Detecting Malicious Authentication Events in SaaS Applications Using Anomaly Detection

Gavin Grisamore
Detecting Malicious Authentication Events in SaaS Applications Using Anomaly Detection

GIAC (GMON) Gold Certification

Author: Gavin Grisamore, gavin@gavsec.com
Advisor: Tanya Baccam

Accepted: 11/10/2019

Abstract

SaaS applications have been exploding in popularity due to their ease of deployment, use, and maintenance. Security teams are struggling to keep pace with the growing list of applications used in their environment as well as with the process of tracking the data these applications hold. Attackers have been taking advantage of these visibility gaps and have targeted SaaS applications regularly. By using log data from the applications themselves, security teams can use anomaly detection techniques to find and respond to such attacks. Anomaly detection allows security teams to more quickly identify and remedy a data breach by condensing large amounts of data into a shortened list of events that are outliers. The detection techniques used can help security teams respond to or prevent the next data breach.
1. Introduction

Software as a Service or “SaaS” has changed the way organizations of all sizes think about the applications that run their businesses. According to the 2017 State of the SaaS-Powered Workplace Report, 73% of organizations say nearly all their apps will be SaaS by 2020 (2017 State of the SaaS Powered Workplace). Additionally, organizations on average use 15 or more SaaS applications (SaaS-Powered Workplace). This increased number of SaaS applications has put a burden on Information Technology (IT) teams to monitor the applications used in their environment. SaaS applications are often employed without the consent of IT. IT teams are then forced to deal with duplicate services, hard-to-trace subscription licenses, massive amounts of data sprawl, and an overwhelming lack of control (SaaS-Powered Workplace).

Since SaaS applications are maintained and operated by a third-party, organizations can no longer rely on traditional network defenses to detect and prevent attacks. Thus organizations are left open to attack by their use of these SaaS web applications. Many SaaS products require little more than a username and password for access. According to the 2019 Verizon Data Breach Report, 53% of data breaches involved the use of hacking techniques and of that 53%, approximately 80% involved the use of stolen credentials against web applications (Data Breach Report). These stolen credentials are often collected across multiple data breaches and are used en masse to facilitate further attacks. Social engineering, mainly through the use of phishing, is another area for concern. The Data Breach Report states that social engineering attacks cause 35% of data breaches, and of that 35%, 90% of breaches involve phishing (Data Breach Report). Hackers and social engineers are increasingly successful in gathering credentials through these methods.

Gavin Grisamore, gavin@gavsec.com
2. Attack Types

Before analysis or implementation of defenses can occur, an understanding of the various types of attacks that affect the authentication mechanism of the SaaS application is required. Authentication mechanisms of a SaaS application represent an internet-facing endpoint that allows users of the applications to access the system. A user often supplies credentials in the form of a username and password combination to access the system. Malicious actors use several techniques to gain access to these credentials. The following attack types highlight the most common exploits against the authentication and authorization mechanisms of a web application.

2.1. Password Guessing

One of the more straightforward approaches used to gain access to a system is to attempt to guess the username and password combination. Poor password hygiene such as using weak or common passwords and sharing passwords across sites make it easier for an attacker to guess a password. According to a Pew Research study on cybersecurity in 2017, 65% of internet users say that memorization is the main or only way they keep track of their online passwords. Furthermore, the study finds that 39% of users use the same (or very similar) passwords for many of their online accounts (Smith, 2017). Since the human memory has limitations, users tend to stick to a simple strategy to remember their passwords. Password reuse is a particular problem because if a service falls victim to a data breach involving passwords, then accounts on another site might be compromised if the user relies on the same or similar password. Collections of these data-breached passwords are readily available and offer a low barrier of entry for attackers and scammers.

Attackers can also leverage social media and other reconnaissance techniques to aid in their password guessing attempts. As stated previously, users often rely solely on their memory to remember a password. Using information that is important to the user such as birthdays, anniversary dates or pet’s names can make it easier to remember; however, if this information is readily available online, an attacker could use it to guess a password. Attackers often use social media as an available resource to retrieve this
information as users typically share this type of information with friends and family. Tools like theHarvester and recon-ng, as shown in Figure 1, make searching easier by automating queries across many search engines and social media sites. An aggregate search also allows attackers to expand to more targets as it makes reconnaissance activities more efficient. Automated reconnaissance tactics combined with poor password hygiene make password guessing a capable attack vector.

