Detecting Account Takeovers and Defending Users

White Paper

Account Takeover (ATO) attacks target active user accounts and cause service disruptions, widespread account lockouts, and prevent customers from using your service. Defending against ATO attacks requires instrumentation of your application's authentication process and sensitive application logic.

Account Takeovers in Action

Account Takeover (ATO) attacks are one of the fastest growing and prevalent problems for most organizations. According to a recent Forrester report, ATO attacks caused at least $6.5 billion to $7 billion in annual losses across financial services, insurance, eCommerce and other industries.[1] In addition to financial loss, organizations face loss of customer trust and their customers' sensitive data.

Organizations are under attack, but today's attacks aren't focused on just servers, but also users. Consider this scenario:

It was a Tuesday morning. Like any other Tuesday, Susan, an experienced security engineer, was looking forward to a day full of meetings and new projects to tackle. But on this particular Tuesday, she quickly realized that everything else was going to have to wait. Something was wrong—very wrong.

It all started as she headed to her desk for the day and walked by the customer care center. Instead of the normal jovial greetings and easy-going morning banter, everyone seemed to be in a panic. Susan overheard bits of conversation: "your password isn't working?" and "...it looks like your account is locked."

Customer calls were backing up the support queue prominently displayed overhead. When she pulled a customer specialist aside and asked what was going on, her growing fear was validated: there was a widespread problem with customer accounts.

Susan ran to her desk and began reviewing and processing monitoring stats, alerts, and firewall logs. Traffic volume seemed normal. No urgent alerts had come in. Network firewall logs looked normal. She dug deeper into the web analytic data and realized that the password reset page was getting significantly more traffic than normal, and most of the traffic was coming from a small IP range in the Ukraine. This was not looking good. After additional analysis, Susan found that so far about a third of their user accounts had been systematically taken over for the better part of the last 22 hours. As her phone rang with the CISO calling to get a status update, the only thought she had was, "Why didn't we detect this sooner?"

There's not a one-size-fits-all ATO defense for web applications. An average authentication failure rate is 33%, but the rate varies widely by industry. Knowing your baseline is critical.

Account Takeover Model

ATO attacks target active user accounts. ATO traffic can cause service disruptions, widespread account lockouts, and prevents customers from using your service. The accounts targeted are accounts held by your customers: real people with financial data and transaction history.

By using ATO techniques, an attacker first analyzes your authentication mechanisms, looking for weaknesses or defensive measures on accounts. The attacker then uses public data dumps or farming techniques to build a user list for your application. When paired with a common password dictionary and lists of other compromised credentials for other sites, attackers are able to execute ATO attacks on a high percentage of accounts. Finally, your application and customer accounts are sold or their contents are harvested for valuable data.

