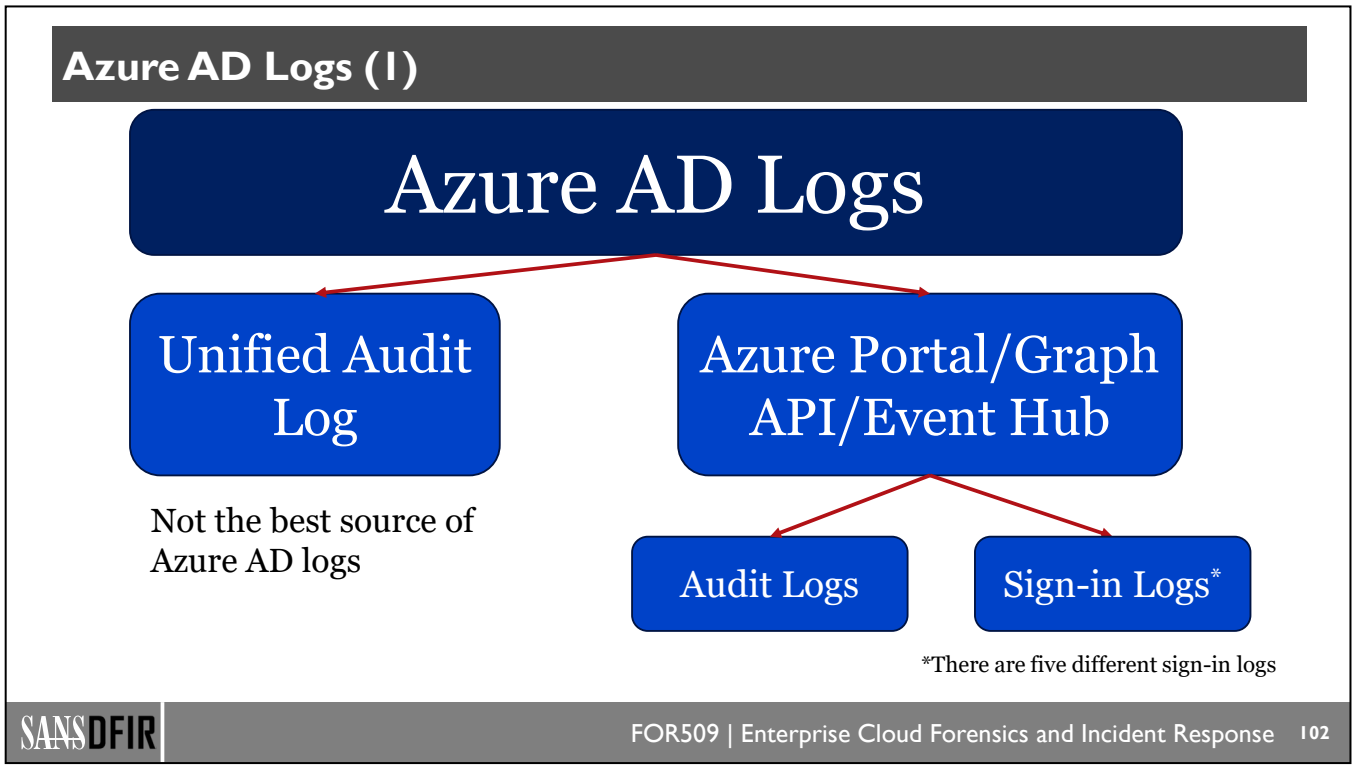
Azure AD is used to manage authentication not only from Microsoft 365 applications but also from Microsoft Azure. There is a trust relationship between the Azure subscription and Azure AD.

Azure AD can also leverage Active Directory Federation Services (ADFS) to authenticate users in hybrid environments. Further, Azure AD can provide authentication for other cloud providers or endpoints that may not be part of a legacy on-prem Active Directory forest.

Another key feature of Azure AD is multifactor authentication (MFA). Azure AD supports different MFA methods, and in today's world there is no excuse for not turning on MFA. A more advanced feature is conditional access which can take into account a number of additional factors before deciding to grant access. For example, one such factor is whether the endpoint is managed or not.

For our purposes, we will look at Azure AD's log entries to understand who logged in and what changes may have been made to Azure AD or the tenant.

1. https://for509.com/aad-intro



Azure Active Directory (Azure AD) logs[1] will be written to both the Unified Audit Log (UAL) and to the Azure portal. In the Azure portal, the logs can be seen in the console, sent to an analytics workspace, written to a storage account, sent to an event hub, or accessed via Graph API. We will discuss these various options in detail in the Azure module.

While the Azure AD logs are written to the UAL, the accuracy has been found to be less than desirable. Further, the limitations in exporting the UAL makes it the least desirable location to acquire these logs.

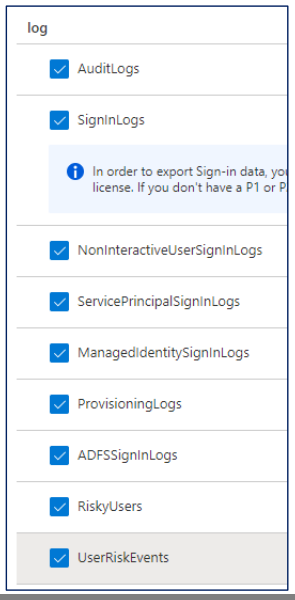
On the other hand, in the Azure portal, you will find two types of Azure AD logs:

1. The audit logs
2. Five different kinds of sign-in logs:
    1. **Sign-in log**: This log will show interactive user sign-ins.
    2. **Non-interactive sign-in log**: These are sign-ins performed by a client app or operating system components on behalf of a user. These sign-ins will use a token rather than an authentication factor such as a password.
    3. **Service principal sign-in log**: These are sign-ins by non-user user account, mainly Graph API applications.
    4. **Managed identity sign-in log**: These are sign-ins that are performed by resources that have their secret managed by Azure.
    5. **ADFS sign-in log**: Sign-ins via Active Directory Federation Services.

1. https://for509.com/sign-in-logs

Azure AD Logs (2)

- Azure AD logs are platform logs
- Two main pillars:
  - Security logs: protect identity
  - Activity logs: understand behavior
- Security Logs:
  - RiskyUsers & UserRiskEvents
  - Identify potentially compromised accounts
- Activity Logs:
  - Audit logs: history of tasks performed at the tenant level
  - Sign-ins: log of who performed the tasks
  - Sign-in logs are subdivided based on the type of sign-in



Azure AD logs[1] are platform logs that provide detailed information about sign-in activity and changes made to Azure AD. There are two categories of Azure AD logs:

1. Security logs, which help identify potentially risky users
2. Activity logs, which help understand your users' behavior

Activity logs are further subdivided into:

1. Audit logs, which provide a history of every task performed at the tenant level
2. Sign-in logs, which show who performed the tasks

Sign-in logs are recorded in five different files to differentiate the type of sign-in.

You should turn on every type of sign-in log, even if you believe that you don't use a certain type of sign-in, to prevent creating a blind spot that may be exploited by a threat actor.

Audit logs are very important, as they will show tasks such as application, user, and service principal creation and modification. These are key activities performed by threat actors once they gain access to your tenant.

1. https://for509.com/aad-logs

Azure AD Log Example

Example of a user login

```
@timestamp 2020-04-11 17:55:29.000 +00:00
workload   AzureActiveDirectory
operation  UserLoggedIn
client_ip  104.238.59.218
user_ids   dcross@pymtechlabs.com
useragent  Mozilla/5.0 (Windows NT 10.0;
Win64; x64) AppleWebKit/537.36 (KHTML,
like Gecko) Chrome/80.0.3987.163
Safari/537.36
```

- For regular login the "useragent" can be helpful in certain investigations (example: password spraying)

Example of a system-generated AAD event

```
@timestamp  2020-04-11 17:40:37.000 +00:00
workload    AzureActiveDirectory
object_id   dcross@pymtechlabs.com
operation   Add member to group.
client_ip   40.126.6.52
modified_properties     {
  "Name": "AccountEnabled",
  "OldValue": "[]",
  "NewValue": "[\r\n  true\r\n]"},
result_status           Failure

user_ids    ServicePrincipal_87b15a38-add8-47ec-aaff-
0a98e8b42edb
```

- IP address belongs to Microsoft
- The UserID is a service principal
- The attempt to add dcross to a group failed

In a large environment, Azure AD can be very verbose. Regular user login entries are straightforward. Others, not so much.

The example on the left shows a regular user login. As expected, the workload is from AzureActiveDirectory since Azure AD performs the authentication. The other interesting fields are the `operation` and `user_ids` fields, which indicate which user is authenticating. The `client_ip` and `useragent` fields may also be useful fields in your investigation.

Applications like SharePoint are constantly "doing things," which creates numerous log entries. They use service principals to perform these operations. Service principals are similar to service accounts.

The example on the right shows a system-generated entry. The key fields are `object ID`, `operation`, `modified_properties`, and `result_status`. The `modified_properties` field will provide information about the change being made.

