KC7

A CYBERSECURITY GAME

Simeon Kakpovi
Greg Schloemer
Emily Hacker

Azure Data Explorer

| New tab | New tab | New tab | kc7round2.eastus.... ✎ |

Add cluster

« ▷ Run ⟳ Recall | **Scope:** @kc7round2.eastus/SecurityLog

▽ Filter...

⌄ ▣ **kc7round2.eastus**
  ⌄ ▤ **SecurityLogs**
    > ▦ AuthenticationEvents
    > ▦ Email
    > ▦ Employees
    > ▦ FileCreationEvents
    > ▦ InboundBrowsing
    > ▦ OutboundBrowsing
    > ▦ PassiveDns
    > ▦ ProcessEvents
  > ▤ **SecurityLogs2**

```
1   Employees
2   | take 10
3
4   Email
5   | take 100
6
```

▦ Table 1   ◎ Stats

sender	recipient
alicia_moreno@envolvelabs.com	maria74@reynolds.net
alma_hibbs@envolvelabs.com	thomasnicholas@gmail.co
angela_carpentier@envolvelabs.com	lawsonanthony@davis.com
anita_lucas@envolvelabs.com	irving_martinez@envolvela
anita_lucas@envolvelabs.com	robert_bjorklund@envolve
assort@gmail.com	mary_laney@envolvelabs.c
beneficent@gmail.com	donald_gabel@envolvelab
beneficent@gmail.com	theodore_kloeck@envolve
bestowingliberality@qq.com	ramona_barnhart@envolve
betty_clark@envolvelabs.com	francisco_mcdowell@envo
brickedsurpassed@aol.com	sarah_perdue@envolvelab

Introduction: Welcome to Envolve Labs

Welcome to Envolve Labs Corporation! 🥳 Today is your first day as a Junior Security Operations Center (SOC) Analyst with our company. Your primary job responsibility is to defend Envolve Labs and its employees from malicious cyber actors.

EnvolveLabs

Envolve Labs is a med-tech startup based in the United States that was founded in 2012. Our mission is to develop a new type of flexible vaccine technology that covers many different viral strains and offers long-lasting immunity (which means no more boosters!) Our initial research has proven this technology is highly effective – we're planning to start production in Q1 2023.

EnvolveLabs has a series of key partners who contribute to the success of our business:

Partner Name	Relationship
We Sell Beakers™ (wesellbeakers.com)	A key distributor of medical-grade laboratory equipment, We Sell Beakers™ provides all of the equipment for our world-class research facilities.
PharmaSupplies, Inc. (pharmasupplies.org)	PharmaSupplies provides Envolve Labs with the core ingredients used in our vaccine products.
Vaccine Distributors Worldwide (vaccinedistributors.com):	EnvolveLabs trusts Vaccine Distributors Worldwide to deliver our temperature and time-sensitive vaccine products to customers around the globe.
Research Compliance, PSC (researchcompliance.com	Research Compliance, PSC is the key legal partner of Envolve Labs who helps ensure compliance with international guidelines and regulations that govern vaccine production.

Until now, we've been laser focused on medical research and meeting production goals. But, as our work becomes more important and successful, we've realized the need to invest more in cybersecurity efforts. That's why we've hired you!

Like all good companies, Envolve Labs collects log data about the activity its employees perform on the corporate network. These security audit logs are stored in **Azure Data Explorer (ADX)** - a data storage service in Azure (Microsoft's cloud). You will use the Kusto Query Language (KQL) to parse through various types of security logs. By analysing these logs, you can help us determine whether we're being targeted by malicious actors.