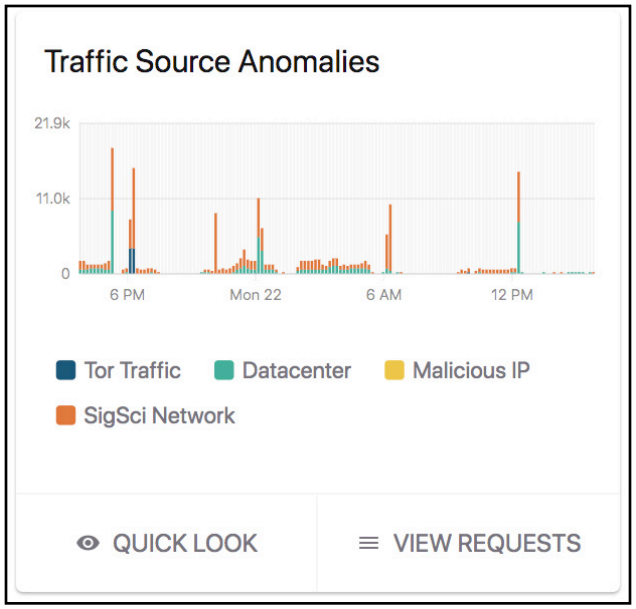
A Common Account Takeover

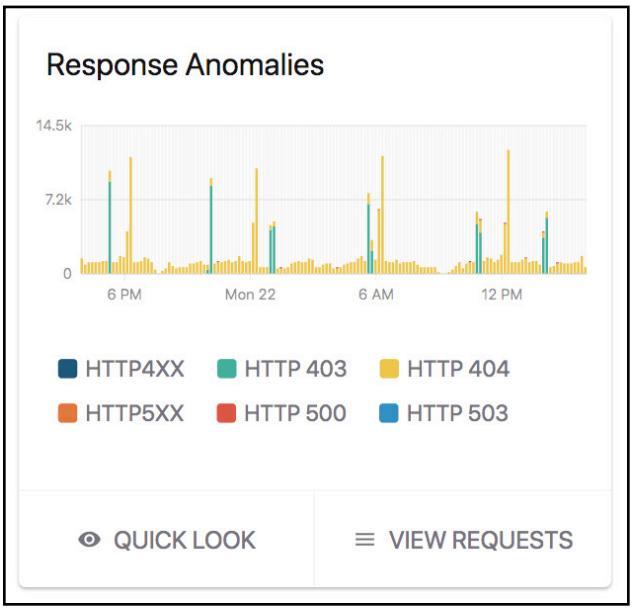
- Attacker visits website and tests authentication flow for weaknesses including: account name or password guessing, account lockout timings, password resets, and more.
- Attacker uses public data dumps and account farming techniques to determine users.
- Using password dictionaries, the attacker is able to gather a percentage of user accounts.
- Accounts or partial information such as credit card and personal data are then sold.

A New Defense Priority

It's no longer good enough to place a traditional web application firewall in place and hope that it stops the attacks against your web applications. In order to protect against account takeover attacks, enterprises must instrument and analyze their user's actions and patterns in detail. Knowing the baseline data for your application and your users' usage patterns is a good place to start asking questions.

- What's the rate of normal authentication traffic?
- What happens in authentication failure modes, and what's the average failure rate?
- What sensitive actions do users take when using the application?
- Does our application alert the engineering or security team when these actions and failures occur?





Every Application is Different

Getting instrumentation in place to know when accounts are under attack is critical, and every application is different. Applications serve different users and are created for different purposes, with each application using a potentially different framework for authentication or being written by different developers.

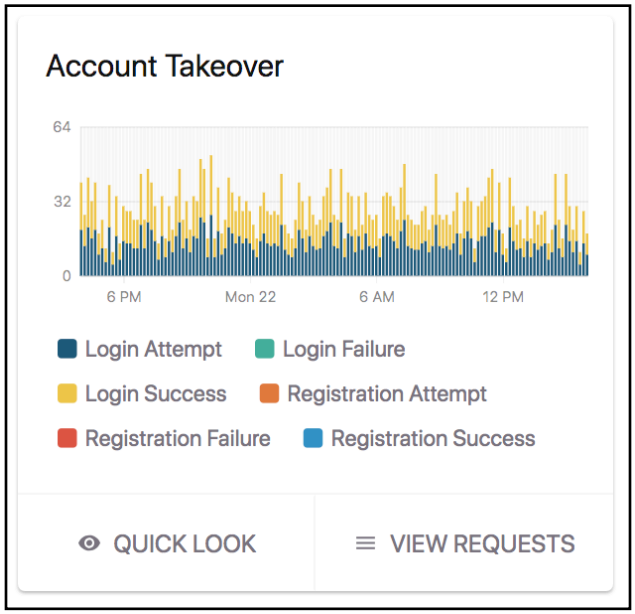
To get a sense of how different applications are, Signal Sciences examined applications and their authentication patterns. Our research focused on three specific areas:

- Authentication failure rates
- Application responses to authentication attempts
- The authentication source for users of the application

These three areas were chosen due to their commonality across all applications. No matter what type or purpose the application has, you can measure its failure rate, responses, and authentication source. The differences between applications in these three areas gives insight into how widely applications vary, and gives hints for what types of instrumentation and defensive measures you should implement.

Failure Rates Vary

In evaluating the usage of applications, our research shows that, on average, any given application experiences a login failure rate of approximately 33% [2]. This means that about a third of users fail authentication under normal traffic patterns. Intranet and partner applications running in internal networks did much better with only a 9% failure rate. Of course every application has unique parameters, which makes your baseline potentially different from these industry averages. Enterprises creating ATO protections must understand their own exact baselines in detail.