In this example, the account dcross@pymtechlabs.com was being added to a group. However, the operation failed. In this case, it failed because the account was already enabled (meaning it was already part of the group).

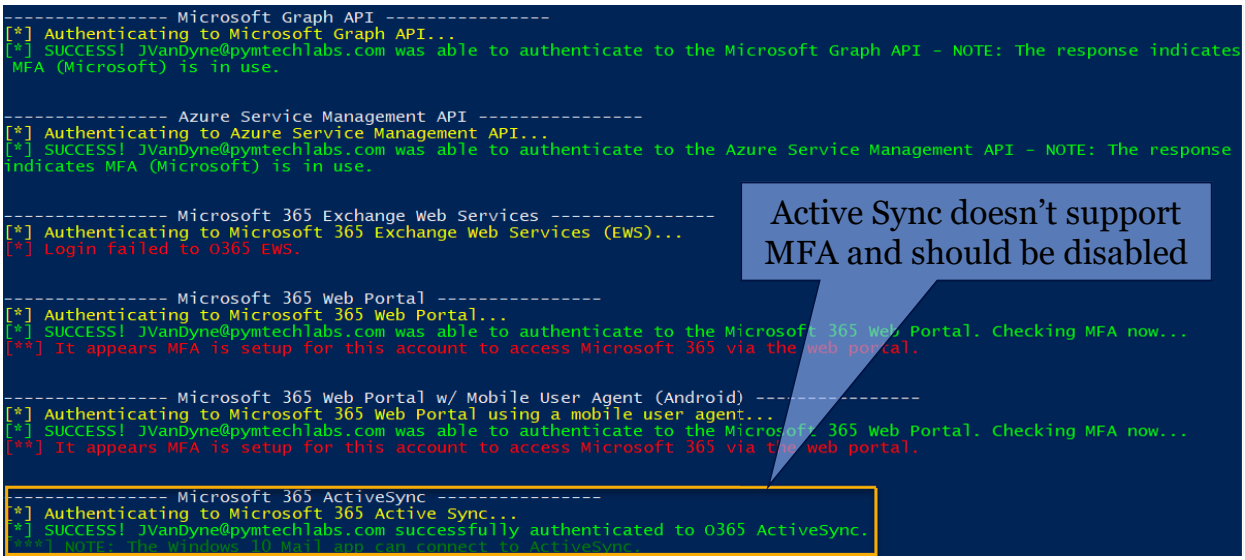
Check MFA

- To secure Azure AD, MFA must be enabled, but is it really turned on?
- So many ways to connect to Azure AD and Microsoft 365
- MFASweep script by Beau Bullock (@dafthack)
- MFASweep will attempt to login via:
  - Microsoft Graph API
  - Azure Service Management API
  - Microsoft 365 Exchange Web Services
  - Microsoft 365 Web Portal
  - Microsoft 365 Web Portal Using a Mobile User Agent
  - Microsoft 365 Active Sync
  - ADFS

There are so many ways to connect to Azure AD and Microsoft 365 that don't always enforce MFA. MFASweep[1] will attempt to log in via seven different methods to verify that MFA is enabled.

1. https://for509.com/mfasweep

MFASweep in Pymtechlabs

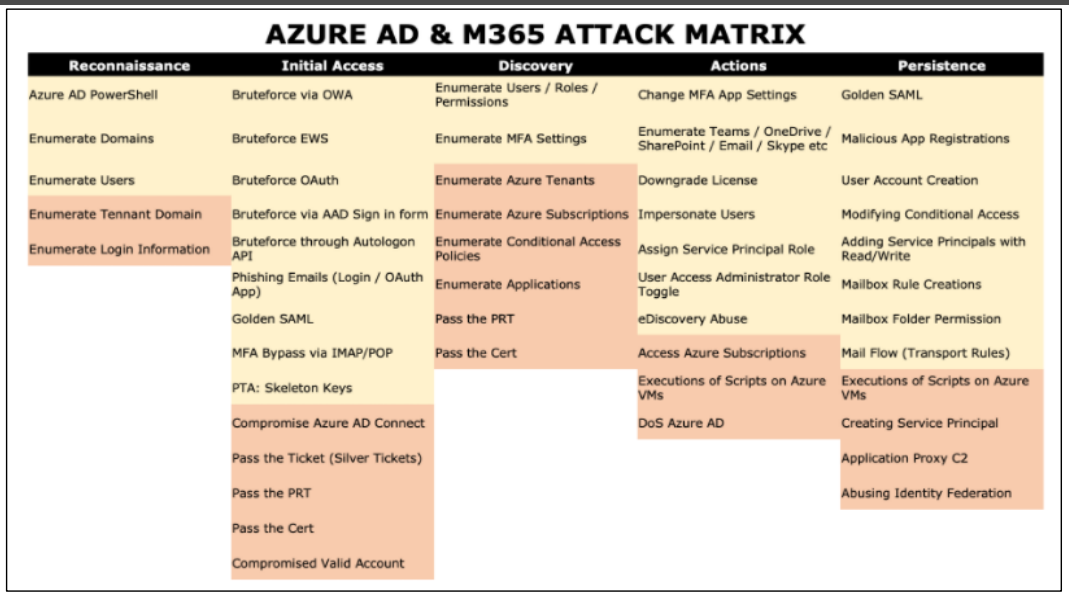


Pymtechlabs enforces MFA; however, the Active Sync protocol is still enabled. This type of "misconfiguration" is frequently exploited by threat actors to check the validity of pilfered credentials.

Microsoft highly recommends disabling Active Sync.[1]

1. https://for509.com/secure-email

Azure AD and M365 Attack Matrix



AZURE AD & M365 ATTACK MATRIX

Reconnaissance	Initial Access	Discovery	Actions	Persistence
Azure AD PowerShell	Bruteforce via OWA	Enumerate Users / Roles / Permissions	Change MFA App Settings	Golden SAML
Enumerate Domains	Bruteforce EWS	Enumerate MFA Settings	Enumerate Teams / OneDrive / SharePoint / Email / Skype etc	Malicious App Registrations
Enumerate Users	Bruteforce OAuth	Enumerate Azure Tenants	Downgrade License	User Account Creation
Enumerate Tennant Domain	Bruteforce via AAD Sign in form	Enumerate Azure Subscriptions	Impersonate Users	Modifying Conditional Access
Enumerate Login Information	Bruteforce through Autologon API	Enumerate Conditional Access Policies	Assign Service Principal Role	Adding Service Principals with Read/Write
	Phishing Emails (Login / OAuth App)	Enumerate Applications	User Access Administrator Role Toggle	Mailbox Rule Creations
	Golden SAML	Pass the PRT	eDiscovery Abuse	Mailbox Folder Permission
	MFA Bypass via IMAP/POP	Pass the Cert	Access Azure Subscriptions	Mail Flow (Transport Rules)
	PTA: Skeleton Keys		Executions of Scripts on Azure VMs	Executions of Scripts on Azure VMs
	Compromise Azure AD Connect		DoS Azure AD	Creating Service Principal
	Pass the Ticket (Silver Tickets)			Application Proxy C2
	Pass the PRT			Abusing Identity Federation
	Pass the Cert			
	Compromised Valid Account			

Lina Lau has done an outstanding job documenting the various attacks that a threat actor may undertake against Azure AD and Microsoft 365. She summarized her research in the "Azure AD & M365 Attack Matrix" shown in this slide.

We highly recommend reading her blog post.[1]

1. https://for509.com/inversecos

Azure AD Incident Response

Key steps to respond to Azure AD attacks:

1. Mobilize the incident response team and secure their communications.
2. Understand your normal user authentication path. Are you using ADFS, SSO, conditional access?
3. Identify and export available logs.
4. Investigate the level of access gained by the attacker.
5. Investigate the extent of the attacker activity.
6. Regain administrative control and remove all attacker access.
7. Monitor for further attacker activity and prepare to rapidly respond.
8. Improve security posture to defend against further attacks.

Source: Will Oram's AzureAD Incident Response GitHub Repo
https://for509.com/azuread-ir

Azure AD incident response is a complex topic and we highly recommend the work of Will Oram. In his GitHub repo,[1] Will provides a wealth of technical information organized around the key steps you will want to take in responding to such an incident:

1. Mobilize the incident response team and secure their communications.
2. Understand how users are authenticated and how Azure AD and Microsoft 365 are configured.
3. Identify and export available logs and configuration information.
4. Investigate the extent of the attacker activity and the access the attacker has gained to the environment.
5. Take immediate containment measures to remove access to known compromised accounts and identities.
6. Perform a more comprehensive review to identify any persistence access the attacker has gained to accounts, systems or data:
    - Hunt for modifications to the configuration of the Azure AD tenant.
    - Hunt for Golden SAML Attacks.
    - Hunt for the compromise of privileged accounts.
    - Hunt for hijacked Azure AD Applications and Service Principals.
    - Hunt for malicious modifications to Exchange Online configuration.
    - Hunt for illicit application consent attacks.
    - Hunt for the compromise of on-premises systems and accounts.
    - Hunt for the compromise of and malicious changes to Azure resources.
7. Monitor for further attacker activity and prepare to rapidly respond.
8. Regain administrative control and remove all attacker access.
9. Assess data accessed and/or exfiltrated by the attacker.
10. Improve security posture to defend against further attacks.

