

# Working with AWS CloudTrail

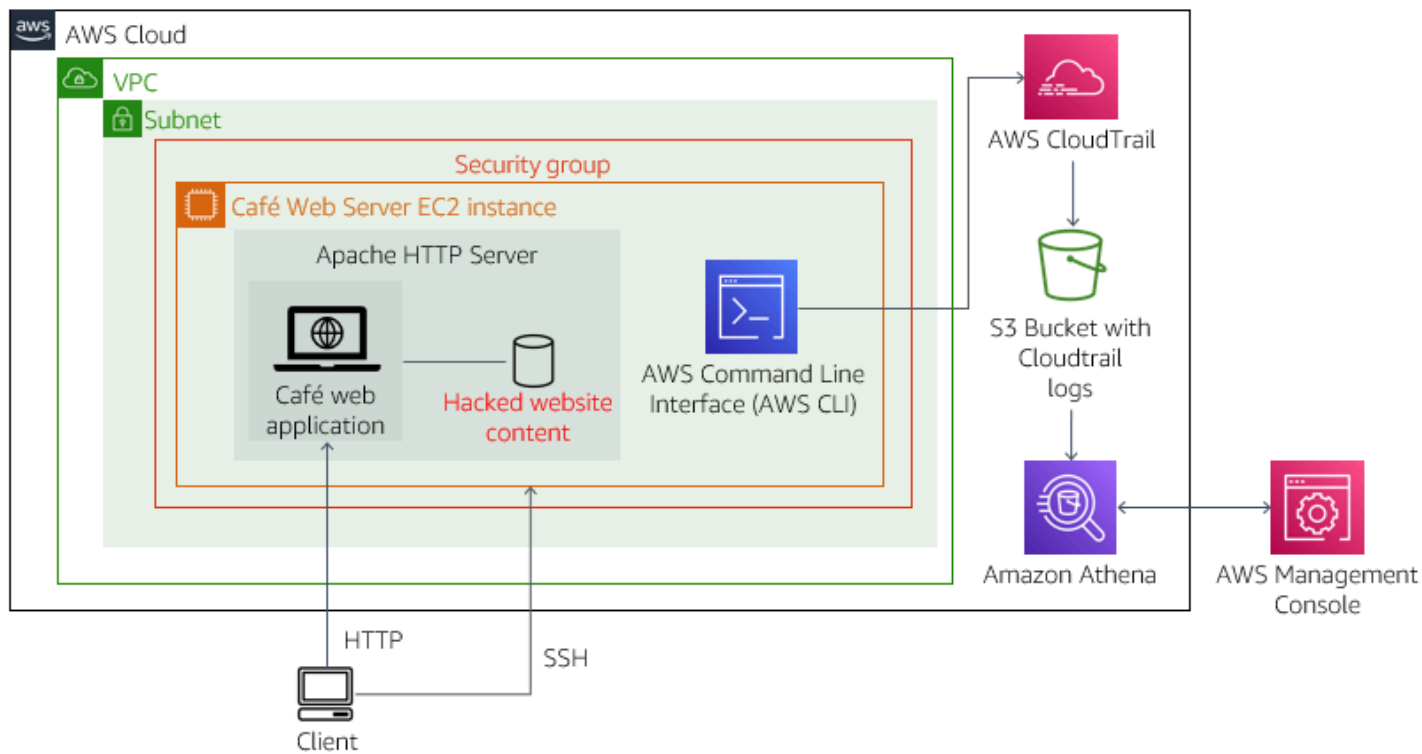
## Activity overview

In this activity, you create an AWS CloudTrail trail that audits actions taken in your account. You then investigate to determine who modified the Café website.

The activity starts with an Amazon Elastic Compute Cloud (Amazon EC2) instance named **Café Web Server**, which runs a web application that hosts the Café website.