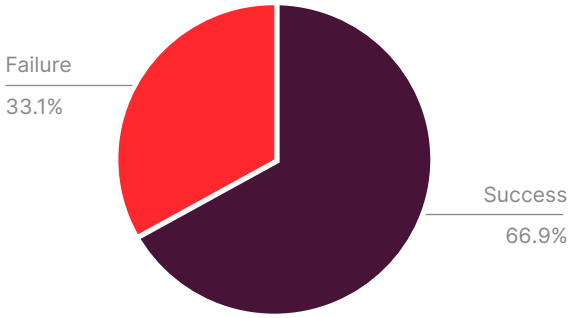


In step two, the SOC manager defines the thresholds and actions if those thresholds are met. In this specific Power Rule, she enters the number of login attempts within a specific timeframe that would trigger the desired action. If there are 20 login attempts within one minute, subsequent login requests from the same IP address will be blocked automatically for one day. The blocking duration can be set for a time period from 10 minutes to 24 hours (one day is the default).

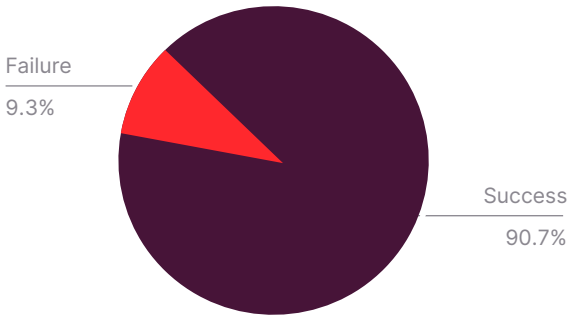
A notification has also been enabled for this rule: when the threshold is met an alert will be sent via Slack, email or other notification means, keeping the DevOps, operations and infrastructure staff aware and engaged in the security of the applications they develop and oversee.

With Signal Sciences integrations, alerts are distributed in an effective and timely manner. Signal Sciences integration and collaborations include PagerDuty, Slack, HipChat, Microsoft Teams, Pivotal Tracker, Jira, or VictorOps.

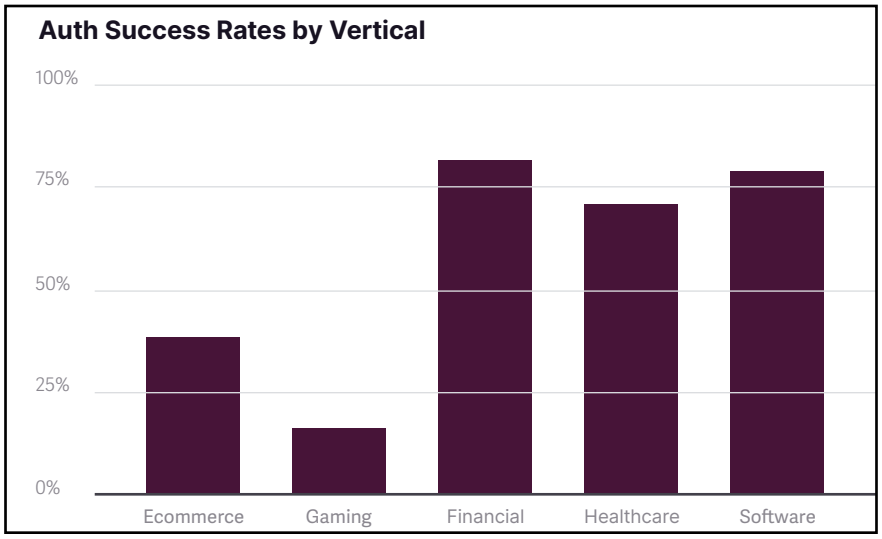
Auth Success and Failures for External Apps