![Screenshot of recon-ng tool](image)

Figure 1: Screenshot of recon-ng tool

### 2.2. Password Reset

Password Reset is another common attack vector against SaaS applications. SaaS applications often make use of email addresses for account sign-up and communication with the user. Applications use email addresses for a variety of purposes from account verification to marketing. Application developers also need to provide a way for users to recover their password should they not remember it. A common technique is to have the

Gavin Grisamore, gavin@gavsec.com
user enter the email address of the account used on the site, and a code or link is sent to
the user’s email address to provide further instructions to reset the password, as depicted
in Figure 2. Another common technique is to have the user answer some previously
defined security questions to perform the password reset. Answers to security questions
can have similar weaknesses as passwords if the questions lead to easily guessable
answers. For example, a user’s answers to “what is your mother’s maiden name?” or
“what is your pet’s name?” can be easily determined through reconnaissance measures
described previously. Sarah Palin, a former Alaskan Governor, fell victim to this type of
attack in late 2008. A hacker used information readily available online such as her
birthday, her zip code and the fact she met her spouse in high school to answer password
reset security questions on her Yahoo email account (Malkin, 2008). If an attacker can
comprise an email account, all accounts tied to it might also be compromised. According
according to a public bulletin posted by the FBI, business email compromise has accounted for over
$12B in exposed dollar loss globally from October 2013 to May 2018 (Business Email
Compromise, 2018). The FBI bulletin illustrates that email attack vectors continue to be a
growing and lucrative endeavor for attackers.

![Password reset form](https://example.com/password-reset-form.png)

*Figure 2: Password reset form*
2.3. Credential Stuffing

Credential stuffing attacks involve the insertion of breached usernames and passwords through automated means to gain access to a web application. Automated tools allow hackers to enter username and password combinations into an application’s password form fields in large numbers. Credential stuffing is a form of brute force attack that leverages the sheer number of breach credentials available to ensure the attack has a higher likelihood of success. Once an attacker gains access to stolen credentials through a first-hand website breach or a password dump site, the attacker can use an account checker tool to test the stolen credentials against a target web application. According to the 2018 Shape Credential Spill Report, 2.3 billion credentials were leaked in 2017. The report goes on to state that consumer banking is one of the hardest impacted, with an average of 232 million attacks per day (Credential Spill Report, 2018).

To further increase the likelihood of a credential stuffing attack succeeding, hackers use proxy services to disguise their attacks. Directing a large number of web requests from a single source against a web application can be easily defended. Attacks that leverage a single or a small number of IP addresses are quickly shut down by the application administrators. By using a series of open anonymous proxies, however, attackers can mask their source IP address and leverage hundreds or thousands of IPs in a single attack. An open proxy is simply a forward proxy that is internet-facing and can be accessible by anyone. Proxy tactics make it difficult for web application defenders because it is challenging to distinguish which requests are a part of the attack and which might be legitimate users. Credential stuffing is a formidable attack vector due to its effective use of automation and lack of defenses by most web application security teams.
2.4. Phishing

Phishing is another dangerous attack vector affecting access control to a SaaS application. A phishing attack consists of deceiving a user into providing credentials to an attacker often through the use of a lookalike login page. Phishing attacks are widely used to gather legitimate account credentials. A report by SANS Institute, “Defending Against the Wrong Enemy 2017 Insider Threat Survey” reported that 80% of respondents had experienced a phishing attack against their organization (Cole, 2017). Hackers use a variety of techniques to make their phishing campaigns successful. First, phishing is predominately delivered by email due to its relative ubiquity. Attackers use language that create a sense of urgency which then incites the user into taking action. Attackers will also often spoof the sending address to appear more legitimate. In addition to the email’s language, the body of the email likely contains a link to an external site. Website links are used in favor of email attachments because they are more likely to pass through email filters. Once the user clicks the link, they are taken to a login page simulating the web application, as shown in Figure 3. More effective login pages use similar domains to the legitimate site and often employ HTTPS protocol to ease the user into a false sense of security.

![Phishing Page Example](image)

*Figure 3: Phishing Page Example*
By combining the variety of techniques, attackers can trick users into providing their credentials. If the user complies, the attacker can now attempt to gain access using the legitimate user credentials. Phishing attacks can be difficult to defend and detect because the login attempt is using valid account credentials without the potential noisy failed login attempts. Phishing attacks are also widely seen because of the relatively low skills required in crafting an email and creating a landing page. Verizon highlights phishing as the top threat vector in data breaches at nearly 40% (Data Breach Report, 2019).

