On App-based Matrix Code Authentication in Online Banking

Vincent Haupert and Tilo Müller
Department of Computer Science, Friedrich-Alexander University Erlangen-Nürnberg (FAU), Erlangen, Germany

Keywords: Mobile Banking, App-based Authentication, Malware, PSD2, Compliance.

Abstract: Owing to their growing popularity, smartphones have made two-step authentication schemes not only accessible to everybody but also inexpensive for both the provider and the end user. Although app-based two-factor methods provide an additional element of authentication, they pose a risk if they are used as a replacement for an authentication system that is already secured by two-factor authentication. This particularly affects digital banking. Unlike methods backed by dedicated hardware to securely legitimize transactions, authentication apps run on multi-purpose devices such as smartphones and tablets, and are thus exposed to the threat of malware. This vulnerability becomes particularly damaging if the online banking app and the authentication app are both running on the same device. In order to emphasize the risks that single-device mobile banking poses, we show a transaction manipulation attack on the app-based authentication schemes of Deutsche Bank, Commerzbank, and Norisbank. Furthermore, we evaluate whether the matrix code authentication method that these banks and Comdirect implement — widely known as photoTAN — is compliant with the upcoming *Revised Payment Service Directive* (PSD2) of the European Banking Authority (EBA).

1 INTRODUCTION

Online banking has become an essential service that virtually every bank offers to its customers and that enjoys wide popularity. In a 2016 representative survey, Bitkom Research (Bitkom e.V., 2016) revealed that 70% of German internet users access their bank's online banking service to check their account balance and to initiate transactions. Multiple surveys indicate, however, that the digital banking activities currently shift towards mobile banking. The annual international report conducted by Bain & Company (Company, 2016) concludes that mobile banking gains traction and that the mobile interactions in some European countries — for example Sweden, the Netherlands, Italy and Spain — already exceeded the interactions using classic online banking. Similar results a survey on behalf of ING (ING, 2016) yields as "the share of mobile device users in Europe who bank by mobile has grown to 47%" and is expected to outrun the usage of traditional online banking in 2017. Furthermore, all studies emphasize that the popularity of local branch banking has declined due to younger customers opting for the increasing convenience of banking services through their smartphone.

The change in people's way of accessing their bank accounts and financial services, that is through their mobile device instead of visiting the bank's local branch, even has led to the emergence of new financial institutions (commonly known as *FinTechs*) like the pan-European banking startup N26 (Number26 GmbH, 2016). By now, every major bank offers a mobile banking application for customers to check their accounts, initiate transactions, and confirm them. The unabated success of smartphones caused many financial institutions to pursue a "mobile first" strategy: Unlike previous authentication procedures used in online banking, recent methods aim at enabling mobile transactions on a single device (*mobile banking*). As opposed to the out-of-band authentication scheme of established procedures, mobile banking no longer requires two separate devices. Instead, the authentication elements are either implemented in two segregated apps, or integrated into a single app.

Although mobile devices are appreciated for providing cost-effective and accessible two-factor authentication as an additional layer of security, it is a matter of concern that smartphones are replacing high-end security solutions basked by dedicated hardware. This development especially affects authentication procedures in online banking. Prior to the introduction of app-based authentication methods, the evolution of the second element used for transaction verification and confirmation was characterized by a steady increase

Haupert, V. and Müller, T.
On App-based Matrix Code Authentication in Online Banking.
DOI: 10.5220/0006650501490160
In *Proceedings of the 4th International Conference on Information Systems Security and Privacy (ICISSP 2018)*, pages 149-160
ISBN: 978-989-758-282-0

in security features. In particular, chipTAN is an established procedure used in online banking. It uses the customer's personal bank card and a dedicated reader device to securely authenticate a transaction.

While it is true that the use of apps as an additional element of authentication can increase the security of systems that were not using a second factor before, it means a step backward for online banking that until lately followed the rule that transaction initialization and confirmation should never take place on the same device. Unlike many other authentication schemes, the security of online payments is subject to national and supra-national regulations. In the following we not only provide evidence that app-based authentication schemes are less secure than their predecessors, but also show that the upcoming EU regulations stipulating strong customer authentication missed the target to account for this decay in security.

1.1 Attacker Model: The Threat of Privileged Malware

The main reason why app-based authentication schemes provide less security than previous methods is that they run on a smartphone. In contrast to methods like chipTAN, a smartphone is not a dedicated but a multi-purpose device. The reason for the success of smartphones is their vast set of features and the possibility to install apps on them. While the interface and attack surface of dedicated hardware devices is tailored to be as small as possible, smartphones have various input channels. Additionally, smartphone operating systems are designed to be modifiable and extensible, making effective security a complex task. This leads to a broad attack surface that is targeted by malware.

Apart from malicious apps that compromise the security and privacy of apps within the security model of the system, malware that attempts to gain root permissions (*privileged malware*) is particularly dangerous. If malware succeeds in executing a privilege escalation exploit (*root exploit*), it gets full control of the system. While root exploits are also often used by power users to gain maximum control over their system, malware deploys them in order to bypass the system's isolation and sandboxing principles. After rooting the victim's device, malware can execute its payload within the maximum privilege level.

That this threat is no fiction and that an app containing a root exploit can actually make its way into the official Google Play Store has been proven in 2014 (Maier et al., 2014). The following year an app called *Brain Test* (Polkovnichenko and Boxiner, 2015) was detected in the Play Store that followed the predicted scenario. The Brain Test app would conceal itself as a functional IQ testing app while trying to root the user's device in the background. Afterwards, it would download a malicious code from an external server to execute it with root privileges. Shortly after the app was removed by Google, 13 similar apps — each with a different name and game logic — were detected in the Play Store (Dehghanpoor, 2016).

Then there are apps based on the malware family *Godless* (Zhang, 2016) or *HummingBad* (Check Point Mobile Research Team, 2016) that also root the device they are run on. According to TrendMicro, "Godless can target virtually any Android device running on Android 5.1 (Lollipop) or earlier", which meant 90% of all Android devices when the article was released on June 21, 2016. By the end of 2015 the HummingBad malware was detected in various apps, and it was estimated that it had already infected and rooted more than 10 million devices (Goodin, 2016a).

