Characterizing Security and Privacy Practices in Emerging Digital Credit Applications

Jasmine Bowers
jdbowers@ufl.edu
University of Florida
Gainesville, Florida

Imani N. Sherman
shermani@ufl.edu
University of Florida
Gainesville, Florida

Kevin R. B. Butler
butler@ufl.edu
University of Florida
Gainesville, Florida

Patrick Traynor
traynor@ufl.edu
University of Florida
Gainesville, Florida

ABSTRACT

Access to credit can provide capital crucial to both businesses and individuals. Unfortunately, for large parts of the developing world, access to credit is not available because customers often lack the traditional data used by lenders to make such decisions (e.g., verifiable payroll statements, property ownership documents). Emerging online credit services address this need through the use of non-traditional creditworthiness data, which many believe to include user geolocation and social network information. While such systems both potentially expand credit availability and improve usability through instant evaluation, their security and privacy practices remain opaque. In this paper, we perform the first comprehensive security analysis of the emerging online credit space. To provide improved transparency, we select 51 representative companies across the industry, analyze their privacy policies and compare them to the sensitive data types mobile applications actually gather. We then evaluate the configuration of connections between mobile apps and their supporting servers to determine whether they securely handle such data. Our analysis demonstrates significant security and privacy issues across this burgeoning industry, including the gathering of previously undisclosed data types and widespread misconfiguration of encryption. We conclude by discussing our efforts to work with partners in and around the industry to improve these issues.

ACM Reference Format:
Jasmine Bowers, Imani N. Sherman, Kevin R. B. Butler, and Patrick Traynor. 2019. Characterizing Security and Privacy Practices in Emerging Digital Credit Applications. In *WiSec '19: Conference on Security and Privacy in Wireless and Mobile Networks, May 15–17, 2019, Miami, FL, USA.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3317549.3319723

WiSec '19, May 15–17, 2019, Miami, FL, USA

ACM ISBN 978-1-4503-6726-4/19/05.
https://doi.org/10.1145/3317549.3319723

1 INTRODUCTION

Mobile phones and networks are transforming the world of finance, creating opportunities for widespread financial inclusion, especially among traditionally overlooked regions and groups. Digital credit offers a particularly important lifeline. Individuals and small businesses in these settings often suffer from the inability to acquire financial assistance. As a consequence, a farmer may not be able to repair the vehicle needed to bring goods to market, a merchant may not be able to fully stock shelves, or an individual may not be able to pay for an unexpected medical procedure.

There are a number of challenges to bringing credit to these settings. Because individuals and businesses in these overlooked regions and groups often lack the types of data that lenders use in more formal settings to make credit decisions (e.g., audited tax forms, pay stubs, property ownership documents, etc.), efforts to simply reapply existing lending mechanisms into this context have failed [20]. A rapidly increasing number of companies have created alternative means of measuring creditworthiness based on observing transactions made through mobile applications and other data. While the types of such data have been considered in general terms (e.g., GPS, social networks), a detailed analysis of which data these applications request has not been conducted.

In this paper, we conduct the first such analysis. We begin by studying the privacy policies of 51 online credit applications to search for specific references to the types of data they request for making credit decisions. We then evaluate Android manifest files to determine the sensitive data types for which applications specifically request access. Finally, we evaluate the transport security provided by these applications as they transport such information back to the online credit providing company. Our analysis demonstrates that while such systems hold the potential to dramatically expand financial inclusion, their significant gathering of data and flawed information security practices may put millions of vulnerable customers at significant risk.

In so doing, we make the following contributions:

- **Privacy Policy Analysis:** We examine both the readability and content of privacy policies for online credit applications to determine 1) how readable the privacy policies are for their target audience and 2) if the privacy policies clearly identify the data they collect.

- **Analysis of Data Collection:** We perform the first industry-wide measurement of the data types collected by online credit applications via analysis of Android manifest files. While our study confirms applications requesting previously known types (e.g., location), we also demonstrate that significant amounts of data not included in the above privacy policies is requested (e.g., access to microphones, access to calendars).
- **Audit of Transport Security Practices:** We evaluate the configuration of mobile applications and web servers to determine the effectiveness of transport security mechanisms in this space. Through a combination of reverse engineering and dynamic analysis, we demonstrate that many services allow deprecated ciphers (e.g., DES, DES40, 3DES), fail to properly authenticate certificates, and use static salt values for passwords.

Understanding and improving the security of this industry is critical. From a usability perspective, such applications allow for users to be instantly evaluated by obviating the need for traditional sources of data, thereby immediately providing qualified candidates with access to funds. Moreover, as many nations race towards credit scores for all citizens [2], it is crucial that the data being used to make such decisions and the security of that data be well-established. Failure to do so may ultimately lead to bias against underrepresented groups and needless exclusion from the growing global financial system.

The remainder of this paper is organized as follows: Section 2 provides an analysis of related work; Section 3 discusses our selection process; Section 4 analyzes the privacy policies, recovers the Android manifest files and compares their contents to those privacy policies; Section 5 provides an analysis of the transport security mechanisms for the mobile applications and web servers implementing these services; Section 6 discusses recommendations; and Section 7 offers concluding remarks.

2 RELATED WORK

Over the years, researchers have conducted in-depth measurement studies on mobile apps. Some assessed mobile apps using various code analysis techniques, while others have assessed the readability, availability and content of privacy policies. This research aims to join these two distinct research areas in order to develop a comprehensive view on how mobile apps, specifically credit granting apps, handle user data.

Harris et at. [14] noted that inadequate privacy protections can lead to abuse and data breaches. Several privacy policy guidelines [17, 28, 30] have been established in the research community, in an effort to improve the usability, design, and authorship of policies. From a nutrition label format [19] to bulleted lists [11] and browser extensions [39], researchers continue to devise new and improved methods of disseminating privacy related information. Reidenberg et al. [29] noted that such efforts have not been implemented at a scale large enough to make a significant impact.

In 2001, Hochhauser [15] assessed the readability of top U.S. financial institution privacy policies, and although the Gramm-Leach-Bliley Act (GLBA) was enacted two years prior in order to hold financial institutions accountable for explaining data sharing practices [9, 37], some policies still remained insufficient. Similarly, in 2004, Jensen [16] analyzed readability, accessibility, content, user needs met, and usability of popular online websites' and health-care sites' privacy policies. The following year, Sheng and Cranor [31] conducted an automated analysis of 50 U.S. finance-related company policies from 1999 to 2005 and found no significant improvement following the GLBA enactment. Recently, in 2016, Cranor [6] also conducted large-scale analysis of sharing practices, third party interactions, and opt-out options covered in over 6,000 policies of financial institutions across the U.S. and in 2017, Bowers et al. [3] compared privacy policies of U.S banks and mobile money apps. Both found issues in the policies regarding sharing practices and user options, such as opt-ing out.