- In **Task 1**, you observe that the website looks normal.
- In **Task 2**, soon after you create a trail with CloudTrail, you notice that the website has been hacked and that part of the hack involved an action during which someone modified the security group settings.
- In **Task 3**, you use a variety of methods to analyze the CloudTrail logs, including the Linux grep utility and the AWS Command Line Interface (AWS CLI).
- In **Task 4**, you use Amazon Athena to search the CloudTrail logs.
  - In the **Challenge** section that concludes Task 4, you work to identify the hacker.
- In **Task 5**, now that you have discovered the culprit, you remove that user's access. You also take steps to reduce the chances that the AWS account and the Café website will be hacked again.

The architectural diagram illustrates the setup that this activity uses.



### Duration

This lab requires approximately **75 minutes** to complete.

## Activity objectives

After completing this activity, you will be able to:

- Configure a CloudTrail trail
- Analyze CloudTrail logs by using various methods to discover relevant information
- Import CloudTrail log data into Athena
- Run queries in Athena to filter CloudTrail log entries
- Resolve security concerns within the AWS account and on an EC2 Linux instance

# Business case relevance

A new request from the Café leadership team



Martha and Frank are concerned because the website was hacked. They are relying on you to discover who did it and to make sure that it does not happen again.

Faythe, Frank, Martha, and others make frequent changes to the website, and sometimes those changes cause issues. Also, this morning, it looks like the website was hacked. Martha and Frank are asking Sofia if there is a way to track what was changed and who made the changes.

Play the role of Sofia, become a detective, and discover the culprit.

## Activity steps

### Launching the activity environment

1. At the top of these instructions, select **Start Lab** to launch your lab.  
  
A Start Lab panel opens displaying the lab status.
2. Wait until you see the message **Lab status: ready**, and then select the **X** to close the Start Lab panel.
3. At the top of these instructions, select **AWS**  
  
This step opens the AWS Management Console in a new browser tab. The system automatically logs you in.  
  
**Tip:** If a new browser tab does not open, a banner or icon at the top of your browser typically indicates that your browser is preventing the site from opening pop-up windows. Select the banner or icon, and choose **Allow pop-ups**.
4. Arrange the AWS Management Console tab so that it displays alongside these instructions. Ideally, you should be able to see both browser tabs at the same time to make it easier to follow the lab steps.

### Task 1: Modifying a security group and observing the website

5. From the **Services** menu, choose **Compute** then the **EC2** service.
6. Choose **Instances**, and then locate and select the **Café Web Server (WebSecurityGroup)** instance.
7. In the **Security** tab, choose the **sg-xxxxxxxxxx** security group.
8. In the **Inbound rules** tab, notice that only one inbound rule has been defined, which is for HTTP access over TCP port 80.
9. Choose **Edit inbound rules**, and then choose **Add rule** and configure the rule as follows:
  - **Type:** Select **SSH**
  - **Port Range:** Enter **22**
  - **Source:** Enter **My IP**
- Important:** Confirm that the TCP port 22 access will be open to only your IP address. The entry should show a Classless Inter-Domain Routing (CIDR) block that has a particular IP address followed by /32, not to all IP addresses (which would be shown by 0.0.0.0/0).
10. At the bottom of the page, choose **Save rules**.
11. Observe the Café website:
  - Choose **Instances**, select the **Café Web Server** instance. Click the **Details** tab and copy the **Public IPv4 address** value
  - Open a new browser tab, and navigate to `http://<WebServerIP>/cafe/` (substitute the `<WebServerIP>` value).


- Notice that the website looks normal. For example, the photos are all appropriate for a bakery café.

## Task 2: Creating a CloudTrail log and observing the hacked website

---

In this task, you create a CloudTrail trail in your AWS account. You also notice that soon after creating the trail, the Café website is hacked.

### Task 2.1: Create a CloudTrail log

12. In the AWS Management Console, from the **Services** menu, select **Management & Governance** then **CloudTrail**. Ignore the AWS Organizations access denied message at the top of the console.
13. On the navigation pane on the left, choose **Trails**.  
  