In August 2016 Check Point announced that it had found a set of four vulnerabilities in "Android devices sporting Qualcomm chipsets" (Donenfeld, 2016). Each of them could be used to gain root permissions on the device. The same line *Dirty Cow* takes, a Linux kernel vulnerability capable of rooting any Android version (Goodin, 2016b). It is merely a matter of time until malware makes use of these vulnerabilities. Owing to the great diversity of device manufacturers that usually ship their own, often modified, versions of Android, there is no centralized update mechanism. Therefore, every manufacturer is responsible for rolling out software and security updates. The cruel reality is that many of these companies take significantly long to release security patches, if they deliver them to the end user at all (Thomas et al., 2015).

Interestingly, the way of monetarization that malware based on Brain Test, Godless, or HummingBad often uses today is displaying advertisements to the user. The rapid spread of mobile banking, however, is set to give rise to mobile banking trojan campaigns. For example, Kaspersky recently reported that the banking malware *Tordow* has significantly evolved (Kivva, 2016). The fact that Tordow roots a user's device will allow "cybercriminals to carry out new types of attacks".

1.2 Related Work

The most important past research in the field of mobile banking security is our 2015 analysis of the Sparkasse pushTAN authentication procedure (Haupert and Müller, 2016). Apart from a transaction manipulation attack, we also mention that an attack which aims at replicating the pushTAN app might be feasible but refrained from executing it. Their first mentioned tran-

saction manipulation attack, however, was realized using the *Xposed* hooking framework on a *SuperSU*-rooted, i. e., heavily prepared device. In their official statement, Sparkasse picked these circumstances up to discount the attack to be only doable under laboratory conditions (Deutscher Sparkassen- und Giroverband, 2015). We also had demanded to fill the regulatory gap that mobile banking authentication schemes were taking advantage of and that is now covered by the *Revised Payment Service Directive* of the European Union. The Regulatory Technical Standards, however, were still in preperation.

Another contribution (Dmitrienko et al., 2014) also deals with the security of the CrontoSign / photoTAN procedure. They already showed in 2014 that an early demo version of the app could not withstand a copy attack. In contrast to recent versions used in the field, however, the procedure did neither yet implement any device binding to mitigate copy attacks nor was it possible to operate it on the same device used to initiate the corresponding transaction. Furthermore, the photoTAN procedure at this time could only be used with two different devices. This, however, has changed in the meantime and single-device transactions form the core of our criticism.

Further research on the security impact of mobile devices with respect to authentication procedures has been conducted in the research paper *How Anywhere Computing Just Killed Your Phone-Based Two-Factor Authentication* (Konoth et al., 2016). They show that the heavy synchronization between a user's devices can invalidate the additional protection of a two-factor authentication scheme by, e.g., also synchronizing a token received via SMS with a user's computer, hence eliminating the separation of channels. In general, the SMS technology — still widely used for authenticating online banking transactions — is well-researched and several issues have been revealed (Mulliner et al., 2013; Reaves et al., 2016; Rao et al., 2016).

1.3 Contributions

In our scenario the victim uses an app-based matrix code authentication scheme for online banking on his or her Android device. The system on the device does not have to be modified; in particular, the device does not have to be rooted. However, a device weakness must be known to gain full access to the system, which is exploited by criminals as described above. In case the device runs both the online banking app and the authentication app, there are no additional requirements. This is also true for authentication schemes that carry out transaction initialization and confirmation within a single app. If the app-based authentication scheme forces another device to initiate a transaction, the knowledge-based authentication factor must additionally be compromised. Based on these assumptions, we make the following contributions:

- First, we show a real-time transaction manipulation attack for the mobile banking use case of the photoTAN procedure of Deutsche Bank, Commerzbank, and Norisbank. The transaction manipulation remains invisible to the victim in all steps of the attack and is entirely technical, meaning that it does not involve social engineering.
- Second, we analyze the compliance of the photoTAN procedure with respect to the forthcoming *Revised Payment Service Directive* of the European banking authority. Although a previous draft suggested that running two authentication factors on the same device may lead to violation of the requirement of the independence of authentication factors, the final draft presumably allows procedures implemented in this way. Furthermore, the photoTAN app alone may not be considered a possession element within the definition of strong customer authorization. In support of this argument, we show that the photoTAN app of Deutsche Bank, Commerzbank, Norisbank, and Comdirect cannot withstand a replication attack.

2 APP-BASED AUTHENTICATION IN DIGITAL BANKING

Particularly in German speaking countries, the second factor procedure used to confirm transactions is called *TAN method*. TAN stands for *transaction authentication number* and is a one-time password (OTP) that is received, processed or even generated by the TAN method after a customer issued a credit transfer. The user afterwards transmits the TAN manually or automatically to the bank's backend causing the transaction to become into effect. Even though modern second-factor authentication schemes do frequently no longer involve a TAN, the expression *TAN method* remained due to historic reasons and denotes a procedure to confirm digital banking transactions.

Although the development of high-end TAN procedures could successfully defeat most threats in online banking, they were expensive either for the bank or for the user. The chipTAN procedure, for example, introduced a dedicated reader to generate TANs in conjunction with the customer's personal bank card. In the past banks supplied their customers with the device free of charge. Today, however, it is common