1. https://for509.com/azuread-ir

AzureAD Incident Response PowerShell Module

"The Azure Active Directory Incident Response PowerShell module provides a number of tools, developed by the Azure Active Directory Product Group in conjunction with the Microsoft Detection and Response Team (DART), to assist in compromise response."[1]

https://github.com/AzureAD/Azure-AD-Incident-Response-PowerShell-Module

1. https://for509.com/azuread-ir-psmodule

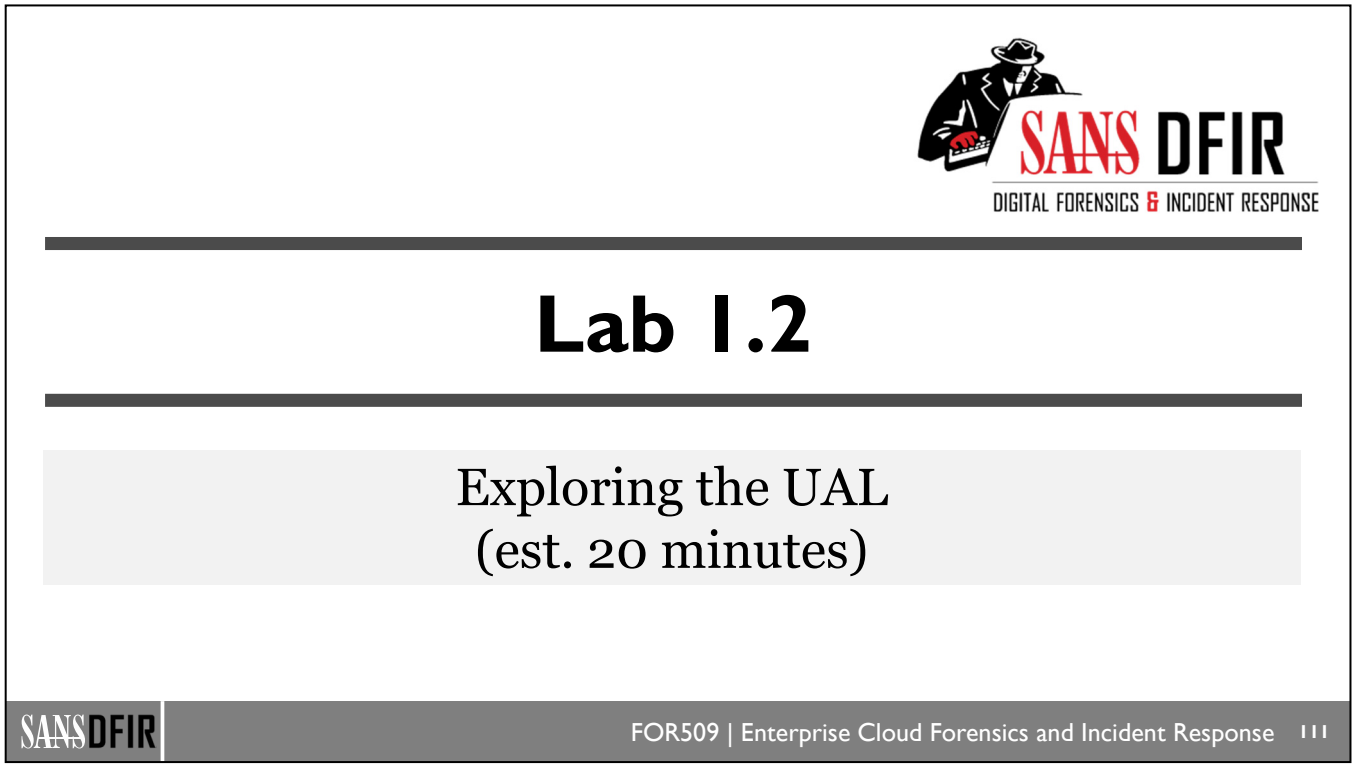
Lab 1.2 Preview

Lab 1.2 will review user activity in Pymtechlabs and search for possible unauthorized access.

The following fields will be important in this lab:

- `client_info_string`
- `client_ip`
- `ips`
- `operation`
- `parameters.ForwardingStmpAddress`
- `source_ip`
- `source_geo.country_name`
- `user_ids`

This page intentionally left blank.



This page intentionally left blank.

FOR509.1: Microsoft 365 and Graph API

Section 1.1: Introducing SOF-ELK®

Section 1.2: Key Elements of Cloud for DFIR

Section 1.3: Microsoft 365 Unified Audit Log

Section 1.4: Microsoft Graph API

This page intentionally left blank.

Microsoft 365 and Graph API Roadmap

1.1: Introducing SOF-ELK®

1.2: Key Elements of Cloud for DFIR

1.3: Microsoft 365 Unified Audit Log

1.4: Microsoft Graph API

- Case Study: SolarWinds
- Microsoft Graph API
- Graph API Process
- Five Steps to Graph API
- Example: Read Email
- Example: Create User
- Microsoft Graph Security API
- **Lab 1.3**: Privilege Escalation with Graph API

This page intentionally left blank.

Case Study: SolarWinds

- Not a cloud-based attack, but cloud used for persistence:
  - Compromised certificate added to a privileged application
  - Generated OAuth token to access resources permitted to the application
  - Application had Mail.Read or Mail.ReadWrite permission, which enabled threat actor to monitor victims' email using the Graph API
- In response to this threat, CISA released a tool called Sparrow to audit possible compromised accounts
- A tool called Hawk can assist with collecting data from Azure
- An excellent analysis of this type of threat was written by Aon's Cyber Labs in their report "Cloudy with a Chance of Persistent Email Access"

The SolarWinds breach made national news and had severe consequences. There were many steps to this breach, but for our purposes the most interesting part is the use of the Microsoft Graph API to access SolarWinds' email for an extended period of time.

As described in the Microsoft report,[1] "the actor has been observed adding credentials to one or more legitimate OAuth Applications or Service Principals, usually with existing *Mail.Read* or *Mail.ReadWrite* permissions, which grants the ability to read mail content from Exchange Online via Microsoft Graph."

At that time, this was the first known instance of a threat actor using this technique.

To assist companies in auditing their environments, CISA released a tool called Sparrow.[2] In addition, Microsoft unofficially released a tool called Hawk[3] on GitHub to facilitate acquiring data from your environment. Both tools will gather a large amount of information that must be analyzed by a human.

For reference, CISA is the United States' Cybersecurity & Infrastructure Security Agency.

Will Oram has published a great article regarding Azure AD Incident Response on his GitHub.[4] His work provides technical information that can help detect potential Azure AD compromises.

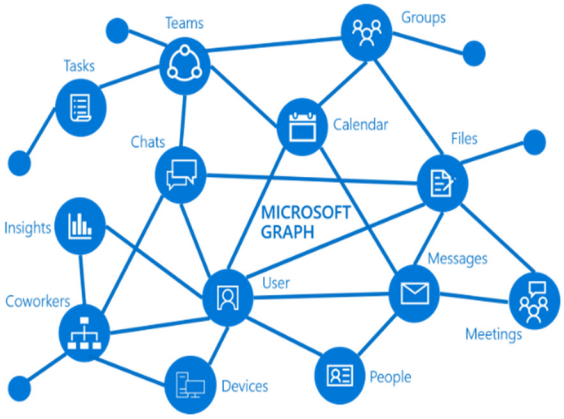
Aon's Cyber Labs has written an excellent report about this type of threats. We highly recommend reading "Cloudy with a Chance of Persistent Email Access".[5]

In this section, we explain the basics of the Microsoft Graph API and review the logs that may exist.

1. https://for509.com/microsoft-solarwinds
2. https://for509.com/sparrow
3. https://for509.com/hawk
4. https://for509.com/willoram
5. https://for509.com/aon-report

Microsoft Graph API

- Unified programming model to access data in Microsoft 365 cloud
- Single access endpoint: https://graph.microsoft.com
- RESTful APIs with support for languages such as:
  - PowerShell
  - PHP
  - Python
  - and many more
- Output structured data



The Microsoft Graph API[1] provides a unified programming model to access resources in Microsoft 365 and Azure. This means that everything you can do with the console, PowerShell, or the command line interface (CLI), you can also do by making a RESTful call to `https://graph.microsoft.com`. As a matter of fact, you can do many more things with the Graph API.

In this section, we will look at two examples. In the first example we will read information (email), and in the second example we will create a new user.

The important question for us as incident responders is, what kind of logs are generated from these actions? While the two examples we selected will have corresponding logs, many read actions do not generate any logs.