You can find full documentation on ADX here: https://docs.microsoft.com/en-us/azure/data-explorer/kusto/query/tutorial?pivots=azuredataexplorer

Objectives

🧠 **By the end of your first day on the job, you should be able to:**

- ✓ Use the Azure Data Explorer
- ✓ Use multiple data sets to answer targeted questions
- ✓ Find cyber activity in logs including: email, web traffic, and server logs
- ✓ Use multiple techniques to track the activity of APTs (Advanced Persistent Threats)
- ✓ Use third party data sets to discover things about your attackers
- ✓ Build a threat intelligence report
- ✓ Make recommendations on what actions a company can take to protect themselves

The attackers have gotten a head start, so let's not waste any more time… time to get to work!

Some important links:
- The scoreboard: kc7cyber.com/scoreboard
- Azure Data Explorer: kc7cyber.com/data

Legend

🎯 **Key Point** – Occasionally, you will see a dart emoji with a "key point." These signal explanations of certain concepts that may enhance your understand of key cybersecurity ideas that are demonstrated in the game.

🤔 **Question** – "Thinking" emojis represent questions that will enable you to demonstrate mastery of the concepts at hand. You can earn points by entering your responses to questions from section 3 in the scoring portal available at kc7cyber.com/scoreboard.

🤫 **Hint** – "Whisper" emojis represent in-game hints. These hints will guide you in the right direction in answering some of the questions.

Section 1: The walkthrough

Getting Set Up in Azure Data Explorer (ADX)

ADX is the primary tool used in the Envolve Labs SOC for data exploration and analysis. The great thing about ADX is that it is used by cyber analysts at many of the smallest and largest organizations in the world.

Let's get you logged in and started with ADX:

1. Go to **kc7cyber.com/data**
2. Login with your microsoft.com account credentials. You can use your Hotmail.com, outlook.com, live.com, or O365 account credentials.

Once you login, you should see a cluster called "kc7cyber.eastus2" has already been added to your account
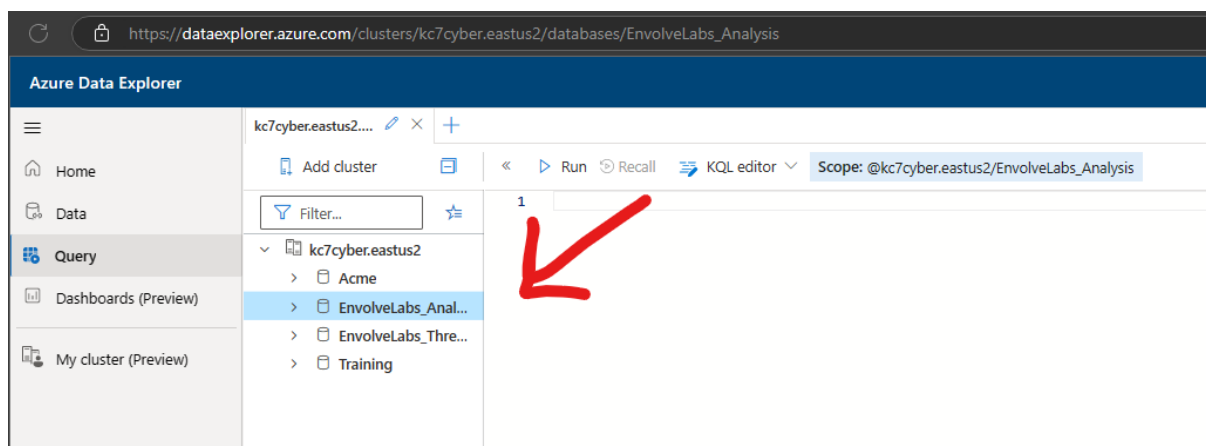
Data in ADX is organized in a hierarchical structure which consists of **clusters**, **databases**, and **tables**.
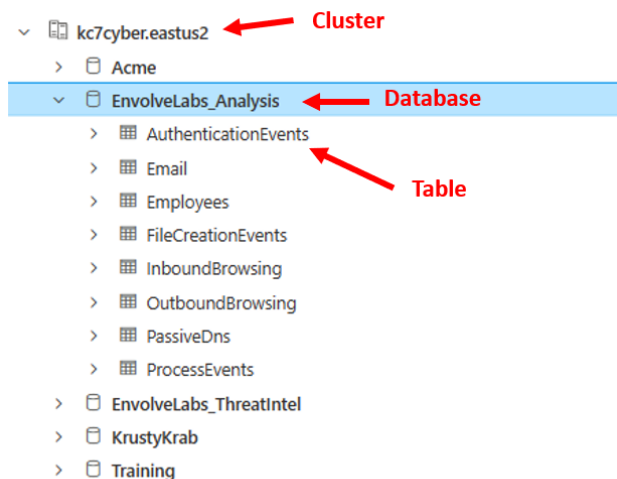
All of Envolve Labs's security logs are stored in a single database – The **EnvolveLabs_Analysis** cluster.

