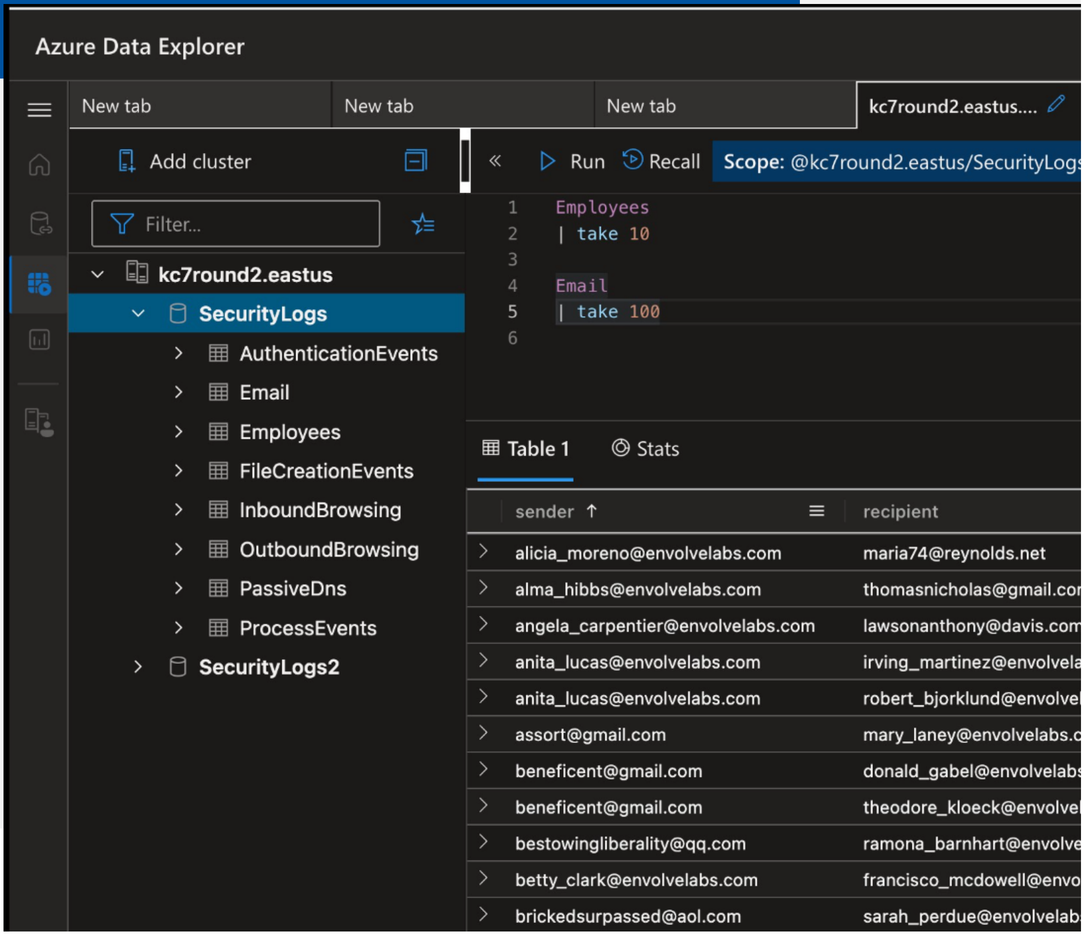
KC7

A Cybersecurity Game



```
Employees
| take 10

Email
| take 100
```

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

kc7cyber.com

KC7 - A Cybersecurity Game

Introduction: Welcome to Krusty Krab

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

The Krusty Krab is a mid-size fast food chain serving the greater Bikini Bottom metropolitan area. The Krusty Krab is best known for its delectable Krabby Patties™, kelp shakes, and sea dogs. Because of its wild success, some of the Krusty Krab's competitors would benefit from stealing its secret formula and reproducing it themselves. Of note, the Chum bucket - a minor competitor - has been less than ethical in its attempts to infiltrate the Bikini Bottom patty market. Your job is to defend the Krusty Krab and its employees from malicious cyber actors looking to steal the secret formula.