If the navigation pane is not displayed, choose the three icon of the horizontal lines  on the top left of the screen.  
If you encounter the following warning *The option to create an organization trail is not available for this AWS account*, you can ignore it.
14. Choose **Create trail**
15. Configure the trail as follows:
  - For **Trail name**, enter `monitor`  
**Important:** Verify that you set the **Trail name** to `monitor`, or this activity will not work as intended.
  - Select ☒ Create a new S3 bucket.
  - For **Trail log bucket and folder**, enter `monitoring####` (the `####` characters are four random digits).
  - For **AWS KMS alias**, enter your initials followed by `-KMS` (for example, `kc-KMS`).
16. Choose **Next**
17. On the **Choose log events** page, choose **Next**
18. On the **Review and create** page, choose **Create trail**
19. Verify that you see your trail on the **Trails** page.

### Task 2.2: Observe the hacked website

20. Return to the browser tab where you have the Café website open, and refresh the page.  
  
**Important:** You might need to wait a full minute before the hack will occur. Also, your browser may be caching the images on this website. Press and hold Shift while you also choose the browser refresh button in order to see the latest changes to the website.  
  
Notice that the website has been hacked. Who put that image there? The image certainly does not look correct.  
  
It is up to you to figure out who hacked the website.  
  
It is good that you enabled CloudTrail before this happened. CloudTrail can give you valuable information about what users have been doing in your account.
21. In the AWS Management Console, browse to the **EC2** service, and observe the **Café Web Server** instance details.  
  
Does anything look suspicious?
22. In the **Security** tab, choose the `sg-xxxxxxxxxx` security group again, and then choose the **Inbound rules** tab.  
  
Where did that extra entry come from?  
  
You still see the entry you created earlier: the rule that opens port 22 to only your IP address. However, you also now see that someone else created an additional inbound rule that allows Secure Shell (SSH) access from anywhere (0.0.0.0/0).  
  
Who added this security hole? You can search the CloudTrail logs to find out.

## Task 3: Analyzing the CloudTrail logs by using grep

---

In this task, you analyze the CloudTrail logs by using the grep Linux utility to see if you can figure out who hacked the website.

## Task 3.1: Connect to the Café Web Server host EC2 instance by using SSH

In this task, you connect to the Café Web Server EC2 instance. You use SSH to connect to the instance.

Windows users should follow Task 3.2 for Windows. Both macOS and Linux users should follow Task 3.2 for macOS/Linux.

[macOS/Linux users: visit this link for login instructions](#)

## Task 3.2 for Windows: SSH

🗨 These instructions are specifically for Windows users. If you are using macOS or Linux, [skip to the next section](#).

23. Select the  drop-down menu above these instructions you are currently reading, and then select . A Credentials window will be presented.
24. Select the **Download PPK** button and save the **labsuser.ppk** file.  
*Typically your browser will save it to the Downloads directory.*
25. Make a note of the **PublicIP** address.
26. Then exit the Details panel by selecting the **X**.
27. Download **PuTTY** to SSH into the Amazon EC2 instance. If you do not have PuTTY installed on your computer, [download it here](#).
28. Open **putty.exe**
29. Configure your PuTTY session by following the directions in the following link: [Connect to your Linux instance using PuTTY](#)
30. Windows Users: [Select here to skip ahead to the next task](#).

## Task 3.2 for macOS/Linux: SSH

These instructions are for Mac/Linux users only. If you are a Windows user, [skip ahead to the next task](#).

31. Read through the three bullet points in this step before you start to complete the actions because you will not be able see these instructions when the **Details** panel is open.
  - Choose the  dropdown menu above these instructions, and then choose . A Credentials window will open.
  - Chose the **Download PEM** button, and save the **labsuser.pem** file.
  - To exit the **Details** panel, choose the **X**.
32. Open a terminal window, and change the `cd` directory to the directory where the labsuser.pem file was downloaded.  
  
For example, run the following command if the file was saved to your Downloads directory:

```
cd ~/Downloads
```

33. To change the permissions on the key to be read only, run the following command:

```
chmod 400 labsuser.pem
```

34. Return to the AWS Management Console, and in the EC2 service, choose **Instances**. Select the check box next to the Café Web Server instance, and choose the **Description** tab.
35. Copy the **IPv4 Public IP** value.
36. Return to the terminal window, and run the following command (replace **<public-ip>** with the actual public IP address you copied):

```
ssh -i labsuser.pem ec2-user@<public-ip>
```

37. When prompted, type  to allow a first connection to this remote SSH server.  
  
Because you are using a key pair for authentication, you will not be prompted for a password.