3. Select your database
   - Expend the dropdown arrow next to the **EnvolveLabs_Analysis** database.
   - Click on the **EnvolveLabs_Analysis** database. Once you've done this, you should see the database highlighted- this means you've selected the database and are ready to query the tables inside.

The big blank space to the right of your cluster list is the *query workspace.* That's where you'll actually write the queries used to interact with our log data.
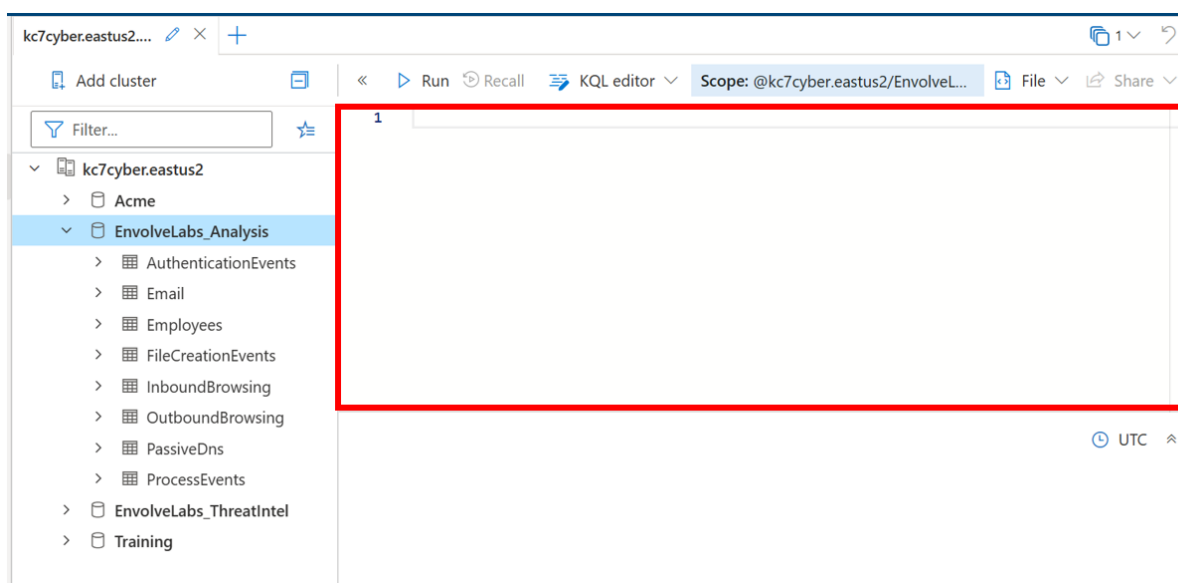
Okay, enough introductions… let's get your hands on the data

First look at the data…

The *EnvolveLabs_Analysis* database contains eight tables. Tables contain many rows of similar data. For security logs, a single row typically represents a single thing done by an employee or a device on the network at a particular time.