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

Partner Name	Relationship
Bikini Bottom Fish (bikini-bottom-fish.com)	Bikini Bottom Fish is a company that specializes in providing fish and seafood products to our company and has been a trusted partner for many years.
Salty Sea Shell LLC (saltyseashell.com)	Salty Sea Shell LLC is a fashion apparel and accessories company that has been a loyal partner of our company, providing us with high-quality products for our customers.
KelpShakez (kelpshake.com)	KelpShakez is a beverage company that has been our partner in providing unique and healthy drink options for our customers, made from kelp and other natural ingredients.
The Chum Bucket	The Chum Bucket is a fast-food restaurant and a key competitor of our business. However, we have worked with them in the past for certain collaborations and events, and they have proven to be a reliable partner on those occasions.

Krusty Krab has been laser-focused on broadening our customer base. Recently, though, we have been getting a lot of attention. All this extra attention has brought along some unwelcome visitors- cyber attackers! That's why we've hired you! To help keep us safe!

Like all good companies, Krusty Krab collects log data about the activity our 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
- Investigate 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

- 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... let's get to work!

Some important links:

The training module: kc7cyber.com/krustykrab

Legend

🎯Key Point – Occasionally, you will see a dart emoji with a “key point.” These signal explanations of certain concepts that may enhance your understanding 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

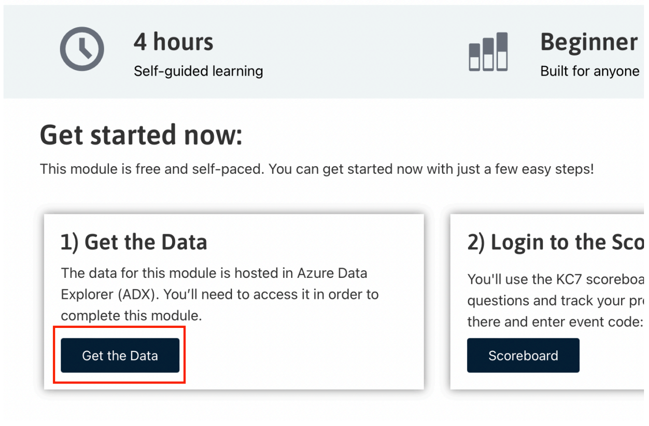
Getting Set Up in Azure Data Explorer (ADX)

ADX is the primary tool used in the Krusty Krab 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 the Krusty Krab training module at kc7cyber.com/krustykrab

On the training module page, you'll see a button that says **Get the Data**. Click this and it will redirect you to ADX! (Note: You'll probably be asked to login with a Microsoft account. You can use an existing personal or organization-issued Microsoft account, or create a new one for free.)



4 hours
Self-guided learning

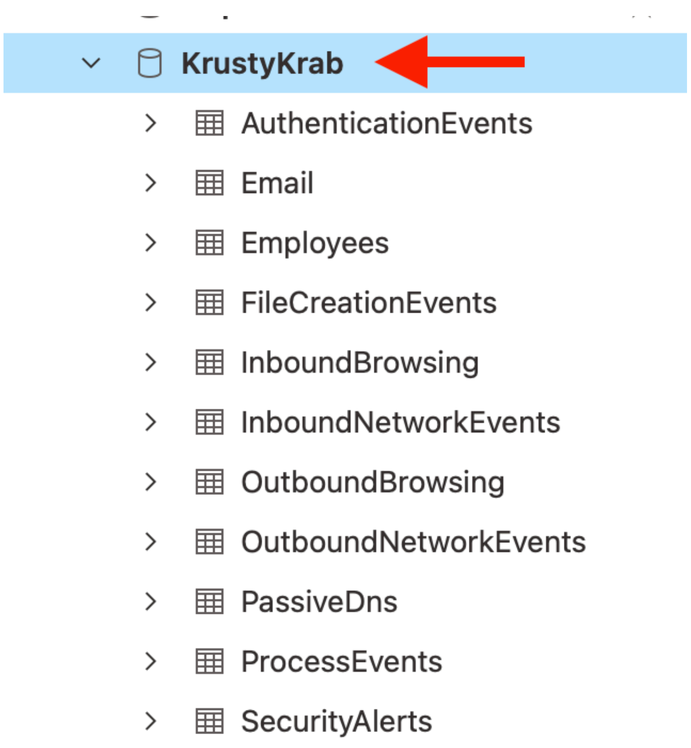
Beginner
Built for anyone

Get started now:
This module is free and self-paced. You can get started now with just a few easy steps!