## Task 3.3: Download and extract the CloudTrail logs

38. Verify that your terminal is connected via SSH to the Café Web Server EC2 instance.

39. Run the following command to create a local directory on the web server to download the CloudTrail log files to:

```
mkdir ctraillogs
```

40. Run the following command to change the directory to the new directory:

```
cd ctraillogs
```

41. Run the following command to list the buckets to recall the bucket name:

```
aws s3 ls
```

42. In the command below, replace *<monitoring####>* with the actual bucket name that starts with **monitoring** (the bucket name is part of the output from the `ls` command that you ran). Run the adjusted command to download the CloudTrail logs:

```
aws s3 cp s3://<monitoring####>/ . --recursive
```

If the command is successful, you should see that a few log files are downloaded.

**Important:** If there was no output in the command line when you ran the last command, it likely means that not enough time has passed since you created the CloudWatch trail. CloudWatch posts logs to Amazon Simple Storage Service (Amazon S3) every 5 minutes. You might need to wait and try running the command again. Do not proceed to the next step until you have downloaded at least one log file.

43. Use the `cd` and `ls` commands repeatedly (or enter `cd` and then press Tab multiple times) as necessary to change the directory to the subdirectory where the logs were downloaded. When you run `ls`, all of the downloaded log files should display. They will be located in an **AWSLogs/<account-num>/CloudTrail/<Region>/<yyyy>/<mm>/<dd>** subdirectory.

Notice that the log files end in `.json.gz`, which indicates that they are compressed as GNU zip files.

44. Run the following command to extract the logs:

```
gunzip *.gz
```

45. Run `ls` again. Notice that all files are now extracted.

## Task 3.4: Analyze the logs by using grep

In this section of the activity, you use the Linux `grep` utility to analyze the CloudTrail logs.

46. To analyze the structure of the logs, do the following:

- Copy one of the file names returned by the `ls` command that you ran.
- Enter `cat` in the terminal window, followed by a space, and then paste the copied file name. Run the command.
- Note that the files are in JavaScript Object Notation (JSON) format. However, it is difficult to read them in this output format.
- Run the `cat` command again, but this time format the output (replace *<filename.json>* with the actual file name):

```
cat <filename.json> | python -m json.tool
```

This format is more readable. You can now also see the structure of the log entries. Notice that each entry contains the same standard fields, including `awsRegion`, `eventName`, `eventSource`, `eventTime`, `requestParameters`, `sourceIPAddress`, `userIdentity`, and more.

The graphic below shows an example log entry.

```
{
  "awsRegion": "eu-west-2",
  "eventID": "81afe62b-c0a1-4878-8f8e-331a7f6e21e9",
  "eventName": "DescribeVpcs",
  "eventSource": "ec2.amazonaws.com",
  "eventTime": "2019-04-23T17:50:49Z",
  "eventType": "AwsApiCall",
  "eventVersion": "1.05",
  "recipientAccountId": "603498349594",
  "requestID": "6509d682-eb54-4e45-8421-a686bd306536",
  "requestParameters": {
    "filterSet": {},
    "vpcSet": {}
  },
  "responseElements": null,
  "sourceIPAddress": "52.94.36.14",
  "userAgent": "signin.amazonaws.com",
  "userIdentity": {
    "accessKeyId": "ASIAYZA20AQNJZJCITXB",
    "accountId": "603498349594",
    "arn": "arn:aws:iam::603498349594:user/awsstudent",
    "invokedBy": "signin.amazonaws.com",
    "principalId": "AIDAI3WOCQLIBJOITYFAQ",
    "sessionContext": {
      "attributes": {
        "creationDate": "2019-04-23T17:27:30Z",
        "mfaAuthenticated": "false"
      }
    },
    "type": "IAMUser",
    "userName": "awsstudent"
  }
}
```

You can now read the log entries. However, the number of entries—even in just this one log file—can be large. You might have downloaded more than one log because new log files are created over time. You need to find a way to search log entries across multiple files and also filter the results.