We currently have eight types of log data. As you'll see in ADX, each log type corresponds to a table that exists in the *EnvolveLabs_Analysis* database:

Table Name	Description
Employees	Contains information about the company's employees
Email	Records emails sent and received by employees
InboundBrowsing	Records browsing activity from the Internet to devices within the company network
OutboundBrowsing	Records browsing activity from within the company network out to the Internet
AuthenticationEvents	Records successful and failed logins to devices on the company network. This includes logins to the company's mail server.
FileCreationEvents	Records files stored on employee's devices
ProcessEvents	Records processes created on employee's devices
PassiveDns (External)	Records IP-domain resolutions

🎯 **Key Point – Over the Horizon (OTH) data**: One of the tables listed above is not like the others – **PassiveDns.** Rather than being an internal security log, PassiveDns is a data source that we've purchased from a 3rd party vendor. Not all malicious cyber activity happens within our company network, so sometimes we depend on data from other sources to complete our investigations.
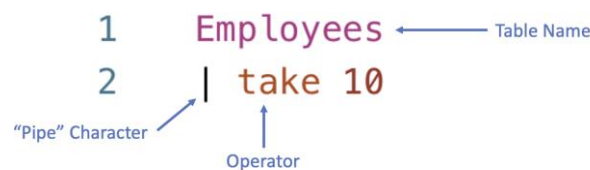
You'll learn more about how to use each of these datasets in just a minute. First, let's just run some queries so you can practice using KQL and ADX.

KQL 101

Type the following query in the workspace to view the first rows in the **Employees** table. Press "run" or "shift + enter" to execute the query.

```
Employees
| take 10
```

This query has a few parts. Let's take a moment to break each of them down:

Query Component	Description
Table name	The table name specifies which table/data source the query will pull data from. All queries must start with a table.
Pipe character (|)	The pipe character indicates the start of a new part of the query. A pipe will be added automatically after typing a table name and pressing enter. You can also add a pipe character manually by holding shift and pressing the backslash (\\) key. That's the one just below the backspace key.
Operator	The operator tells the query what exactly you want to do. The first operator you've learned is take, which simply *takes* a given number of rows and shows you the data there. You'll learn and practice using more operators soon!

The `take` operator is a powerful tool you can use to explore rows in a table, and therefore better understand what kinds of data are stored there.

🎯 **Key Point – What to do when you don't know what to do:** Whenever you are faced with an unfamiliar database table, the first thing you should do is sample its rows using the `take` operator. That way, you know what fields are available for you to query and you can guess what type of information you might extract from the data source.

The **Employees** table contains information about all the employees in our organization. In this case, we can see that the organization is named "Envolve Labs" and the domain is "envolvelabs.com".

1. 🤔 *Try it for yourself! Do a* `take` `10` *on all the other tables to see what kind of data they contain.*

You can easily write multiple queries in the same workspace tab. To do this, make sure to separate each query by an empty line. Notice below how we have separated the queries for the **Employees**, **Email**, and **OutboundBrowsing** tables by empty lines on lines 3 and 6.

```
1    Employees
2    | take 10
3
4    Email
5    | take 10
6
7    OutboundBrowsing
8    | take 10
```

When you have multiple queries, it's important to tell ADX which query you want to run. To choose a query, just click on any line that is part of that query. Once you've selected a query, it will be highlighted in blue, as seen on lines 4 and 5 above.

Finding Out "How Many": The count Operator

We can use count to see how many rows are in a table. This tells us how much data is stored there.

```
Employees
| count
```

2. 🤔 *How many employees are in the company?*

Filtering Data With the *where* Operator

So far, we've run queries that look at the entire contents of the table. Often in cybersecurity analysis, we only want to look at data that meets a set of conditions or criteria. To accomplish this, we apply filters to specific columns.

We can use the where operator in KQL to apply filters to a particular field. For example, we can find all the employees with the name "Linda" by filtering on the *name* column in the **Employees** table.

where statements are written using a particular structure. Use this helpful chart below to understand how to structure a where statement.

```
where  field   operator  "value"
where  name    has       "Linda"
```

```
Employees
| where name has "Linda"
```

The has operator is useful here because we're looking for only a *partial* match. If we wanted to look for an employee with a specific first and last name (an *exact* match), we'd use the == operator:

```
Employees
| where name == "Linda Holbert"
```

3. 🤔 *Each employee at Envolve Labs is assigned an IP address. Which employee has the IP address: "192.168.0.191"?*

While performing their day-to-day tasks, Envolve Labs employees send and receive emails. A record of each of these emails is stored in the **Email** table.