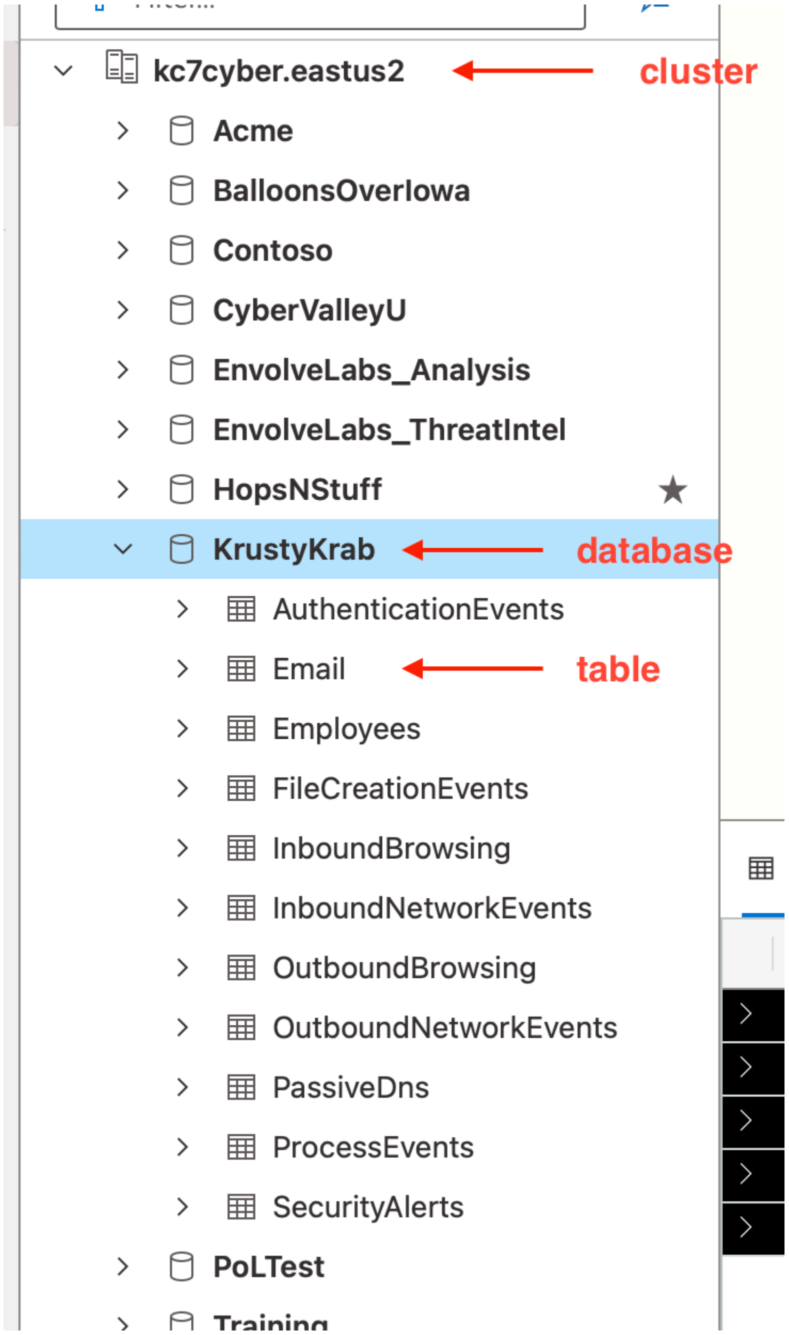
1) Get the Data
The data for this module is hosted in Azure Data Explorer (ADX). You'll need to access it in order to complete this module.

2) Login to the Sco
You'll use the KC7 scoreboa
questions and track your pr
there and enter event code:

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**.

