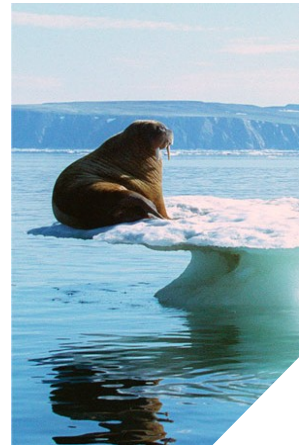
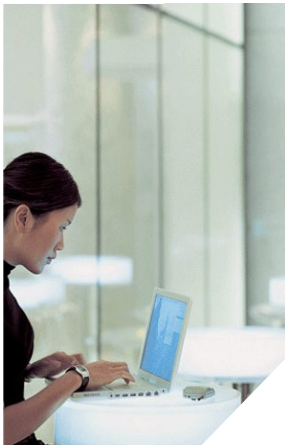




Mobile Application Security





© 2012, Cryptomathic A/S. All rights reserved

Jægergårdsgade 118, DK-8000 Aarhus C, Denmark

This document is protected by copyright. No part of the document may be reproduced in any form by any means without prior written authorization of Cryptomathic.

This document is provided “as is” without warranty of any kind.

Cryptomathic may make improvements and/or changes in the product described in this document at any time. The document is not part of the documentation for a specific version or release of the product, but will be updated periodically.

Date: 2012-02-22
Doc. title: Mobile Application Security
Doc. version: 1.0

CRYPTOMATHIC OFFICES

Canada

Cryptomathic Security Corp.
204 rue St-Sacrement, Suite 300
Montreal, Quebec
H2Y 1W8
Canada
Tel: +1 514-871-9398
Fax: +1 514-937-6140

Germany

Cryptomathic GmbH
Bretonischer Ring 7
D-85630 Grasbrunn
Germany
Tel: +49 (0)89 451874-0
Fax: +49 (0)89 451874-1

USA

Cryptomathic, Inc.
111 North Market Street, Suite 300
San Jose, CA 95113-1116
USA
Tel +1 408-981-4745

Denmark – HQ - Research & Development

Cryptomathic A/S
Jægergårdsgade 118
DK-8000 Aarhus C
Denmark
Tel: +45 8676 2288
Fax: +45 8620 2975

United Kingdom – Innovation

Cryptomathic Ltd
327 Cambridge Science Park
Cambridge
CB4 0WG, UK
Tel: +44 (0)1223 225350
Fax: +44 (0)1223 225351

www.cryptomathic.com



Table of Contents

- 1 Introduction: 4**
- 2 Analysis in mobile attack/defense – a threat model 5**
 - 2.1 Status quo 5
 - 2.2 Threat model 6
- 3 Cryptomathic Mobile Security Strategy – an evolving plan 9**
 - 3.1 Phase 1 Security Architecture 11
 - 3.2 Phase 2 Plans 13
 - 3.2.1 Data Obfuscation 14
 - 3.2.2 Code Obfuscation 15
 - 3.2.3 Limits of Secure Coprocessors 15
 - 3.2.4 Dependence upon HTTPS 16
 - 3.2.5 Clock Tampering and File Migration 16
 - 3.3 Phase 3 Plans and Beyond 17
 - 3.3.1 Breaking criminal economic models 17
 - 3.3.2 Non-standard crypto 18
 - 3.3.3 Threat Monitoring and Intelligence 19
- 4 Cryptomathic Offerings 20**
 - 4.1 The Cryptomathic Authenticator 20
 - 4.2 PKI Solutions 22
 - 4.3 Benefits of Working with Cryptomathic 22
- 5 About Cryptomathic 23**
 - 5.1 Company Background 23



1 Introduction:

Cryptomathic have a suite of mobile applications designed for user and transaction authentication, including mobile TOTP, HOTP, OCRA and CAP transaction signing (hereinafter Cryptomathic Mobile AuthApp). Our mobile applications have been designed in accordance with an *evolutionary mobile security strategy* that is designed to retain an appropriate level of security in a fast-changing environment where platforms and operating systems dramatically change over a cycle of months rather than years.

