Learning from Learning: Detecting Account Takeovers by Identifying Forgetful Users

Sean McElroy
Learning from Learning: Detecting Account Takeovers by Identifying Forgetful Users

GIAC (GCIH) Gold Certification

Author: Sean McElroy, me@seanmcelroy.com
Advisor: David Hoelzer

Accepted: October 12, 2020

Abstract

By measuring a user’s increasingly familiarity with a web application over time, outliers in use may indicate account takeover fraud. Credential stuffing attacks are increasing in frequency, allowing threat actors to use data breaches from one source to perpetuate another. While multi-factor authentication remains a crucial preventative measure to protect against credential stuffing, the availability of credential data sets with contact information and the correlation with demographic data can allow threat actors to overcome it through interactive social engineering. Concurrently, alternative defense mechanisms such as network source profiling and device fingerprinting lose effectiveness as privacy-protecting technologies reduce the observable variability between legitimate and fraudulent user sessions. This paper explores the potential of clickstream data containing logs of users’ navigation through a web application as an alternative defense to detecting account takeover activity for digital banking platforms. By identifying when users are exhibiting learning behaviors, the detection of such behaviors for established users may provide an indicator of compromise.
Learning from Learning: Detecting Account Takeovers by Identifying Forgetful Users

1. Introduction

Credential stuffing attacks take advantage of a common human habit related to passwords: 65% of users reuse the same passwords across multiple systems (The Harris Poll, 2018). Instead of relying on low password complexity, as many brute force attacks do, a credential stuffer reuses usernames and passwords disclosed in previous data breaches against a different target system. Because these attacks only attempt to access a system with a single credential for each user, they often do not trigger account lockout systems that a brute force methodology would. This may be costing U.S. financial institutions alone up to $50 million per day (Shape Security, 2018). When sourced from a distributed network, such as a botnet, and with activity spread across several days or weeks, credential stuffing attacks can be challenging to distinguish from actual users’ failed login attempts. Worse, network providers report observing billions of such credential stuffing attempts monthly and warn that the rate of incidents is increasing substantially (Bolstridge, 2018).

1.1. Challenges for Digital Banking

One effective defense against credential stuffing is implementing a mandatory or risk-based two-factor authentication (2FA) mechanism. Even when an attacker might have a username and password pair, if they are unable to receive an out-of-band one-time password or otherwise complete a device or biometric factor of authentication, knowledge alone is insufficient to compromise a user’s account. However, threat actors motivated for financial gain (71% of all breaches) are increasingly successful in using stolen credentials, including credential attacks, as a prelude to interactive social engineering techniques against digital banking platforms (Verizon Data Breach Report, 2019).

Fraudsters can use the subtle difference in website responses to determine which of their stolen credentials are valid on their target site, forge a Caller ID name to mimic the targeted financial institution, and pose as a call center representative to induce a victim to read back a one-time password so it can be typed in by the attacker. Such methods are often successful because many such systems’ defenses hinge on a strong perimeter. However, once authenticated, a threat actor may have access to steal identity

Author Name, email@address
information or move money. Fraudsters have been successful by circumventing perimeter defenses completely by targeting the weakest link – the human element. While ‘Nigerian prince’ scams may have limited effectiveness in 2020, by posing as known acquaintances on social media or the institution itself, fraudsters remain capable of socially engineering credentials from users.

Advanced adversaries filter their potential targets by ZIP code, neighborhood, or other indicators of potential wealth to prioritize their manual social engineering campaigns. They may only use stolen account access to view trial deposit amounts used to verify account ownership by a remote. From a financial institution’s point of view, the user may only appear to have logged in from another location, completed a 2FA challenge, and then issued an increasing series of ACH debits to slowly drain an account to avoid transaction fraud detection systems.

1.2. Existing Detection and Prevention Methods

Banks and credit unions have long been targets of attackers, ranging from individuals (Sheridan, 2019) to highly sophisticated and well-funded nation-state actors (FireEye, Inc., 2018). In response to these threats, the financial services industry has a comprehensive overlay of regulatory requirements and guidance, regular examinations from increasingly savvy IT government agency auditors. A rich ecosystem of commercial solution providers offers integrable services for digital banking platforms. These products include perimeter defenses, such as Imperva’s Bot Detection, LexisNexis Risk Solutions’ Threat Metrix risk detection, and factors of authentication such as RSA SecurID. However, such methods often presume the victim is not a party to the attack, which is the case when they are compromised through interactive social engineering.