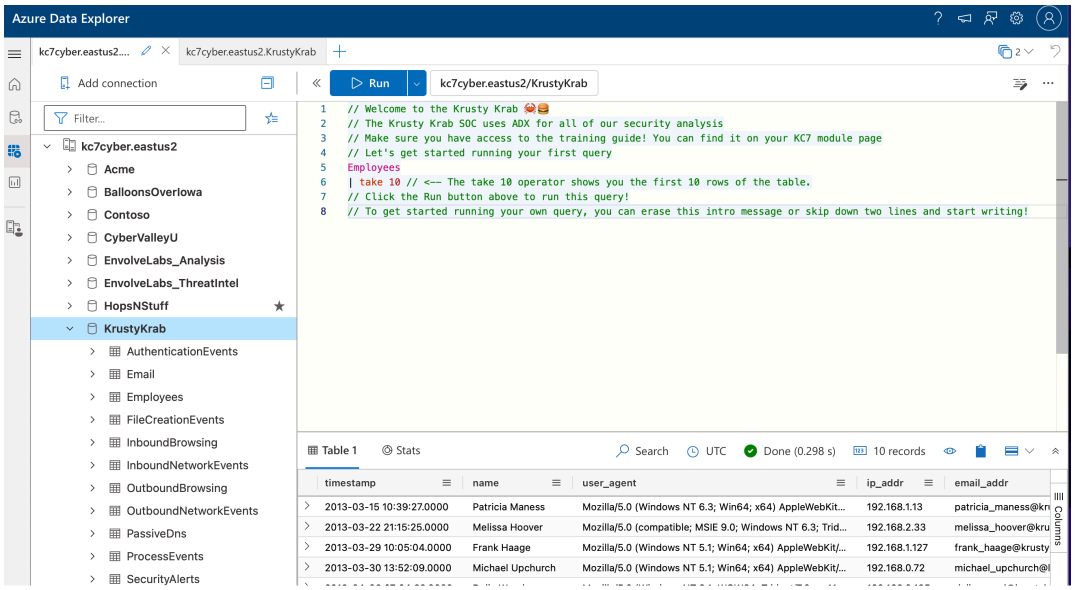


All of Krusty Krab's security logs are stored in a single database – the Krusty Krab database.

2. Select your database.
- Expand the dropdown arrow next to the Krusty Krab database. - Click on the **KrustyKrab** 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.

Note: It's very important that you use the Krusty Krab database for all questions while you're investigating activity at Krusty Krab! If you choose the wrong database, you won't be able to answer questions correctly.

The big 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.



```
// Welcome to the Krusty Krab 🦀🍔
// The Krusty Krab SOC uses ADX for all of our security analysis
// Make sure you have access to the training guide! You can find it on your KC7 module page
// Let's get started running your first query
Employees
| take 10 // <-- The take 10 operator shows you the first 10 rows of the table.
// Click the Run button above to run this query!
// To get started running your own query, you can erase this intro message or skip down two lines and start writing!
```

timestamp	name	user_agent	ip_addr	email_addr
2013-03-15 10:39:27.0000	Patricia Maness	Mozilla/5.0 (Windows NT 6.3; Win64; x64) AppleWebKit...	192.168.1.13	patricia_maness@kru
2013-03-22 21:15:25.0000	Melissa Hoover	Mozilla/5.0 (compatible; MSIE 9.0; Windows NT 6.3; Trid...	192.168.2.33	melissa_hoover@kru
2013-03-29 10:05:04.0000	Frank Haage	Mozilla/5.0 (Windows NT 5.1; Win64; x64) AppleWebKit/...	192.168.1.127	frank_haage@krusty
2013-03-30 13:52:09.0000	Michael Upchurch	Mozilla/5.0 (Windows NT 5.1; Win64; x64) AppleWebKit/...	192.168.0.72	michael_upchurch@l

Currently, you'll see there's a message there welcoming you to Krusty Krab! Click the blue Run button above the query workspace to run your first query! Once you've done that, you can erase the welcome message by highlighting it and pressing backspace or delete on your keyboard.

Okay, enough introductions… let's get your hands on the data.

First Look at the data...

The **KrustyKrab** database contains nine 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 nine types of log data. As you'll see in ADX, each log type corresponds to a table that exists in the **KrustyKrab** database:

Table Name	Description
Employees	Contains information about the company's employees
Email	Records emails sent and received by employees
InboundNetworkEvents	Records inbound network events including browsing activity from the Internet to devices within the company network
OutboundNetworkEvents	Records outbound network events including 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
SecurityAlerts	Records security alerts from an employee's device or the company's email security system

🎯**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.