Microsoft maintains a GitHub repository with documentation and other resources dedicated to the Graph API.[2]

1. https://for509.com/graph-api
2. https://for509.com/graph-api-github

REST API

- Microsoft Graph is a RESTful API that allows you to interface directly with the Microsoft 365 cloud
- Simple API calls are made using http methods:
  - GET is used to fetch data from Microsoft Graph
  - POST is used to create a new entity
  - PATCH is used to make changes to an entity
  - DELETE is used to remove an entity

Action (Example)	HTTP Request
List of users	GET /users
Create user	POST /users
Update user properties	PATCH /users/{id | userPrincipalName}
Delete user	DELETE /users/{id | userPrincipalName}

Entities Examples

- Users
- Files
- SharePoint sites
- Email
- Calendar
- Team chats
- 100s more

REST is an acronym for **RE**presentational **S**tate **T**ransfer. Microsoft Graph API supports these methods:

- GET: Used to fetch data from Microsoft Graph
- POST: Used to create a new entity
- PUT: Used to modify an entity with existing data (not used a lot in Graph API, hence omitted from table above)
- PATCH: Used to change an entity (note that you only need to specify new or changed information)
- DELETE: Used to remove an entity

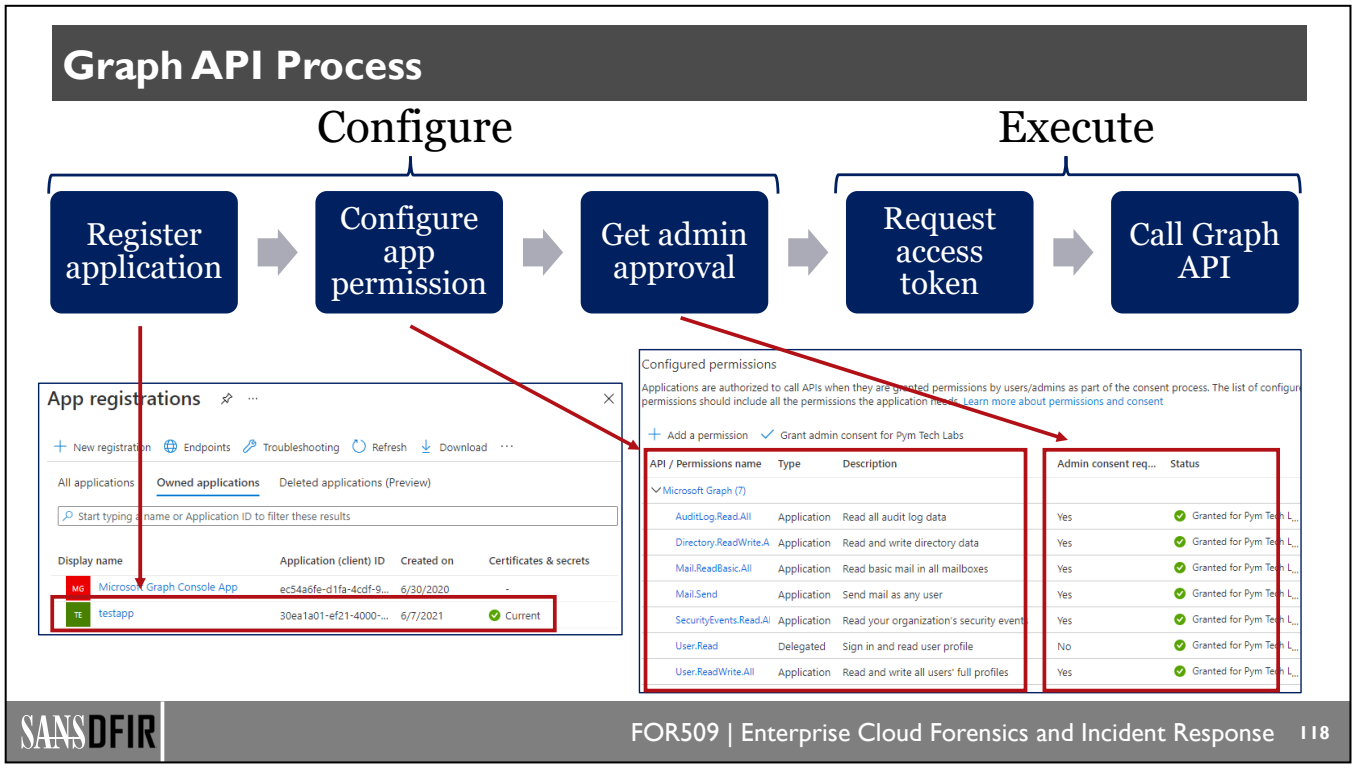
The format of a query is:

{**method**} https://graph.microsoft.com/{**version**}/{**resource**}?{**query-parameters**}

- Method: GET, POST, PATCH, DELETE, as explained above
- Version: v1.0 (production) or beta (preview with new features)
- Resource: Entity that is being queried
- Query-parameters: Optional, depending on the query

For example, to see security alerts in your tenant, you could write the following query:
GET https://graph.microsoft.com/v1.0/security/alerts

There are hundreds (possibly thousands) of entities that can be addressed with the Graph API. The Microsoft documentation shows which method to use for each action.



To use the Microsoft Graph API, you will need to follow these five steps:

1. Register the application in Azure AD.
2. Configure app permission.
3. Get Microsoft 365 global admin approval.
4. Request access tokens from Azure AD.
5. Call the Graph API.

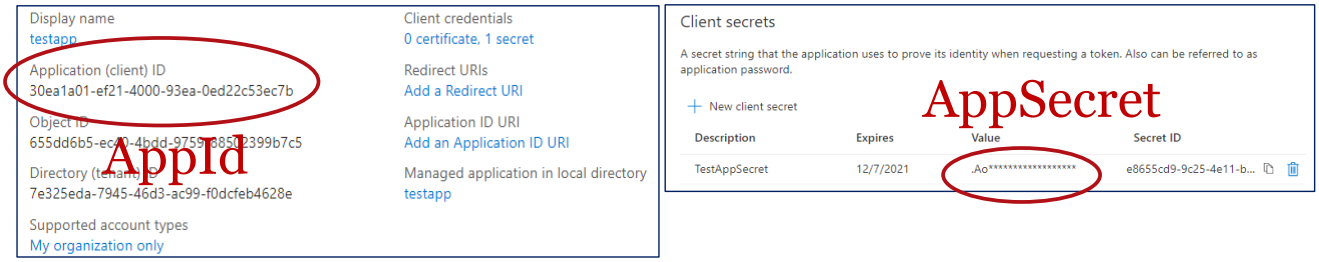
Steps 1 through 3 will configure the application, create a service principal, assign the appropriate permissions, and, if necessary, obtain consent from the global admin. There are a lot of nuances regarding API permissions. We will limit ourselves to a high-level discussion in the next few slides.

In steps 4 and 5, an access token is requested (based on the predefined client secret) and one of the many hundreds of Graph API functions can now be called.

We will now examine each step.

Step 1: Register App

Azure Console: App registrations



PowerShell code

```
$AppId = '30ea1a01-ef21-4000-93ea-0ed22c53ec7b'
$AppSecret = '.Ao8A_ZP.nNGC30n2.xxxxxxxxxxxxx'
$Scope = "https://graph.microsoft.com/.default"
$TenantName = "pymtechlabs.com"
$Url = "https://login.microsoftonline.com/$TenantName/oauth2/v2.0/token"
```


The first step is to register a new application in Azure Active Directory. This step limits who is able to access your tenant via the Graph API.

When you register the application, you can create a client secret, which will be needed for authentication. Be sure to keep that client secret well protected, as this is highly coveted by threat actors. You also have the option to use a certificate instead of a secret.

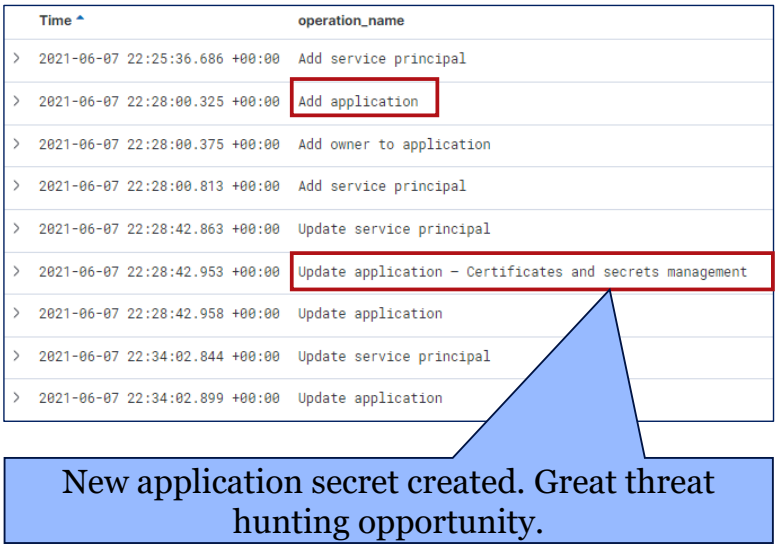
To start our script, we define a number of variables with our AppId and our AppSecret. Using these two pieces of information, we will be able to request an access token.

