Creating a dashboard with SumoLogic to monitor suspicious AWS console logins

dev.classmethod.jp/articles/sumologic_custom_dashboard

江口佳記

October 31, 2019

This is Eguchi from the Operations Department.

I previously wrote an introductory article on using SumoLogic to obtain and visualize CloudTrail information from our core AWS accounts.

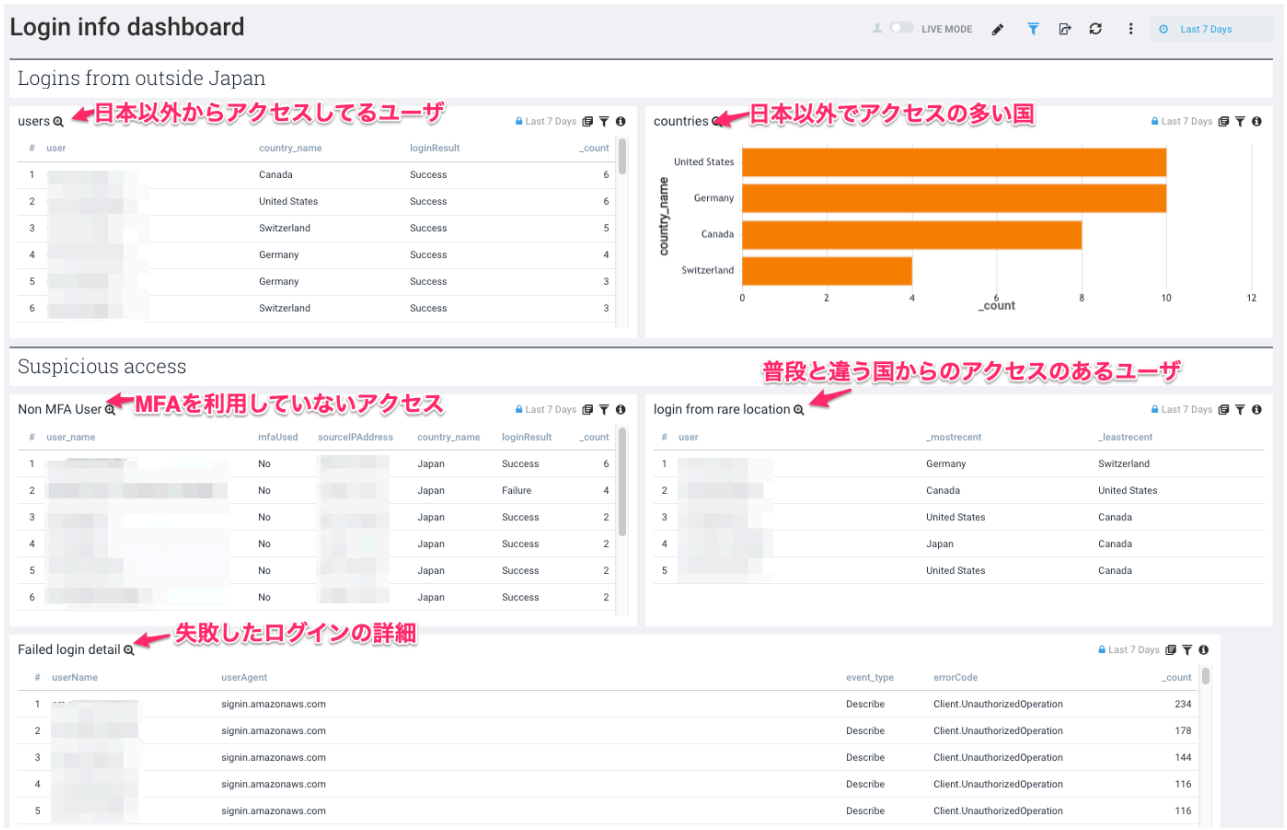
> Visualizing information about our company's core AWS environment with Sumo Logic (CloudTrail Edition)

The article above introduces how to dig deeper and investigate by writing queries whenever you notice something of interest. However, in actual operation, you will want to have a comprehensive dashboard so that you can immediately identify problems. One of the attractions of SumoLogic is that it has a fairly comprehensive default dashboard, but this time I would like to try customizing the dashboard to suit my own purposes. The purpose is, as the title of the article suggests, to "monitor suspicious AWS console logins."

I will also introduce the queries for each panel, so I hope that it will be helpful for you to use queries in Sumo (although I am still not very familiar with Sumo queries, so they may not be very neat).

The dashboard I created

I'm still experimenting, but the current state of things is as follows. It's a tool that can visualize things, but it's a bit strange that it's mostly just a table...



There are comments in the images, but we will explain each panel below.

Logins from outsite Japan

Displays information about access from outside Japan. Our company has overseas branches, and employees on business trips abroad also access the site, so access from overseas does not necessarily mean it is suspicious. However, since the majority of our employees are in Japan, it is still meaningful to check the access from overseas and its frequency.