Recently, the idea of ‘continuous authentication’ has gained the interest of security practitioners. Continuous authentication considers a broad set of behaviors profiled for a user, including access patterns, to provide more than authentication factors to score the certainty the user’s identity matches expected activities. This idea is appealing because it incorporates or relies on a rich context beyond individual knowledge or possession-based factors of authentication. As digital banking continues to shift to mobile form factors, the device sensors on smartphones allow for intriguing use cases,
from motion-based profiling via accelerometer readings to gait analysis as users move about the physical world (Abuhamad, Abusnaina, Nyang, & Mohaisen, 2020). Profiling users in this manner can be problematic, since device manufacturers continue to make changes to limit continued access to these sensors to preserve privacy and conserve battery power. General solutions that operate at the physical device layer may not be consistently accessible, but the principle of establishing a continuous authentication methodology within an application may hold merit in identifying threat actors attempting to take over accounts through credential stuffing attacks.

1.3. Opportunities in Learnability

When reviewing audit logs of a consumer’s account takeover, for those incidents where an attacker interactively accessed an account with a stuffed credential, certain patterns of behavior are anecdotally apparent. Users who are familiar with a system appear to exhibit goal-directed behavior when navigating web interfaces. In contrast, users who access a digital banking platform for the first-time appear to meander through screens, discovering features and paths before requesting an action. Software usability is a concept familiar to industrial engineering and is formally defined by ISO/IEC 25010:2011. A component of usability is learnability, which is defined in part as the “degree to which a product or system can be used … to achieve specified goals of learning to use the product or system with effectiveness” (International Organization for Standardization, 2011).

Learnability is certainly important for a product. Design teams should measure the time it takes a user to become familiar with and quickly accomplish tasks using the technology. A rich history of physiological response measurement and analysis exists in industrial engineering and human factors research (Akamatsu, Green, & Bengler, 2013). Measuring learnability has also become important in software engineering. Advances in eye-tracking, expression monitoring, and machine learning have extended these techniques that companies incorporate into their product’s lifecycles (Juin, Diah, Ismail, & Adam, 2017). As an internationally and well-defined usability concept with mature assessment techniques established, measuring learnability may provide an opportunity...
not only to measure the time it takes for a user to learn how to use a system’s features, but it may also indicate “re-learning.”

Prior academic research has investigated measures for indications of re-learning to identify potential “learning issues” that may stem from system usability design flaws (Marrella & Catarci, 2018). Machine learning and anomaly detection have long held the interest of researchers and practitioners alike in detecting digital banking fraud, with varying levels of success. However, an opportunity exists to determine whether the identification of learning behaviors among frequent users could be indicators not of application usability flaws, but fraudulent activity. If applied in the context of online financial services, it could provide a way to overtly deny account takeover attempts or covertly classify suspect sessions for fraud monitoring.

2. Research Method

2.1. Source Data Set

To examine the question of whether the detection of learning behaviors in user activity data can predict fraud, the researcher obtained the permission of a digital banking solution provider to obtain access to a repository of anonymized “clickstream” data. The data was labeled based on whether a session was associated with a financial institution report of fraudulent activity. The researcher obtained: a full of this clickstream data in CSV format that contained an incrementing database identifier for each user session, the click date and time, a GUID representing the user, the URL path for the HTTPS request, the user’s IP address, the user-agent string, and a GUID representing the unique session for the user. Importantly, the platform’s application server generated these logs and not third-party analytics services that operate solely on the user’s client, such as Google Analytics. Because many user-agents and browser extensions block or degrade the effectiveness of user profiling and clickstream tracking for enhanced privacy, using those sources may skew the analysis of specific user segments.