In addition to privacy policy research, several researchers have assessed the security practices of finance apps. In 2015, Reaves et al. [27] analyzed mobile money Android apps and found several vulnerabilities, including improper certificate validation, poor cryptography, and information leakage. Two years later, they conducted a follow-up study and the results [26] showed that most apps remained vulnerable. In 2016, Castle et al. [4] assessed financial services Android apps permissions and data handling practices and found some permissions to be concerning, e.g., access to the microphone and flashlight. In addition, nearly 10% of the apps lacked HTTPS URLs, which meant those apps were potentially sending sensitive information without encryption.

In an effort to increase app analysis research, many app analysis tools have been developed over the past two decades. Reaves et al. [25] categorized tools published since 2010 and point out the unique capabilities of each. The static analysis tool Mallodroid [8] was designed to detect man-in-the-middle (MITM) attacks due to inadequate use of SSL/TLS. Epicc [22] was designed to detect inter-component communication (ICC) vulnerabilities, while DroidSafe [13] analyzes static information flow of an app and detects leaks of sensitive user data. Flowdroid [1] facilitates static taint app analysis, where taint analysis involves tracking the flow of tainted information through an app. Along with FlowDroid, DroidBench, a test suite of app, was developed to evaluate all taint analysis tools. Similar to EPICC, Amandroid [38] analyzes inter-component control as well as the flow and context of data to determine where the app is vulnerable.

This research aims to contribute to both research areas in an effort to bridge the gap and present a comprehensive view of what credit granting apps collect, how they address it in their policies, and how they protect it within their apps.

3 ONLINE CREDIT PROVIDERS

Around the world, digital lending companies are starting up and growing. Many new companies are reaching out to customers at the base of the economic pyramid. Digital loans are offered through mobile phone apps or through websites (including websites optimized for mobile). The loan products supported through digital means are varied. Many are aimed at consumers; others focus on small enterprises. Amounts and loan tenures vary from very short term "nano" loans of a few dollars to medium term small business loans of a few hundred or thousands of dollars. Some companies have grown to substantial, even massive, scale, while others are just

Company	Country
Airtel	Democratic Republic of Congo
Atom	United Kingdom
Azimo	United Kingdom
BlueVine	United States
Branch	Kenya
C2FO	United States
Coins	Philippines
CommonBond	United States
Creditas	Brazil
CrowdEstates	United Kingdom
EcoCash Loans	Zimbabwe
Equitel	Kenya
Equity Direct	Kenya

Company	Country
Farm Drive	Kenya
FastPay	United States
GetBucks	South Africa
GoPaysense	India
IndiaMart	India
Insikt	United States
InstaPaisa	India
InvoiNet	Argentina
Jimubox	China
Kabbage	United States
KCB	Kenya
Koopkrag	South Africa
Kopo Kopo	Kenya

Company	Country
Kubo Financiero	Mexico
Lending Club	United States
Lulalend	South Africa
M-Pawa	Kenya
M Co-op Cash	Kenya
Micromobile	Kenya
MoneyTap	India
OnDeck	United States
Pay Your Tuition Funds	United States
PesaPata	Kenya
Prosper	United States
Puddle	United States
Road Loans	United States

Company	Country
Saida	Kenya
Salud FÃącil	Mexico
SMECorner	India
Social Lender	Nigeria
Suregifts Redemption	Nigeria
Tala	Kenya
Taplend	United Kingdom
Tencent	China
Upstart	United States
WeFinance	United States
Yoco	South Africa
Zidisha	Kenya

Table 1: Digital lenders evaluated in this study

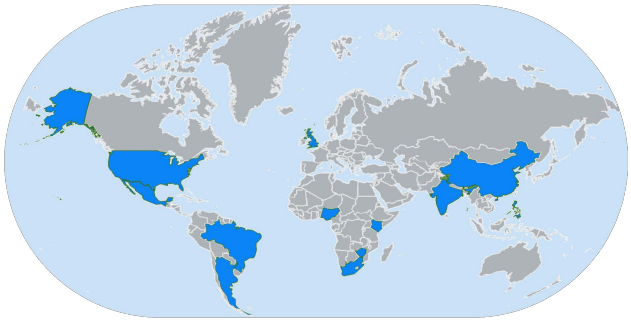

Figure 1: Countries represented by our selection of digital credit companies

starting out. All these companies have at least two things in common: they use online and mobile tools to connect with customers and they use a range of customer data, obtained electronically, in making their credit decisions.

Digital credit companies exist in a wide array of markets across the world. While many focus on small businesses, others work with individuals. Our analysis characterizes the security and privacy practices of a wide swath of this industry, including both leading names in the industry (such as Tencent in China and Lending Club in the U.S.) and smaller efforts (such as Coins in the Philippines and Branch in Kenya). Table 1 lists the 51 (37 international and 14 U.S.) digital credit companies we analyzed for this project.

We considered multiple factors in our selection. First, all the analyzed companies have a mobile application or a version of their website optimized for mobile devices. Second, these companies offer a broad range of geographic coverage, spanning 13 countries across 5 continents (Figure 1). This study is not meant to cover every existing digital credit provider, but only a representative sample, both by region and type of product (SME lending, P2P lending, payroll lending, etc.).

4 CLARITY OF PRIVACY POLICIES

Big Data promises to deliver customized services at an accelerated speed, improve customer satisfaction through personalized product design and increase profits through targeted advertisements. However, access to the customer data needed to facilitate these improvements may also have unintended consequences. Personal information that some individuals may prefer to keep private (e.g., being pregnant, sexual orientation, behavior patterns) has become discoverable through the use of modern data analytics techniques [21]. Financial information, affiliations (political, religious, etc.) and other data may also become involved. As such, many consumers look to the acquisition and use of their data with apprehension.

Privacy policies can help alleviate some of this fear by offering a public explanation of how a company intends to handle user data, use it, share it, the conclusions that may be drawn from it, and what rights customers have to correct such information. If a company follows its own privacy policy, the actual risks of improper data use may be reduced. Unfortunately, no two privacy policies are the same, and many are vague, unclear or incomplete. This makes it easier for a company to do what they'd like while also following their privacy policy.

Our first exercise in this study was to characterize privacy policies for digital lenders. We seek to determine whether such policies discuss critical issues, how they compare against traditional financial offerings, and the accessibility or readability of such policies for customers from different backgrounds.

4.1 Methodology

Our study of privacy policies addressed two high-level questions:

(1) How readable are these policies for their target audiences?
(2) Do the privacy policies published by digital lenders clearly identify the data they collect?

4.2 Privacy Policy Specifications Considered For This Analysis

In addition to the previously mentioned policy analysis, the app manifest files were reviewed to see what permissions were being requested. Figure 4 shows the permission groups and the frequency of each. Most apps require access to the device's identity, location,

phone call capability, photos/media/files, storage, and device ID and call information. Although such permission groups may be requested for legitimate reasons (e.g., phone call capability for automatically dialing customer service), all of these can be misused to leak private information about the customer. Therefore it became pertinent to investigate if the privacy policies mentioned any of the permission groups requested by name, just as a user might, within the privacy policy.