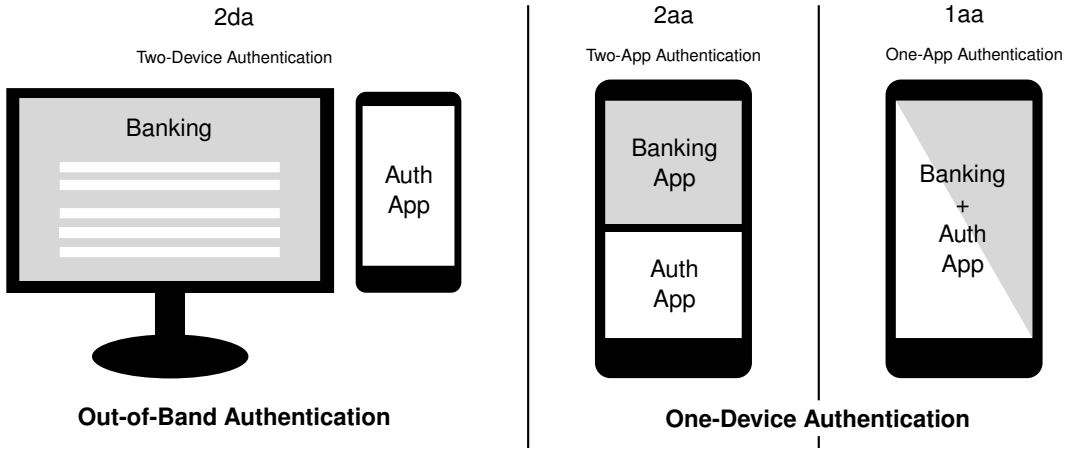

Figure 1: The different types of app-based authentication methods. The two-device authentication (2da) scheme, which makes use of two independent devices, is suitable for out-of-band authentication of transactions. Two-app authentication (2aa) and one-app authentication (1aa) were developed in order to enable mobile banking on one device. While 2da and 2aa use two different apps for authentication, 1aa issues and confirms transactions only within a single app.

practice to pass the acquisition costs to the end user. In contrast, smartphone-based authentication schemes allow both the financial institution and the end user to reduce costs. Banking apps are usually freely available to the customers, guaranteeing high acceptance rates.

In contrast to previous authentication methods, app-based authentication can be divided into three categories: *two-device*, *two-app*, and *one-app* authentication schemes, as shown in Section 2. All three types of apps display transaction details to the user in order to get a second confirmation of the transaction. Only if the user confirms the transaction via a second channel does the transaction come into effect. In the following each type of app-based authentication is described:

Two-Device Authentication (2da). This mode of authentication is largely similar to established methods like mTAN and chipTAN, as it is a true two-factor authentication scheme using two independent devices. First, the user logs into the banking app or web interface to issue a transaction order. Second, the user uses an independent device to confirm the transaction. The delivery of transaction details to the authentication app differs across vendors, but it is dependent on whether the method is an online or offline procedure. An offline procedure obtains the transaction details through an input different from the network channel. The user often has to scan a matrix barcode using the smartphone camera. The authentication app then extracts the transaction details from the obtained image and displays them to the user. As the procedure takes place offline, a TAN is displayed after the transaction has been confirmed, and the user has to manually transfer the TAN to the banking app or web interface.

Two-App Authentication (2aa). In contrast to 2da, the 2aa method does not rely on two different devices but two different apps running on the same mobile device. To initiate a transaction the user opens the banking app and enters his or her login credentials. After sending a transfer order, the banking app opens the authentication app. Depending on whether the authentication app works online or offline, the banking app sends the transaction details to the authentication app based on app-to-app communication, or the authentication app receives the transfer data over the network from the banking server. Likewise, when the user confirms the transaction, the authentication app either sends the TAN via app-to-app communication, or directly confirms the transaction over the network.

One-App Authentication (1aa). As the name suggests, this method does both transaction initialization and confirmation not only on the same device but also inside the same app. When a customer uses this app to issue a transaction, he or she is no longer required to use a different app. Instead, the app shows the confirmation dialog right after the transaction submission. In contrast to 2da and 2aa, this method only displays the transaction details but never shows a TAN to the user as that would not add value to the procedure.

3 THE photoTAN METHOD

The photoTAN procedure is a TAN method which is based on CrontoSign, a visual signing technology developed by Cronto (Cronto, 2011). In 2008, Commerzbank was the first bank that experimented with Cronto's technology, using it as a secure, cost-effective, and usable second-factor authorization met-

hod (Cronto, 2008). To bring it in line with the already existing naming scheme for previous methods, it was labeled photoTAN.

The photoTAN method is a popular app-based authentication procedure based on matrix code scanning. Even though mostly German and Swiss banks have adopted the photoTAN method, it is also used internationally, presumably under different names. We have chosen Deutsche Bank and Commerzbank, along with their direct banking subsidiaries Norisbank and Comdirect, because they all play a significant role in the German banking landscape. With respect to their balance sheet total (Bundesverband deutscher Banken e.V., 2015), Deutsche Bank and Commerzbank are Germany's largest banks, while Norisbank and Comdirect are popular direct banks.

3.1 Order and Activation

In order to use the photoTAN procedure to legitimize transactions, the photoTAN app of the respective bank and an activation graphic is needed to initialize the app. While one might download and install photoTAN immediately, all analyzed banks send the graphic printed on a postal letter. As such, the delivery takes at least one or two days. After receiving the activation letter and installing the app, one can begin the activation process.

First, the user has to scan the activation graphic found on the bank's postal letter. Afterward, the customer is asked to log into the online banking app to add a new photoTAN device through the TAN administration web page. Depending on the bank, the remainder of the procedure continues differently. If using photoTAN by Deutsche Bank or Norisbank, the last step generated a 12-digit numerical token, and the photoTAN app prompts the user to enter and send it through the online banking. Thereafter, the online banking app shows another photoTAN graphic which the user must scan to generate a 7-digit TAN to transfer it to the online banking and complete the activation process. In the case of Commerzbank and Comdirect, the process is slightly different: Instead of showing a token to the user after scanning the activation graphic, the online banking asks the user to scan another photoTAN graphic right away. This generates a 7-digit TAN just like in the last step of the activation procedure of Deutsche Bank and Norisbank. In all cases, the photoTAN app is assigned a unique identifier that consists of five uppercase alphabetic letters. Even though multiple devices might be registered with the same activation letter, they do not generate the same TAN. Therefore, each transaction can be confirmed by multiple TANs. It is also noteworthy that registering an additional device does not require the confirmation of an already activated device.

The photoTAN procedure is a strict offline method. By implication, the app cannot send any data over the network back to the bank's server. The only possibility to transfer any device information is by coding the information inside the activation code of the TAN. Apparently, only Deutsche Bank and Norisbank receive data from the photoTAN app as neither Commerzbank nor Comdirect use an activation code at all. Although ultimately unknown, as a deep analysis of the protocol was out of scope, it is likely that the photoTAN app of Commerzbank and Comdirect is immediately activated after scanning the activation graphic while Deutsche Bank and Norisbank transfer additional information coded inside the 12-digit number. This code offers the possibility to carry only very limited amounts of data. After this step, the process continues equally, and the scanned PhotoTAN graphic serves to confirm the activation. Due to its short length of only 7 digits, it is highly unlikely that the TAN transfers any device information back to the online banking without substantially decreasing the entropy of the actual payload. In the end, it seems most probable that Commerzbank and Comdirect do not obtain any additional information. Even though Deutsche Bank and Norisbank ascertain further device information, their activation process does not involve any seed. As a consequence, the last step in the activation always generates the same TAN if performed on the same device again. This behavior differs from Commerzbank and Comdirect, as their activation process always yields a different TAN, even if repeated on the device.