### 2.5. Software Vulnerability

A code level vulnerability is another option attackers might exploit to gain access to a web application. A software vulnerability affecting access control can take many forms. One option might include looking for logic flaws in the authentication mechanisms such as the password form or the password reset page. Application developers who do not follow best practices might experience manipulation of the authentication and authorization components. The Open Web Application Security Project (OWASP) provides recommendations and best practices to follow. Another option might be to perform a session hijacking type of attack, which targets the authorization or session tokens of a valid user. One example might be a cross-site scripting attack in which an attacker might send a user a crafted link with a malicious JavaScript payload. If the user clicks the link, the JavaScript executes the instructions designed by the attacker (Figure 4). The software vulnerability attack vectors are numerous as they could target the authentication or authorization mechanisms as well as target either the client or the server.

Gavin Grisamore, gavin@gavsec.com
2.6. Insider Threat

In addition to external attack vectors, web applications can also suffer from insider threat attacks. Insider threats are highly dangerous and can be challenging to detect. According to the insider threat survey, 38% of survey respondents said they did not have effective ways to detect them (Cole, 2017). An insider threat is merely a person inside an organization that abuses their access either maliciously or accidentally. For example, an employee might attempt to take a company’s intellectual property with them after being terminated. Similar to software vulnerabilities, this category includes many different attack vectors. Web applications are vulnerable because they may contain sensitive information for which the employee might already have access. It might also be challenging to determine if an insider threat attack has occurred because security teams are relying on audit logs provided to them by the third-party application. These logs might not be readily available or might not be detailed enough to assist in an investigation. Defenders of a web application also cannot often rely on similar detection mechanisms used for external attacks. With external attacks, a defender might be able to investigate clues in unusual or never-before-seen IP addresses, anomalous user-agents, or manipulated web requests. Insider threats can originate from within company networks and can leverage company equipment and authorized access to systems.

Gavin Grisamore, gavin@gavsec.com
3. Setting Up the Attacks

Before attack detection can take place, selecting a target web application and data setup must occur. The Okta web application is the data source for analysis. Okta is a SaaS single sign-on and company directory application. Companies leverage Okta to allow for centralized control over authenticating to various other applications. While Okta itself is a SaaS application, the risk for Okta is more significant because it allows a single account to access several other SaaS applications, which offers a high reward potential for attackers. If an Okta account was to become compromised, all applications linked to Okta, both SaaS or internal applications, could be compromised, causing downstream impact. Okta also offers robust logging, which is vital for detecting application intrusions. The Okta application used for testing is a production environment used by approximately 400 users daily. A production environment provides the most accurate representation of baseline employee login behavior for analysis. The Okta SaaS application offered the best solution to test the model, as it not only offers an attractive target for attackers but also provides detailed logging to capture the data points required.

For the research, several simulated attacks are launched against the Okta web application. The attacks include a password reset attack, a credential stuffing attack, and a phishing attack. The password guessing attack was omitted from testing due to its similarity to the credential stuffing attack as well as the high likelihood of causing an account lockout. While software vulnerabilities, and insider threats are also valid and severe attacks, the researcher felt that too many variables and assumptions would be involved to accurately simulate these types of attacks.

All attacks are targeting the authentication mechanism of Okta to simulate unauthorized access control. All attacks use an open anonymous proxy server. The proxy server performs the attacking web requests on the attacker’s behalf, which changes the source IP address of the web request to the proxy versus the attacker’s machine. To further decrease the level of detection, proxies that use residential IP addresses perform the attacks. In addition to the use of a proxy, user-agent spoofing simulates changing of the attacker’s device. By combining both the use of a proxy server and user-agent spoofing, the researcher can align to commonly exploited attacks.

Gavin Grisamore, gavin@gavsec.com
3.1. **Okta Password Reset Attack**

A password reset attack is simulated by using a victim’s email account tied to Okta that has been compromised by an attacker. The attacker performs the password reset initiation and subsequently changes the Okta password. The attack targets two victims over a week. A valid password reset event demonstrates a successful compromise.

3.2. **Okta Credential Stuffing Attack**

A credential stuffing attack is simulated using compromised usernames and passwords from a password dump site of a previous data breach. The attack targets three victims over two weeks consisting of four failed login attempts followed by a successful one. The assumption in this situation is that an attacker might have several username and password combinations to try against a web application, which might result in several failed login attempts. The researcher also chose to keep failed login attempts under five to avoid account lockouts. A valid login event shows a successful compromise under the assumption that one of the breached credentials is accurate.

3.3. **Okta Phishing Attack**

A phishing attack consists of credentials stolen from an Okta user through the use of a phishing email and landing page. To simulate compromise, the attacker makes several successful login attempts over a week for one victim. A successful login event from the attacker represents a valid compromise of the Okta account.