Auth Success and Failures for Internal Apps

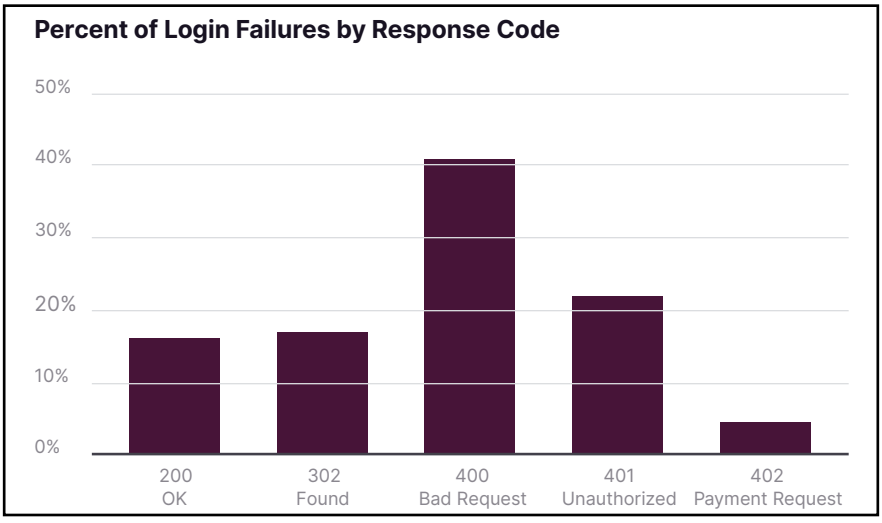


But the story isn't just internal versus external applications. There's significant variation in authentication failure rates based on industry vertical. While the average failure rate is 33% across all verticals, failure rates for individual verticals ranged as high as 84% or as low as 18%, giving a wide variance to the data. Financial and healthcare organizations saw authentication failure rates better than the overall average, while gaming saw very high failure rates for authentication. Authentication baselines aren't one-size-fits-all, which is why knowing your application's baseline for success and failure is an important starting point to stopping account takeover attacks.



There is No Standard Authentication Approach

Another interesting data point is the type of application responses for login successes and failures. It is tempting to think a "normal" application would return an http status code of 200 OK for a successful login, with an unsuccessful login returning an http status code of 401 Unauthorized. While this is sometimes the case, it's not a hard and fast rule. Sometimes failed login attempts return http 4XX failure codes, but sometimes they return 200 OK or temporary redirect via a 302 Found http code.



Our results demonstrate that every application responds differently, and there isn't a right or wrong answer to how applications should act. How login failures are handled can be due to development language being used, the authentication framework of choice, or organizational decisions of the development and operations teams. Our research even demonstrated response variation between applications within a single enterprise. Making assumptions organization-wide can be very detrimental to ATO protection capabilities, driving home the point that we should know our normal failure mode on a per-application basis. Understanding this norm gives your team the capability to react to responses and data that falls outside of these normal expectations.

Traffic Source as a Possible Indicator of ATO

The last area we evaluated is what network source a user might be coming from. Depending on the application, network sources can indicate whether there is an account takeover attempt. For a site that never gets traffic from other data center sources or data that originates from a TOR network point, a spike in authentication failures from these sources is worth looking into.

It's easy to assume that users coming from data centers or TOR are up to no good. However, our research showed that for some applications, these two traffic sources were normal. Across our enterprise sample set, data center traffic ranged up to 21% of successful authentication attempts, and TOR traffic was around 2%. These sources may be tempting to rule out, but it's important to know what's normal for your application and your users and make informed decisions and actions against these norms.

Since there isn't a one-size-fits-all approach to handling authentication, determining what's normal versus malicious must be adaptable on an application-by-application basis. Normalizing applications, even in the same organization, often isn't possible due to resource constraints. Because of these factors, finding a protection solution that can provide a flexible platform for application security defense is a necessity.

Instrumentation that Matters

In addition to authentication, every application has unique logic. Banking applications transfer funds, link accounts or change beneficiaries; mobile games provide virtual goods and connect players; ecommerce optimizes for customer reviews and payment handling. In each of these cases, being able to detect authentication successes and failures is important; but knowing that sensitive areas of your application are being attacked is critical.

There are two main methods to add instrumentation to applications: in-application event monitoring and server-based logging. Generally, a mix of both of these approaches happens organically. As new features are developed, in-application monitoring is set up to emit events to a monitoring stack. These are really useful events, often geared toward emitting business-level metrics like completed transactions, but they easily span to account-level security concerns, e.g. they can identify application-specific login failures or user profile changes. In-application monitoring can be really helpful in fighting account takeover attempts, however they generally require code rewrites and additional resources to manage.