3.2 Usage and Modes of Operation

The photoTAN image is a matrix code that contains the transaction data and additional metadata to ensure its integrity. Recent versions of the photoTAN method of Deutsche Bank, Commerzbank, and Norisbank offer two different ways of receiving the payload: (1) Scanning a photoTAN image with the device camera and decoding its payload, and (2) receiving the photoTAN payload directly via app-to-app communication (mobile banking). Both modes are illustrated in Figure 2. In any case, the photoTAN app uses its cryptographic key received during the app's activation process to decrypt the payload and to generate a TAN that corresponds to the transaction details.

The first mode of operation is used for the out-of-band approach involving two devices (2da). After the customer has sent a transfer order using the transaction initiation channel — which is the banking app or web interface — , a matrix code is displayed. The

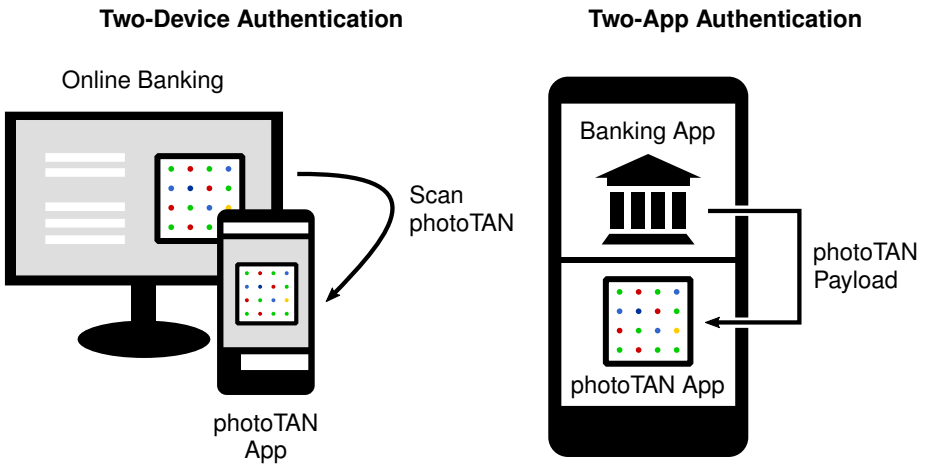

Figure 2: The two different operation modes of the photoTAN method. Either the user scans a matrix code with the device camera (2da), or the decoded photoTAN payload is directly transferred from the banking to the photoTAN app (2aa).

photoTAN app generates a TAN from the decoded transaction details of the matrix code. Finally, the user manually transfers the TAN to verify the transaction details and to finally confirm the transaction.

The second option implements the 2aa scheme to facilitate mobile banking transactions. The customer uses the banking app to fill in the transaction details and send them to the banking server. As a mobile device with an integrated camera cannot scan its own display, the banking app transfers the decoded photoTAN payload to the photoTAN app via app-to-app communication. Thereafter, the app displays the transaction details and asks the user for confirmation. But instead of transferring the TAN back manually, the photoTAN app automatically sends it to the banking app. Ironically, the term "photoTAN" entirely loses its justification when used in this mode of operation because no matrix code scanning is involved.

As of February 2017, mobile banking transactions are currently supported by Deutsche Bank, Norisbank, and recently also by Commerzbank. Comdirect only supports out-of-band photoTAN transactions. Currently, there is no photoTAN implementation that integrates both authentication steps in one app, as evident from N26.

3.3 Security Features

In this section we provide an overview of the security features of the photoTAN procedure and the different apps offered by Deutsche Bank, Commerzbank, Norisbank, and Comdirect. As some attacks also involve the respective banking apps, their security is also addressed, if necessary. The following description of security properties is summarized in Table 1.

No Access Barrier. None of the analyzed photoTAN derivates restricts the photoTAN app by explicitly authenticating the customer. On the one hand, this allows quick access to the photoTAN app, but on the other, this means there is no additional security barrier to physical access attacks for users without screen lock.

Fingerprinting. To mitigate replication attacks, all variants employ device fingerprinting to bind certain device properties to the installed and activated app. The fingerprinting step, however, only relies on the `IMEI` and the `ANDROID_ID`. In the case of the Comdirect photoTAN app, fingerprinting is solely based on the `ANDROID_ID`. Both values can be easily forged.

Repackaging Protection. Repacking is the process of decoding, modifying, and encoding an existing app. Ultimately, the app is signed with a new key. More often than not, repackaging is used to trojanize existing apps and spread them using third-party stores. Furthermore, repackaging is an important assistant for dynamic analysis and reverse engineering. To mitigate repackaging, apps check their own signature at runtime to spot modifications. Even though the photoTAN apps carry sensitive information, only Commerzbank and Comdirect have taken active measures to prevent it. These banking apps do not account for repackaging, which means none of them employs any mitigation technique.

Rooting Policy. In the Android universe, the process of gaining system privileges and installing the `su` binary to permit apps to ask for root permissions is called *rooting*. As this process could disable important security anchors and features, many apps dealing with sensitive data employ a restrictive usage policy for rooted devices. Nonetheless, only the photoTAN app of Commerzbank enforces a restrictive rooting policy, whereas Comdirect only shows a message hinting

Table 1: Overview of the security features of different Android photoTAN derivates and their corresponding banking apps. The first sub-column deals with the respective banking app while the second refers to the bank's photoTAN app.