47. Consider how you want to target the search. You are not interested in everything that is happening in this account. Instead, your interest is in an action that was taken on a particular EC2 instance (that is, the web server that was hacked).

Start by filtering the log results where the **sourceIPAddress** matches the IP address of the Café Web Server instance.

Run the following command to set the `WebServerIP` address as a variable that you can use in future commands (replace `<WebServerIP>` with the actual IP address that displays to the left of these instructions):

```
ip=<WebServerIP>
```

48. Run the following command:

```
for i in $(ls); do echo $i && cat $i | python -m json.tool | grep sourceIPAddress ; done
```

The command you ran does the following:

- It creates a **for** loop that includes the names of the files in the current directory.
- During each iteration of the for loop, it echoes the file name and then prints the contents of the file in JSON format.
- Only the lines of JSON that contain the `sourceIPAddress` tag are printed.

Note that there are several log entries in the trail where the **sourceIPAddress** was the Café Web Server instance.

49. Run a similarly structured command but where the command returns the **eventName** of every captured event:

```
for i in $(ls); do echo $i && cat $i | python -m json.tool | grep eventName ; done
```

The command you ran follows the same logic as the command you ran before, but this time, it filters log entries for the **eventName**.

The results of the previous command contain different details. Many **describe** and **list** actions were recorded, and they look relatively harmless. However, if you scroll through the list, you notice that occasional **update** actions were also recorded. You could use a text editor like `vi` to open a log that contains a recorded event that you want to know more about. You can then search for that `eventName` and look at the details.

However, you might benefit from using a different tool other than `grep` to locate these log entries more easily.

## Task 3.5: Analyze the logs by using AWS CLI CloudTrail commands

Another approach you can use to analyze CloudTrail logs is to use AWS CLI CloudTrail commands.

50. Open the [AWS CLI Reference page for CloudTrail](#).

51. Choose the **lookup-events** command to see details about the command.

- Notice that you can look up events based on one of eight different attributes, including AWS access key, event name, user name, and others.
- In the AWS CLI Command Reference page, scroll to the **Example**, which shows how to filter the trail for console logins. Run that command in your terminal window:

```
aws cloudtrail lookup-events --lookup-attributes AttributeKey=EventName,AttributeValue=ConsoleLogin
```

The results indicate there have been no console login events or that the only user who has logged into the console is the same user that you are logged into the console as

However, there are other ways to modify resources on AWS instead of using the console. The hacker might have used a different approach.

52. Run the following command to find any actions that were taken on security groups in the AWS account:

```
aws cloudtrail lookup-events --lookup-attributes AttributeKey=ResourceType,AttributeValue=AWS::EC2::SecurityGroup --output text
```

Something in this result set might contain some information that would help you discover what happened, but there might be too many results for you to easily identify the issue.

Perhaps you can narrow the search results further so that you get only the results related to the security group that is used by the web server instance.

53. Run the following commands to find the security group ID that is used by the Café Web Server instance, and then echo the result to the terminal:

```
region=$(curl http://169.254.169.254/latest/dynamic/instance-identity/document|grep region | cut -d '"' -f4)
sgId=$(aws ec2 describe-instances --filters "Name=tag:Name,values='Cafe Web Server'" --query 'Reservations[*].Instances[*].SecurityGroups[*].[GroupId]' --region $region --output text)
echo $sgId
```

Notice that a single security group ID was found.

54. Now use the security group ID that the previous command returned to further filter your AWS CLI CloudTrail command results:

```
aws cloudtrail lookup-events --lookup-attributes AttributeKey=ResourceType,AttributeValue=AWS::EC2::SecurityGroup --region $region --output text | grep $sgId
```

You could keep experimenting with different commands to filter the log results. However, you might wonder whether there is a better tool or solution for reading these logs. AWS has the AWS Partner Network (APN), where companies specialize in helping AWS customers with this challenge. See <https://aws.amazon.com/cloudtrail/partners/> for a listing of APN Partner solutions.

The APN Partner solutions suit the needs of many AWS customers. However, for the purposes of this activity, there is one additional approach to examining CloudTrail log files that you might use, and it uses another AWS service. In the next task, you explore CloudTrail logs by using Athena.