users

The combination of user, country, and login success/failure is displayed in descending order of access. This information is displayed on the assumption that if an access attempt is made from a region other than your usual location and the login fails, it is highly suspicious. The query is as follows. `| where country_code<>"JP" and !isNull(latitude)`If you delete this command, access from Japan will also be included in the calculation.

```
_sourceCategory = cloudtrail  "ConsoleLogin"
| parse "\"eventName\":\"*\"" as eventName nodrop
| where eventName="ConsoleLogin"
| json "sourceIPAddress"
| parse "\"userName\":\"*\"" as user_name nodrop
| json field=_raw "userIdentity.principalId" as principal_id nodrop
| parse regex field = principal_id ":(?<user_principal>.+)" nodrop
| if (user_name="", user_principal, user_name) as user
| json field=_raw "responseElements.ConsoleLogin" as loginResult nodrop
| parse "\"MFAUsed\":\"*\"" as mfaUsed nodrop
| count by sourceIPAddress, user, loginresult
| lookup latitude, longitude, country_code, country_name, region, city, postal_code
from geo://location on ip = sourceIPAddress
| where country_code<>"JP" and !isNull(latitude)
| fields user, country_name, loginresult, _count
| sort by _count, country_name asc, user, loginresult
```

counties

This is a count of access numbers by country. You should be cautious if there are a lot of accesses from regions where there are no overseas branches. The query is as follows. Here too, if you remove the command | where country_code<>"JP" and !isNull(latitude)`, accesses from Japan will also be included in the count.

```
_sourceCategory = cloudtrail  "ConsoleLogin"
| parse "\"eventName\":\"*\"" as eventName nodrop
| where eventName="ConsoleLogin"
| json "sourceIPAddress"
| parse "\"userName\":\"*\"" as user_name nodrop
| json field=_raw "userIdentity.principalId" as principal_id nodrop
| parse regex field = principal_id ":(?<user_principal>.+)" nodrop
| if (user_name="", user_principal, user_name) as user
| json field=_raw "responseElements.ConsoleLogin" as loginResult nodrop
| parse "\"MFAUsed\":\"*\"" as mfaUsed nodrop
| count by sourceIPAddress, user, loginresult
| lookup latitude, longitude, country_code, country_name, region, city, postal_code
from geo://location on ip = sourceIPAddress
| where country_code<>"JP" and !isNull(latitude)
| count by country_name | sort by _count
```

Non-MFA User

Since we require MFA to be used for access to the AWS console, we need to be wary of access from accounts that do not use MFA. Therefore, we display the source IP address, the country from which the access is coming, whether the login was successful, and the number of accesses, so that we can consider whether the access is reasonable. The query is as follows.

```
_sourceCategory = cm-core/cloudtrail  "ConsoleLogin"
| parse "\"eventName\":\"*\"" as eventName nodrop
| where eventName="ConsoleLogin"
| json "sourceIPAddress"
| parse "\"userName\":\"*\"" as user_name nodrop
| json field=_raw "userIdentity.principalId" as principal_id nodrop
| parse regex field = principal_id ":(?<user_principal>.+)" nodrop
| if (user_name="", user_principal, user_name) as user
| json field=_raw "responseElements.ConsoleLogin" as loginResult nodrop
| parse "\"MFAUsed\":\"*\"" as mfaUsed nodrop | where mfaUsed!="Yes"  | lookup
latitude, longitude, country_code, country_name, region, city, postal_code from
geo://location on ip = sourceIPAddress| count by
user_name,mfaUsed,sourceIPAddress,country_name,loginResult| sort by _count
```

Login from rare location

This is a bit tricky information, as it tallys up the country of origin for each user and displays the name of the country of origin if there is access from a country different from the country of their usual access (the country with the most accesses during the period). Due to query limitations, only the two countries with the most and least accesses are displayed, so if there are accesses from three or more countries, information other than these two countries will not be displayed.

I wondered if I could create an information panel that would allow me to narrow down the search to "users accessing from a different environment than usual," and after much trial and error, I finally created one, but I'm not entirely sure if it's the right solution. Ideally, I'd like to display the country from which each user logged in and the number of accesses they made, but I'm not yet able to achieve that with the SumoLogic query statistics command. This may be because I haven't fully mastered SumoLogic's queries, so there's still room for further investigation. The query is as follows:

```
_sourceCategory = cloudtrail  "ConsoleLogin"
| parse "\"eventName\":\"*\"" as eventName nodrop
| where eventName="ConsoleLogin"
| json "sourceIPAddress"
| parse "\"userName\":\"*\"" as user_name nodrop
| json field=_raw "userIdentity.principalId" as principal_id nodrop
| parse regex field = principal_id ":(?<user_principal>.+)" nodrop
| if (user_name="", user_principal, user_name) as user
| json field=_raw "responseElements.ConsoleLogin" as loginResult nodrop
| parse "\"MFAUsed\":\"*\"" as mfaUsed nodrop
| lookup latitude, longitude, country_code, country_name, region, city, postal_code
from geo://location on ip = sourceIPAddress | withtime country_name
| most_recent(country_name_withtime),least_recent(country_name_withtime) by user |
where _mostrecent != _leastrecent
```

Failed login detail

It displays detailed information about communications that have failed due to errors. This information only shows errors for all operations, including those other than console logins. This extracts information that can capture mass access from suspicious tools, such as the one found at the end of the previous article . The query is as follows:

```
_sourceCategory = cloudtrail
| json "userAgent","errorCode" | json "userIdentity.userName" as userName| parse
"\"eventName\":\"*\"" as event_name
| parse regex field=event_name "^(?<event_type>[A-Z][a-z]+?)[A-Z]"
| count by  userName, userAgent, event_type,errorCode
| where _count > 10 | sort _count
```

Conclusion

Although I've created it for now, I feel like GuardDuty will be able to detect a lot of unusual access, so I'm not sure if it's worth creating such a detailed dashboard. However, the queries displayed on this panel may also be useful for investigating the details of events detected by GuardDuty. I hope to be able to refine the information while also listening to the opinions of internal experts.

This concludes our introduction to SumoLogic custom dashboards.