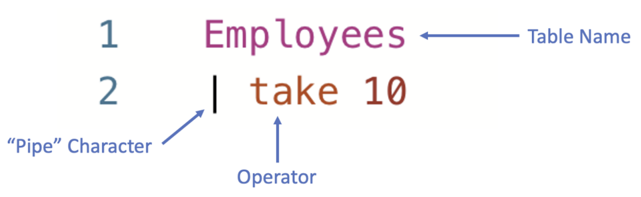
You'll learn more about how to use each of these datasets in just a minute. First, let's just run some queries so you can practice using KQL and ADX.

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:


```
Employees
| take 10
```

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.

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 – If you don't know what to do, first take 10**: 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 "Krusty Krab" and the domain is "krustykrab.com".

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

Make sure you record your answer to all the questions from KQL 101 in the scoreboard at scoreboard.kc7cyber.com.

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 **OutboundNetworkEvents** tables by adding empty lines between them.

```
Email
| take 10

Employees
| take 10

OutboundNetworkEvents
| 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.

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 "Candace" 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	"Candace"

```
Employees
| where name has "Candace"
```

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 == "Candace Ligget"
```

3. 🤔Each employee at Krusty Krab is assigned an IP address. Which employee has the IP address: "192.168.1.191"?

While performing their day-to-day tasks, Krusty Krab 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 "Candace Ligget":

```
Email
| where sender == "candace_ligget@krustykrab.com"
```

4. 🤔How many emails did Hector Duncan 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 "krustykrab.com"
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 "krustykrab.com" 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 "krustykrab.com" 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 distinct senders were seen in the email logs from bikini-bottom-fish.com?

Tracking Down a Click: OutboundNetworkEvents Data

When employees at Krusty Krab browse to a website from within the corporate network, that browsing activity is logged. This is stored in the **OutboundNetworkEvents** table, which contains records of the websites browsed by each user in the company. Whenever someone visits a website, a record of it is 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. When a user visits a site, sometimes data from a lot of other sources are loaded as well. For example, images, assets, and other content may be hosted on content delivery network (CDN), which is used to deliver and load content quickly on a website. Sometimes, advertisements will also load from a particular website as well.

If we want to figure out what websites Kory Burton visited, we can find their IP address from the Employees table.

```
Employees
| where name == "Kory Burton"
```

The query above tells us their IP address is "192.168.2.57". We can take their IP address and look in the **OutboundNetworkEvents** table to determine what websites they visited.

```
OutboundNetworkEvents
| where src_ip == "192.168.2.57"
```

6. 🤔How many unique websites did “Christopher Lynch” 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 “scary”? (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 IP did the domain “scarynight.com” resolve to?

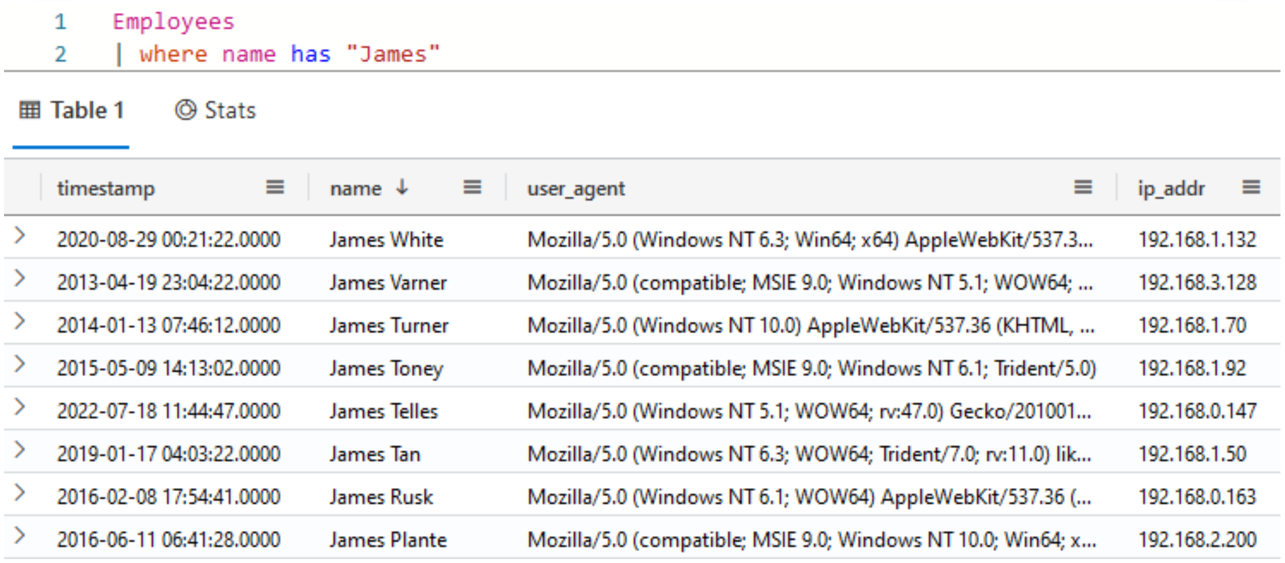
🤯**Let statements – making your life a bit easier:**