We believe this is the right approach for banking and government customers, and in this discussion we aim to explain why. The document first covers the current state of the art in mobile attack and defence, and works through the common threats. It then describes the Cryptomathic security strategy split into several iterative phases and details the technical security mechanisms used in the first iteration of the plan.





2 Analysis in mobile attack/defense – a threat model

2.1 Status quo

One must be bullish to operate in the mobile space, since many of the innovations and changes to the way which we compute have taken place over just a handful of years. But there is more than just blind optimism driving this surge: just as mobile has been appealing to developers because of homogeneity of platforms, and easy distribution channels for software with low barrier to entry, the same savings have been afforded to the manufacturers. But this trend toward more similar, closed and regulated platforms helps manufacturers with security too, since they can focus their efforts more narrowly. As a result the current mobile malware scene is not just a reflection of the adoption and changing use of the platform but also of the design and peculiar properties of this new platform. When considering an Apple iPhone for example, maybe a better comparison than to call it a miniature PC is to call it a miniature Sony Playstation 3.

Currently malware has some presence on Android, but is almost non-existent on iPhone. Typical malware functionality includes surreptitious sending of SMS messages and calling of premium rate numbers, click diversion (for stealing advertising revenue) and a little keylogging/SMS interception for harvesting credentials and SMS-based OTPs. But nearly all this malware operates within bounds of *requested permissions*, that is, the user clicks to agree to grant the application the permission it needs to perform these malicious acts, and very few currently escalate their privileges using root exploits. Thus the primary attack vector is to advertise the malware in the official appstore (or a third party store when available), and to have it installed by consent.



There is a very active mobile security research community, which arguably surges ahead of the criminal community (unlike the Anti-Virus developers which were playing catch up on PC), that have written about and demonstrated much more advanced and powerful attacks than what is generally seen in the wild. There is some evidence of adoption of research ideas by hackers, but still the general malware present is not advanced.



The same bullish view is not given by the companies looking to expand their anti-virus and protection software suites to mobile platforms, who raise fear, uncertainty and doubt about future trends and point towards the extremely fast rate of malware development. However this malware development rate is comparable to the rate of the platform itself. Reading for example the Juniper Networks mobile threat report¹, one sees a bait-and-switch approach where the user is scared with the malicious capability of commercial spyware packages, and then switched to expect the proliferation levels of less severe malware. The commercial spyware packages such as Mobispy all require direct physical access to install and usually rooting/jailbreaking² of the phone first.

So in summary, it appears that currently the operating system controls hold effectively at preventing applications from exceeding their authorised permissions, and that the major problem is the perennial challenge of educating users to make wise decisions regarding which apps to install. While the latter issue is of course a threat for deploying mobile authentication for use in banking/government, note that it means that users who did manage to install the legitimate app are not threatened by malware.

Furthermore, for so long as malware authors can continue to create revenue by exploiting non-escalating compromises (e.g. compromising the web browser but not the whole operating system) to perform actions such as click diversion and advertising, they are not motivated to develop more advanced attacks which threaten the security of existing installed apps. This partitions the malware community and means fewer hackers are working on the most powerful cracks.

2.2 Threat model

Let's now consider the threat model – the set of threats – against which a mobile app might need to have protection. In a full model process these threats are derived from a variety of attacker goals, foremost being monetary gain but also including revenge, anarchy, curiosity, perceived public good. The attackers themselves are also classified/grouped by resource levels³ and goal.

Threat	Goal/Resource	Notes
Malware attack	G: Monetary Gain R: Large black-market economy	Malware attack remains the primary threat for authentication mobile apps. Regardless of installation vector (phishing, app store poisoning, drive-by website) the result is similar and those deploying the attack are likely from the same criminal economy. Resistance comes from technical phone measures, user education and distribution channel policing.

¹ Juniper Networks Malicious Mobile Threats Report 2010/2011

² *Rooting* and *Jailbreaking* refer to the owner gaining full administrative access to the phone in a way that circumvents the interests and security policy of the phone operating system manufacturer.