We note that mentioning the name of the permission group requested was sufficient to credit the privacy policy with addressing a given permission. This should not be construed as an indication that such policy comprehensively covers a topic; rather, our goal is to determine if coverage of the permission group was even considered. We adopt this approach because we seek empirical evidence of compliance with the guidelines we have mentioned in Section 2; arguing for or against the quality of coverage beyond this is an important task, but one better undertaken by more specialized researchers (e.g., policy specialists, consumer rights advocates). Accordingly, the results herein should be viewed as a starting point for discussion and are by no means an endorsement of any of these policies.

4.2.1 Permission Groups. The permission groups used in the manifest files were predefined by Google [12]. The list of permission groups below only includes those requested in the manifest files of the apps.

Device & App History: Allows the app access to sensitive log data, web bookmarks, web history, currently running apps and internal system state.

Identity: Allows the app to determine the device ID.

Calendar: Allows the app read and write access to the calendar.

Contacts: Allows the app to access saved information about the contacts stored on that device.

Locations: Allows the app to get the location of the device using GPS or network location sources.

SMS: Allows the app access to incoming, outgoing, and stored SMS messages.

Phone: Allows the app to access call log, add voicemails, control outgoing calls, get the device phone number and cellular network information.

Photos/Media/Files: Allows the app access to stored photos, files and other media.

Storage: Allows the app to access information in the storage and add files to the storage.

Camera: Allows the app access to the camera.

Microphone: Allows the app to record audio.

Wi-Fi: Allows the app access to the wireless connection information.

Call: Allows the app access to the call log, and information about incoming and outgoing calls.

4.3 Analysis of Readability

The applied linguistics community has developed several tests to automatically measure the readability of text. These tests produce a "grade level" that is meant to reflect the suggested level of education needed to fully comprehend the body of text. For example, although a gold-standard technique is still debated, it has become common in certain fields to use several tests for one study [10]. Our analysis includes calculating scores based on the following readability scoring mechanisms: Flesch-Kincaid Grade Level, Gunning Fog Index, Coleman-Liau Index, SMOG Index, and Automated Readability Index. Once individual scores were tallied, we calculated the overall average score. In addition, we also calculated the estimated time-to-read and word count.

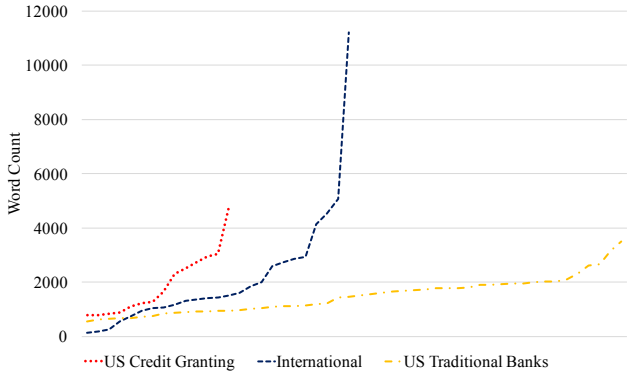

Figure 2: Word Count

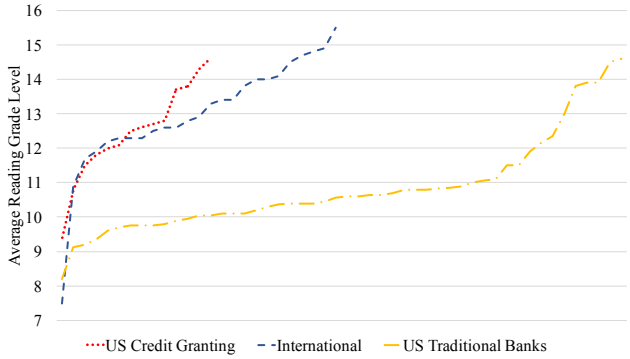

Figure 3: Average Reading Grade Level of Privacy Policies

	International Digital Lenders		U.S. Digital Lenders		U.S. Banks
Min	7.5	Atom	9.4	Road Loans	8.2
Max	15.5	Invoinet	14.6	Fast Pay	14.6
Med	12.9		12.6		10.6

Table 2: Privacy Policy Reading Grade Level

4.4 Results

4.4.1 Reading Grade Level Scores. Privacy policy reading grade level statistics represent the average reading grade levels of our three sets of policies (see Figure 3). As a point of reference, it is widely assumed that the average American citizen reads at approximately an 8th grade level. Our measurement shows that the grade levels of the digital lenders policies were higher than the U.S. traditional bank policies which we include as a baseline to show impact of regulation, e.g. through the GLBA. As shown in Table 2, the

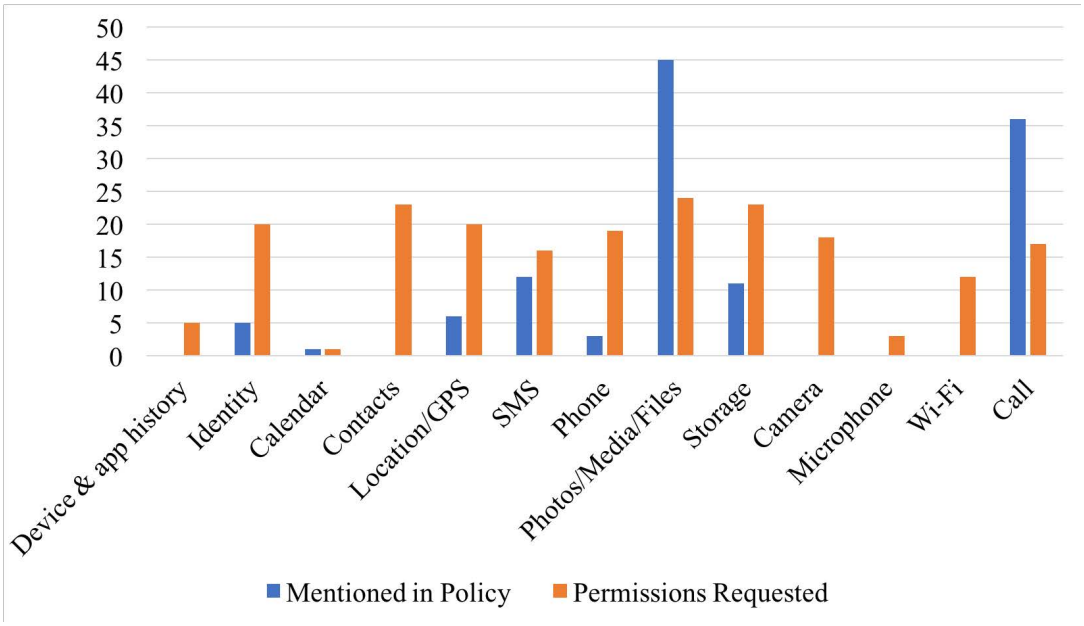

Figure 4: A comparison of permissions requested (26 apps) vs. permission-based content mentioned in the policies (50 policies). In two cases, references to permissions in the privacy policies outnumber the number of apps that request the permission. However, in most, the number of apps that requested permissions was either equal to or greater than the number of policies that mentioned the permissions.