A variety of constituents utilize digital banking, including end-user consumers and account aggregators that log in on behalf of users using their credentials. Also, synthetic transaction monitoring tools, which authenticate with test accounts to measure...
Learning from Learning: Detecting Account Takeovers by Identifying Forgetful Users

the platform's performance and responses, can skew the data set. Similarly, other sources of automated access, such as dynamic application security testing tools, may represent a significant number of sessions over time and may not be representative of user behavior. For this reason, the researcher used the IP address and user-agent strings to filter out aggregation and performance-monitoring platforms. Similarly, the researcher used tools including `sed`, `grep`, `cut`, and `grepcidr` to exclude activity from the institution and platform provider itself to exclude testing and support activity from this analysis.

The researcher also wrote a series of scripts to normalize paths for path analysis, including replacing GUID and numerical resource identifiers to placeholders, such as condensing `/accounts/activity/f5ca9a9c-b806-44e3-bfa2-fc791c486cb` to `/accounts/activity/*GUID*`. Because the IP address and user-agent data are unnecessary for this experiment, the researcher securely deleted it once IP-based filters were applied.

By grouping 64,747,197 individual navigation events in 19.4 GB of logs, such as “User navigated to /accounts”, into 10,387,421 unique session identifiers, the researcher prepared session-indexed paths for the experiment. The researcher split the resulting population of session events into two data sets. The first set contains sessions of users who have logged in at least ten times prior, but four times in the preceding 90 days (the “infrequent users’ sessions” set). The second set contains sessions for users who logged in at least ten times prior and four or more times in the preceding 90 days. By dividing the population of user sessions by the anticipated learning behavior activity of each segment’s population, the researcher measured the relative strength of the detection method for each.

2.2. Experiment Design

2.2.1. Preliminary Observations

To construct a potential measure of learning behavior, the researcher created an algorithm based on observation and analysis of the first-time sign-on data and made several observations when searching for potential differences between infrequent user sessions and engaged users’ sessions. First, the average number of pages visited per session was higher (9.3 vs. 6.3) among infrequent users. Second, infrequent users were
more likely to navigate areas providing infrequently changed settings, such as user profile information (7.0% vs. 3.2% sessions). Third, engaged users were more likely to modify banking alerts and log in to view or change alerting settings (3.9% vs. 1.6% of sessions).

Unexpectedly, in the search for “meandering” behaviors, where a user returned to the same pages more than once in a given session, the average number of total pages vs. unique pages visited per session was consistent at a 2.0 ratio among all user classes. A significant number of users accessing the system (52.9%) log in and view information on a comprehensive dashboard and log out after reviewing only one or two total pages per session. For this reason, such sessions lacked the resolution required to discern whether users were familiar with a system or did not need to learn it because their reason to log in was satisfied without needing to navigate through additional screens.

2.2.2. Detection Algorithm

Based on the observations mentioned above, the researcher designed and implemented an algorithm to produce a 0 or 1 as to whether it detected a learning behavior within each of the infrequent user and frequent user data sets when a session met three of the four following criteria:

1. The total session time divided by the number of pages was higher than an average of 5 seconds per page.
2. The total pages visited in the session were higher than 7.
3. Users visited help content or entered non-transactional search terms in the application search textbox at the top of the digital banking application.
4. Users viewed profile pages or user settings pages that contained options that are infrequently changed.

Known potential indicators of fraud, such as adding new external accounts or bill pay payees, modifying contact settings, or disabling alerts, were not included in the algorithm. The researcher excluded these indicators since the purpose of the experiment is to determine if the detection of general learning behavior in established and engaged users serve as a fraud detection mechanism, not to detect fraud-related activity directly through this experiment. The researcher implemented this algorithm as a custom .NET

Author Name, email@address
Core program using C# to process the source data set into various reports for analysis in Microsoft Excel.

2.3. Limitations of Approach

The design of this experiment has inherent limitations that other researchers may consider when replicating or implementing it in other environments. First, web applications vary widely in their design, and page-level navigation may not provide sufficient specificity to measure learning behaviors. For example, suppose a URL with a path of /home contains rich functionality that allows a user to perform many actions without navigating to separate web pages. In that case, a URL-based log may not represent important activity that may distinguish learning behaviors. Without sufficient optionality and variance in a data set, this technique may be less effective.