	Deutsche Bank		Commerzbank		Norisbank		Comdirect	
Enforces Out-of-Band	☐		☐		☐		■	
Analyzed Version	2.6.0	2.1.7	4.0.1	7.1.7	2.6.0	2.1.7	2.1.5	6.0.6
Denies Backup	☐	☐	☐	■	☐	☐	☐	☐
Anti-Rooting	☐	☐	☐	■	☐	☐	☐	▨
Anti-Repackaging	☐	☐	☐	■	☐	☐	☐	■
Obfuscation	☐	▨	■	■	☐	▨	▨	▨
Fingerprinting	–	ID, IMEI	–	ID, IMEI	–	ID, IMEI	–	ID
TLS Pinning	■	–	☐	–	■	–	☐	–

at the risks involved. Deutsche Bank and Norisbank neither advise about nor restrict access to rooted devices. None of the banking apps actually checks for rooting.

Prevent Backup. Android offers the option of creating backups of apps and their data. As this feature represents a legitimate, non-root opportunity to access an app's data, it poses the risk of the data getting compromised. As this could happen due to a system feature, it can also be disabled using the `allowBackup` flag in the Android manifest. Despite the risks, only Commerzbank has disabled the option of creating backups.

TLS Pinning. While the photoTAN method operates completely offline, the respective banking apps themselves must retrieve data from the internet. It is important that all apps — especially those that receive or send sensitive information — use TLS-encrypted connections. An attacker might still use a man-in-the-middle (MITM) attack to compromise the integrity and confidentiality of a connection. To prevent this kind of attack, an app can pin a specific certificate used for network communication. Even though MITM attacks against TLS-encrypted connections have been known for years, only Deutsche Bank and Norisbank use certificate pinning to prevent MITM attacks targeting their banking apps.

Obfuscation. In order to reverse-engineer, understand, and modify the logic of a program, its code can be disassembled for static analysis. In the case of Android, apps are not delivered as machine code for the target architecture but as Java bytecode. The latter contains significantly more metadata, thereby easing the reverse engineer's analysis and allowing automatic decompilers to produce results that are close to the original source code. Obfuscation is the process of making mainly static but also dynamic analysis harder by removing or modifying metadata and introducing additional code to conceal the idea and logic behind a particular piece of code. Even though the default Android build configuration provides for ProGuard (Lafortune,), not all apps use it. ProGuard is primarily a code minifier aimed at improving performance. However, certain features, such as the function renaming employed by ProGuard, also have a significant obfuscating effect. The Deutsche Bank and Norisbank photoTAN apps, as well as the Comdirect banking and photoTAN app, make use of this. Both the Commerzbank banking and photoTAN apps are processed with tools that employ obfuscation techniques that go beyond ProGuard. The banking apps of Deutsche Bank and Norisbank are not protected at all.

Third-Party Protection. For enhanced security, third parties offer solutions to provide apps with additional safeguards. Promon Shield (Promon AS, 2016), a product designed by the Norwegian company Promon, offers protection against various threats — including most of those noted so far — without much interaction from the developers of the app. Even though research has shown that app transformations can be powerful, it is difficult for Promon Shield to provide effective protection against real attacks without causing false positives (Commerzbank A.G., 2016). The only app that uses such a solution is Commerzbank which introduced Promon Shield with a recent update of their photoTAN app.

In summary, the photoTAN method has a very permissive security model. One security decision is to not secure the app and its data using an additional login screen, even though many other app-based TAN methods like pushTAN employ such protection. As stated before, this decision does not only influence who can use the app but also prevents encryption of the user's credentials. The only safeguard that mitigates a naive replication attack is the app's device fingerprinting. The device fingerprinting involves the `IMEI` (hardware property) and the `ANDROID_ID` (software property), but both values are common device and system properties

that can easily be replicated.

Commerzbank and Comdirect have taken more measures to ensure the security of the system and the integrity of their apps. The Comdirect photoTAN app does not only warn the customer about the risks of rooting, but also checks the integrity of the app at runtime, and more precisely during registration. When the user scans the matrix code, it also checks the signature of the app. Comdirect sends the expected signature of the app along with the payload of the matrix code. Commerzbank provides protection against repackaging and uses a restrictive rooting policy, as enforced by the third-party security module developed by Promon.

Last but not least, please note that the photoTAN procedure is not only available as a smartphone app but also as dedicated hardware (Cronto, 2011). Naturally, our statements about the security features of app-based authentication cannot be transferred to the photoTAN hardware device. Quite the contrary, a dedicated photoTAN device — available for all three analyzed banks — offers excellent security properties largely similar to those of chipTAN.

4 TRANSACTION MANIPULATION ATTACK AGAINST photoTAN

This section describes a real-time transaction manipulation attack that we implemented against the photoTAN procedure of Deutsche Bank, Commerzbank, and Norisbank. The attack cannot be used against Comdirect, as Comdirect does not support mobile banking on a single device. For the other three banks, however, the attack manipulates the transaction data the victim (1) sends during initialization (banking app), (2) sees during verification (banking app), and (3) sees during confirmation (photoTAN app).

4.1 Banking App: MITM

In order to manipulate the data the user sends and sees inside the banking app, we use a TLS man-in-the-middle (MITM) attack. The Android system — just like any other operating system — ships a bundle of certificates it regards as trusted. To get the system to trust the certificate presented by the MITM proxy, the attacker needs to install it. This process is straightforward as the system regards certificates that reside in a specific system directory as trusted. Owing to the attacker's privilege level we assume in our attacker model, files can be placed in any location.

Especially applications dealing with sensitive data are developed with MITM attacks in mind. Consequently, manufacturers employ certificate pinning to protect their apps against such attacks. This technique causes the app to only trust a specific set of certificates instead of solely relying on the system's trust settings. Although certificate pinning effectively protects an app against MITM attacks, only the banking apps of Deutsche Bank and Norisbank employ this method. The banking app of Commerzbank does not pin its certificate. But also disabling the certificate pinning the Deutsche Bank and Norisbank is possible, because both use a flag that controls if the application should quit due to the detection of a certificate error or not. Therefore, our patch does not stop the apps from detecting the error but simply prevents any consequences of it. Furthermore, neither of the two apps performs any repackaging checks. Therefore, an attacker only needs to introduce a patch that toggles the flag to make the apps accept connections with the attacker server.

Besides the possibility to manipulate the data a user sends and receives, we were able to eavesdrop on the user's login credentials. The latter was true for all the analyzed banks including Comdirect.