🎯 **Key Point – User Privacy and Metadata:** As you can imagine, some emails are highly sensitive. Instead of storing the entire contents of every email sent and received within the company in a database that can be easily accessed by security analysts, we only capture email metadata.

Email metadata includes information like: the time the email was sent, the sender, the recipient, the subject line, and any links the email may contain. Storing only email metadata, rather than entire contents, helps protect the privacy of our employees, while also ensuring that our security analysts can keep us safe. Sometimes even metadata can reveal sensitive information, so it's important that you don't talk about log data with other employees outside the SOC.

We can find information about the emails sent or received by a user by looking for their email address in the *sender* and *recipient* fields of the **Email** table. For example, we can use the following query to see all the emails sent by "Michael Montello":

```
Email
| where sender == "michael_montello@envolvelabs.com"
```

4. 🤔 *How many emails did Betty Parrish receive?*

Easy as 1, 2, 3… Compound Queries and the distinct Operator

We can use the `distinct` operator to find unique values in a particular column. We can use the following query to determine how many of the organization's users sent emails.

```
1    Email
2    | where sender has "envolvelabs"
3    | distinct sender
4    | count
```

This is our first time using a multi-line query with multiple operators, so let's break it down:

In **line 2**, we take the Email table and filter the data down to find only those rows with "envolvelabs" in the sender column.

In **line 3**, we add another pipe character (|) and use the distinct operator to find all the unique senders. Here, we aren't finding the unique senders for all of the email senders, but only the unique senders that are left after we apply the filter looking for rows with "envolvelabs" in the sender column.

Finally, in **line 4**, we add another pipe character (|) and then use the count operator to count the results of lines 1-3 of the query.

5. 🤔 *How many users received emails with the term "vaccine" in the subject?*

Tracking Down a Click: OutboundBrowsing Data

When employees at Envolve Labs browse to a website from within the corporate network, that browsing activity is logged. This is stored in the **OutboundBrowsing** table, which contains records of the websites browsed by each user in the company. Whenever someone visits a website, a record of it stored in the table. However, the user's name is not stored in the table, only their IP address is recorded. There is a 1:1 relationship between users and their assigned IP addresses, so we can reference the **Employees** table to figure out who browsed a particular website.

If we want to figure out what websites Annie Jackson visited, we can find her IP address from the Employees table.

```
Employees
| where name == "Annie Jackson"
```

The query above tells us her IP address is "192.168.3.168". We can take her IP address and look in the **OutboundBrowsing** table to determine what websites she visited.

```
OutboundBrowsing
| where src_ip == "192.168.3.168"
```

6. 🤔 How many unique websites did "Keith Mitchell" visit?

What's in a Name? All about Passive DNS Data

Although domain names like "google.com" are easy for humans to remember, computers don't know how to handle them. So, they convert them to machine readable IP addresses. Just like your home address tells your friends how to find your house or apartment, an IP address tells your computer where to find a page or service hosted on the internet.

🎯 **Key Point – Practice Good OPSEC:** If we want to find out which IP address a particular domain resolves to, we could just browse to it. But, if the domain is a malicious one, you could download malicious files to your corporate analysis system or tip off the attackers that you know about their infrastructure. As cybersecurity analysts, we must follow procedures and safeguards that protect our ability to track threats. These practices are generally called operational security, or OPSEC.

To eliminate the need to *actively resolve* (that is- directly browse to or interact with a domain to find it's related IP address) every domain we're interested in, we can rely on *passive DNS* data. Passive DNS data allows us to safely explore domain-to-IP relationships, so we can answer questions like:

- *Which IP address does this domain resolve to?*
- *Which domains are hosted on this IP address?*
- *How many other IPs have this domain resolved to?*

These domain-to-IP relationships are stored in our **PassiveDns** table.

