DIGITAL FINANCIAL SERVICES
RESEARCH GROUP

RESEARCH BRIEF

September 2016

Let’s Talk Money: Evaluating the Security Challenges of Mobile Money in the Developing World

Sam Castle, Fahad Pervaiz, Galen Weld, Franziska Roesner, and Richard Anderson

DIGITAL FINANCIAL SERVICES
RESEARCH GROUP

This research brief provides an overview of our recently published paper “Let’s Talk Money: Evaluating the Security Challenges of Mobile Money in the Developing World.” The full paper was published in the 2016 ACM Symposium on Computing and Development (DEV) and is available for download on our website.[1]

1. Introduction

Digital money drives modern economies, and the global adoption of mobile phones has enabled a wide range of digital financial services in the developing world. Where there is money, there must be security, yet prior work on mobile money has identified discouraging vulnerabilities in the current ecosystem.

The situation, however, may not be as dire as it seems—many issues can be resolved by security best practices and updated mobile software. Using a systematic threat model to frame our assessment, we have diagnosed the problems from two directions: (1) a large-scale analysis of existing digital financial service products and (2) a series of interviews with seven developers and designers in Nigeria, Kenya, Uganda, Zimbabwe, and Colombia.

We’ve concluded that although attack vectors are present in many apps, solutions are feasible and service providers are generally making intentional, security-conscious design decisions.

2. DFS and Security

Digital financial services (DFS) constitute a rapidly growing industry that provides access to formal financial instruments through mobile technology. These services operate on existing mobile networks and are managed by mobile network operators, banks, and third-party software companies. One critical aspect is that these services allow individuals with no formal financial history to establish an account, often without needing to travel to a physical bank location.

The basic services offered by DFS applications include monetary deposits, withdrawals, and person-to-person transactions. Many include additional value-added services, such as government-to-person payments, loans, and payments for goods and utilities. Deposits and withdrawals are almost always handled by a network of agents, typically shop owners or community members, who are employed by the service operator. They accept cash deposits in exchange for digital currency transferred to the user’s account, and they keep stores of cash for users to withdraw their digital balances.

Strong security and privacy measures are critical to expanding DFS products to the world’s poor and unbanked. For a person who has been living their entire life with structured financial institutions, falling victim to a security failure may permanently divert them from that particular bank or service. For a person with no former structured financial experience, running into a security failure may deter them

[1] http://dfs.cs.washington.edu/resources.html

from formal financial systems as a whole. For these reasons, we consider security and privacy issues to be especially paramount in the context of the developing world.

3. Existing Security Concerns

Prior work has examined the security features and vulnerabilities in existing Android applications for DFS in the developing world. This work highlighted 4 broad areas of concern: SSL/TLS certificate verification, non-standard cryptography, access control, and information leakage.

1. SSL/TLS is the standard cryptography used in the Internet—it is the "HTTPS" security. This both encrypts traffic and verifies that the user is sending data to the correct recipient. Methods for implementing this are straightforward, but when done improperly, a large range of vulnerabilities emerge. Adversaries can potentially read all data from the user, send fake data to either party, or secretly pose as the service provider. Many apps incorrectly use SSL/TLS or neglect to use it altogether.

2. Non-standard cryptography: Rather than adhering to the standard SSL/TLS protocol, it is possible to create custom protocols from scratch. Doing so is extremely challenging, as the smallest detail can compromise the entire system, and prior work has shown that time and again, non-standard approaches fail to provide adequate security. Past research discovered that applications are attempting this approach, without success.

3. Access Control: The way a user logs in to an account is important in several ways, including registration requirements, the login method (such as PIN, password, or biometrics), the way user information is transmitted over the network, and account recovery processes. On the user-facing side, PINs are a common authentication mechanism, yet they can be easy to guess, especially when services start users with the default PIN of '1234' (yes, this really happens). On the back end, some apps naively store user credentials on the device or transmit user credentials over unencrypted channels.

4. Information leakage: Applications often record user details, from usernames and email addresses to transaction history and passwords, on the user's device. It may be possible for other apps on the device to read this information---a clear and dangerous violation of user privacy. The severity of this issue is dramatically reduced by using up-to-date mobile phone software.

4. Security Threat Model

Before assessing security vulnerabilities, it is essential to understand the range of possible attacks and any potential adversaries. This process, known as threat modeling, is common within the computer security community. Having an understanding of potential security threats helps developers choose which security features to implement and allows researchers and quality assurance teams to ground their security analyses in reality.