## Task 4: Analyzing the CloudTrail logs by using Athena

As you experienced in the previous task, it can be difficult to find specific information within a very large dataset. CloudTrail logs are verbose for a reason: you might want to know every relevant detail about a particular action that was taken in your AWS account. However, using command line tools to filter the logs can be tedious.

It would be convenient if all the log data were in a database and you could use structured query language (SQL) queries to search for the log entries that you are most interested in. Athena provides such a solution. Athena is an interactive query service that makes it easy to analyze data in Amazon S3 by using standard SQL.

In this task, you use Athena to analyze your CloudTrail logs.

## Task 4.1: Create the Athena table

55. From the AWS Management Console **Services** menu, choose **CloudTrail** to open the CloudTrail console.
56. In the navigation pane, choose **Event history**.
- Notice that CloudTrail provides this event history interface where you can apply filters and conduct a basic search based on parameters, such as **Event name** or **Resource type**. The **Event history** page can be a useful tool, and you are free to explore it. However, in this activity, you use Athena.
57. From the **Event history** page, click **Create Athena table**
- **Storage location:** Choose the **monitoring####** S3 bucket where you configured CloudTrail to store log files.
58. Take a moment to analyze how the Athena **CREATE TABLE** statement is formed.
- It creates a database column for each of the standard name-value pairs in each JSON-formatted CloudTrail log entry. Refer to the image of the JSON format of a typical log entry in Task 3.4 to confirm this information.
  - At the bottom of the **CREATE TABLE SQL** statement, notice the **LOCATION** statement. This indicates the Amazon S3 location where the table data will be stored. In this case, the data is already there. You are defining the table schema that will be used to parse existing JSON-structured data.
  - For details on CloudTrail record structure, see <https://docs.aws.amazon.com/awscloudtrail/latest/userguide/cloudtrail-event-reference.html>.
  - For details on how this Athena table was created, see the **CREATE EXTERNAL TABLE** document at <https://docs.aws.amazon.com/athena/latest/ug/cloudtrail-logs.html>.
59. After you are done analyzing the **CREATE TABLE** details, choose **Create table**.
- The table is created with a default name that includes the name of the S3 bucket.
60. From the **Services** menu, choose **Analytics** then the **Athena** service.

## Task 4.2: Analyze logs using Athena

The advantage of using Athena is that you can now run SQL queries over your log data.

61. If you do not already see the **Athena Query Editor**, choose **Explore query editor** and it should then display.
- If a **Tutorial** screen appears, choose the **X** in the top corner to exit out of it.
62. In the left panel of the **Athena Query Editor**, you should see the **cloudtrail\_logs\_monitoring####** table.
- Select **+** beside the table to reveal the column names.
- Analysis:** Notice how each standard child element that exists in a CloudTrail log record in JSON format has a corresponding column name in this database. The **useridentity** database column is a struct type, because it contains more than a single name-value pair. Similarly, the **resources** database column is an array.
63. Start by setting up a query results location and then running a simple query to get an idea of the data that is available in the logs.
- On the menu bar at the upper right of the page, choose **Settings** followed by **Manage**.
- Set **Location of query result** to `s3://monitoring####/results/` and replace *monitoring####* with the name of the bucket you created earlier.
  - Choose **Save**.
  - Select the **Editor** table and paste the following SQL query into the **Query 1** panel. Replace *####* with the numbers in your actual table, and choose **Run**.

```
SELECT *
FROM cloudtrail_logs_monitoring####
LIMIT 5
```

This query returns five rows of data. Look at the result set (scroll to the right in the **Results** panel to see additional column data).

Focus on the columns **useridentity**, **eventtime**, **eventsource**, **eventname**, and **requestparameters**, which contain the most valuable information to help you find the origin of the hack.

The **useridentity** column has many details that make it more difficult to read though. You now return only the user name for that column.

64. Run a new query that selects only those columns that were previously mentioned. This time, limit the results to 30 rows:



```
SELECT useridentity.userName, eventtime, eventsource, eventname, requestparameters
FROM cloudtrail_logs_monitoring####
LIMIT 30
```

You should now be able to find out who modified the security group that is associated with the Café Web Server instance.