7. 🤔 *How many domains in the **PassiveDns** records contain the word "vaccine"? (hint: use the* `contains` *operator instead of* `has`. *If you get stuck, do a* `take` 10 *on the table to see what fields are available.)*

8. 🤔 *What IPs did the domain "biotechenvolv.science" resolve to?*

🧑‍🦲 **Let statements – making your life a bit easier:**

Sometimes we need to use the output of one query as the input for a second query. The first way we can do this is by manually typing the results into next query.

For example, what if we want to look at all the web browsing activity from employees named "Linda"?

First, you would need to go into the **Employees** table and find the IP addresses used by these employees.

```
1    Employees
2    | where name has "Linda"
```

⊞ Table 1   ◎ Stats

	timestamp	name	user_agent	ip_addr	email
>	2012-12-04 12:51:51.409473	Linda Osman	Mozilla/5.0 (Macintosh; Intel Mac ...	192.168.3.60	linda_
>	2013-09-22 14:25:13.939571	Linda Uerkwitz	Mozilla/5.0 (Linux; Android 4.2) Ap...	192.168.3.131	linda_
>	2015-03-25 11:50:38.009644	Linda Vauters	Mozilla/5.0 (Linux; Android 2.3.3) A...	192.168.2.15	linda_
>	2015-09-01 13:06:53.140583	Linda Loy	Mozilla/5.0 (Macintosh; Intel Mac ...	192.168.2.218	linda_
>	2016-08-25 13:42:59.359341	Linda Ramirez	Mozilla/5.0 (X11; Linux x86_64) App...	192.168.3.36	linda_
>	2019-11-23 13:19:15.689430	Linda Luna	Mozilla/5.0 (iPad; CPU iPad OS 9_3_...	192.168.3.118	linda_

Then, you could manually copy and paste these IPs into a query against the **OutboundBrowsing** table. Note that we can use the *in* operator to choose all rows that have a value matching any value from a list of possible values. In other words, the == (comparison) operator looks for an exact match, while the *in* operator checks for any values from the list.

```
1    OutboundBrowsing
2    | where src_ip in ("192.168.3.60","192.168.3.131","192.168.2.15","192.168.2.218","192.168.3.36","192.168.3.118")
```

⊞ Table 1   ◎ Stats   🔍 Search   🕐 UTC   ✅ Done (0.133

	timestamp	method	src_ip	user_agent	url
>	2022-01-02 20:41:55.900806	GET	192.168.3.1...	Mozilla/5.0 (iPad; CPU iPad OS 9_3_6 like Mac OS X) AppleWebKit/533.1 (KHTML, like Gecko) CriOS/38.0.821.0 Mobile/74...	https://buil
>	2022-01-03 04:07:11.130219	GET	192.168.3.1...	Mozilla/5.0 (Linux; Android 4.2) AppleWebKit/531.1 (KHTML, like Gecko) Chrome/59.0.896.0 Safari/531.1	https://ridc
>	2022-01-03 05:39:32.280788	GET	192.168.3.36	Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/534.0 (KHTML, like Gecko) Chrome/35.0.849.0 Safari/534.0	https://lact

Although this is a valid way to get the information you need, it may not be as elegant (or timely) if you had 100 or even 1000 employees named "Linda."