³ It is worth noting that the landscape of threats in computer security in general has changed considerably since the arrival of the *advanced persistent threat* (APT), whose most unique and concerning characteristic is persistence of attack on a single target organisation, which may be the focus of attack for several years (the technology level of exploits/vulnerabilities in the attack rarely seems to be as high as initially thought).



Borrowed phone	<p>G: Revenge, monetary gain</p> <p>R: single layperson + commercial spyware market</p>	<p>The attacker might have brief direct access to the phone of a family member or colleague. Here the individual's resources are very limited but they may buy/licence quite advanced spyware. Best security is afforded through platform lockdown to prevent any type of spyware being installed, and user authentication before granting access to the authentication token (e.g. a PIN). Commercial spyware manufacturers can possibly be pressured to ensure their products cannot be used for stealing auth credentials (e.g. interception phone # black list, monitoring blacklists)</p>
Stolen phone	<p>G: Monetary Gain</p> <p>R: Small black-market economy</p>	<p>The attacker might steal the individual's phone either by mugging or pick pocketing. Here research shows that a majority of users will notice the theft within an hour, so the challenge is to ensure that credentials cannot be stolen, sold and abused all within the timeframe before reporting. Measures to damage efficiency of the criminal economy will help here. Some phones now have remote kill switches and tracking in addition. Loss of personal data and email access on the phone for use in identity theft is also a risk, but mainly to the user and less to the organisation relying upon the mobile for authentication.</p>
Reputational attack	<p>G: Perceived public good, anarchy</p> <p>R: Large organisation, top staff, limited budget</p>	<p>Researchers, pressure groups and lobbyists may take a dislike to a particular larger project (particularly those projects related to personal data centralisation and privacy) and attack the authentication mechanism as a way of highlighting risk or simply because it is there. Here what is important is that the architecture is seen to be secure and that security claims can be justified and defended. Likely the attack will come via the media. It may be necessary to prepare and brief spokespersons on the long-term mobile security strategy and to consider when briefing the difference between protecting the overall bottom line and the loss to the individual – assurance of fair dispute resolution mechanisms is important. The cruder (but also very effective) defence is stonewalling.</p> <p>Of most interest resistance to this sort of attack comes from careful, clean and elegant engineering of the system, which has wide ranging design consequences. Security through obscurity, obfuscation and so forth should be employed with caution.</p>
White-hat hacking	<p>G: Curiosity, perceived public good</p> <p>R: Small number of highly trained individuals</p>	<p>An individual or small group of researchers who make a living from finding and reporting/selling vulnerabilities may test the application and find a vulnerability. Researchers frequently spend most time exploring systems which are present, easy to experiment upon and in the public eye, so this is much more relevant for mobile app security than back end server stuff.</p> <p>Here the main risk arises from failing to acknowledge the vulnerability or engage properly with the researcher, where the situation might rapidly escalate into public disclosure of the vulnerability including exploit details. Best resistance is afforded from proper organisational procedures to correctly route enquiries to those who can make the right judgement call.</p>



The table shows that a good mobile security strategy must defend both against specific mobile threats but also against more generic threats such as reputational attack and white-hat hacking which have increased prevalence and importance in the dynamic mobile market. However it also shows that attacks involving direct physical contact (theft and borrowing) are of limited interest beyond achieving due diligence (giving the customer tools to protect their credentials from family/colleagues) mainly due to lack of scalability and ease of credential cancelling and reissue after theft.

We will return to this taxonomy of threats throughout the remainder of the document as we cover the mobile security architecture, and consider which features address which threats.





3 Cryptomathic Mobile Security Strategy – an evolving plan

The Cryptomathic mobile security strategy is an iterative evolutionary strategy. It is *iterative* in that it uses distinct phases of security improvements, and relies on frequent updates to software and protocols. It is *evolutionary* in that it adapts and responds to developing threats which due to the complexity of the market do not necessarily develop in a predictable way or according to a predictable timescale.