Second, different classes of users may learn systems in different ways. For instance, college-aged digital banking users may learn systems faster or explore navigational features of web applications in very different ways than retired users. Different usage patterns may be influenced not only by the user’s age or general familiarity with technology concepts but also by varying spending patterns and net worth, which drive differences in the use of financial services. A struggling credit union member living paycheck to paycheck may log in very frequently to check a balance, yet they still may be unfamiliar with many other features they use less. With a high variance of user activity between separate classes of users, averaging activity together, as this study did for the “engaged users’ sessions” data set, may adversely impact the efficacy of this detection method in practice.

Third, users leveraging assistive technologies, such as screen readers, may appear to use systems in very distinct ways. As an example, users with a visual impairment who depend on an audio readout of a web page to make a selection may “learn” a system very differently than one operating primarily from visual design paradigms and cues. In addition, many user activity data sets, including the one used in this research, do not include indicators when assistive technologies are used. Without a way to control for these variables when building a model, algorithms may encode biases that fail to find
meaningful fraud indicators that would help protect users with visual or mobility impairments.

3. Findings and Discussion

3.1. Dimensional Analysis

A significant portion of this research effort involved the acquisition and preparation of data. The final data set was formatted in a single result file with each record representing a single session. Each record contained fields for the dimensions required for this analysis, including the user’s navigational path during the session, as exemplified in the following figure.

Before the researcher applied the algorithm to the labeled data set of fraudulent sessions (n=62), he performed an analysis of each dimension to visually determine whether each supports the hypothesis that each may indicate a learning behavior exhibited by users with less familiarity. The time spent per page between user navigation events aligned with expectations that first-time users would spend more time navigating through the site than experienced users frequently logging in, presumably because new users are reading more text and are learning the visual cues of the system. While this behavior differentiated around the 8 second average for the tested web application, 41% of sessions had an average of 7 seconds per page or less. This suggests that for user sessions where the user is actively engaged and average navigation time per page is low,
this dimension may be significant to separating those who are familiar with the application from those who are still learning how to use it.

![Figure 2. Learnability as measured by page navigation time](image)

Similarly, evaluating the number of pages a user of a given data set navigated to in a given session, visualized as a cumulative percentage of sessions, suggested the second criteria of the detection algorithm was useful. As expected, first-time users explored more of the digital banking web application, with the last 10% of first-time logins viewing 20 or more pages, but 90% of returning frequent users viewing eight pages or less. Because different functions, such as editing a scheduled banking transfer, require a minimum number of pages regardless of familiarity, the graph of this dimension suggests its value as an independent assessment criterion is limited to goal-directed sessions where the user’s intent and action can be discerned and measured categorically. For the purposes of this research, however, these page views are averaged by the data set as depicted in Figure 3.
Figure 3. Learnability as measured by per-session page depth

For the third criterion, the measure of the percentage of user sessions for each data set of first-time, infrequent, and frequent users also appeared to fit the hypothesis. First-time users were 5.5 times more likely, and infrequent users were 2.3 times more likely to view FAQs or to search for content in an application search textbox than frequent users (2.2% vs. 1.0% vs. 0.3% of sessions, respectively). This visualization supports the supposition that users who use the system more frequently have learned how to use features that new or infrequent users may explicitly need help with.
Finally, first-time users were five times more likely to navigate to rarely-used profile page settings. Infrequent users were twice likely to do the same, as compared to frequent users of the digital banking platform (2.2% vs. 1.1% vs. 0.35%, respectively). Logically, users verify their information and explore these areas, but over time, rarely revisit them unless they have a goal to update settings that affect the overall behavior of their experience, rather than research specific transactions. Because of the stark 5X difference in the activity, given fraudulent sessions often view or change contact settings under these profile areas, this researcher assessed this measure might be an especially strong candidate to differentiate between legitimate and fraudulent sessions for frequent users of digital banking.