We can accomplish this in a more elegant way by using a let statement, which allows us to assign a name to an expression or a function. We can use a `let` statement here to save and give a name to the results of the first query so that the values can be re-used later. That means we don't have to manually type or copy and paste the results repeatedly.

```
1    let linda_ips = Employees
2    | where name has "Linda"
3    | distinct ip_addr;
4    OutboundBrowsing
5    | where src_ip in (linda_ips)
6
```

On the left of the `let` statement is the variable name ("linda_ips" in this case). The variable name can be whatever we want, but it is helpful to make it something meaningful that can help us remember what values it is storing.

```
1   let linda_ips = Employees
2   | where name has "Linda"
3   | distinct ip_addr;
4   OutboundBrowsing
5   | where src_ip in (linda_ips)
```

On the right side of the `let` statement in the expression you are storing. In this case, we use the `distinct` operator to select values from only one column – so they are stored in an array – or list of values.

```
1   let linda_ips = Employees
2   | where name has "Linda"
3   | distinct ip_addr;
4   OutboundBrowsing
5   | where src_ip in (linda_ips)
```

The `let` statement is concluded by a semi-colon.

```
1   let linda_ips = Employees
2   | where name has "Linda"
3   | distinct ip_addr;
4   OutboundBrowsing
5   | where src_ip in (linda_ips)
```

After we store the value of a query into a variable using the `let` statement, we can refer to it as many times as we like in the rest of the query. The stored query does not show any output. Remember, however, that your KQL query must have a tabular statement – which means that you must have another query following your `let` statement.

9. 🤔 How many unique URLs were browsed by employees named "Karen"?

🎯 **Key Point – Pivoting:** Part of being a great cyber analyst is learning how to use multiple data sources to tell a more complete story of what an attacker has done. We call this "pivoting." We pivot by taking one known piece of data in one dataset and looking in a different dataset to learn something we didn't already know. You practiced this here when we started in one dataset – the Employees table – and used knowledge from there to find related data in another source – OutboundBrowsing.


Section 2: Introducing the hackers

Now that you've completed your initial round of training, you are ready to work your first case in the SOC!

A security researcher tweeted that the domain "*immune.tech*" was being used by hackers. Apparently the hackers are sending this domain inside credential phishing emails.

> ⚠️ NOTE! This domain and others encountered in this game are fictional and are not representative of actual malicious activity,

According to OSINT research your colleagues conducted, this domain may be used as part of a phishing campaign with the following stages:
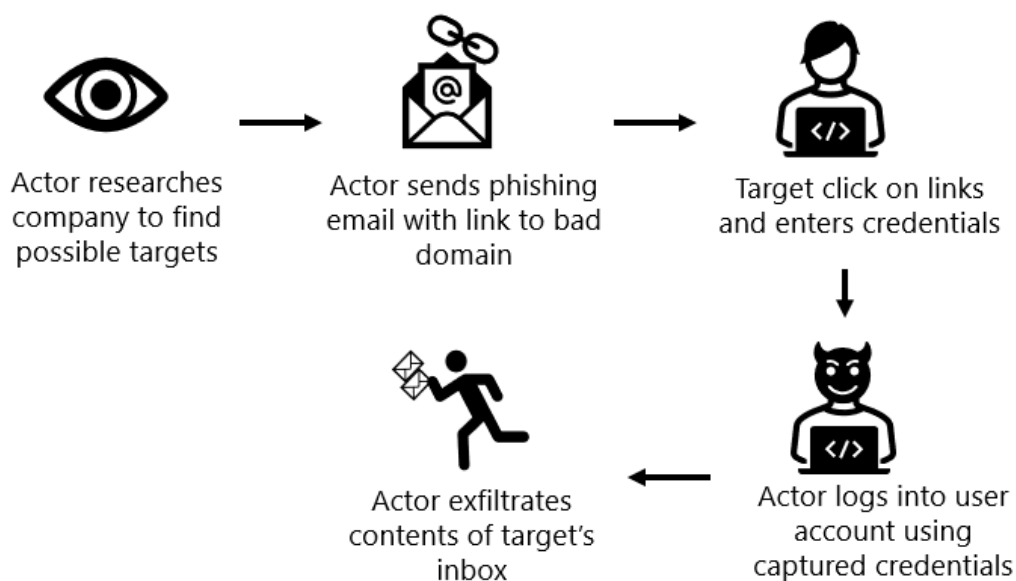
🎯 **Key Point – Open Source Intelligence (OSINT):** Security researchers and analysts often use free, publicly available data, like Twitter! We call this public data OSINT, and it can be a great way to get investigative leads. Like all public data sources on the internet, you should follow up any OSINT tip with rigorous analysis, rather than blindly trusting the source.