Our strategy currently has two defined phases – phase 1 and 2, and we make a more general discussion of phase 3 and beyond, and the goals and technologies likely to play strongly during this phase.

Through sheer crystal ball gazing one can estimate the duration of effectiveness of the first phase, and on the assumption that the trend toward closed platforms which only run manufacturer-approved applications continues for consumer electronics devices (iPhone, PS3, XBOX and so forth), the first phase could last as long as 2—5 years.

The evolving plan is in sympathy with the rate of change of software and hardware platforms in the phone industry. There are multiple incentive-aligned drivers for this rate of change:

- Operating system vendors releasing new versions to
 - close jailbreak and security loopholes that allow users to install unapproved software
 - correct bugs or performance issues
 - add new features to be innovative or match competition
- Phone manufacturers
 - promoting their new handset models
 - delivering more powerful CPU/GFX to the platform for gaming, necessitating API and OS updates

In concrete terms there is a new iPhone software version approximately every six weeks and a new Android version about every three months. Figure 1 shows frequency of update:

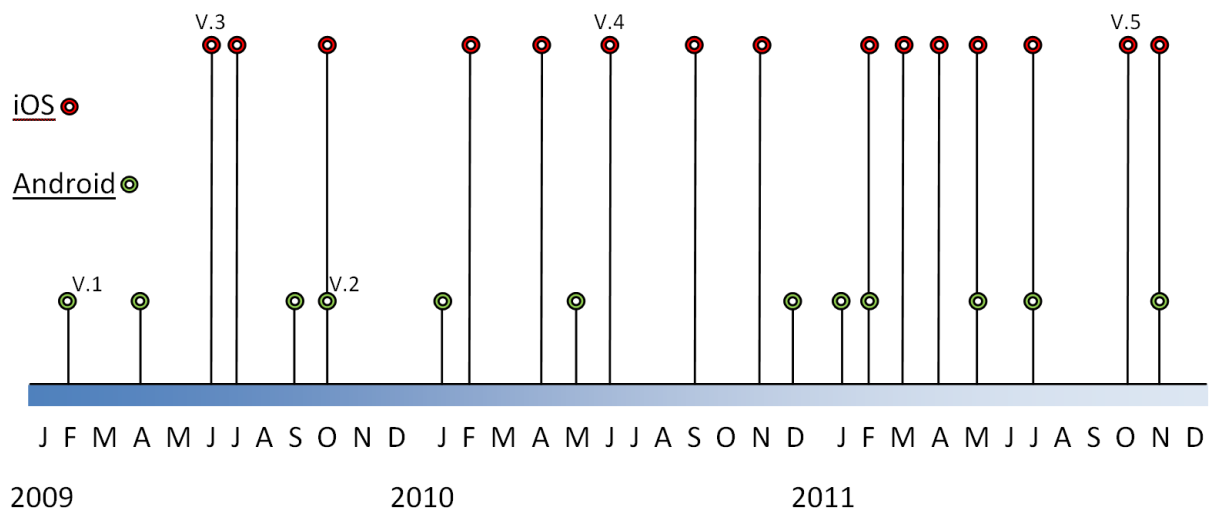


Figure 1 – Mobile OS Update Frequency



Given this natural rate of flux (which is also a source of unpredictability) it is perfectly reasonable to expect to update a security app to a new version several times a year, certainly once a quarter, if not more frequently. Mobile phone app stores can ensure that users are sufficiently reminded to go ahead and install updates, and the positive feedback of updates yielding new features or levels for games, and fixing issues related to new OS versions means the user is likely to make a positive choice to upgrade rather than to ignore it. They might upgrade in the end only simply to silence the notification area which otherwise will permanently remind them.