4.2 PhotoTAN App: Repackaging

To forge the data the photoTAN app presents to the user during transaction confirmation, an attacker needs to either modify the environment the app is running in, hook particular app methods, or patch the app's code statically. We decided to modify the app statically, as this has the least impact on the system and represents the method a real attacker would most likely choose in practice. This process was straightforward for the photoTAN apps of Deutsche Bank and Norisbank because none of them protects itself against repackaging. The photoTAN app of Commerzbank and Comdirect, however, required extra work to disable their repackaging protection.

Commerzbank. The repackaging protection of Commerzbank's photoTAN app is provided by Promon Shield. The protection solution by Promon is integrated into the app and delivered in the form of a native library that loads when the app starts. As the repackaging protection is part of the native library, which itself is obfuscated and uses tamper resistance to spot modifications, it would be rather hard to patch the library. Another idea is to remove Promon Shield entirely from the app, but even though this would be theoretically possible, it is also assumed to be rather difficult because Promon strips the app of all strings and outsources them to the native library. The Java code then queries these strings at runtime using a defined index. The easiest way to disable Promon's repackaging pro-

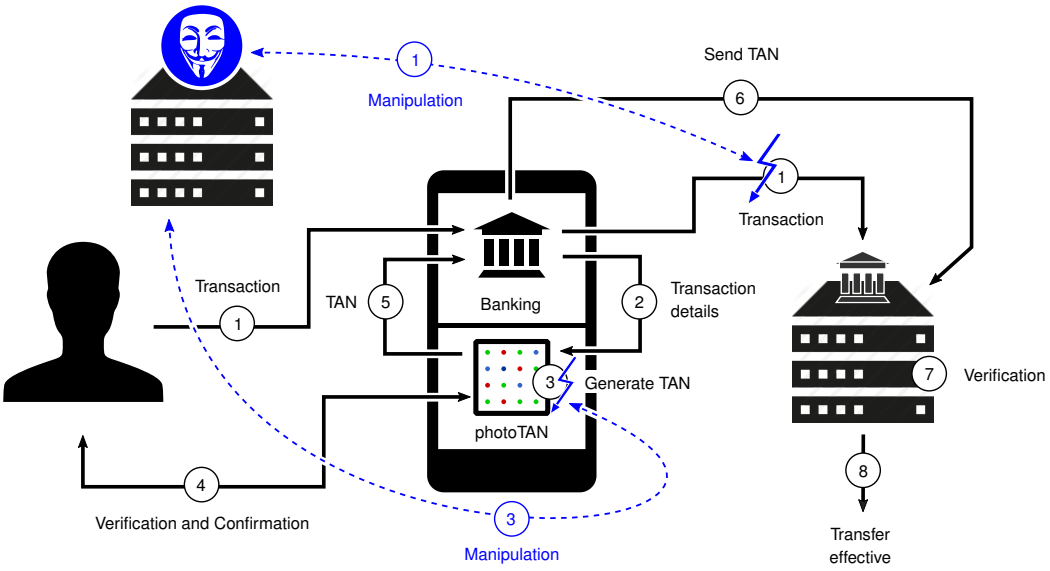

Figure 3: Overview of the steps involved in the implementation of real-time transaction manipulations.

tection is using `LD_PRELOAD`. The only hook required to disable the repackaging protection was to return a file descriptor to the original app whenever Promon tried to open the repackaged app. When the system ran our modified app, Promon Shield reads the app that had not been tampered with; hence, all security checks were passed, and the app continued to execute both its own and the attacker's code. This approach disabled the core security features of Promon in less than 100 lines of C.

Comdirect. Unlike Commerzbank, the Comdirect photoTAN app implemented its own routines to check the integrity of an app. Unlike Promon, which checks the app's signature during app startup, the Comdirect photoTAN app is not shipped with a hardcoded certificate. Instead, parts of the app's signing certificate are checked with values encoded in the matrix code. If the repackaging check cannot validate the app's signature, the app crashes without a warning. After we learned how the repackaging protection works, patching it was straightforward too: A function of the app responsible for returning the signing certificates is compared with the values found inside the matrix code. Instead of retrieving the certificates at runtime, we patched the app to statically return the certificate the original app was signed with.

After the repackaging protection was defeated, we could patch the photoTAN app to display arbitrary data. All of the analyzed banks limit the transferred data to the minimum required to verify a credit transfer, namely its IBAN and amount. As a result, we only needed to forge those two values to conceal transaction manipulations. This could be achieved by statically injecting additional code right before the IBAN and the amount are displayed to the client. The injected code searches our attacker server for details of the manipulated transaction, and so the photoTAN app eventually processes different transaction details as those the user sees during confirmation.

4.3 The Attack

With the ability to modify the transaction initialization, verification, and confirmation process, we are able to execute real-time transaction manipulations. The attack visualized in Figure 3 works as follows:

1. The victim, Bob, wants to execute a mobile banking transaction to Alice using a malware-infected Android device. To complete the transaction, Bob opens the banking app and provides the required details (beneficiary, IBAN, amount, and reference). When he submits the transaction details, however, they are not delivered to the bank's server. Instead, all requests are routed to our attacking server where we cannot only eavesdrop on the victim's credentials but also modify the transaction. First, the original transaction Bob issued is saved for later use, and then Bob's transaction is replaced with our own transaction that benefits our own bank account.
2. After submitting the transaction, the banking app prompts Bob to verify and confirm his transaction using the photoTAN app. As Bob is using the banking app and the photoTAN app on the same device, the photoTAN app opens automatically. When he starts the verification process, the banking app sends the decoded matrix challenge with our for-

ged transaction details to the photoTAN app via app-to-app communication.
3. Next, the photoTAN app displays the transaction details and generates a corresponding TAN. Bob would normally spot the fraud when the photoTAN app displays transaction details as different from his input. As the photoTAN app is attacker-controlled, however, it retrieves the transaction details that Bob originally entered from our attacking server.
4. Bob verifies the transaction, and since everything visually appears as expected, he confirms the transaction.
5. The TAN is automatically transferred back into the banking app.
6. Bob must confirm once again in order to send the TAN to the bank server.
7. The bank server checks if the TAN matches the expected TAN.
8. Finally, it passes verification, and the transaction is eventually completed.

Note that Bob confirmed a different transaction than he had actually intended without realizing it during or after the attack. As the victim's phone was compromised, we could even change the transaction overview to hide the tampering. The only way for the victim to spot the fraud was consulting the bank's online banking from an independent device.