Remember the AppId because we will see it again and again in the logs—it's the critical piece of information we need in order to track what this app in doing in our environment.

Register App Logs

When we register an app, many tasks are automatically executed. These can be seen in the logs:

- Add a service principal
- Add application
- Create a client secret
- Global admin consent



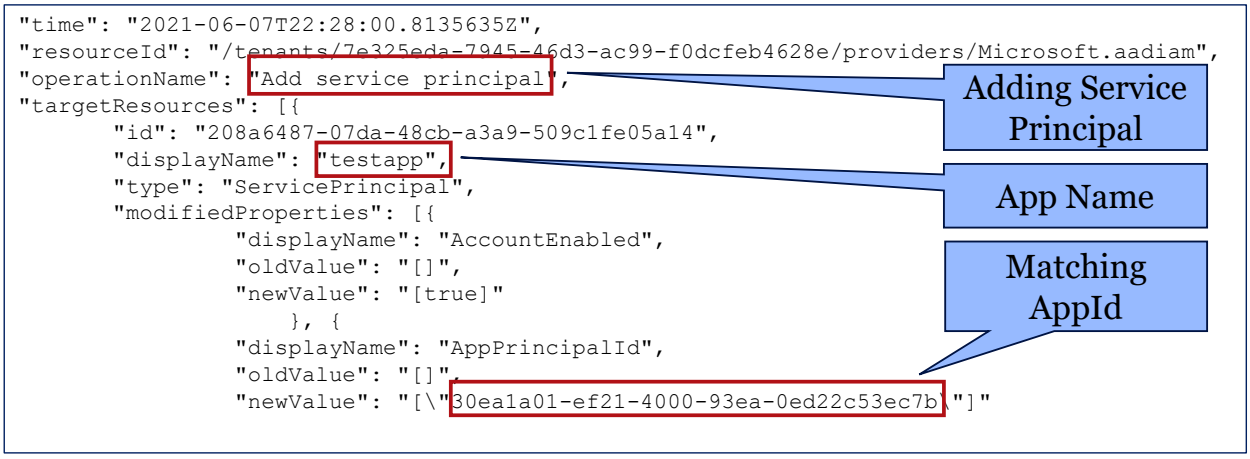
Before we proceed, let's look at our logs. You can see that registering an application results in many log entries:

- Creation of a service principal
- Creation of the application
- Association of the service principal and the owner to the application
- Creation of the application secret and associated updates to the application and service principal

These log entries will be present in the Azure Active Directory audit log.

Register App Log Details

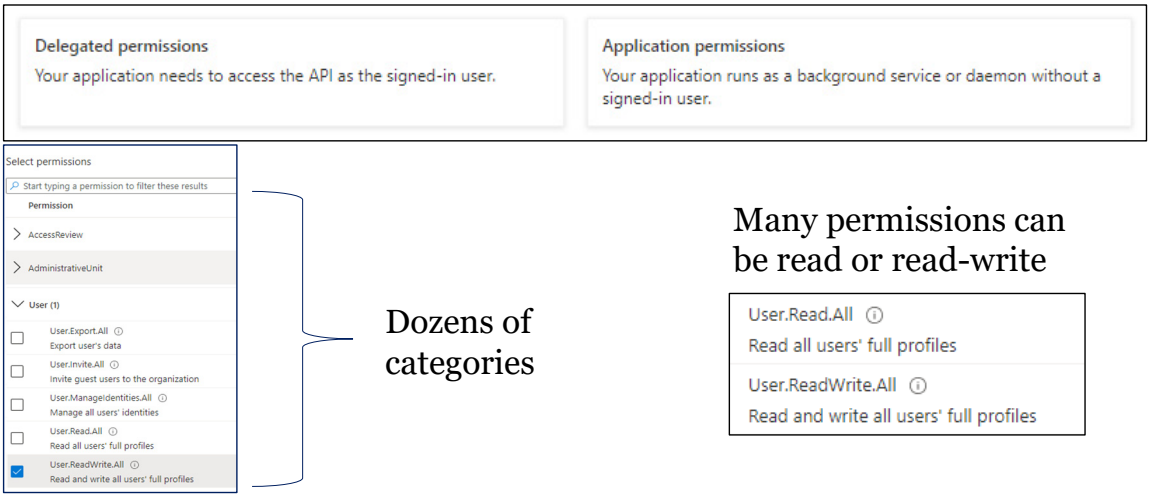
Example detailed log entry for the “add service principal” step



If we expand one of these log entries, we recognize the same information we saw in the console. Namely, we see the AppId and the name of the application. As we track the actions taken by this application, the AppId and the displayName will be key pieces of information.

Step 2: Configure App Permissions

Two levels of access: delegated and application. Application level is the most permissive.



We now have to decide what permissions to grant our application. There are two levels of permissions:

1. Delegated permissions
2. Application permissions

Delegated permissions are less powerful, as they limit access to the context of the signed-in user. Threat actors will favor application permissions because they have global access.

The next step is to select a permission. Each category has numerous options and, in many cases, within each option you will have the choice between Read and ReadWrite.

On the left side of the slide, you can see a small example of the permission categories.

If we were to select the User category, we would then have the choice between `User.Read.All`, which is a read-only permission, and `User.ReadWrite.All`, which is a read-write permission. To create a new user, we will need `User.ReadWrite.All` permission. This won't be sufficient, and we will also need access to the directory with `Directory.ReadWrite.All` permission.

To select the correct permissions for your application, see the Microsoft documentation for the specific action(s) you wish to perform with your application.

Most Risky Permissions

There are hundreds of possible permissions. Apps that are granted application-level permissions may have some of the most risky permissions:

- BitlockerKey.Read.All
- Chat.*
- Directory.ReadWrite.All
- eDiscovery.*
- Files.*
- MailboxSettings.ReadWrite
- Mail.ReadWrite & Mail.Send
- Sites.*
- User.*

*includes different permissions

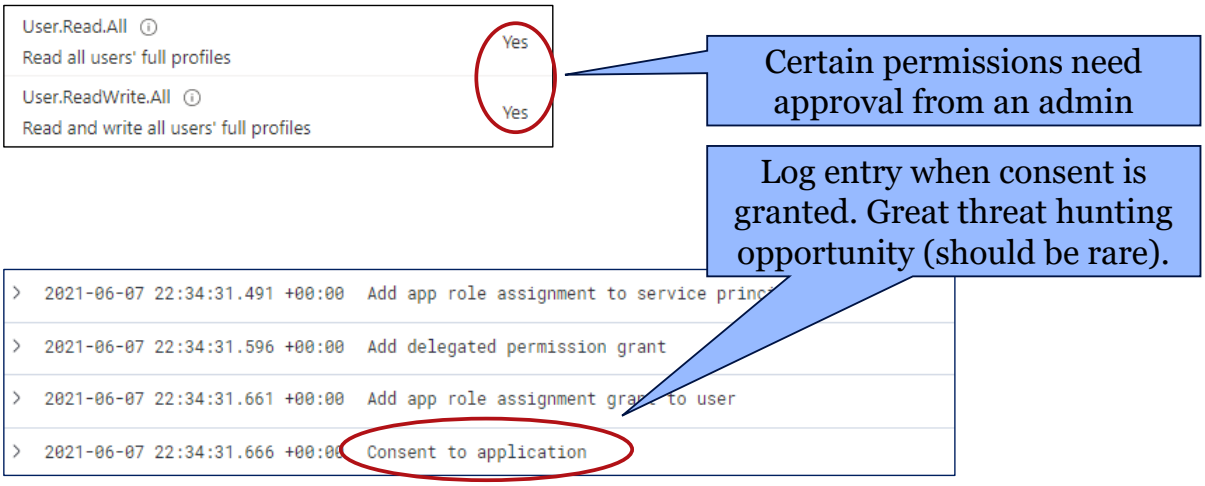
Extremely Dangerous:
- AppRoleAssignment.ReadWrite.All
- RoleManagement.ReadWrite.Directory

In its App Consent Grant incident response playbook, Microsoft lists eight categories that are particularly risky. Threat actors are drawn to these permissions because they allow them to manipulate identities, access mailboxes, and access data.

Depending on the threat actor's objectives, there are many other permissions that could be of high value. However, the ones listed by Microsoft are a good starting point, and you should limit these permissions to as few applications as possible.