Meanwhile malware evolves at increased rates too, and especially when considering drive-by infections from websites (the user is infected automatically upon visiting a site due to a browser vulnerability) the trend on the PC platform is toward polymorphic generation of a unique malware binary for each infectee, which has almost totally undermined virus infection statistics collection. On mobile the trend is toward automatic reverse engineering of existing applications, addition of the malware and then resubmission to the store under a different name. The same attacker might submit 50-100 new apps to the marketplace which rip off other apps in this way. Although the fake marketplace publishing account may be identified, the cost of manufacturing a new identity has to remain low (to encourage legitimate developers to participate), and the attacker may likely be in an uncooperative jurisdiction. Or the attacker might steal credentials from a legitimate but lapsing developer to launch the attack.

The future for mobile might also bring with it new hardware-backed security features, be it new TPMs (trusted platform modules), proprietary cryptoprocessors (such as found in the iPhone), SIMs or Secure Elements (SEs) which are part of the GlobalPlatform TEE⁴ trust architecture for mobile payments. Our plan can take advantage of these emerging security technologies but with two important caveats.



The first caveat is the issue of *shared risk* where if you rely on a security technology which is used by another party for a different purpose, then you both share risk of compromise. An example is that to depend on infrastructure used for digital rights management (DRM) as an authentication provider is to make your security come under daily attack from organisations and groups that wish to freely pirate movies/games or simply to evade region lock or accessory control. This adversary may be far more powerful and widespread than the native criminal adversary for hacking banking authentication, for example, and able to operate with total impunity in many jurisdictions. Therefore adopting a new security technology on the mobile platform may create shared risk depending on the source of the tech and this must be weighed up before making the decision

⁴ Trusted Execution Environment



The second caveat is the issue of *negotiating access*. A secure capability may exist on the phone but may require cooperation of both handset manufacturer and mobile network operator in order to have access to load a trusted application. Any iterative plan considering relying on new secure elements must be mindful of the likely challenges facing a company adopting the technology from a business and economic perspective. This is particularly relevant to those creating payment or transaction authentication applications as other access-granting companies may demand payment in the form of a transaction fee. Thus for a new technology to be part of the plan there needs to be a credible route for it to become widely available, such that one is not a hostage to the service provider.

3.1 Phase 1 Security Architecture

The Cryptomathic mobile app registration workflow is shown in Figure 2 below, which forms the backbone of the security architecture. The registration workflow delivers a long term key K to the mobile app which is shared with the authentication service provider and can be used to implement specific authentication protocols such as OATH HOTP or TOTP, or used to bootstrap delivery of other credentials. The app is delivered via an app store containing a hardwired public key K_{pub} corresponding to the server. The server keeps the private half of the key in a Hardware Security Module (HSM), which is a high performance secure coprocessor that prevents theft of the key.