	International Digital Lenders		U.S. Digital Lenders		U.S. Banks
Min	151	Invoinet	788	Road Loans	557
Max	11210	KCB	4847	Prosper	3494
Med	1436		1470		1488

Table 3: Word Count Statistics

median of the international policies is higher than the median of both sets of U.S. policies. With a median of 13.1 years, customers would need at least one year of higher education to fully grasp the meaning of the policies. In contrast, average school attainment in Nigeria and the Democratic Republic of the Congo, for example, is 9 years [5]. Hence, it is important that providers and policy makers aim for content that users of varying educational backgrounds can read and understand.

4.4.2 Word Count. Figure 2 shows the word counts of all three sets of privacy policies. Aside from one outlier policy of over 11,000 words, the word counts ranged between 0 and 6000. As shown in Table 3, the median word counts of each data set were relatively close. However, the minimums were far less similar, with Invoinet having a word count of 151. In addition, Invoinet only covered 2 of the 11 categories of policy content. We found that policies of lower word count are more inclined to cover less user privacy content than those with higher counts. In contrast, we cannot conclusively say that longer policies are fully comprehensive. However, longer policies can reflect increased effort put forth to fully cover critical policy topics.

4.4.3 Request and Mention of Permissions. Although most of the apps requested access to *Storage*, *Location*, and *Contacts* permission groups, only a small number of companies mentioned these in their privacy policy. As shown in Figure 4, this trend was seen across the board for all of the permission groups requested with the exception of *Photos/Media/Files* and *Calls*. 47% of the apps listed the *Photos/Media/Files* permission group in their manifest file, while 90% of them mentioned it in their privacy policy. It is unclear why this data is mentioned in the privacy policy and not collected, but may exist simply such that future versions of the software that may select this data may simply do so.

The remaining permission groups were requested by more apps than those that mentioned it in their privacy policy. For example, 46% of the apps requested access to *Storage*; however, only 22% of the privacy policies mention it.

Additionally, the reasons for access to *Microphone* (9.4%), *Device & App History*[1] (15.6%) and *Camera* (56.3%) were neither discussed in the privacy policy nor obvious in their intentions. Although, each were requested by at least three apps, none of the apps mentioned the permission group in the privacy policy.

5 SECURITY ANALYSIS

Security is a critical requirement any time money is involved. Digital financial services offer consumers greater physical security than borrowing, saving or transacting in cash; this increased security greatly contributes towards improving the livelihood of lower

[1]This permission can allow an application to gain access to sensitive logs, learn the identity of the other applications running on the phone and read web bookmarks.

income people. However, security risks do not disappear when lending moves online. As money flows online, adversaries arise that focus their efforts on whatever vulnerabilities providers leave open. Theft and fraud are enabled when data falls into adversarial hands. Moreover, breaches to online lending services may further endanger customers because of the wider range of personal data they may collect (e.g., social networks, GPS information). As such, it is critical that these services provide strong, best-practice protections for their customers.

Given that money is potentially transferred between parties using these services and that we have demonstrated that sensitive data is also being transmitted, it is critical to characterize the transport security of online credit applications. This section provides such an analysis, and uncovers the security practices for mobile applications, affiliated websites and back-end servers supporting this industry.

5.1 Methodology for Analysis of Security

Our study sought to measure the security of the connection between customers' mobile devices and the server within the digital lender's network that process and store data. While it is possible to measure the security of many parts of an online credit system, the most critical is ensuring that connections between mobile devices and the service provider's servers are secure. Accordingly, we focused our attention here.

Measuring the security of connections between users and digital credit providers is step one in assessing the practices of this industry. If a digital credit provider fails to adequately protect data in this space, an adversary could recover potentially sensitive consumer information with very little effort. Such a breach can obviously entail financial loss. However, it is critical to note that the wide array of data collected by these services could further harm a customer. For instance, GPS data could be used to track specific individuals and target them for extortion or harm.

Note that although correctly protecting communications, as discussed in this paper, is a good first step, a positive evaluation here should not be viewed as an endorsement of all security practices of the studied companies; rather, it merely represents that this one aspect is done well. Multiple additional analyses of internal policies and controls, each of which represent additional significant efforts, remain projects for future research.

Our experiments sought to answer three specific questions about the state of secure communications use by digital credit providers. They are:

(1) Do mobile devices properly use strong encryption algorithms to protect the confidentiality and integrity of all communications? (Devices)
(2) Do mobile devices properly verify that they are communicating with the correct server? (Authentication)
(3) Are the servers configured to use strong encryption algorithms to protect the confidentiality and integrity of all communications? (Servers)

We break down our methodology with respect to each of these questions below.

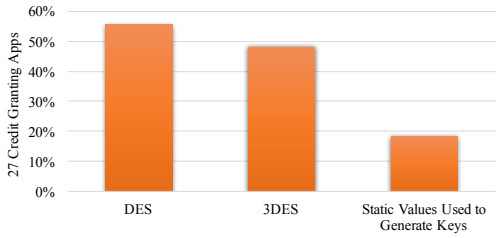

Figure 5: Applications with weak cryptographic parameters. Seventeen of the 27 applications offer demonstrably bad cipher options.

Devices: Of the 51 digital credit providers in this study, 27 offer a mobile application for Android phones.[2] We downloaded a copy of each of these applications to computers in our lab.

```
public DesEncrypter(String arg7) {
    super();
    this.salt = new byte[]{-87, -101, -56, 50, 86, 53, -29, 3};
    this.iterationCount = 19;
```

Figure 6: An example of a hardcoded salt value

We recovered over one million lines of code from the 26 mobile apps via the JEB Decompiler tool [23]. Our in-depth static analyses searched for specific categories of operations, including data encryption and password handling. In the case of encryption, we specifically searched for the use of DES and 3DES cryptographic standards, which are both outdated and now deprecated [7, 32]. The use of either of these ciphers would void any protections application designers attempted to build to protect the confidentiality of data. For password handling, we are particularly interested in the use of salts, which should be random and unique to each user; if they are not, any protection they may provide is defeated.

Authentication: We measure the strength of authentication via certificate handling in mobile applications.

We accomplish this through the use of the Mallodroid tool [8]. Mallodroid takes as input an application and reverse engineers it to produce Java code. Mallodroid then searches that code for certificate handling routines and determines if they are written in potentially dangerous ways. When the tool indicates a possible problem, we manually analyzed the routine in question and determined if a problem indeed exists, and whether that routine is called by the application (and is not dead code). While the use of the Mallodroid tool is relatively fast, it provides limited insight into the code and still requires a significant amount of evaluation time by a security engineer [25].

Servers: Even with close attention to detail for security on the mobile device, failure to provide similar attention in the backend servers can similarly expose sensitive user data. Because the software engineers writing the code for mobile phones are often different from the individuals configuring servers, it is critical to check both sides of the connection.