Step 3: Get Admin Approval

- Admins who can grant permissions are Global Administrator, Application Administrator, and Cloud Application Administrator



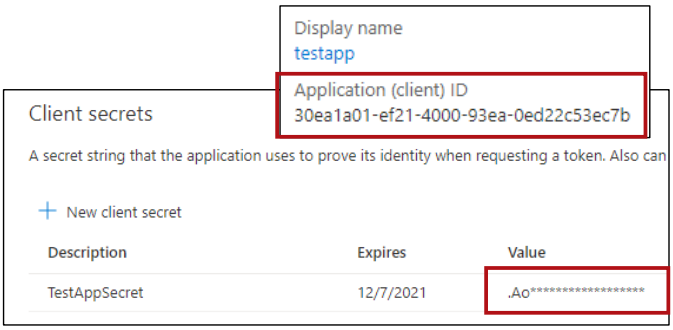
Certain permissions require a second step, where an administrator must consent. Administrators with that authority are Global Administrators, Application Administrators, and Cloud Application Administrators. While Global Administrators are usually highly protected roles, the other two are less well known. Be sure to audit accounts with all three roles and enable MFA on these accounts.

In the logs, when an administrator consents to a permission, you will get an entry called “Consent to application.” These are entries that are worth auditing because they should be very rare.

Step 4: Request Access Token

```
$tenantId = "pymtechlabs.com"
$clientID = "30ea1a01-ef21-4000-93ea-0ed22c53ec7b"
$clientSecret = (ConvertTo-SecureString "Ao**********" -AsPlainText -Force )
$Scope = "https://graph.microsoft.com/.default"
$authToken = Get-MsalToken -ClientId $clientID -ClientSecret $clientSecret
    -TenantId $tenantId -Scopes $Scope
```

By providing the application ID (ClientID) and the client secret, we make a call to the Graph API and obtain an authorization token



We are ready for the next step of our script. We will now request the access token using the PowerShell `Get-MsalToken` cmdlet.

Here is a sample access token:

```
AccessToken                   : eyJ0eXAiOiJKV1QiLCJub25jZSI6InVnT3NISlAzTmt5d21OYzVuMDVfcHQtamIyVWlPWldEWjR0N0xvSXV3Nm8i
                                LCJhbGciOiJSUzI1NiIsIng1dCI6Im5PbzNaRHJPRFhFSzFqS1doWHNsSFJfS1hFZyIsImtpZCI6Im5PbzNaRHJP
                                RFhFSzFqS1doWHNsSFJfS1hFZyJ9.eyJhdWQiOiJodHRwczovL2dyYXBoLm1pY3Jvc29mdC5jb20iLCJpc3MiOiJ
                                odHRwczovL3N0cy53aW5kb3dzLm5ldC83ZTMyNWVkYS03OTQ1LTQ2ZDMtYWM5OS1mMGRjZmViNDYyOGUvIiwiaWF
                                0IjoxNjI1OTQ5NDkyLCJuYmYiOjE2MjU5NDk0OTIsImV4cCI6MTYyNTk1MzM5MiwiYWlvIjoiRTJaZ1lOQnJlNzc
                                xQUhPdTVObFUxVmdSL2V0Y0FBPT0iLCJhcHBfZGlzcGxheW5hbWUiOiJ0ZXN0YXBwIiwiYXBwaWQiOiIzMGVhMWE
                                wMS1lZjIxLTQwMDAtOTNlYS0wZWQyMmM1M2VjN2IiLCJhcHBpZGFjciI6IjEiLCJpZHAiOiJodHRwczovL3N0cy5
                                3aW5kb3dzLm5ldC83ZTMyNWVkYS03OTQ1LTQ2ZDMtYWM5OS1mMGRjZmViNDYyOGUvIiwiaWR0eXAiOiJhcHAiLCJ
                                vaWQiOiIyMDhhNjQ4Ny0wN2RhLTQ4Y2ItYTNhOS01MDljMWZlMDVhMTQiLCJyaCI6IjAuQVVVQTJsNHlma1Y1MDB
                                hc21mRGNfclJpamdFYTZqQWg3d0JBay1vTzBpeFQ3SHRGQUFBLiIsInJvbGVzIjpbIlVzZXIuUmVhZFdyaXRlLkF
                                sbCIsIlNlY3VyaXR5RXZlbnRzLlJlYWQuQWxsIiwiTWFpbC5SZWFkQmFzaWMuQWxsIiwiRGlyZWN0b3J5LlJlYWR
                                Xcml0ZS5BbGwiLCJNYWlsLlNlbmQiLCJBdWRpdExvZy5SZWFkLkFsbCJdLCJzdWIiOiIyMDhhNjQ4Ny0wN2RhLTQ
                                4Y2ItYTNhOS01MDljMWZlMDVhMTQiLCJ0ZW5hbnRfcmVnaW9uX3Njb3BlIjoiTkEiLCJ0aWQiOiI3ZTMyNWVkYS0
                                3OTQ1LTQ2ZDMtYWM5OS1mMGRjZmViNDYyOGUiLCJ1dGkiOiJzWHpYbjl3cmkwYWtJeURXc1VmUUFBIiwidmVyIjo
                                iMS4wIiwid2lkcyI6WyIwOTk3YTFkMC0wZDFkLTRhY2ItYjQwOC1kNWNhNzMxMjFlOTAiXSwieG1zX3RjZHQiOjE
                                1NzM1MjYwODB9.IvAqgpcICrt9EwPudp5FkRC2kU5nd2n1XlNS89JlNd-yHsChMe_BHGlAhI9Dz_vUa2X4IisH3X
                                At5Ol9SXbC_HfliSMVUavTJWcgt-mTWf4XvPfyMchVgmpuEYu6HFKVzf5tkgnsDmV13W66X72Rdh9QOCgSrBz5Fg
                                OuvVeE5ltkk7gMxdhIwhWJqRldkd9E5KZkC6Os9UO44iXf0seGCA9r6CHbVgPtxH6fMO6ZPcmpSr7FOeV8sANIZT
                                dzJ8-dGEqoj2IhvfZcxGnSJ5ROqCdaAyu4GuQIYCaYuqcn6zsFWUAxFETGI3CRBZGppOoP4j9MQfskDUUvW2UkMt
                                qzQg
IsExtendedLifeTimeToken       : False
UniqueId                      :
ExpiresOn                     : 7/10/2021 9:43:11 PM +00:00
```

You can also get a refresh token that your app can use to acquire additional access tokens after the current access token expires. Refresh tokens are long-lived and can be used to retain access to resources for extended periods of time. These are of great interest to threat actors.

Step 5: Call Graph API

```
$Headers = @{
        "Authorization" = "Bearer $($authToken.AccessToken)"
        "Content-type"  = "application/json"
         }
$apiUri = "https://graph.microsoft.com/v1.0/auditLogs/signIns"
$response = Invoke-RestMethod -Headers $Headers -Uri $apiUri -Method GET
```

$Headers	Contains the access token we requested in the previous slide
@apiUri	Contains the API call
$response	Contains the results of the API call

There are thousands of potential API calls. The Microsoft documentation will show the required parameters. API calls that read data (like the one above) are pretty simple. API calls that write data need additional code blocks and are passed to `Invoke-RestMethod` as a `-body` parameter.

We are now ready to call the Graph API with our request. Based on the Uniform Resource Identifier (URI) that we call, we can retrieve various information from the Azure cloud. The Microsoft Graph REST API v1.0 Reference[1] lists the thousands of possibilities. Microsoft also offers a beta API that includes APIs currently in preview.

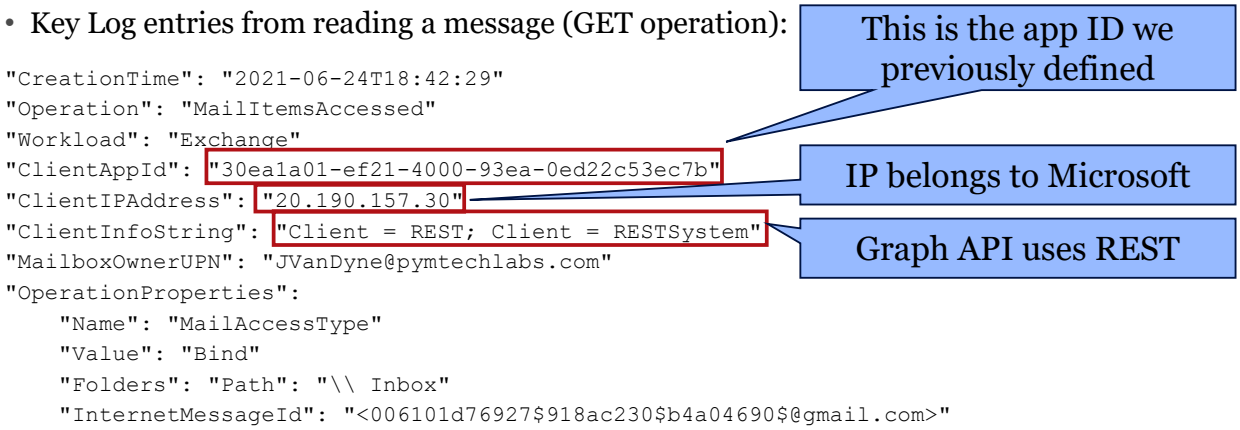
The simplest option is to request some information (read-only), which simply requires the URI. In some cases, the request can be made more specific by appending a parameter to the URI. In this sample request we specified that we want the sign-in logs. A different example would be to request the directory audit logs by specifying the URI: `https://graph.microsoft.com/v1.0/auditLogs/directoryAudits`

When performing a write request, we need to create an array to store the required parameters. This information is then passed with the `-body` option in the `Invoke-RestMethod` call, as we will see in our example to add a user.