CIA Triad

Computer security goals are often modeled around the "CIA" triad: confidentiality, integrity, and availability. Confidentiality is akin to privacy, meaning the protection of any sensitive or identifying information about a customer, such as biometric data, account balance, and passwords. Integrity refers to the accuracy and trustworthiness of data—if a user initiates a transaction to pay a merchant, it is important that the correct amount of money is sent to the correct recipient. Availability is the need for access to services within a reasonable time frame; for example, a customer's money should be available for withdrawal when they need it.

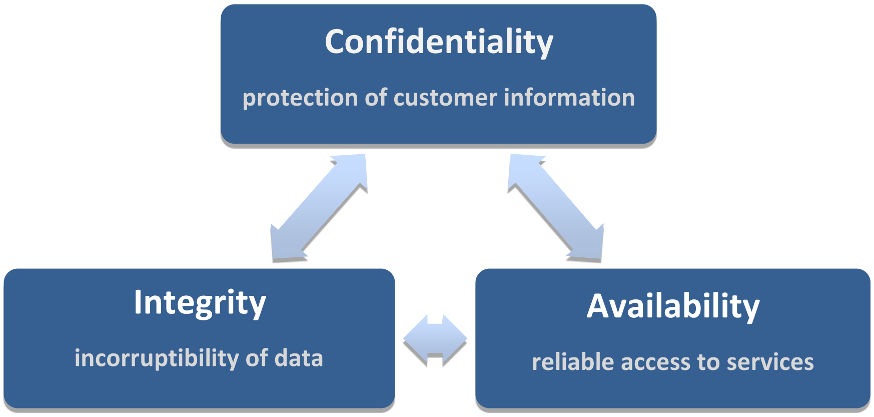
Confidentiality
protection of customer information
Integrity
incorruptibility of data
Availability
reliable access to services

Figure 1: Security goals represented by the CIA triad.

Potential Adversaries:

The following list summarizes any actors who may be motivated to compromise any of the security goals described above.

- *Customers* or other company outsiders may attempt to steal money or information from unsuspecting users, or they may steal money and services from the organization itself. For attacks on users, *friends and family members* pose unique security challenges because they may have access to personal information, such as the user's password, device, or account recovery details.
- *Agents* act as intermediaries between users and their accounts, and agents can abuse this position in ways such as charging extra fees or secretly depositing money into their own accounts.
- *Organization employees*, beyond agents, may have access to confidential information, such as user account credentials or lingering security vulnerabilities. A rogue employee may use this knowledge to steal money or sell sensitive information to other adversaries.

With the CIA triad and the list of potential adversaries, we can use the following threat model to summarize concrete possible attacks by which potential adversaries may try to compromise these security goals.

	Attack Name	*Description*
Confidentiality	**External Apps**	In early versions of Android, apps can read private data stored by other apps on the same device.
	External Libraries	Developers often include 3rd party libraries in their applications for social media, advertisements, cryptography, etc. Such libraries can introduce unintentional vulnerabilities or actively-malicious code, and researchers have found that legitimate libraries may be duplicated and repackaged with malware. Advertising and analytics libraries are also known to track user data.
	SMS Intercept	When apps communicate sensitive data via SMS, adversaries can intercept the SMS to learn private information about an individual or to take control of a user's account activity. SMS communications have known vulnerabilities, and Android has known issues in communications between apps.
Integrity	**Server Attack**	An adversary gains unauthorized access to the service's server. This includes complete control of the server as well as gaining access to partial server logs, database information, or proprietary source code.
	Man-in-the-Middle (MITM)	An adversary is able to intercept network traffic between the client and server. This allows the adversary to observe any transmitted information and to also send fake data to either party.
	Authentication Attack	There are many ways in which an adversary can gain unauthorized access to a user's account. These attacks are facilitated by services with unlimited login attempts, weak password reset procedures, and accounts where the user ID is the phone number, which is often considered public information.
	SMS Spoof	This fraud occurs when a service uses SMS to communicate with users. When used for receipts, one user can send an SMS to another user to "confirm" a transaction, when in fact no money has been transferred. Fraudsters may also be able to pose as the organization to engineer phishing attacks.
	Agent-driven Fraud	Many people, due to illiteracy or general fear of making a mistake, trust agents to process transactions on their behalf. This enables a variety of attacks from agents targeting customers, such as stealing money intended for deposits (SMS spoofing) or charging additional, unlawful fees. Customers can also defraud agents with counterfeit currency or physical force.
	Fake Accounts	If new accounts are easy to obtain, fraudsters will have more opportunities to create disposable accounts for scams. Attacks which rely on fake accounts can be mitigated by strict ID requirements for account setup and a system for reporting and disabling fraudulent accounts.
Availability	**Data Loss**	Rather than gaining access to sensitive information, the adversary destroys or corrupts business data. This may range from a complete database wipe to erasing the data of a single user.
	Denial-of-Service (DoS)	Targeting a service's connection to the server with useless traffic can block actual traffic from reaching the server. This is an attack on both the service provider and the customer—the organization loses revenue and reputation, and the customer cannot access their account.
	Theft of Services	Customers can target organizations by gaining free use of services that would normally require payment. For example, apps may include zero-rated URLs, which can be extracted by tech-savvy individuals to bypass paywalls and browse on the web without paying for a service bundle.
	Device Theft	If an adversary is able to steal a user's physical device, then the thief may be able to gain access to funds or private information. Some services bind accounts to a user's device in order to bypass password login procedures, which would allow adversaries to easily compromise accounts on lost or stolen devices.