## Challenge: Identify the hacker

In this section of the activity, you try to discover the log entry that includes the essential information about who hacked the website. Specific steps are not provided. Instead, you must experiment with running different queries until you find the information that you are looking for.

Tips:

- **Tip 1:** Look at the data that was returned by the last command that you ran. Even if none of the log entry details that display are the log entry you are looking for, they still give you an indication of what kinds of data the different columns contain. Don't be afraid to experiment with running modified SQL queries. Choose the **+** icon next to **New query 1** to create a second query tab. This way, you can preserve older queries without deleting them.
- **Tip 2:** Try filtering by events that are related to the Amazon EC2 service. Remember that you can add **WHERE** clauses, such as `WHERE eventsource = 'ec2.amazonaws.com'`
- **Tip 3:** To ensure you are querying the entire log set, remove the **LIMIT** clause from your query.
- **Tip 4:** Take a look at the kind of data that is captured in the **eventname** column. Can you further refine your SQL query so that it looks for only events that contain the word `Security`? Remember that SQL allows you to use compound **WHERE** clauses that look for pattern matches (for example, `WHERE columnName = 'some value' AND otherColumnName LIKE '%part of some value%'`).
- **Tip 5:** After you have successfully filtered all security-related actions in the log, analyze the **eventnames** further. Do any of them look suspicious? Can you adjust the **WHERE** clause to search for a particular **\*eventname**?
- **Tip 6:** If you are still looking for the entry that shows who opened port 22 to the world, here is a general query that is often useful to run. This query might help identify the action:

```
SELECT DISTINCT useridentity.userName, eventName, eventSource FROM cloudtrail_logs_monitoring#### WHERE
from_iso8601_timestamp(eventtime) > date_add('day', -1, now()) ORDER BY eventSource;
```

This query returns a list of all users who were active in the account in the past day and the distinct actions they have taken.

You have successfully completed the challenge if you can identify the following information:

- The name of the AWS user who created the security hole in the Café Web Server security group
- The exact time that they hacked the security group
- The IP address from which they hacked it (copy this value to a text file for later reference)
- The method they used to perform the hack (console or programmatic access)

Congratulations! You have successfully uncovered the identity of the hacker!

## Task 5: Analyzing the hack further and improving security

In this last task, you work to secure both your AWS account and the web server instance.

### Task 5.1: Check the OS users

65. In the terminal where you have an active SSH session to the web server instance, run the following command to find out who has recently logged into this operating system (OS):

```
sudo aureport --auth
```

There is evidence that a user other than ec2-user has logged in. Who is that chaos-user?

66. Run the **who** command to figure out who is currently logged in:

```
who
```

The user is still logged in! Get them off this instance right away!

67. Run the following command to try to remove the chaos-user OS user:

```
sudo userdel -r chaos-user
```

That didn't work because they are still logged in. However, it did return the process number they are connected as.

68. In the command below, replace **ProcNum** with the process number returned by the last command. Run the adjusted command to stop the process that has the active chaos-user login session:

```
sudo kill -9 ProcNum
```

69. Run the **who** command again to verify that the chaos-user OS user is no longer connected:

```
who
```

Now you (the ec2-user) should be the only user connected.

70. Run the following command to try to delete the chaos-user again:

```
sudo userdel -r chaos-user
```

It should succeed this time.

71. Run the following command to verify no other suspicious OS users who can login:

```
sudo cat /etc/passwd | grep -v nologin
```

Note that the grep part of the command you just ran filtered out the OS users who do not have a login.

The **root**, **sync**, **shutdown**, and **halt** users are all standard OS users in Amazon Linux, so there are no other concerning user logins on this instance.

## Task 5.2: Update SSH security

72. Analyze SSH settings on the instance.

You have removed the OS user who hacked into the instance, but how did they manage to connect to the EC2 instance by using SSH in the first place?