All three attack types are designed to mimic real-world attacks. Attackers often use various infrastructure such as a proxy to mask their source IP address and spoof other details about their request such as their user-agent. While these attacks are frequent and relatively unsophisticated, they are often the starting point to a data compromise.

Gavin Grisamore, gavin@gavsec.com
4. Gathering Log Data

The primary data source for the research analysis is log data provided by the Okta application. Okta provides event details across their application for not only authentication and authorization, but also for multi-factor events, application access, and administrator auditing. Okta logs are available using their application programming interface (API) and return in JavaScript Object Notation (JSON) format. JSON is a more readable way to structure data by using name/object pairs. A sample name object pair is as follows:

"ipAddress":"192.168.0.2"

The notation above signifies that “ipAddress” is the name of the object, and “192.168.0.2” is the value. This simple structure is easy to parse both manually and automatically to gain insight into the data.

Okta offers logs available via an API; however, they need to be accessed from the application to be loaded into a log management system or security information and event management (SIEM) system. The operation to gather logs from a SaaS application puts the effort on the organization’s engineering teams to maintain this log aggregation service, perform log transformation such as deduplication or filtering out irrelevant content, and to publish the log data to the log management or SIEM system. For this research, Sumo Logic is serving as the log management store as well as the SIEM for analysis. An operation is already in place to move logs from Okta to Sumo Logic.

Gavin Grisamore, gavin@gavsec.com
Detecting Malicious Authentication Events in SaaS Applications

The data set for analysis will consist of 30 days of production Okta authentication logs. The following is a sample authentication event indicating a successful login attempt:

```json
{
  "actor": {
    "id": "99999999999999999999",
    "type": "User",
    "alternateId": "sample@example.com",
    "displayName": "John Doe",
    "detailEntry": null
  },
  "client": {
    "userAgent": {
      "rawUserAgent": "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_14_6) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/76.0.3809.100 Safari/537.36",
      "os": "Mac OS X",
      "browser": "CHROME"
    },
    "device": "Computer",
    "ipAddress": "96.52.49.18",
    "geographicalContext": {
      "city": "New York",
      "state": "New York",
      "country": "United States",
      "postalCode": "10010",
      "geolocation": {
        "lat": 40.745,
        "lon": -73.9869
      }
    }
  },
  "displayMessage": "User login to Okta",
  "eventType": "user.session.start",
  "outcome": {
    "result": "SUCCESS",
    "reason": null
  },
  "securityContext": {
    "asNumber": 1000,
    "asOrg": "acme telecom",
    "isp": "acme telecom corp.",
    "domain": ".",
    "isProxy": false
  },
  "severity": "INFO",
  "uuid": "99999999-9999-9999-9999-999999999999",
  "version": "0",
  "target": null
}
```

The dataset consists of authentication events such as login and password reset events. Application access events are not included, so the dataset remains general enough to apply to other SaaS applications. The data consists of approximately 7,200 log events.

Gavin Grisamore, gavin@gavsec.com
5. Data Analysis

With the data collected and the attacks performed, analysis can be conducted to see what anomaly detection methods are useful in finding the attacks. Each log event that the SaaS application produces generates several data points that aid analysis. Furthermore, data that can be further transformed to provide additional insights is also valuable. Based on the collection of data, the researcher can hypothesize that the more data elements, such as IP address and user agent used in the analysis, the higher the likelihood of attack discovery. The hypothesis is tested against each analysis method.

5.1. SIEM & Log Analysis

According to SIEM market analysis, organizations have invested $2.59 Billion in SIEM technologies to assist with enterprise security initiatives and detecting threats to their businesses (SIEM, 2018). Organizations can leverage these existing investments to build detections for anomalous activity against SaaS applications. For this research, Sumo Logic will function as the existing log management and SIEM system.

The analysis starts with all events from Okta being sent to Sumo Logic. Sumo Logic is then able to parse the log data to identify the various data values with the events that occurred. For an illustration of Sumo Logic parsing an Okta event log, see Figure 5.
Detecting Malicious Authentication Events in SaaS Applications

Figure 5 – Sumo Logic Okta Event Detail

Sumo Logic, like many log management and SIEM systems, rely on SQL-like queries to search through the logs which allows for operations to narrow down the search. Constructing the initial query consists of various log reduction techniques as well as statistical analysis to locate the attacker’s events. Simplified log reduction is a common SIEM and log management functionality and is a crucial differentiator versus using command-line tools for log analysis. Two advanced capabilities that Sumo Logic offers are LogReduce and LogCompare. These options allow an analyst to limit the number of logs that need to be searched or differentiate one set of log events from another. LogReduce is particularly useful if an analyst needs to sort through many different types of events. In the case of Okta, analysts might be reviewing all logs, including authentication events, authorization events, application access, and administrative events. LogReduce allows analysts to find anomalous log events that are different from one another to limit the number of events to search through. LogCompare looks to consolidate similar log events and highlight comparisons and differences at a field or value level. LogCompare looks for differences in log events over time; however, it does not assist in this use case as only specific values such as IP address and user-agent are changing within a log message. Many of these changes are benign and would not