[2]The companies not analyzed in this section operate through webpages, rather than mobile apps. Therefore, we were unable to examine their operations without explicit access to code from the company.

```
public String getApiKey() {
    String v1 = this.settings.getString("api_key", "");
    String v2 = !TextUtils.isEmpty(((CharSequence)v1)) ? new DesEncrypter("KOPOKOPOWORLDDOMINATION").decrypt(v1) : "";
    return v2;
}
```

Figure 7: An example of an app that uses a static string to generate keys

```
private String decryptPassword(String arg9) {
    try {
        SecretKey v2 = SecretKeyFactory.getInstance("DES").generateSecret(new DESKeySpec("i-cry-when-angles-deserve-to-die".getBytes("UTF-8")));
        byte[] v1 = Base64.decode(arg9, 0);
        Cipher v0 = Cipher.getInstance("DES");
        v0.init(2, ((Key)v2));
        arg9 = new String(v0.doFinal(v1));
```

Figure 8: This company also uses a static string to generate its cryptographic key.

```
public class EncryptionHelper {
    private Context a;
    private static String b;
    private static byte[] c;
    private static EncryptionHelper d;
    private static final String e;

    static {
        EncryptionHelper.b = "AES";
        EncryptionHelper.c = new byte[]{-52, 51, -68, -121, -44, -114, -59, -20, -79, 22, 34, -77, -48, -75, 45, 93};
        EncryptionHelper.e = EncryptionHelper.class.getName();
```

Figure 9: An MNO app uses a static cryptographic key across all users. This code was taken from an open source project, meaning that this application uses the same cryptographic key as that application.

```
static {
    CipherSuite.INSTANCES = new ConcurrentHashMap();
    CipherSuite.TLS_RSA_WITH_NULL_MD5 = CipherSuite.of("SSL_RSA_WITH_NULL_MD5", 1);
    CipherSuite.TLS_RSA_WITH_NULL_SHA = CipherSuite.of("SSL_RSA_WITH_NULL_SHA", 2);
    CipherSuite.TLS_RSA_EXPORT_WITH_RC4_40_MD5 = CipherSuite.of("SSL_RSA_EXPORT_WITH_RC4_40_MD5", 3);
    CipherSuite.TLS_RSA_WITH_RC4_128_MD5 = CipherSuite.of("SSL_RSA_WITH_RC4_128_MD5", 4);
    CipherSuite.TLS_RSA_WITH_RC4_128_SHA = CipherSuite.of("SSL_RSA_WITH_RC4_128_SHA", 5);
    CipherSuite.TLS_RSA_EXPORT_WITH_DES40_CBC_SHA = CipherSuite.of("SSL_RSA_EXPORT_WITH_DES40_CBC_SHA", 8);
    CipherSuite.TLS_RSA_WITH_DES_CBC_SHA = CipherSuite.of("SSL_RSA_WITH_DES_CBC_SHA", 9);
    CipherSuite.TLS_RSA_WITH_3DES_EDE_CBC_SHA = CipherSuite.of("SSL_RSA_WITH_3DES_EDE_CBC_SHA", 10);
```

Figure 10: An app that allows for the use of export DES (40-bit keys), DES, and 3DES, all of which are considered weak ciphers

We measure the configuration of the server via the Qualys SSL Test [24]. Qualys provides a free service that attempts to connect to a target server using all possible configurations that have historically been approved for TLS/SSL connections. The output of the Qualys SSL Test is a grade level, similar to those used in traditional "report cards". Many previously allowed versions of these protocols and their parameters, while believed to be secure in the past, are now known to be insecure. Such insecurity can lead to an adversary being able to observe, modify and inject their own traffic between a user and the server. As such, performing this test helps us concretely characterize the security standing of an organization's communications.

5.2 Results

Devices: In order to find weak uses of cryptography, our reverse engineering started by first searching for instances of DES/3DES or static encryption strings. In Figure 5, we show our findings in the 27 U.S. and international apps. Most critically, 17 (63%) of the 27 apps offer demonstrably dangerous cipher options. Over half of the apps use DES, an algorithm that has been deprecated for over a decade -- well before the creation date of these apps. In many of these apps (nearly 20%), we also found disastrously incorrect cryptographic use, leading to substantial risk to consumers.

As an example of static parameters causing cryptographic problems, in Figure 6 we show where one international company's app generates a secret DES key with a static salt -- that is, one that is always the same. As mentioned previously, salts were designed to add randomness; an adversary who knows this value will have a substantially easier time recovering secret data.

This is not the only place where this weakness occurs in the app, however. Figure 7 is another snapshot of the code found in the app. As denoted by the red arrow, the same string is used as a passphrase to create a secret key. Due to this, all installations of this application will generate the same key, and any adversary with this string can likewise generate the key. Accordingly, communications from this app can be intercepted and decrypted.

Figure 8 is a snapshot of a similar use of cryptography in a U.S.-based lender's app. As shown in the figure, the static string

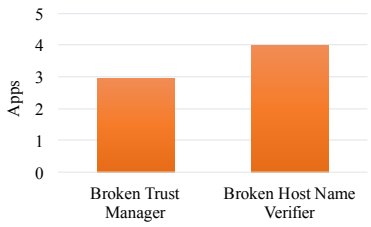

Figure 11: Mallodroid discovered four applications that failed to properly handle certificates.

"i-cry-when-angles-deserve-to-die" [3] is used to generate a secret key, causing the same vulnerability as in the previous example. Another example, an MNO app, contains the same vulnerability, as shown in Figure 9. The arrow points to the actual key used in the application. Worse still, we searched the web for the list of numbers comprising this key and found them in an unrelated, open-source project [18] . The secrecy of this key (and as a result, the security of this cryptosystem) is completely non-existent. Not only has the company failed to produce a random value for key generation, the key itself is both static and hardcoded.

In a different example from an international app, the application specifically allows both DES and 3DES, as shown in Figure 10. Allowing these deprecated algorithms can allow an adversary to recover sensitive communications through a *downgrade attack*. Such attacks are simple to configure and execute while the victim is unaware that their data is being compromised. More critically, all of the above options are demonstrably weak -- they allow both null MD5 and null SHA1 configurations, neither of which provide encryption. All eight of the ciphers have been deemed broken and hence should not be used for encryption.

Authentication: We ran Mallodroid on each of the 27 Android apps. Mallodroid detects broken trust managers and host name verifiers. These functions are critically important to ensuring the security of a TLS session, but when broken often break the security in subtle but catastrophic ways.

Trust managers accept or reject presented credentials and manage trust material in order to make trust decisions. Host Name Verifiers determine if a URL's hostname matches its respective server's hostname. If they do not match, the verifier takes necessary steps to determine if the connection should be allowed. As shown in Figure 11, three of the apps had broken trust managers and four had broken host name verifiers.