5 LEGAL CONFORMITY WITH UPCOMING EU REGULATIONS

The security of online transactions is subject to national and supranational regulations. EU Directive 2015/2366 (European Union, 2015) was issued on January 12, 2016, and it will come into effect on January 13, 2018. Being the successor to the *Payment Service Directive* (PSD), it is better known as the *Revised Payment Service Directive*, or just PSD2. The directive will make strong customer authentication mandatory for all payment services and platforms, especially mobile devices. On February 23, 2017, the EBA has released the final draft of its regulatory technical standards (RTS) (European Banking Authority, 2017) that provide a more detailed description of the requirements of strong customer authentication (SCA). Much to our surprise, the EBA significantly relaxed the rules defined in the previous draft (European Banking Authority, 2016), thus having introduced a longer lasting weakening effect on the security requirements of European online and particularly mobile banking.

Article 4(30) PSD2 demands strong customer authentication based on two or more elements, categorized as knowledge (something only the user knows), possession (something only the user possesses), and inherence (something only the user is).

5.1 The photoTAN App as a Possession Element

Section 4 showed that it is particularly dangerous to use one-device mobile banking. Although this is a valid scenario, currently only 13% of German online banking users initiate and confirm transactions on the same device (Bitkom e.V., 2016). By implication, the vast majority of photoTAN customers of Deutsche Bank, Commerzbank, and Norisbank still use out-of-band authentication in order to legitimize their transactions with two independent devices. Even though using the photoTAN app in conjunction with another device is substantially more secure, it cannot be compared to the level of security that dedicated hardware gives.

The photoTAN procedure makes use of a knowledge element during transaction initialization — the login credentials — and a possession element during transaction confirmation — the activated photoTAN app. In the past it allowed for substantial divergences in the interpretation of the essential features of the individual factors. With respect to the possession element, the EBA RTS require it to be "designed to prevent replication of the elements".

Owing to the weak device binding of the photoTAN app, we were able to copy all the analyzed photoTAN apps from a victim's device to our attacker device. After cloning, the photoTAN app on both devices generated the same TAN for a specific transaction. The photoTAN method fails to guard against our replication attack because of two reasons. First, the photoTAN app only relies on the device's `IMEI` and the system's `ANDROID_ID`. Both of these are common values for implementing device binding and any attacker would look into them first. Second, cloning of the photoTAN app is successful because it has no access protection. If the photoTAN app were secured by knowledge or an inherence element, the app could store its data in an encrypted format.

5.2 Independence of the Elements in Mobile Banking

Apart from the description of the elements used for strong customer authentication, the EBA also specifies the requirements with respect to their independence. This is particularly relevant for single-device mobile banking transactions.

While the RTS draft found in the EBA discussion paper still suggested strong security standards that re-

quire the channel used for transaction initialization and confirmation to "be independent or segregated", the final report withdrew this demand. The same line the restatement requiring the elements to be implemented in "discrete trusted execution environments" takes: This formulation suggested that one-device mobile banking transactions should only be compliant on devices and apps leveraging explicit hardware support like ARM TrustZone. The final version, however, only requires that the initialization and confirmation logic make "use of separated secure execution environments".

Although the final RTS demands "that the breach of one does not compromise the reliability of the others, in particular when any of these elements are used through a multi-purpose device", the EBA also clarifies that a "mobile phone which can be used for both giving the instruction to make the payment and for being used in the authentication process" does comply with the requirements of SCA. Furthermore, the authentication procedure needs to ensure that neither the user nor a third party has modified the software or device. Alternatively, the scheme needs to take measures to "mitigate the consequences of such alteration".

Within the purview of the EBA regulations, the banking and the photoTAN app can be likely described as independent even though we oppose the EBA's decision. However, if the regulation requiring the authentication procedure to detect and mitigate device modifications is currently met, remains debatable:

- The breach of the banking app can compromise the photoTAN app and vice versa. It is true that the sandboxing mechanism of the prevailing mobile operating systems mitigates the risk of a successful attack against one app — due to an app-specific vulnerability — also compromising the other app. If the system layer of the device is compromised, however, both elements are affected equally, and an attacker can gain access to the knowledge and possession factor at the same time. This is not only true for all analyzed banks but also for any multi-factor authentication scheme implemented on a single multi-purpose device. It remains unclear if implementing the authentication procedure using two separate apps is a strict guideline or if one-app authentication schemes could also be regarded as compliant if they take — yet to be definded — software-based measures.
- None of the analyzed apps provides protection against a device that has been compromised by a third party with privileged malware. Some of the apps — Commerzbank and Comdirect — reduce the tampering of their photoTAN app. All of the analyzed banking apps, however, are totally exposed to attacks as they do not apply effective safeguards against system-level malware. In general, however, this is an impossible task to perform on today's commodity smartphones without hardware support for strong isolation like the ARM TrustZone.
- Only Commerzbank and Comdirect deal with the alteration of the system by detecting rooted devices. In such a case, the photoTAN app by Commerzbank refuses to run, and the photoTAN app by Comdirect advises the user. However, the photoTAN apps of Deutsche Bank and Norisbank do not take any measures in this respect. Moreover, none of the banking apps addresses system alterations at all. Again, it is generally impossible to effectively scan the system layer from an unprivileged app.

6 CONCLUSION

The transaction manipulation attack demonstrated that running the banking and the photoTAN app on the same device cannot technically be regarded as secure. Compromising one authentication channel immediately leads to compromising the other authentication channel. Even though the photoTAN procedure has flaws that are specific to it, the core issues lie in the conception of app-based authentication and are thus common to all authentication schemes of this kind.

We appreciate the recent efforts of the European Union and the EBA that defined common standards for the security of online payments. Unfortunately, the EBA has refrained from defining clear limits, particularly for single-device mobile payments that can at best keep an illusion of two-factor authentication. The final draft poorly accounts for the threats that mobile devices already face but also the ones they will face in the future, namely banking trojans that explicitly attack single-device mobile banking. As a result, banks will continue to affirm the same statement they gave when we confronted them with our findings through the public media (Tanriverdi, 2016): There are no claims known to date.