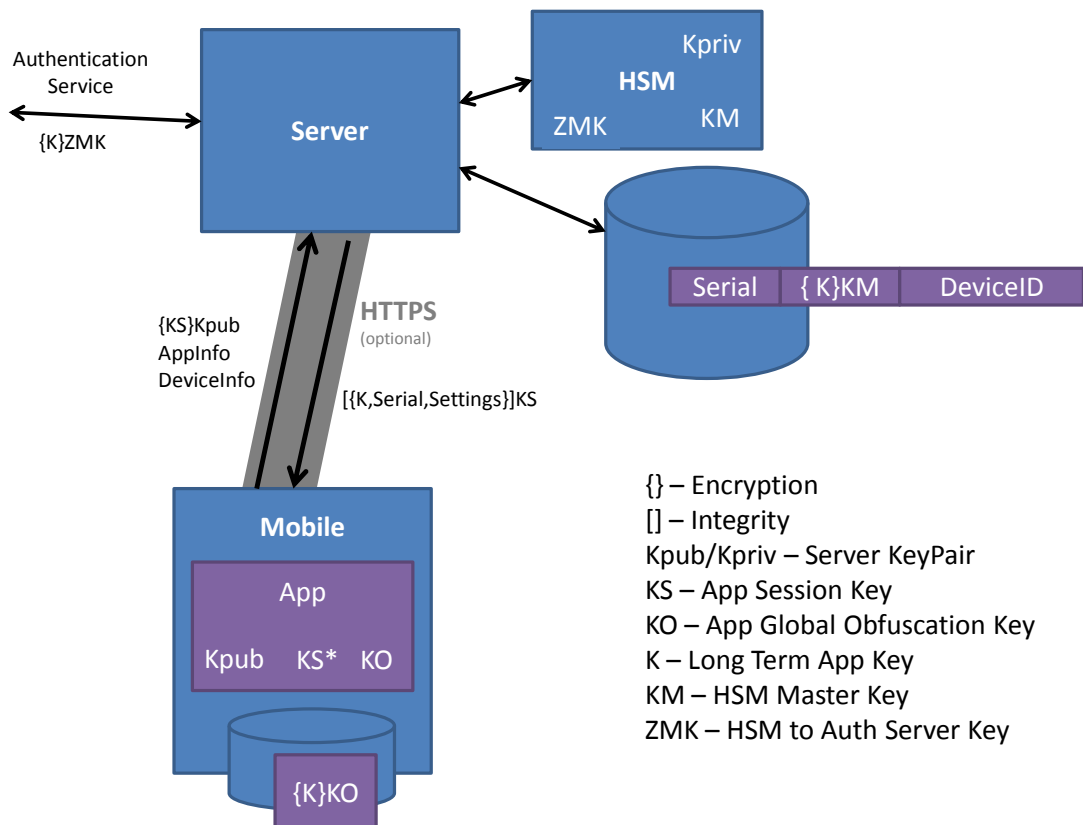


Figure 2 – Current mobile security architecture

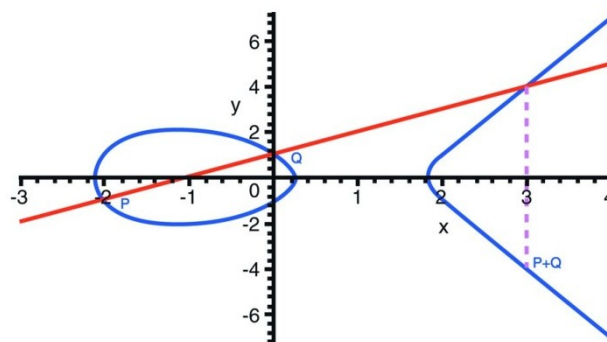
The workflow is as follows:

1. The authentication service provider publishes the app to an app store.



2. The user downloads the app onto their mobile device.
3. The user launches the app which detects first install and enters an initialisation phase
4. The app generates a session key KS^* and encrypts this under the hardcoded public key of the server, K_{pub} .
5. The app delivers the session key to the server using either HTTP or HTTPS, depending on the URL provided for the server which is also hardcoded into the app.
6. The server receives the session key and at this point generates a unique serial number for the mobile (ideally just a long random number).
7. The server sends a request to the HSM to generate a long-term key K for the phone, and to encrypt this key, the serial number and some server-chosen configuration parameters under the session key.
8. The HSM returns the key K encrypted in a message to return to the mobile, and the same key K under a local storage master key KM , suitable for storage in a long-term database.
9. The server returns the message to the mobile app via the HTTP/HTTPS link and the app uses the session key to decrypt and integrity check the message, recovering the key K and the configuration data.
10. The mobile app stores the long term key K and configuration data in local storage accessible only to that specific app, obfuscated by encryption using a hardcoded key KO . The key KO is derived from hashing, using SHA1, the phrase “This global obfuscation key created in accordance with threat model.”, thus defending against reputational attacks.
11. The server can subsequently deliver the key K to another service under a transport key (a Zone Master Key – ZMK) or can provide authentication services itself.

This registration workflow delivers a shared secret between two parties which is connected with a unique identifier. It is then up to the authentication service provider to allow the user to bind together their authentication token identifier with their existing (or new) account.



Elliptic Curve

The workflow is designed to use standard cryptography, namely RSA or ECC and AES encryption with data formats which are suitable for use with an HSM at server side. This allows the registration server to scale to support millions of registrants without making the server itself vulnerable (as the HSM can protect the keys). The same cryptographic



algorithms can also be implemented efficiently and effectively in pure software on the mobile device (not relying on underlying libraries which might change) therefore making the mobile application security core self-contained.