The consequences of these broken functions is severe: adversaries that can impersonate the provider's service to the app (e.g., on a coffee shop WiFi network) can intercept the sensitive communications, record, and tamper with messages between the app and the provider's legitimate service. These apps perform weak or no verification of the service and in many cases will blindly accept a connection to any server which answers. Without this verification, the user is never notified of a problem and the app appears to work normally.

[3]Misspelled lyrics to the song "Chop Suey!" by the rock band "System of a Down".

Servers: Apps often contact many different servers. Each server may be configured differently, so we first extracted all of the URLs and other server information from each app. We then characterized the range of these configurations using the Qualys SSL scanner, as described above. Figure 12 shows the best and worst Qualys score of all URLs found in the assessed apps. The results show a wide range of scores, with company 2 showing the widest range of configuration failures, from A+ to T (or "not trusted"). The lack of consistency even among the same providers is particularly concerning as it signals that the provider might not have a configuration management process, which may prevent (or alert engineers to) errors such as weak Diffie-Hellman key exchange parameters, acceptance of deprecated algorithms, and certificate not trusted errors.

We also evaluated the international company web sites from our original list of 51 companies for TLS/SSL configuration errors. As shown in Figure 13, a much higher percentage of company websites had properly configured TLS/SSL connections. However, there are also a number of important negative observations. First, note that a score of "T" was a result of an "untrusted" website, hence yielding a "failing" score. Twelve (or 24%) of the websites we measured had demonstrably vulnerable configurations (e.g., use broken encryption ciphers, are susceptible to interception, etc) and received failing scores. Seven (26%) of the 12 websites with failing scores were deemed "Not Trusted ("T")" altogether due to either an invalid certificate, invalid configuration, or unknown certificate authority. This figure includes two of the seven websites with a score of "F" that were also untrusted. The other five had significant security issues, including insecure or weak ciphersuites, and several were vulnerable to attacks including the POODLE Attack [36] (a result of using SSL3), the Heartbleed Attack [35], OpenSSL Vulnerability (CVE 2014-0224) [33], and (CVE 2016-2107) [34]. In addition, several were using deprecated versions of both SSL and TLS.

6 DISCUSSION

6.1 Notification and Impact

This research was conducted over the course of approximately 18 months. During that time, we worked carefully to ensure that all of our findings were correct, and also to ensure that the impacted parties have sufficient time to address as many of these issues as possible.

Prior to the submission of this paper, we worked closely with *anonymized for submission* to reach out to all of the companies evaluated in this paper. Through this collaboration, we were able to provide unredacted versions of our research to each of the companies, and also provided them with a solutions guide to address the most common and serious problems that we encountered. After multiple webinars and conference calls, companies have been given six weeks to address these issues before a redacted whitepaper is made publicly available.

These efforts have already demonstrated a number of measurable improvements in the security standings and processes of these and other companies within the industry. *Anonymized for submission*

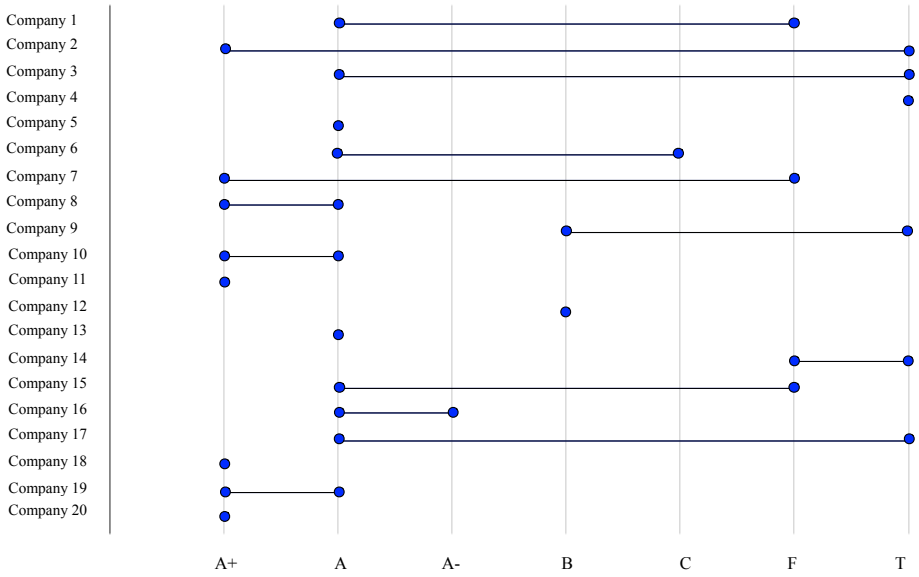

Figure 12: Best and Worst Qualys scores (all valid links found in apps)

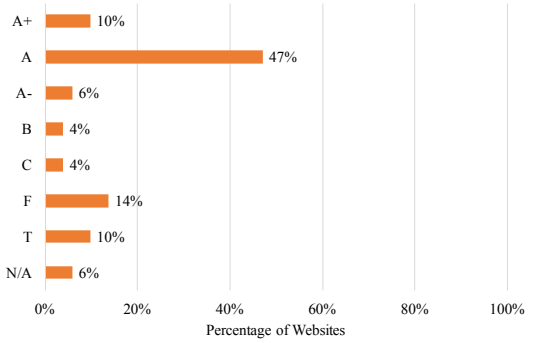

Figure 13: Qualys test results for digital credit provider websites. 12 of 51 websites received a failing grade.

has already planned its first industry-wide summit, and many companies have already agreed to make security a priority.[4] However, it is important that we make clear recommendations within this document so that future studies can determine how the industry has responded to our recommendations.

6.2 Privacy Policy Recommendations

Our analysis of the privacy of digital credit providers provided an analysis of readability and then characterized the data types collected. In general, we discovered that the policies of internationally-based digital credit providers were longer and more difficult to read; virtually all were less understandable than the privacy policies of traditional U.S.-based financial institutions. Moreover, as our technical analysis confirms, few of the policies provide many specifics regarding the types of data that will be collected or how user data will be handled.

Along those lines, companies should explain how customers can dispute incorrect claims and data. For these companies, it is important to note how a customer could possibly dispute an incorrect credit score or any information that could lead to an unfavorable credit granting decision. Due to this, we searched each policy for the word *dispute*, *credit score* and *access*. We searched for these words to determine if a policy discussed if a user had access to data collected about them, if policies mentioned collecting or sharing credit scores since it would be necessary for credit granting, and finally, if there was any mention of how customers could dispute incorrect data. The results of this keyword search showed that more than 80% of the policies mentioned the word acces or detail customer access to data. However, only 14% of the policies state that they collect or share credit scores. Even more problematic, only 10% of the companies explicitly discuss how customers can dispute incorrect data about themselves or provide information regarding disputes in general.

Policies should explicitly identify the sensitive user data they intend to collect, and how disputes can be made, so that users can make informed decisions when selecting a lender. Beyond this, the industry should strive for further clarity in the creation of these policies, including writing policies that match the reading level and language of their intended markets. This recommendation would not only provide greater clarity for users, but would also make digital lenders better at communicating privacy policies than traditional banks, furthering their value proposition as a more convenient and beneficial alternative.