Logging is another common approach. It's accomplished by parsing application and server logs to identify flows through the system. Logs contain information about a user's source and their general path through the application; but details are often omitted for security (e.g. it's bad practice to log a POST body or query strings) and the risk of logging too much. By the nature of logging protocols and systems, they're not in real time.

Monitoring and logging add instrumentation, but they're unable to provide defense.

Defend What Matters

There's another approach to add instrument applications and defend against account takeover attacks. Signal Sciences provides a web protection platform that instruments your application—including authentication flows—and provides defense at the same time. Through a mix of approaches, Signal Sciences provides complete coverage of your web applications without expensive code rewrites, and adds runtime defense that you can depend on.

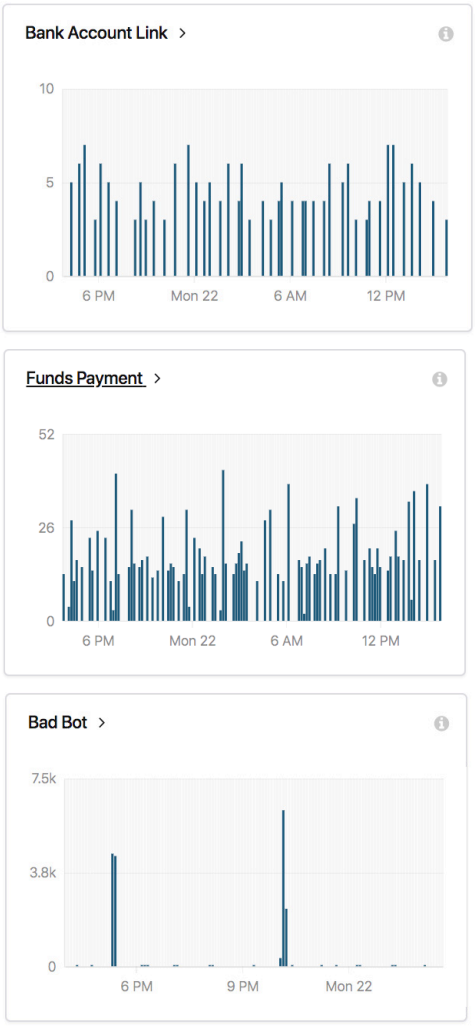
Signal Sciences spans the breadth of your applications, pinpointing application logic flaws and problems based on your unique business logic. One of our customers, Jon Oberheide, Co-founder and CTO of Duo Security, says it best: "The Signal Sciences approach gives us situational awareness about where and how our applications are attacked so that we can best protect ourselves and our customers."

Using the Signal Sciences, organizations can instrument any flow that exists in their application logic. This allows them to gain insight into their application and bring awareness to the development, security and operations teams.

Signal Sciences defends what matters.

Signal Sciences Next-Gen WAF and RASP

Signal Sciences takes a unique approach to web application security. Our platform identifies common web application attack vectors like XSS, SQLi and other OWASP Top Ten attacks—but it doesn't stop there. Using our Power Rules, users can detect business logic flaws, user account takeovers, or monitor any application flow they desire. Whatever you need to watch more closely, you can.



Signal Sciences monitors, alerts and defends whatever it is that you care about most: from account linking, to payment flows, or even keeping malicious bots at bay.

We work with organizations every day to stop account takeovers and make them a thing of the past. They use our flexible platform to identify login attempts, password reset flows, and suspicious network flows. Combined with our out-of-the-box instrumentation and defense, in a short amount of time, with no changes to any authentication or application code, you can establish account takeover defense you can depend on.

Never wonder "Why didn't we detect this sooner?" again.

Methodology

Signal Sciences utilized a random sampling from a 30-day timeframe (June 1 – July 1, 2017). The data encompasses login attempts, login successes, login failures and several other authentication based data points available in Signal Sciences.

Endnotes

[1] The Forrester Wave(TM): Risk-Based Authentication, Q3 2017

[2] Signal Sciences utilized random sampling from a thirty day timeframe (approximately June 01, 2017 - July 1, 2017). The data encompasses login attempts, login successes, login failures and several other authentication based data points in Signal Sciences next-gen WAF and RASP.

Please visit www.signalsciences.com to
learn more about our platform.