1. https://for509.com/graph-reference

Example: Reading Email

- Three useful API calls to read email:
  - List folders: `https://graph.microsoft.com/beta/users/$usr/mailFolders`
  - List messages: `https://graph.microsoft.com/beta/users/$usr/messages`
  - Retrieve a message: `https://graph.microsoft.com/beta/users/$usr/messages/<message id>`
- Key Log entries from reading a message (GET operation):




```
"CreationTime": "2021-06-24T18:42:29"
"Operation": "MailItemsAccessed"
"Workload": "Exchange"
"ClientAppId": "30ea1a01-ef21-4000-93ea-0ed22c53ec7b"
"ClientIPAddress": "20.190.157.30"
"ClientInfoString": "Client = REST; Client = RESTSystem"
"MailboxOwnerUPN": "JVanDyne@pymtechlabs.com"
"OperationProperties":
    "Name": "MailAccessType"
    "Value": "Bind"
    "Folders": "Path": "\\ Inbox"
    "InternetMessageId": "<006101d76927$918ac230$b4a04690$@gmail.com>"
```


As discussed in the SolarWinds case study, the threat actors established long-term access in order to monitor email activity. This can be accomplished with the help of three URIs:

- List folders: `https://graph.microsoft.com/beta/users/$usr/mailFolders`
- List messages: `https://graph.microsoft.com/beta/users/$usr/messages`
- Retrieve a message: `https://graph.microsoft.com/beta/users/$usr/messages/<message id>`

Note that $usr stands for the userid of the mailbox owner.

Once a list of messages has been acquired, it's just a matter of retrieving each one using the third URI.

When looking at the corresponding log entry, you will notice three key pieces of information to indicate that the Graph API was used:

1. The client application ID
2. The client info string, which is set to "REST"
3. The client IP address, which belongs to Microsoft

Unfortunately, we don't get the IP address from the threat actor in this log entry, which is the case for all Graph API calls.

The message access is a bind operation, which will give us the individual message ID that was accessed.

Since the Graph API action is against a mailbox, the logs will be found in the Unified Audit Log.

Example: Adding a User (1)

API to access Azure AD user accounts: `https://graph.microsoft.com/v1.0/users`

Key Log entries from creating a user (POST operation):

```
"creationtime": "2021-06-22T01:38:29"
"operation": "Add user."
"workload": "AzureActiveDirectory"
"actor": {nested JSON field which we are simplifying}
      "ID": "testapp"
      "ID": "30ea1a01-ef21-4000-93ea-0ed22c53ec7b"

<Continued on the next slide>
```

Creating a new user in Azure Active Directory

App name and app ID

This log entry doesn't provide any other indication that it was issued by the Graph API such as the `ClientInfoString` that we saw in the previous example.

The second example is to add a new user to our Azure tenant. The only difference is that we need to provide parameters with the new user's information. We create an array with that information and convert it to JSON:

```
$person = @{
  "accountEnabled" = "true"
  "displayName" = "Hydra"
  "mailNickname" = "Hydra"
  "userPrincipalName" = "Hydra@pymtechlabs.com"
  "passwordProfile" = @{
    "forceChangePasswordNextSignIn" = "false"
    "password" = "HdrwillNev3rdie!"
  }
}

$body = $person | ConvertTo-Json
```

When we call the Graph API, we simply add this information as an option:

```
Invoke-RestMethod -Uri $Uri -Headers $Header -Method Post -Body
$body
```

Example: Adding a User (2)

```
"modifiedProperties": [{
       "displayName": "AccountEnabled",
       "oldValue": "[]",
       "newValue": "[true]"
                   }, {
       "displayName": "DisplayName",
       "oldValue": "[]",
       "newValue": "[\"Hydra\"]"
                   }, {
       "displayName": "MailNickname",
       "oldValue": "[]",
       "newValue": "[\"Hydra\"]"
                   }, {
       "displayName": "UserPrincipalName",
       "oldValue": "[]",
       "newValue": "[\"Hydra@pymtechlabs.com\"]"
                   }, {
       "displayName": "UserType",
       "oldValue": "[]",
       "newValue": "[\"Member\"]"
```

We see a new account named `Hydra@pymtechlabs.com` was created using the Graph API.

In the second part of the log entry, we see the new user's information.

A similar entry will also be created in the Azure audit logs that we will discuss in the Azure section of this class.

What's Logged?

- There is no official documentation about which Graph API calls are logged.
- As a rule of thumb, we observed that "Read" calls **are usually *not*** logged.
- The result of "Write" calls **is usually** logged to one or more logs, such as the Unified Audit Log or various Azure logs.
- For example, testing has shown that the following calls aren't logged:
  - OneDrive file access
  - OneDrive file delete
  - Uploading a user's profile picture

We looked at two examples that generated logs: reading email and adding an Azure Active Directory user. However, there is no official documentation that indicates which Graph API call generates a log entry.

As a rule of thumb, it appears that most "Read" calls do not generate log entries. On the other hand, "Write" calls will generate a log entry either in the UAL or the appropriate Azure log. Some may generate an entry in multiple logs. For example, creating a user would generate an entry both in the UAL and in the Azure audit log.

Some results can be a bit unexpected. We wouldn't expect to get a log entry from accessing a file on OneDrive since it's a "Read" operation. However, it would be good to get a log entry when deleting a file, but that's not the case.

Uploading a user's profile picture is a write operation which we hope would generate a log entry. However, our testing shows that's not the case.

Experimentation is the only way to figure out which API calls create log entries.

Microsoft Graph Security API

Graph API can also be used to query the various security providers

Security Provider
Azure Security Center
Azure Active Directory Identity Protection
Microsoft Cloud App Security
Microsoft Defender for Endpoint
Microsoft Defender for Identity
Azure Information Protection
Azure Sentinel

Common Use Cases
List and update alerts
List and get secure scores
List, get, update secure score control profiles

URI: https://graph.microsoft.com/v1.0/security

Many resources (including sample code) in Microsoft Graph GitHub: `https://for509.com/security-api-github`

Microsoft also offers the Graph Security API.[1] This API provides a unified interface to many of the Microsoft security solutions:

- Azure Security Center
- Azure Active Directory Identity Protection
- Microsoft Cloud App Security
- Microsoft Defender for Endpoint
- Microsoft Defender for Identity
- Azure Information Protection
- Azure Sentinel

Microsoft has a dedicated GitHub repository for the Graph Security API where it provides documentation and sample code.[2]

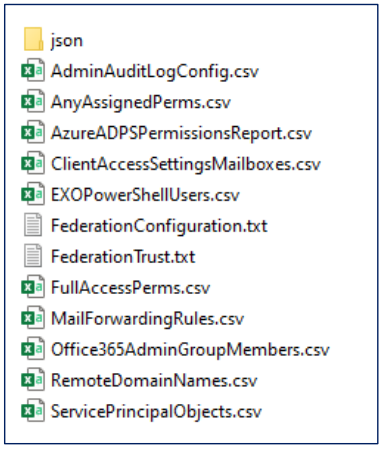
The Graph Security API is a great way to incorporate various security alerts into your own applications. This feature is targeted at software developers rather than incident responders, but you should know it exists if you get the opportunity to participate in strategic security planning for your company.

1. https://for509.com/security-api
2. https://for509.com/security-api-github

Investigate Your Environment

CrowdStrike Reporting Tool for Azure (CRT)

- CRT queries numerous configurations in Azure AD/M365
- Reports in CSV and JSON
- `AzureADPSPermissionsReport` will provide information about Graph API apps
- All reports should be reviewed as they contained valuable security configurations



There are so many possible configurations in Azure AD and Microsoft 365 that it can be very challenging to audit them. CrowdStrike has written an excellent script that can help.[1]

This script will provide information regarding the following permissions:

Exchange Online (M365)

- Federation Configuration
- Federation Trust
- Client Access Settings Configured on Mailboxes
- Mail Forwarding Rules for Remote Domains
- Mailbox SMTP Forwarding Rules
- Mail Transport Rules
- Delegates with 'Full Access' Permission Granted
- Delegates with Any Permissions Granted
- Delegates with 'Send As' or 'SendOnBehalf' Permissions
- Exchange Online PowerShell Enabled Users
- Users with 'Audit Bypass' Enabled
- Mailboxes Hidden from the Global Address List (GAL)
- Collect administrator audit logging configuration settings

Azure AD

- Service Principal Objects with KeyCredentials
- O365 Admin Groups Report
- Delegated Permissions & Application Permissions

We highly recommend running this script in your environment and spending time reviewing the results.