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 the next query.

For example, what if we want to look at all the web browsing activity from employees named “James”?

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

```
Employees
| where name has "James"
```



```
Employees
| where name has "James"
```

Table 1 | Stats

timestamp	name	user_agent	ip_addr
2020-08-29 00:21:22.0000	James White	Mozilla/5.0 (Windows NT 6.3; Win64; x64) AppleWebKit/537.3...	192.168.1.132
2013-04-19 23:04:22.0000	James Varner	Mozilla/5.0 (compatible; MSIE 9.0; Windows NT 5.1; WOW64; ...	192.168.3.128
2014-01-13 07:46:12.0000	James Turner	Mozilla/5.0 (Windows NT 10.0) AppleWebKit/537.36 (KHTML, ...	192.168.1.70
2015-05-09 14:13:02.0000	James Toney	Mozilla/5.0 (compatible; MSIE 9.0; Windows NT 6.1; Trident/5.0)	192.168.1.92
2022-07-18 11:44:47.0000	James Telles	Mozilla/5.0 (Windows NT 5.1; WOW64; rv:47.0) Gecko/201001...	192.168.0.147
2019-01-17 04:03:22.0000	James Tan	Mozilla/5.0 (Windows NT 6.3; WOW64; Trident/7.0; rv:11.0) lik...	192.168.1.50
2016-02-08 17:54:41.0000	James Rusk	Mozilla/5.0 (Windows NT 6.1; WOW64) AppleWebKit/537.36 (...	192.168.0.163
2016-06-11 06:41:28.0000	James Plante	Mozilla/5.0 (compatible; MSIE 9.0; Windows NT 10.0; Win64; x...	192.168.2.200

Then, you could manually copy and paste these IPs into a query against the **OutboundNetworkEvents** 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

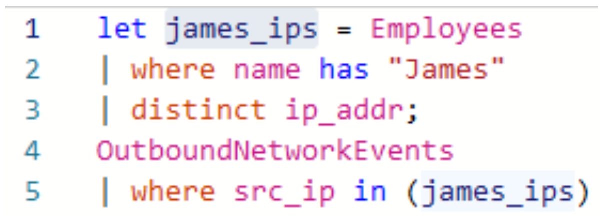
```
OutboundNetworkEvents
| where src-ip in ("192.168.1.132",
    "192.168.3.128",
    "192.168.1.70",
    "192.168.1.92",
    "192.168.0.147",
    "192.168.1.50",
    "192.168.0.163",
    "192.168.2.200")
```

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 “James.”

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.

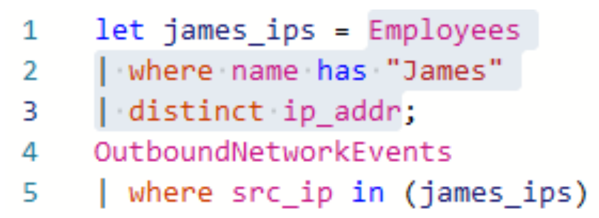
```
let james_ips = Emplyoees
| where name has "James"
| distinct ip_addr;
OutboundNetworkEvents
| where src_ip in (james_ips)
```

On the left of the `let` statement is the variable name (`james_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.