🧐 Answer the following questions related to this tip:

1. Which users in our organization were sent emails containing the domain *immune.tech*?

2. Did we block any of the emails containing that domain? Who actually received one of these emails? (hint: the "accepted" field in the **Email** table tells you whether or not the email was blocked. Blocked emails will show as false).

3. What other domains shared the same IPs as *immune.tech?* Can you find the full list of domains associated with this actor based on PassiveDns data? (hint: you can use the **in** operator to check for multiple values in a field. E.g. where field in ("x", "y", "z")

4. What email addresses did the hackers use to send these domains?

5. Did users click on any of the links in the phishing emails?

6. Did any user have their credentials stolen? How do you know?

🧐 **Hint:** In order to have their credentials stolen, a user would need to browse to the credential harvesting site and enter their username and password. After this, the actor might try to login to the user's account using the stolen credentials. You can find details about login activity in the **AuthenticationEvents** table.

7. Did any user have contents exfiltrated (stolen) from their mailbox? How do you know? What risk is posed to the company by this content being stolen?

Section 3: Hackers sending malware docs

After digging for a bit on the phishing activity, you come across another tweet from a threat intelligence vendor SolitaryStrike:

🧐 Use the tipper above to answer the following questions:
You can optionally submit your answers to the scoreboard at
https://kc7cyber.azurewebsites.net/ to get feedback and earn points.

1. How many emails contained the domain notice[.]io?

2. What email address sent the domain notice[.]io

3. What was the subject line of the emails containing the domain notice[.]io?

4. What is the name of the user who clicked on the notice[.]io link?

5. At what timestamp did the user above download the file: "Critical_Security_Path.docx"?

6. How many emails were sent to your organization on January 9th by users at wesellbeakers.com?

7. What other domains are hosted on the same IPs as notice[.]io?

8. What email address is seen sending emails containing one of the domains identified in question 7?

9. How many users downloaded the files observed in the emails from question 8?

10. One of the files observed in question 9 - IMPORTANT_INSTRUCTIONS.pptx - was seen in two separate emails. What are the subject lines of these emails?

11. Which compromised pharmasupplies.org email address was used to send a link to scanverify.com?

12. How many IPs has scanverify.com resolved to?

13. Consider the email address you found in question 11. What other domain did this email address send?

14. What is the name of the file hosted on scanverify.com?

15. Which .pptx file was used to target Gerald Kempinski and Kenny Salcido?

16. Which actor IP was used to search EnvolveLabs' website for the term "helpdesk ticket system"?

17. How many total emails were sent to your organization by this actor?

18. Which .dll file was dropped on a victim machine shortly after the user downloaded the malicious zip : EnvolveLabs_Research_Tool.7z

    🧑 **Hint**: Files that are created on employees' devices are captured in the **FileCreationEvents** log. Try looking there to see which employees downloaded this file.

19. Which six letter reconnaissance command was executed on the Machine of the user that loaded the implant above?

    🧑 **Hint**: Try narrowing down on one particular device that downloaded the EnvolveLabs_Research_Tool.7z file. Then, look in both the **FileCreationEvents** and **ProcessEvents** logs to find new files and processes created <u>around the time</u> when the file was downloaded.

20. A malicious file 'infector.exe' is observed performing suspicious actions on multiple devices. What process_commandline associated with this file is being used for persistence on the devices?

    🧑 **Hint**: Actors establish persistence so they can come back later and conduct manual tasks (called hands-on-keyboard activity) within your company's network. Try looking for systems creating connections to external domains and IPs, or unusual behaviors like creation of scheduled tasks.