Gavin Grisamore, gavin@gavsec.com
Detecting Malicious Authentication Events in SaaS Applications

necessarily indicate malicious behavior. While these capabilities can be powerful in reducing the log volume in order to search through and find differences, the search dataset is filtered to only include password reset and login authentication events.

The more appropriate search operation for these use cases is using outlier detection against time-series data. Time-series data is the process of organizing the data into blocks of time and then comparing them against each other using statistical calculations. For example, the researcher can create a time-series set of data of the Okta logs group by day. Sumo Logic collects the log timestamps and groups the login events that correspond to 24 hours. Once the data is grouped in this way, variables such as IP address can be analyzed to see if they have changed. The outlier search operations have several input options to configure:

- **Window Size**: The value indicates how many blocks of time the query should search back through. In this case, the window size was set to 5 to look back over the last 5 days of activity.
- **Threshold**: The standard deviation value. Standard deviation is set to 3 to include 99.73% of possibilities for a normal distribution.
- **Consecutive**: The value indicates how many times the value needs to trigger to alert. The consecutive value is set to 1 to trigger on the first new value.
- **Direction**: The value indicates the direction to be analyzed is a positive and/or negative deviation from the mean. The direction in this case is both positive and negative.

With these inputs, the outlier search operation calculates the following:

- **Mean**
- **Upper limit**: Mean + Threshold * Standard Deviation
- **Lower limit**: Mean – Threshold * Standard Deviation
- **Indicator**: Alert if a value is outside the upper or lower limit

With the search established, a determination is made on which values of the Okta logs will be analyzed. The first value that is considered is the user field. The user field is the first dimension to be calculated because the use case is to determine anomalous

Gavin Grisamore, gavin@gavsec.com
behavior for each login. Other values such as a new IP address is a useful indicator, but only if it is compared against a single user. An IP addresses analyzed in aggregate across all users might not standout when compared against so many other addresses. Analyzing these values against a single user allows the query to calculate averages on a much smaller set of data points to reduce false positives.

After selecting on the user as the first dimension, the second dimension is to leverage the client’s IP addresses to detect anomalies. To further enrich this analysis, Sumo Logic offers a geographic IP lookup database. While geographic IP databases are not always accurate or up to date, they can serve as a valuable tool to enrich the dataset. By combining the time-series calculation, with the user and IP address outlier search, a query can be conducted. Figure 6 shows an identified user’s IP enriched with geographic IP data and plotted on a map. The map visually indicates some anomalies in the south and south-central regions of the United States. Figure 7 shows the count of the IPs in question. The masked IPs correspond to IP addresses that are known to the organization such as the office location and virtual private network (VPN).

Figure 6: Anomalous User Geo IP Map
Sumo Logic was able to determine several users that were anomalous based on their past login behavior. The query was successful in identifying several of the attacks based on user and IP address alone; however, several false positives were identified, which may include employees traveling or working from home.

The last component of the query is accomplished by determining if the device changed by using the user-agent string in the HTTP header. The user-agent is a way for the client to tell the server what type of device it is so the webserver can return content most appropriate for that device. The user-agent can be parsed to include the operating system of the client as well as the type and version of the browser the client is using to make requests. By using the same technique to analyze user-agents as was performed with IP addresses, users logging in from new devices can be identified in relation to their typical login behavior.

Combining all techniques, Sumo Logic can determine when an individual user logs in from a new device and a new location. A filtering process can be implemented to

Gavin Grisamore, gavin@gavsec.com
eliminate known locations as a way to reduce false positives. While each of the techniques are easy to implement, combining them and using the knowledge of the enterprise environment proves to be an effective way of analyzing anomalous login behavior.

5.2. Amazon Web Services IP Insights

Amazon Web Services is a popular cloud infrastructure provider that not only offers computer and database resources for organizations, but also offers machine learning and data science analysis tools. AWS has developed an algorithm that analyzes IP addresses using unsupervised machine learning to determine the usage pattern of IPv4 addresses. The IP insights algorithm is used to build a model of the Okta log data and report any anomalous IP address usage.