You have been careful about who has access to the key pair file. However, maybe you should check the SSH settings on this instance.

```
sudo ls -l /etc/ssh/sshd_config
```

Notice the last modified timestamp for the file. This file was modified today! That is concerning.

73. Run the following command to edit the SSH configuration file in the VI editor:

```
sudo vi /etc/ssh/sshd_config
```

- Analyze the details of this file. Enter `:set number` to see the line numbers in this file.
- Notice on line 61 that password authentication is enabled. That is definitely not a security best practice! That means that anyone who knows (or can correctly guess) the username and password combination of an OS user can remotely access this instance without using an SSH key pair. This setting needs to be corrected.
- Move your cursor (using the arrow up or down keys) to the **PasswordAuthentication yes** line and comment it out.

**Tip:** Enter `a` on your keyboard to enter **edit mode** in VI, and add a `#` character at the start of the line.

- Next, move your cursor to the **#PasswordAuthentication no** line (line 63) by using the arrow keys and uncomment this line (remove the `#` character).
- Choose the **Esc key** on your keyboard to exit edit mode.
- **Save** the changes, and exit the VI editor using the `:wq` command.

74. Run the following command to restart the SSH service so that the changes go into effect:

```
sudo service sshd restart
```

Note: If running the command above interrupts your SSH connection, reestablish the SSH connection before continuing on to the next step.

75. Finally, in the EC2 console, return to the Web Server security group settings.

With the Web Server security group selected, go to the **Inbound** tab, and choose **Edit**.

Delete the inbound rule that allows port 22 access from 0.0.0.0/0 (the one the hacker created).

76. **Save** the change.

Nice work! You have kicked the hacker out of this instance and remove the login account that they used. You also updated the SSH settings so that only users who have the correct key pair and the same source IP address as you can connect to it.

## Task 5.3: Fix the website

Now that the hacker no longer has access to this instance, you can fix the issue with the website.

77. Run the following command to navigate to the directory where the website image files are held and review the contents:

```
cd /var/www/html/cafe/images/  
ls -l
```

It looks like the hacker created a backup of the original file.

78. Run the following command to restore the original graphic on the website.

```
sudo mv Coffee-and-Pastries.backup Coffee-and-Pastries.jpg
```

79. To test the fix, reload the <http://WebServerIP/cafe> website in the browser.

**Tip:** You may need to press and hold the Shift key and choose the browser refresh button to see the change.

That looks better!

## Task 5.4: Delete the AWS hacker user

Recall that the hacker not only accessed the EC2 instance hosting the website but also managed to run an AWS CLI command that opened port 22 in the security group to the entire internet. In this step, you remove the chaos AWS Identity and Access Management (IAM) user from the account.

80. In the AWS Management Console, choose the **Services** menu, and choose **IAM**.

81. Choose the **Users** link, and select the check box next to the chaos user.

82. Choose **Delete**, enter the users name and select **Delete**.

Nice work! That chaos user shouldn't be causing any trouble in the AWS account anymore.

**Update from Café**



Everyone at the Café is relieved that Sofia was able to uncover the identity of the person who committed the hack and remove their access to the web server and to the AWS account.

In the end, the team members were lucky that it looks like the hacker was just trying to have fun. However, they all know that the hacker could have caused serious damage. Everyone on the team at the Café who participates in updating and maintaining the website now knows how important it is to keep the site secure. They are also definitely going to continue to use CloudTrail as a key tool for auditing activity on their AWS account.

# Activity complete

---

Congratulations! You have completed the lab.

83. Choose  at the top of this page, and then choose  to confirm that you want to end the lab.

A panel appears indicating that **DELETE has been initiated... You may close this message box now.**

84. Choose the **X** in the top-right corner to close the panel.

## Additional resources

---

For more information about AWS Training and Certification, see <https://aws.amazon.com/training/>.

*Your feedback is welcome and appreciated.*

If you would like to share any suggestions or corrections, please provide the details in our [AWS Training and Certification Contact Form](#).

© 2022 Amazon Web Services, Inc. and its affiliates. All rights reserved. This work may not be reproduced or redistributed, in whole or in part, without prior written permission from Amazon Web Services, Inc. Commercial copying, lending, or selling is prohibited.