REFERENCES

Bitkom e.V. (2016). Digital Banking. Accessed: 25 September 2016.

Bundesverband deutscher Banken e.V. (2015). Zahlen, Daten, Fakten der Kreditwirtschaft. Accessed: 25 September 2016.

Check Point Mobile Research Team (2016). From HummingBad to Worse: New In-Depth Details and Analysis of the HummingBad Android Malware Campaign. Accessed: 11 September 2016.

Commerzbank A.G. (2016). Commerzbank photoTAN - Android Apps on Google Play. Accessed: 12 October 2016.

Company, B. . (2016). Customer Loyalty in Retail Banking: Global Edition 2016.

Cronto (2008). Commerzbank and Cronto Launch Secure Online Banking with photoTAN. Accessed: 20 September 2016.

Cronto (2011). Cronto Launches World's First Visual Transaction Signing Hardware. Accessed: 4 October 2016.

Cronto (2011). CrontoSign. Accessed: 02 October 2016.

Dehghanpoor, C. (2016). Brain Test re-emerges: 13 apps found in Google Play. Accessed: 16 September 2016.

Deutscher Sparkassen- und Giroverband (2015). Stellungnahme zur Angreifbarkeit von App-basierten TAN-Verfahren. Accessed: 19 November 2017.

Dmitrienko, A., Liebchen, C., Rossow, C., and Sadeghi, A. (2014). On the (in)security of mobile two-factor authentication. In Christin, N. and Safavi-Naini, R., editors, *Financial Cryptography and Data Security - 18th International Conference, FC 2014, Christ Church, Barbados, March 3-7, 2014, Revised Selected Papers*, volume 8437 of *Lecture Notes in Computer Science*, pages 365–383. Springer.

Donenfeld, A. (2016). QuadRooter: New Android Vulnerabilities in Over 900 Million Devices. Accessed: 11 September 2016.

European Banking Authority (2016). EBA consults on strong customer authentication and secure communications under PSD2.

European Banking Authority (2017). EBA paves the way for open and secure electronic payments for consumers under the PSD2.

European Union (2015). Directive (EU) 2015/2366 of the European Parliament and of the Council of 25 November 2015 on payment services in the internal market, amending Directives 2002/65/EC, 2009/110/EC and 2013/36/EU and Regulation (EU) No 1093/2010, and repealing Directive 2007/64/EC (Text with EEA relevance). 337:35–127.

Goodin, D. (2016a). 10 million Android phones infected by all-powerful auto-rooting apps. Accessed: 11 September 2016.

Goodin, D. (2016b). Android phones rooted by "most serious" Linux escalation bug ever. Accessed: 31 October 2016.

Haupert, V. and Müller, T. (2016). Auf dem Weg verTAN: Über die Sicherheit App-basierter TAN-Verfahren. In Meier, M., Reinhardt, D., and Wendzel, S., editors, *Sicherheit 2016: Sicherheit, Schutz und Zuverlässigkeit, Beiträge der 8. Jahrestagung des Fachbereichs Sicherheit der Gesellschaft für Informatik e.V. (GI), 5.-7. April 2016, Bonn*, volume 256 of *LNI*, pages 101–112. GI.

ING (2016). ING International Survey: Mobile Banking 2016.

Kivva, A. (2016). The banker that can steal anything. Accessed: 22 September 2016.

Konoth, R. K., van der Veen, V., and Bos, H. (2016). How anywhere computing just killed your phone-based two-factor authentication. In Grossklags, J. and Preneel, B., editors, *Financial Cryptography and Data Security - 20th International Conference, FC 2016, Christ Church, Barbados, February 22-26, 2016, Revised Selected Papers*, volume 9603 of *Lecture Notes in Computer Science*, pages 405–421. Springer.

Lafortune, E. Proguard. Accessed: 12 October 2016.

Maier, D., Müller, T., and Protsenko, M. (2014). Divide-and-conquer: Why android malware cannot be stopped. In *Ninth International Conference on Availability, Reliability and Security, ARES 2014, Fribourg, Switzerland, September 8-12, 2014*, pages 30–39. IEEE Computer Society.

Mulliner, C., Borgaonkar, R., Stewin, P., and Seifert, J. (2013). Sms-based one-time passwords: Attacks and defense - (short paper). In Rieck, K., Stewin, P., and Seifert, J., editors, *Detection of Intrusions and Malware, and Vulnerability Assessment - 10th International Conference, DIMVA 2013, Berlin, Germany, July 18-19, 2013. Proceedings*, volume 7967 of *Lecture Notes in Computer Science*, pages 150–159. Springer.

Number26 GmbH (2016). N26 - Banking by Design. Accessed: 10 October 2016.

Polkovnichenko, A. and Boxiner, A. (2015). BrainTest – A New Level of Sophistication in Mobile Malware . Accessed: 16 September 2016.

Promon AS (2016). Promon SHIELD™ - Rock-Solid App Security! Accessed: 12 October 2016.

Rao, S. P., Kotte, B. T., and Holtmanns, S. (2016). Privacy in LTE networks. In Yan, Z. and Wang, H., editors, *Proceedings of the 9th EAI International Conference on Mobile Multimedia Communications, MobiMedia 2016, Xi'an, China, June 18-20, 2016*, pages 176–183. ACM.

Reaves, B., Scaife, N., Tian, D., Blue, L., Traynor, P., and Butler, K. R. B. (2016). Sending out an SMS: characterizing the security of the SMS ecosystem with public gateways. In *IEEE Symposium on Security and Privacy, SP 2016, San Jose, CA, USA, May 22-26, 2016*, pages 339–356. IEEE Computer Society.

Tanriverdi, H. (2016). Mobiles Banking: Hacker knacken Photo-Tan-App. *Süddeutsche Zeitung*, 72(241).

Thomas, D. R., Beresford, A. R., and Rice, A. C. (2015). Security metrics for the android ecosystem. In Lie, D. and Wurster, G., editors, *Proceedings of the 5th Annual ACM CCS Workshop on Security and Privacy in Smartphones and Mobile Devices, SPSM 2015, Denver, Colorado, USA, October 12, 2015*, pages 87–98. ACM.

Zhang, V. (2016). 'GODLESS' Mobile Malware Uses Multiple Exploits to Root Devices. Accessed: 11 September 2016.