The IP insights model consists of two separate phases for interacting with the model. First is the training phase, which consists of providing the model the data used to establish a baseline set of behaviors for the model to compare against for the actual data set. Okta logs containing login and password reset events are collected over a week to train the model. Approximately 2,800 events are collected during this timeframe. The model expects the data in CSV format with two columns of data. The first data element must consist of a unique identifier of entity identification value. For the purposes of this research, the Okta username acts as the entity value. The second field value in the CSV is the IP address. The Okta client IP address is collected and used. The training data set is complete with the Okta username acting as the entity ID and the Okta client IP address.

To set up the model and begin training data, AWS offers a service called SageMaker for which the IP insights model is a component. SageMaker builds the infrastructure necessary to perform the machine learning and operational steps. Python programming language is used to provide the necessary parameters for the model as well as direct the locations for the input and output datasets. The model is trained on the week’s worth of Okta events prepared previously. Once the training is complete, the primary dataset can be used for analysis. The primary dataset consists of the same structure as the training data, but with more data points. The primary dataset consists of

Gavin Grisamore, gavin@gavsec.com
approximately 8,000 events. The primary analysis consists of inputting the primary dataset into the model and outputting a value for each log event and consolidating it to a single output file.

The output file consists of a single score known as a dot_product value. This value represents how unusual the entity and IP address combinations compare to the input and training data. The value is unbound, which means there is no range for which the value might fall. In this case, the values ranged from -6.12 to 0.03, with a mean of -0.16. By setting an upper and lower limit using three standard deviations away from the mean, a basis is established for how anomalous the score value is. Figure 8 shows the normal distribution breakdown of anomaly scores.

![Attack Detection](image)

*Figure 8. Amazon IP Insights Model Anomaly Score Distribution*
This figure shows that a majority of the IP addresses have an anomaly score between 1 and -1. The black dotted line indicates the lower third standard deviation limit, while the solid red lines indicate the attacking IP addresses. The third standard deviation threshold is used as an indicator of what qualifies as anomalous in this case. The IP insights model was able to correctly identify the attacks as anomalous compared to the general population; however, the model also identified several false positives. There could be several reasons for the high number of false positives. One possibility is that the model was only trained on a week’s worth of log data. More log data, including several weeks or months of logs to train the model, would likely reduce false positives.

The SageMaker IP insights model has shown to be an effective way to identify anomalous IP addresses from data sets with limited testing. Use cases and datasets vary widely by organization, and more extensive testing is necessary before operating these models in a production environment.

5.3. DIY Model

Taking the knowledge gained from using the SageMaker IP Insights model, a custom model was created to further enhance detection accuracy. This model was built using the same SageMaker infrastructure, but the model consists of more factors beyond just IP addresses. Applying several of these factors can not only make the model more accurate in identifying true positives, but can also reduce the number of false positives as seen by using the IP insights model alone.

Since log data from SaaS application varies widely in terms of data points, the research attempted to select those that would be the most generally available within the SaaS logs themselves. An organization can use external databases to enrich the datasets if the values are not native to the log data. The first additional factor to consider is user-agent. User-agents can provide the device’s operating system type and version, as well as the browser type and version. User-agents can also be spoofed to limit detection by an attacker or for legitimate testing purposes. For this model, only user-agent string matching is used to determine the device has changed.

Gavin Grisamore, gavin@gavsec.com
Another data point available is geographic IP lookup. Okta natively provides geo IP data in their logs, but this information can be gathered if it isn’t included in the log data. Several SIEMs, including Sumo Logic, provide lookups to geographic IP databases. Organizations can also build integrations with geo IP databases directly to leverage this information. The accuracy of these databases are not 100%, but they can provide a general sense of location based on ARIN WHOIS registration information and other factors the database provider collects. The last factor added to the model is the autonomous system number or ASN. ASN numbers are assigned by ARIN to internet service providers. ASN numbers should further enhance the model by looking for changing internet service providers or requests coming from cloud infrastructure providers.

With all these data points, a model can be built with the following information:

<table>
<thead>
<tr>
<th>Datapoint</th>
<th>Example Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>IP Address</td>
<td>52.14.242.23</td>
</tr>
<tr>
<td>ASN Number</td>
<td>AS16509</td>
</tr>
<tr>
<td>User Agent</td>
<td>Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/67.0.3396.99 Safari/537.36</td>
</tr>
<tr>
<td>Country</td>
<td>US</td>
</tr>
<tr>
<td>Region</td>
<td>Great Lakes</td>
</tr>
<tr>
<td>City</td>
<td>Columbus</td>
</tr>
<tr>
<td>Postal Code</td>
<td>43201</td>
</tr>
</tbody>
</table>

*Figure 9. Example DIY Model Data Elements*

Combining these data points allows the model to look for changes across multiple variables. While a single value change might not indicate suspicious behavior, multiple changing values deviating from normal user behavior might be worth investigating. For simplicity, the same IP insights model was used as before but accounts for changes across multiple data points. More changes to the data points yield a lower anomaly score. Running the model against the same input data produces results with fewer false positives. Figure 10 shows the normal distribution breakdown of anomaly scores for the second model.