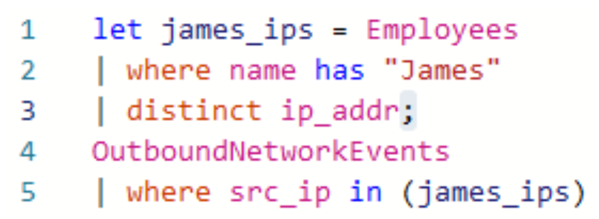
```
let james_ips = Employees
| where name has "James"
| distinct ip_addr;
OutboundNetworkEvents
| where src_ip in (james_ips)
```

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.



```
let james_ips = Employees
| where name has "James"
| distinct ip_addr;
OutboundNetworkEvents
| where src_ip in (james_ips)
```

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



```
let james_ips = Employees
| where name has "James"
| distinct ip_addr;
OutboundNetworkEvents
| where src_ip in (james_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 – OutboundNetworkEvents.

Section 2: Start Hunting!

You've finished your training and you're ready to get to work protecting Krusty Krab.

Work with your team to complete as many challenge questions from the remaining sections in the scoreboard as possible! The goal is to score as many points as you can. There are a lot of questions (the attackers have been busy), so you probably won't be able to answer them all. Just do as many as you can!

As you answer the questions, we will take you on a journey exploring the data and discovering what actions the adversaries have taken. However, you should remember that this is only one of many paths you can take through the data. As you go, don't forget to pay attention to the details along the way. What patterns do the attackers exhibit that could help you track them better? Do they like to use certain words, themes? Or do they make mistakes? Keeping track of these patterns will help you build the full picture of what happened.

Use the provided Actor Preview document to keep track of what you know about the attacker. Building a good profile, timelining the attacker's activity, and forming a list of indicators of compromise (IOCs) will help you keep track of the attacker. KC7 models some of the techniques used by these attackers from real world threat actors, so it may be a helpful resource for you in the future when you are investigating a real security incident.

Now, get out there and keep us safe! The whole company is counting on you. No pressure😊.

Glossary

Watering hole: A type of attack where a hacker compromises a website that is frequently visited by a specific group of users, such as employees of a certain organization, and then infects their devices with malware when they visit the site.

Phishing: A type of attack where a hacker sends an email or other message that appears to be from a legitimate source, such as a bank or a colleague, and tries to trick the recipient into clicking on a malicious link, opening an attachment, or providing sensitive information.

Credential theft: A type of attack where a hacker steals or obtains the username and password of a user or an administrator, and then uses them to access their accounts or systems.

Password spray: A type of attack where a hacker tries to guess the passwords of multiple accounts by using common or weak passwords, such as “password” or “123456”, instead of targeting one account with many password attempts.

Reconnaissance: The process of gathering information about a target system, network, organization, or user before launching an attack. This can include scanning for open ports, identifying vulnerabilities, mapping network topology, collecting email addresses, etc.

Supply chain compromise: A type of attack where a hacker infiltrates the software development process or distribution channel of a trusted vendor or partner and inserts malicious code into their products or services. This way, the hacker can compromise the customers or users who install or use those products or services.

Malware: A general term for any software that is designed to harm or disrupt a system, network, device, or data. Malware can include viruses, worms, trojans

Command and control: A type of server or network that is used by hackers to communicate with and control their malware or botnets on compromised systems, networks, or devices. Command and control servers can send commands, receive data, update malware, or launch attacks.

Adversary in the middle: A type of attack where a hacker intercepts and modifies the communication between two parties, such as a user and a website, without their knowledge. Adversary in the middle attacks can be used to steal or alter data, redirect traffic, inject malware, or impersonate either party.

Top level domain: The highest level of domain names in the Internet’s domain name system.

Top level domains are the last part of a domain name after the dot, such as .com, .org, .edu, etc. Top level domains can indicate the type or purpose of a website or its geographic location.

Resources

Understanding KQL operators: https://learn.microsoft.com/en-us/azure/data-explorer/kusto/query/datatypes-string-operators