[4]Should this paper be accepted, we will continue to update this section of the paper through the camera-ready deadline to provide positive examples of companies making meaningful changes.

6.3 Technical Security Recommendations

We found widespread misuse of cryptographic algorithms, including the use of algorithms that have long been publicly deprecated. We discovered that 17 of the 27 apps offered such dangerous options. Our results for authentication were better, but 4 of the 27 apps still exhibited significant vulnerabilities that would allow an adversary to impersonate the server and potentially trick clients into exposing their data. Finally, we measured server TLS/SSL configuration and demonstrated a wide range of security (even among different servers involved in a single app). At least eight of the apps had entirely vulnerable configurations that represent a significant threat to their users. Poor security configuration also impacted digital credit provider websites, with approximately one quarter also receiving a failing grade.

At the time of this paper's submission, the strongest possible relevant standards for secure communications are TLS 1.2 and TLS 1.3. Digital credit lenders should eliminate the use of all other versions as soon as is possible. Factors that may slow down such changes include legacy devices; however, a measurement study of TLS versions available to users would easily allow for a reasonable timeline to be developed. The use of TLS alone is not enough -- it must be correctly parameterized. Applications should use strong encryption ciphers and hashing algorithms (e.g., "AES_256_CBC_SHA256"). Such algorithms can be efficiently implemented in both client and server with negligible impact to performance. Support for weak ciphers (e.g., DES, 3DES) and weak hashing algorithms (e.g., MD5) must be eliminated immediately from the servers of all digital credit providers, as they provide a false sense of security. Such configuration should be uniformly applied across servers and mobile devices. Finally, digital financial applications should not rely on third party advertising or metrics libraries, some of which our analysis shows poor information security practices. All such functionality should be developed and protected by the digital credit lenders themselves.

Management should also take an active role in ensuring these technical improvements. For instance, managers could regularly request (or even perform themselves) the Qualys analysis on public facing servers. Concrete deliverables such as this ensure that management can begin to measure their organizations security standing.[5] Management should also consider engaging in discussions around issues such as patching strategies, asset management and deprecation of old technologies.

7 CONCLUSIONS

Credit can be a powerful tool to empower consumers or help build businesses. For many people, digital credit systems make it possible to access credit for the first time and for many others digital credit increases the convenience and flexibility of the credit they can use, especially data held on mobile devices. As such, it is critical to determine how data is treated and protected in such systems, and to measure the policy and infrastructure that is deployed to ensure that potential harm is minimized. Our analysis provides significant insights into the security and privacy practices of the rapidly growing online credit industry. Moreover, through our partnerships, we continue to work with vulnerable companies to dramatically increase protection of consumer data. In a world where even large and seemingly protected financial companies are being breached, such protections have never been more critical.

[5]We do not pretend that security metrics are a solved problem, or that any one measurement means an organization is secure. However, defining concrete measurements that are regularly observable is a good starting point.

8 ACKNOWLEDGMENTS

We gratefully acknowledge the generous financial support provided by the Center for Financial Inclusion at Accion, without which this work would not have been possible. We would particularly like to thank Sonja Kelly, Director of Research, and Pablo Anton Diaz, Research Manager, for not only helping us to work productively with security stakeholders around the world, but also for their tireless efforts to ensure that these issues are prioritized and addressed. This work was supported in part by the National Science Foundation under grant numbers CNS-1526718, and CNS-1540217. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation.

REFERENCES

[1] Steven Arzt, Siegfried Rasthofer, Christian Fritz, Eric Bodden, Alexandre Bartel, Jacques Klein, Yves Le Traon, Damien Octeau, and Patrick McDaniel. 2014. FlowDroid: Precise Context, Flow, Field, Object-sensitive and Lifecycle-aware Taint Analysis for Android Apps. In *Proceedings of the 35th ACM SIGPLAN Conference on Programming Language Design and Implementation (PLDI '14)*. ACM, New York, NY, USA, 259–269.

[2] Rachel Botsman. 2017. Big data meets Big Brother as China moves to rate its citizens. http://www.wired.co.uk/article/chinese-government-social-credit-score-privacy-invasion.

[3] Jasmine Bowers, Bradley Reaves, Imani N. Sherman, Patrick Traynor, and Kevin Butler. 2017. Regulators, Mount Up! Analysis of Privacy Policies for Mobile Money Services. https://www.usenix.org/conference/soups2017/technical-sessions/presentation/bowers. In *Thirteenth Symposium on Usable Privacy and Security (SOUPS 2017)*. USENIX Association, Santa Clara, CA, 97–114.

[4] Sam Castle, Fahad Pervaiz, Galen Weld, Franziska Roesner, and Richard Anderson. 2016. Let's Talk Money: Evaluating the Security Challenges of Mobile Money in the Developing World. In *7th ACM Symposium on Computing for Development (DEV)*.

[5] Central Intelligence Agency. [n. d.]. The World Factbook. https://www.cia.gov/library/publications/the-world-factbook/fields/2205.html.

[6] Lorrie Faith Cranor, Pedro Giovanni Leon, and Blase Ur. 2016. A Large-Scale Evaluation of U.S. Financial Institutions' Standardized Privacy Notices. http://doi.acm.org/10.1145/2911988. *ACM Trans. Web* 10, 3, Article 17 (August 2016), 33 pages. https://doi.org/10.1145/2911988

[7] Division, Computer Security and Laboratory, Information Technology and National Institute of Standards and Technology and Commerce, U. S. Department of. [n. d.]. Update to Current Use and Deprecation of TDEA | CSRC. https://goo.gl/B73ACB.

[8] Sascha Fahl, Marian Harbach, Thomas Muders, Lars Baumgärtner, Bernd Freisleben, and Matthew Smith. 2012. Why Eve and Mallory Love Android: An Analysis of Android SSL (in)Security. In *Proceedings of the 2012 ACM Conference on Computer and Communications Security (CCS '12)*. ACM, New York, NY, USA, 50–61.

[9] Federal Trade Commmission. [n. d.]. Gramm-Leach-Bliley Act. https://www.ftc.gov/tips-advice/business-center/privacy-and-security/gramm-leach-bliley-act.

[10] Thomas A Fergus, David P Valentiner, Patrick B McGrath, Stephanie L Gier-Lonsway, and Hyun-Soo Kim. 2012. Short forms of the social interaction anxiety scale and the social phobia scale. *Journal of personality assessment* 94, 3 (2012), 310–320.

[11] Joshua Gluck, Florian Schaub, Amy Friedman, Hana Habib, Norman Sadeh, Lorrie Faith Cranor, and Yuvraj Agarwal. 2016. How Short Is Too Short? Implications of Length and Framing on the Effectiveness of Privacy Notices. https://www.usenix.org/conference/soups2016/technical-sessions/presentation/gluck. In *Twelfth Symposium on Usable Privacy and Security (SOUPS 2016)*. USENIX Association, Denver, CO, 321–340.