Gavin Grisamore, gavin@gavsec.com
Figure 10. DIY Model Anomaly Score Distribution

The second model was able to produce more accurate results while lowering the amount of false positives captured. All attacks tracked measured three standard deviations below the mean, which was the alerting threshold. False positives also decreased by 41%. While the DIY model was more accurate in reducing false positives, it still captured 40 unique false positives. Adding more data elements to the model did prove the research hypothesis that more data points would increase the model’s accuracy; however, depending on the use case, false positives remain high.

Gavin Grisamore, gavin@gavsec.com
5.4. Comparison

By analyzing all three techniques, there are definite positives and negatives that can be gleaned from the analysis. Using a SIEM or log management tool provides the most flexibility for data analysis. Log data can vary significantly in content as well as file format. Therefore, having the ability to drill into logs quickly through automatic log parsing is advantageous. The expertise needed to pivot through and analyze logs is relatively low compared to building a machine learning model. When it comes to incident response scenarios, time is of the essence. Gathering and analyzing data quickly, even if it is manual, is also helpful. SIEM and log management tools fall short when attempting to use anomaly detection to pick out events of interest. While Sumo Logic was able to reduce the amount of logs needed to search through and determine some range of outliers, a security analyst would still need to review the results and further filter the logs manually.

The machine learning model, however, was better able to assist an analyst by providing single events that are suspicious. Both models were able to identify the attacks in the simulation and can be further tuned to be more accurate as more data is fed into the model. The machine learning models also have several downsides. First, machine learning models require expertise to build, tune, train, and deploy. Organizations might not have this expertise in house, and it might be difficult or cost-prohibitive to outsource.

Furthermore, machine learning models can be expensive to run. While the researchers used a dataset containing less than 10,000 events, more massive datasets require more expensive infrastructure to run. Depending on the needs of the organization, the model detecting anomalous login behavior against a SaaS application might not be enough of a cost-benefit if it is too expensive to run. All techniques were successful in finding the attacks; unfortunately, the techniques used had high cost and time considerations. Ultimately, security teams need to determine which model produces the most cost benefits for attack mitigation. Figure 11 summarizes each solution in terms of the level of effort to construct and interrupt the results as well as how the solutions performed during testing.

Gavin Grisamore, gavin@gavsec.com
Detecting Malicious Authentication Events in SaaS Applications

<table>
<thead>
<tr>
<th></th>
<th>Log Analysis</th>
<th>IP Insights</th>
<th>DIY</th>
</tr>
</thead>
<tbody>
<tr>
<td>Level of Effort and</td>
<td>Low</td>
<td>High</td>
<td>High</td>
</tr>
<tr>
<td>Knowledge to Construct</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Level of Effort to</td>
<td>Medium</td>
<td>Low</td>
<td>Low</td>
</tr>
<tr>
<td>Interrupt Results</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Analysis Flexibility</td>
<td>High</td>
<td>Low</td>
<td>Low</td>
</tr>
<tr>
<td>True Positives</td>
<td>Medium</td>
<td>High</td>
<td>High</td>
</tr>
<tr>
<td>False Positives</td>
<td>Medium</td>
<td>High</td>
<td>High</td>
</tr>
<tr>
<td>False Negatives</td>
<td>Low</td>
<td>Low</td>
<td>Low</td>
</tr>
</tbody>
</table>

Figure 11. Anomaly Detection Solution Summary

6. Conclusion

As SaaS applications have grown in popularity among businesses, they have become an increasing target of attacks. From intellectual property to personally identifiable information, SaaS applications offer rich data for attackers to steal. Organizations need new ways of combating the threat of yet another data breach. Investing in anomaly detection techniques can prove valuable to security teams looking to secure the growing number of SaaS applications under management. Advancements in machine learning technologies and the commercialization of their use has lowered the barrier of entry to learn and leverage these new tools. The research has shown that the application of these new strategies can identify anomalous behavior from application logs. While more work is required to makes these techniques operational in a production environment, it is a solid start for security teams looking to take advantage of machine learning models.