Table 1: Possible Attacks

5. Product Analysis

We studied 197 Android apps to search for known indicators of potential security concerns, including the presence of HTTP web URLs (instead of HTTPS), unnecessary device permissions, and the use of advertising and tracking libraries. The use of such techniques is widespread---for example, a significant number of apps request permission to use the phone's flashlight for unknown reasons---yet this only indicates the potential for vulnerabilities. Understanding the extent of security exploits in practice requires additional research.

We conducted an in-depth manual analysis of product websites for 71 services, including both Android and USSD-only services. This analysis covered aspects of human-factor design, including the use of SMS for communications, procedures for resetting passwords, identification requirements for account registration (KYC), and the ability to send funds to non-users of the service. Overall, services appear to make conscious, intentional design decisions, but some practices stand out as disconcerting. For example, several services reset the user's PIN to "1234" after a simple phone call to verify relatively minor information, such as the account-holder's current address. It is not difficult to imagine a scenario where a malicious actor could take advantage of this process to engineer a targeted attack.

6. What Developers Say

We conducted semi-structured interviews with seven application developers and designers in Nigeria, Kenya, Uganda, Zimbabwe, and Colombia. Interviews discussed developer experience, organizational structure, organizational training and resources, and security processes.

As a general impression, the developers are technically competent and operate within highly-structured organizations. Based on the existence of security vulnerabilities throughout the current app ecosystem, one possible conclusion is to cast blame on the developers. Through discussions, we learned that the situation is markedly more nuanced. Institutional factors, such as outdated regulations or differing priorities among partner organizations, may lead to security risks just as easily as developer negligence. Ultimately, the most critical factor is that organizations, which are operating inside complex, ever-changing landscapes, have the ability to make informed and intentional design decisions.

The developers we interviewed noted the presence of dedicated security teams within their organizations, but requisite qualifications for such teams were unclear and inconsistent. There is a notable lack of standardization for security experts. One counterexample is the security certification issued by the Kenya Bankers Association. Adopting similar standards elsewhere would likely improve the consistency in the market.

The developers displayed technical competence, yet there is an apparent lack of resources describing best practices in DFS, both in technical computer programming and in high-level security design decisions. This leads developers to online forums, which are known to be unreliable. This also leads designers to experiment, make mistakes, and iteratively improve their product.

7. Conclusion

Prior research along with our own large-scale analysis identified numerous potential security risks in apps. Solutions, however, are feasible and within reach, aided by the presence of established organizations. Recommended improvements with the most potential for immediate impact include:

- Mandating updated Android versions, which are currently supported by over 95% of phones worldwide.
- Correct implementation of SSL/TLS.
- Standardizing security qualifications through national-level or market-level certifications.
- Providing resources for DFS security best practices to replace unreliable online forums.
- Building security validation tools in addition to existing security implementation tools.

Tools and known best practices exist, but they may be ignored due to complex institutional factors or a lack of available documentation. Rather than focusing on building new tools and expecting developers to use them, it is essential to learn the real issues, whatever they may be.

For a more in-depth analysis, read the full paper online at http://dfs.cs.washington.edu/resources.html.