1. https://for509.com/crowdstrikecrt

AzureADPSPermissionsReport

Permissions report for Pymtechlabs Graph API apps

PermissionType	ClientDisplayName	ClientAppId	Permission
Application	testapp	30ea1a01-ef21-4000-93ea-0ed22c53ec7b	Mail.ReadBasic.All
Application	testapp	30ea1a01-ef21-4000-93ea-0ed22c53ec7b	Directory.ReadWrite.All
Application	testapp	30ea1a01-ef21-4000-93ea-0ed22c53ec7b	Sites.ReadWrite.All
Application	testapp	30ea1a01-ef21-4000-93ea-0ed22c53ec7b	Contacts.ReadWrite
Application	testapp	30ea1a01-ef21-4000-93ea-0ed22c53ec7b	Calendars.ReadWrite
Application	testapp	30ea1a01-ef21-4000-93ea-0ed22c53ec7b	Mail.Send
Application	testapp	30ea1a01-ef21-4000-93ea-0ed22c53ec7b	MailboxSettings.ReadWrite
Application	testapp	30ea1a01-ef21-4000-93ea-0ed22c53ec7b	AuditLog.Read.All
Application	for509	183867b2-c2bf-4564-ae48-cddb111e8926	Directory.Read.All
Application	PymChatBot	e62f08d4-c577-4abb-ab56-ee08a0e42fc1	Directory.Read.All
Application	PymChatBot	e62f08d4-c577-4abb-ab56-ee08a0e42fc1	Mail.Read

This screenshot shows the Graph API apps configured in the Pymtechlabs environment. You can clearly see the permissions assigned to each app and assess if they are appropriate or excessive based on your security policy.

For example, the app called “testapp” being granted the “Sites.ReadWrite.All” permission should be reviewed as this permission is extremely broad.

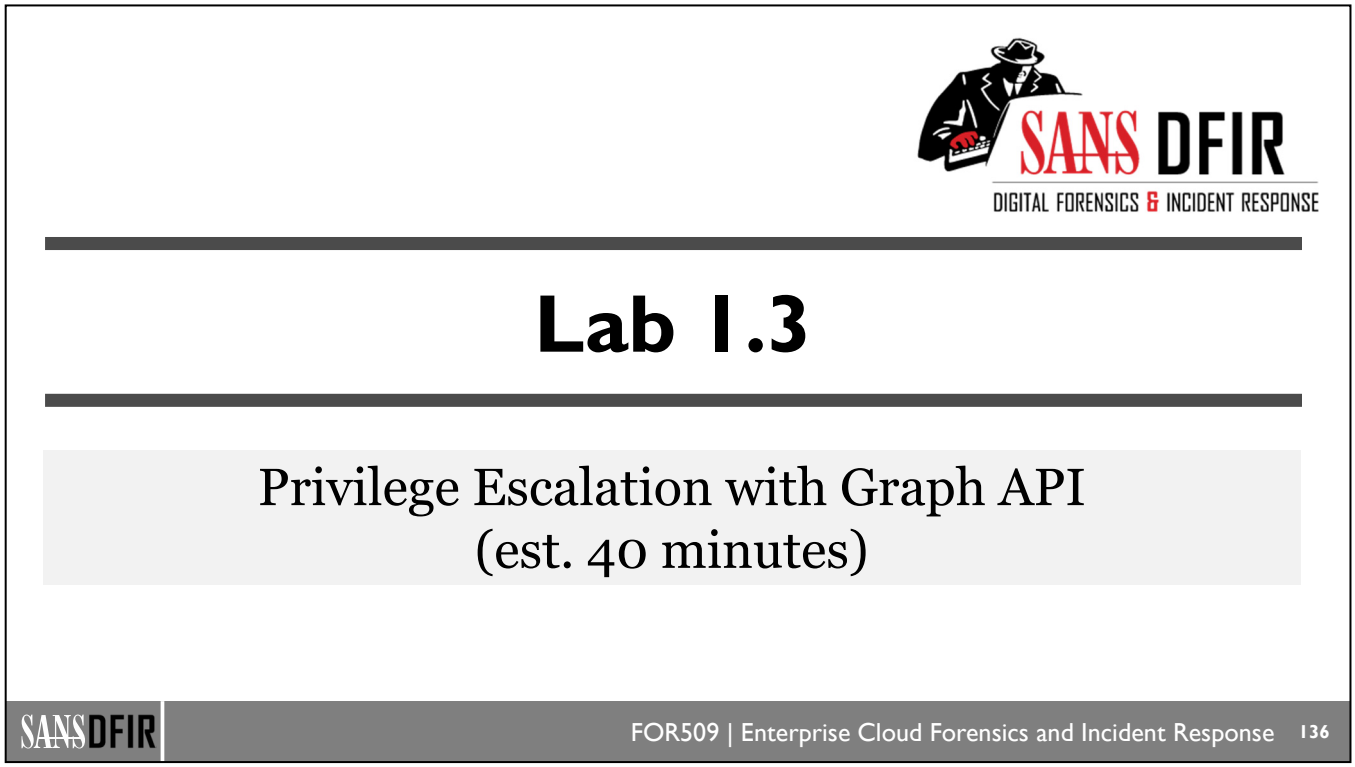
Lab 1.3 Preview

Lab 1.3 will investigate a privilege escalation incident achieved via Graph API.

The following fields will be important in this lab:

- `identity`
- `initiating_user_principal_name`
- `operation_name`
- `source_ip`
- `source_geo.as_org`
- `target_resource_modification.*`
- `target_resources.*`

This page intentionally left blank.



This page intentionally left blank.

Lab 1.3 Debrief

- Analysis is non-linear when compared to the threat actor's actions.
  - You will discover events out of order, so creating a timeline is critical.
- Adding a user to the Global Admin group is noisy and most likely not the best course of action for the threat actor. Watch for more subtle permissions being added to existing Graph API applications.
- Global Admin permission will trigger many warnings such as an email to existing administrators and possibly a "triggered PIM alert":

```
"resourceId": "/tenants/7e325eda-7945-46d3-ac99-f0dcfeb4628e/providers/Microsoft.aadiam",
"operationName": "Triggered PIM alert",
"category": "AuditLogs",
"identity": "Azure AD PIM",
"targetResources": [{
               "id": "TooManyGlobalAdminsAssignedToTenantAlert",
               "displayName": "There are too many global administrators",

(Log entry was removed from the lab data)
```


Lab 1.3 analyzes a specific abuse of Graph API permissions discussed by SpecterOps[1]. In their article, SpecterOps discusses how to perform the attack. As DFIR investigators, we look at the aftermath of the attack and try to understand what actions the threat actor took. One of the main observations is that you will not discover the threat actor's actions in the same order as they were performed. This highlights the need for creating a timeline as you discover various artifacts.

These are the steps of this attack in chronological order. You will notice that in the lab we discovered the log entries showing the privilege escalation of JanetVanDyne (JVD) and Hydra together even though these steps were quite distant from each other in the attacker's roadmap:

Step 1: Connect to Azure as JVD
Step 2: Connect to AzureAD as JVD with PowerShell
Step 3: Add a secret to app: QuantumApp
Step 4: Connect to Azure with QuantumApp secret
Step 5: Get bearer token for QuantumApp
Step 6: Grant QuantumApp the permission "RoleManagement.ReadWrite.Directory"
Step 7: Get bearer token for QuantumApp with new permission
Step 8: Promote JVD to Global Admin
Step 9: Add permission "User.ReadWrite.All" to QuantumApp
Step 10: Add permission "Directory.ReadWrite.All" to QuantumApp
Step 11: Get a bearer token for QuantumApp with the new permissions
Step 12: Create new user: hydra@pymtechlabs.com
Step 13: Promote Hydra to GA

From a threat actor point of view, while obtaining Global Admin may be seen as nirvana, it's actually very dangerous. Because the Global Admin role is so powerful, it's actively monitored by Microsoft. When an account is given that role, existing global admins will receive an email. In addition, Azure AD Privileged Identity Management (PIM) is likely to issue a warning. An experienced threat actor is more likely to add more subtle yet powerful permissions to an existing Graph API application in order to minimize attention.

1. https://for509.com/specterops

Course Resources and Contact Information

Here is my lens. You know my methods. —Sherlock Holmes

AUTHOR CONTACT
Pierre Lidome
plidome@gmail.com
Twitter: @texaquila

SANS INSTITUTE
11200 Rockville Pike, Suite 200
North Bethesda, MD 20852
301.654.SANS(7267)

DFIR RESOURCES
digital-forensics.sans.org
Twitter: @sansforensics

SANS EMAIL
GENERAL INQUIRIES: info@sans.org
REGISTRATION: registration@sans.org
TUITION: tuition@sans.org
PRESS/PR: press@sans.org

This page intentionally left blank.