Figure 4. Variances in Help Content Use by User Class
3.2. Hypothesis Testing

The researcher processed both source data sets to determine two Odds Ratios (OR) with the algorithm prepared. The Odds Ratio is the ratio of the probability of an event occurring in one group to the odds of it occurring in another (Buis, 2017). To test the hypothesis that learning behaviors, as detected by the algorithm, may be indicators of fraudulent behavior, the Odds Ratio for the population of a data set must be greater than 1.0. The values used to calculate the Odds Ratio to test this hypothesis can be visualized as a 2 x 2 table:

<table>
<thead>
<tr>
<th></th>
<th>Fraud Reported</th>
<th>No Fraud Reported</th>
</tr>
</thead>
<tbody>
<tr>
<td>Algorithm detected learning behavior</td>
<td>$F_L$</td>
<td>$G_L$</td>
</tr>
<tr>
<td>No learning behavior detected by the algorithm</td>
<td>$F_N$</td>
<td>$G_N$</td>
</tr>
</tbody>
</table>

By applying the learning behavior algorithm to the population of sessions in a data set and calculating these four independent values, the Odds Ratio would be:

$$ OR = (F_L/G_L)/(F_N/G_N) $$
In this context, the Odds Ratio is the ratio of the probability that the algorithm will detect a learning behavior in a fraudulent session to the probability that it will not detect a learning behavior in a session later reported as fraudulent. For example, if 3% of sessions flagged as containing a learning behavior were fraudulent, but only 2% of sessions of those not flagged were fraudulent, OR₁ would be 1.5. If this were true, the Odds Ratio would be greater than 1.0, which would indicate that learning behaviors are associated with an increased incidence of fraud. If the OR were < 1.0, the algorithm would not be an effective method of detecting the learning behaviors hypothesized to be exhibited by users with less familiarity or usage of the system. The findings must show OR > 1.0 to suggest this mechanism may be useful for detecting learning behaviors that are also indicative of digital banking fraud.

After the experiment, the learning behavior detection algorithm’s output resulted in the following coefficients:

<table>
<thead>
<tr>
<th></th>
<th>Fraud Reported</th>
<th>No Fraud Reported</th>
</tr>
</thead>
<tbody>
<tr>
<td>Infrequent Users’</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Sessions</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Algorithm detected</td>
<td>10</td>
<td>50,716</td>
</tr>
<tr>
<td>learning behavior</td>
<td></td>
<td></td>
</tr>
<tr>
<td>No learning behavior</td>
<td>10</td>
<td>1,517,423</td>
</tr>
<tr>
<td>detected by the algorithm</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Frequent Users’</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Sessions</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Algorithm detected</td>
<td>31</td>
<td>32,178</td>
</tr>
<tr>
<td>learning behavior</td>
<td></td>
<td></td>
</tr>
<tr>
<td>No learning behavior</td>
<td>16</td>
<td>3,304,295</td>
</tr>
<tr>
<td>detected by the algorithm</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

As a result, the Odds Ratio for infrequent users’ sessions was OR₁ = 29.92. For frequent users’ sessions, the ratio was OR₂ = 205.38. Both Odds Ratios are greater than 1.0, indicating the detection of learning behaviors using the dimensions defined in this research can indicate fraud. Moreover, since OR₂ > OR₁, this strongly suggests that the ability to detect fraud is improved for users who frequently utilize the system.

Increasing user engagement with the application has many commercial benefits for the provider in retaining customers and expanding relationships as the institution can better know and cater to the user’s needs. Moreover, this research indicates there is a
Learning from Learning: Detecting Account Takeovers by Identifying Forgetful Users

mutual benefit in that as users learn a system and regularly engage with it, they improve their knowledge of how to use it. Consequently, they provide the platform data that can be used to indicate account takeover fraud risk.

4. Recommendations and Implications

As this research demonstrates, application usage data can be leveraged to detect fraud by identifying learning behaviors. Collecting the requisite data, operationalizing the analysis of it, and properly leveraging results in post-detection actions is critical to realizing the value of this detection methodology.

4.1. Recommendations for Practice

This research used data widely available in web page analytics solutions, namely, user and session identifiers, date and timestamps, and page navigation paths. Many applications log this data in server-side repositories for diagnostic purposes. However, to leverage auditing to mitigate fraud losses, this user session data must be analyzed after the session has ended but within a period sufficient to allow for analysis of flagged user sessions and to stop a loss before it is realized. An appropriate place to add learning behavior detection routines may be an enrichment step in an existing logging pipeline, such as through a custom Logstash filter.