[12] Google. 2018. Review App Permissions thru Android 5.9 - Android. https://goo.gl/xr4gWv.

[13] Michael I. Gordon, Deokhwan Kim, Jeff Perkins, Limei Gilham, Nguyen Nguyen, and Martin Rinard. 2015. Information Flow Analysis of Android Applications in

DroidSafe. In *Proceedings of the ISOC Network and Distributed Systems Symposium (NDSS)*.

[14] Andrew Harris, Seymour Goodman, and Patrick Traynor. 2012-2013. Privacy and Security Concerns Associated with Mobile Money Applications in Africa Mobile Money Symposium 2013. *Washington Journal of Law, Technology & Arts* 8 (2012-2013), 245–264.

[15] Mark Hochhauser. 2001. Lost in the Fine Print: Readability of Financial Privacy Notices. https://www.privacyrights.org/blog/lost-fine-print-readability-financial-privacy-notices-hochhauser.

[16] Carlos Jensen and Colin Potts. 2004. Privacy Policies As Decision-making Tools: An Evaluation of Online Privacy Notices. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*.

[17] Maritza Johnson, John Karat, Clare-Marie Karat, and Keith Grueneberg. 2010. Optimizing a Policy Authoring Framework for Security and Privacy Policies. In *Proceedings of the Symposium on Usable Privacy and Security (SOUPS)*.

[18] Juspay Technologies. [n. d.]. Github: ec-android-demo. https://github.com/juspay/ec-android-demo.

[19] Patrick Gage Kelley, Joanna Bresee, Lorrie Faith Cranor, and Robert W. Reeder. 2009. A "Nutrition Label" for Privacy. In *Proceedings of the Symposium on Usable Privacy and Security (SOUPS)*.

[20] Ethan Loufield, Dennis Ferenzy, and Tess Johnson. 2018. Accelerating Financial Inclusion with New Data. *Mainstreaming Financial Inclusion: Best Practices* (may 2018).

[21] Mary Brophy Marcus. 2016. Fitbit Fitness Tracker Detects Woman's Pregnancy. https://www.cbsnews.com/news/fitbit-fitness-tracker-tells-woman-shes-pregnant/.

[22] Damien Octeau, Patrick McDaniel, Somesh Jha, Alexandre Bartel, Eric Bodden, Jacques Klein, and Yves Le Traon. 2013. Effective Inter-component Communication Mapping in Android with EPIC: An Essential Step Towards Holistic Security Analysis. In *Proceedings of the USENIX Security Symposium*.

[23] PNF Software. [n. d.]. JEB by PNF Software: Reverse Engineering for Professionals. https://www.pnfsoftware.com/.

[24] Qualys SSL Labs. [n. d.]. SSL Server Test. https://www.ssllabs.com/ssltest/.

[25] Bradley Reaves, Jasmine Bowers, Sigmund Albert Gorski III, Olabode Anise, Rahul Bobhate, Raymond Cho, Hiranava Das, Sharique Hussain, Hamza Karachiwala, Nolen Scaife, Byron Wright, Kevin Butler, William Enck, and Patrick Traynor. 2016. *Droid: Assessment and Evaluation of Android Application Analysis Tools. *ACM Comput. Surv.* 49, 3, Article 55 (Oct. 2016), 30 pages. https://doi.org/10.1145/2996358

[26] Bradley Reaves, Jasmine Bowers, Nolen Scaife, Adam Bates, Arnav Bhartiya, Patrick Traynor, and Kevin R. B. Butler. 2017. Mo(Bile) Money, Mo(Bile) Problems: Analysis of Branchless Banking Applications. *ACM Trans. Priv. Secur.* 20, 3, Article 11 (Aug. 2017), 31 pages. https://doi.org/10.1145/3092368

[27] Bradley Reaves, Nolen Scaife, Adam Bates, Patrick Traynor, and Kevin R.B. Butler. 2015. Mo(bile) Money, Mo(bile) Problems: Analysis of Branchless Banking Applications in the Developing World. In *24th USENIX Security Symposium (USENIX Security 15)*. USENIX Association, Washington, D.C., 17–32.

[28] Robert W. Reeder, Clare-Marie Karat, John Karat, and Carolyn Brodie. 2007. Usability Challenges in Security and Privacy Policy-authoring Interfaces. In *Proceedings of the International Conference on Human-computer Interaction*.

[29] Joel R. Reidenberg, N. Cameron Russell, and Thomas B. Norton. 2016. Rating Indicator Criteria for Privacy Policies. In *Twelfth Symposium on Usable Privacy and Security (SOUPS 2016)*. USENIX Association, Denver, CO. https://www.usenix.org/conference/soups2016/workshop-program/wpi/presentation/reidenberg

[30] Florian Schaub, Rebecca Balebako, Adam L. Durity, and Lorrie Faith Cranor. 2015. A Design Space for Effective Privacy Notices. In *Symposium On Usable Privacy and Security (SOUPS)*.

[31] Xinguang Sheng and Lorrie Faith Cranor. 2005. Evaluation of the Effect of US Financial Privacy Legislation Through the Analysis of Privacy Policies. *ISJLP* 2 (2005), 943.

[32] Gayle Swenson. 2005. NIST Withdraws Outdated Data Encryption Standard. https://www.nist.gov/news-events/news/2005/06/nist-withdraws-outdated-data-encryption-standard.

[33] U.S. Computer Emergency Readiness Team. [n. d.]. CVE-2014-0224 Detail. https://nvd.nist.gov/vuln/detail/CVE-2014-0224.

[34] U.S. Computer Emergency Readiness Team. [n. d.]. CVE-2016-2107 Detail. https://nvd.nist.gov/vuln/detail/CVE-2016-2107.

[35] U.S. Computer Emergency Readiness Team. [n. d.]. OpenSSL 'Heartbleed' Vulnerability (CVE-2014-0160). https://www.us-cert.gov/ncas/alerts/TA14-098A.

[36] U.S. Computer Emergency Readiness Team. [n. d.]. SSL 3.0 Protocol Vulnerability and POODLE Attack. https://www.us-cert.gov/ncas/alerts/TA14-290A.

[37] U.S. Government Publishing Office. [n. d.]. Public Law 106 - 102 - Gramm-Leach-Bliley Act. https://www.gpo.gov/fdsys/pkg/PLAW-106publ102/content-detail.html.

[38] Fengguo Wei, Sankardas Roy, Xinming Ou, and Robby. 2014. Amandroid: A Precise and General Inter-component Data Flow Analysis Framework for Security Vetting of Android Apps. In *Proceedings of the ACM Conference on Computer and Communications Security (CCS)*.

[39] Sebastian Zimmeck and Steven M. Bellovin. 2014. Privee: An Architecture for Automatically Analyzing Web Privacy Policies. https://www.usenix.org/conference/usenixsecurity14/technical-sessions/presentation/zimmeck. In *23rd USENIX Security Symposium (USENIX Security 14)*. USENIX Association, San Diego, CA, 1–16.