Gavin Grisamore, gavin@gavsec.com
References


2018 Credential Spill Report (Rep.). (n.d.), Shape Security


Amazon SageMaker IP Insights Model. (n.d) Retrieved from
https://docs.aws.amazon.com/sagemaker/latest/dg/ip-insights.html

Business Email Compromise The 12 Billion Dollar Scam. (2018, July 12) retrieved from:


Log Compare. (n.d.) Retrieved from
https://help.sumologic.com/05Search/LogCompare/LogCompare-Syntax

Log Reduce. (n.d.) Retrieved from
https://help.sumologic.com/05Search/LogReduce/Detect-Patterns-with-LogReduce

Malkin, Michelle. (2008, September 17). The story behind the Palin e-mail hacking.


Gavin Grisamore, gavin@gavsec.com
Detecting Malicious Authentication Events in SaaS Applications


<table>
<thead>
<tr>
<th>Event Name</th>
<th>Location</th>
<th>Dates</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>SANS Northern VA - Reston Spring 2020</td>
<td>Reston, VAUS</td>
<td>Mar 02, 2020 - Mar 07, 2020</td>
<td>Live Event</td>
</tr>
<tr>
<td>SANS Munich March 2020</td>
<td>Munich, DE</td>
<td>Mar 02, 2020 - Mar 07, 2020</td>
<td>Live Event</td>
</tr>
<tr>
<td>SANS Secure Japan 2020</td>
<td>Tokyo, JP</td>
<td>Mar 02, 2020 - Mar 14, 2020</td>
<td>Live Event</td>
</tr>
<tr>
<td>SANS St. Louis 2020</td>
<td>St. Louis, MOUS</td>
<td>Mar 08, 2020 - Mar 13, 2020</td>
<td>Live Event</td>
</tr>
<tr>
<td>SANS Dallas 2020</td>
<td>Dallas, TXUS</td>
<td>Mar 09, 2020 - Mar 14, 2020</td>
<td>Live Event</td>
</tr>
<tr>
<td>SANS Prague March 2020</td>
<td>Prague, CZ</td>
<td>Mar 09, 2020 - Mar 14, 2020</td>
<td>Live Event</td>
</tr>
<tr>
<td>SANS Wild West Hackin Fest 2020</td>
<td>San Diego, CAUS</td>
<td>Mar 10, 2020 - Mar 11, 2020</td>
<td>Live Event</td>
</tr>
<tr>
<td>SANS Doha March 2020</td>
<td>Doha, QA</td>
<td>Mar 14, 2020 - Mar 19, 2020</td>
<td>Live Event</td>
</tr>
<tr>
<td>SANS Norfolk 2020</td>
<td>Norfolk, VAUS</td>
<td>Mar 16, 2020 - Mar 21, 2020</td>
<td>Live Event</td>
</tr>
<tr>
<td>SANS Kuwait March 2020</td>
<td>Salmiya, KW</td>
<td>Mar 21, 2020 - Mar 26, 2020</td>
<td>Live Event</td>
</tr>
<tr>
<td>SANS 2020</td>
<td>Orlando, FLUS</td>
<td>Apr 03, 2020 - Apr 10, 2020</td>
<td>Live Event</td>
</tr>
<tr>
<td>SANS Muscat April 2020</td>
<td>Muscat, OM</td>
<td>Apr 04, 2020 - Apr 09, 2020</td>
<td>Live Event</td>
</tr>
<tr>
<td>SANS Riyadh April 2020</td>
<td>Riyadh, SA</td>
<td>Apr 04, 2020 - Apr 16, 2020</td>
<td>Live Event</td>
</tr>
<tr>
<td>SANS Bethesda 2020</td>
<td>Bethesda, MDUS</td>
<td>Apr 14, 2020 - Apr 19, 2020</td>
<td>Live Event</td>
</tr>
<tr>
<td>SANS Minneapolis 2020</td>
<td>Minneapolis, MNUS</td>
<td>Apr 14, 2020 - Apr 19, 2020</td>
<td>Live Event</td>
</tr>
<tr>
<td>SANS Brussels April 2020</td>
<td>Brussels, BE</td>
<td>Apr 20, 2020 - Apr 25, 2020</td>
<td>Live Event</td>
</tr>
<tr>
<td>SANS Blue Team Summit &amp; Training 2020</td>
<td>OnlineKYUS</td>
<td>Mar 02, 2020 - Mar 09, 2020</td>
<td>Live Event</td>
</tr>
<tr>
<td>SANS OnDemand</td>
<td>Books &amp; MP3s OnlyUS</td>
<td>Anytime</td>
<td>Self Paced</td>
</tr>
</tbody>
</table>