Not all web applications will benefit from this algorithmic detection method. Applications that lack feature breadth or those that provide only linear flows, such as a checkout process where there is little optionality or provide little user choice, will not require users to learn how to navigate through or use the system to a degree sufficient enough to detect distinct learning behaviors. Application security teams interested in identifying learning behavior dimensions and designing quantitative measures should collaborate with product design teams on initial and ongoing efforts. Measuring usability and learnability are concerns for product interface designers, and the feedback from focus groups and quantitative usability measures can inform security implementations of this technique.

Author Name, email@address
4.2. Implications for Future Research

While this research focused on learning behaviors broadly, these behaviors likely vary significantly by user segment and medium. For instance, an elderly consumer accessing a web application on a desktop computer in an office may exhibit different behaviors than an active teenage user on a mobile device. Accounting for the variances in application usage, these differences in form factors, timing, and the inherent time it takes different groups to learn and become familiar with an application may produce a more accurate model for detecting the differing behaviors a subsequent account takeover session would exhibit.

Usability and learnability measuring tools in the context of user interface design and product management may supplement security-drive modeling of learning behavior. For instance, at least one such commercial tool provides for the detection of frustrated users through “rage, dead and error clicks or high rates of abandoned forms” (FullStory, 2020). These behaviors, which may not be recorded by server-side application event auditing mechanisms, if incorporated, may further enhance the modeling of learning behavior for the purposes of fraud detection.

Finally, this research was focused on the population of session events available at the conclusion of a user’s session to render a detection result. In systems where user requests are made, but there is sufficient time to review post-session detection alerts to halt a fraud loss, this methodology may integrate well into existing back-office processing routines. However, some systems may incur fraud losses before operators can stop them, such as in real-time payment scenarios or when in systems where users are delegated to a single sign-on operated by a third-party provider. For those systems, researching methods to identify learning behaviors reliably before a session was concluded may be significantly more difficult, but may add value to systems commonly targeted by fraudsters.
5. Conclusion

Account takeover fraud will be a significant challenge for online service providers for a long time to come. With so many avenues available to threat actors to circumvent technical controls by socially engineering providers and their users, a layered approach to continuously authenticating and assessing users is necessary to recognize and stop the fraud losses that account takeovers can create. This research demonstrates that by measuring when a user is learning the system, that measurement can help detect fraud when such behaviors are observed in sessions for users who have established familiarity or mastery of a user interface.

Notably, the results of this fraud detection method in the context of digital banking can be implemented with existing application usage data common to many types of web applications. Formulating measures of learning behavior is not so complicated as to require data analytics expertise. With consistent and comprehensive server-side application logging, basic analytical tools can implement this technique. Operationalizing this detection method at scale generally does not require esoteric or expensive machine learning or anomaly detection models or tools. Security analysts have an opportunity to work closely with product managers, user interface designers, and engineering teams to mitigate the potential for fraud by collectively learning how to recognize when users are learning.
References


Learning from Learning: Detecting Account Takeovers by Identifying Forgetful Users


Author Name, email@address
Appendix

1. Source code for data transformation is made available under the GPL 3.0 license at https://github.com/seanmcelroy/clickstream-fraud
## Upcoming SANS Training

Click here to view a list of all SANS Courses

<table>
<thead>
<tr>
<th>Course</th>
<th>Location</th>
<th>Date</th>
<th>Format</th>
<th>Pacing</th>
</tr>
</thead>
<tbody>
<tr>
<td>SANS Essentials Australia 2021</td>
<td>Melbourne, AU</td>
<td>Feb 15, 2021 - Feb 20, 2021</td>
<td>Live Event</td>
<td></td>
</tr>
<tr>
<td>SANS OnDemand</td>
<td>OnlineUS</td>
<td>Anytime</td>
<td>Self Paced</td>
<td></td>
</tr>
<tr>
<td>SANS SelfStudy</td>
<td>Books &amp; MP3s OnlyUS</td>
<td>Anytime</td>
<td>Self Paced</td>
<td></td>
</tr>
</tbody>
</table>