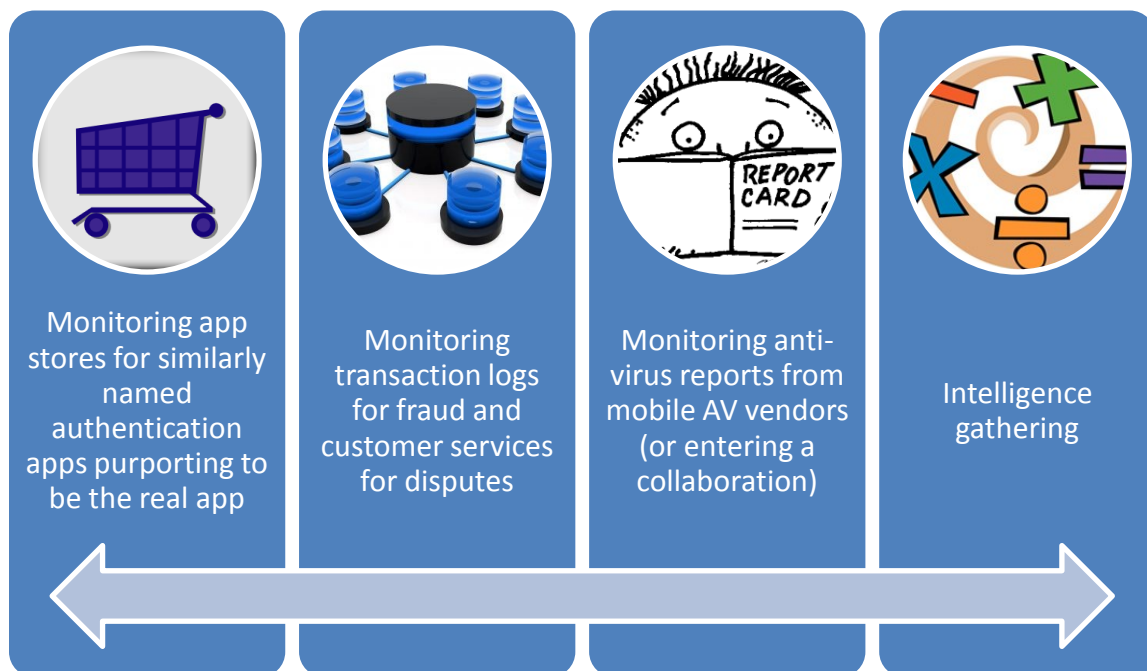
If the authentication service provider wishes to change the registration site they can either do this with DNS mapping, or by publishing an application update. The same process allows the hardcoded public key K_{pub} to be changed. The mobile device also sends model information and serial number during the registration process to enable statistics on mobile platform usage to be collected by the server.

The application is configured not to store its authentication key in an area of the filesystem which can be backed up, and therefore when a phone is replaced or recovered from backup it is necessary to re-register the phone.

In essence, the phase 1 security approach is to rely totally on operating system file access controls to prevent the key from being stolen. Thus for any phone where malware typically operates within granted app permissions this key will remain secure (since there is no permission for “override access to all files on filesystem”), but for malware which has root access, this key will be trivially recoverable. Keychain/keyring storage of the key is deliberately not used as for many authentication algorithms temporary access to the key is equivalent to theft of the key (see phase 2 for further discussion) so currently confers little advantage. However use of keychain may incur interaction with cloud-based backup services and here it is preferable to avoid user credentials from being stored centrally with a third-party service provider.

3.2 Phase 2 Plans

In the second phase of Cryptomathic’s security architecture we assume that real users of the authentication service are starting to come under attack, and that this attack is detectable. If the attack is a Malware threat, it could be detected by:





Due to the architecture of the app, it is likely that when the second phase is reached, root exploits of phones are available in sufficient quantity to be available on the black market to attackers. The second phase requires countermeasures to be deployed to limit the effectiveness of such malware at stealing credentials until the operating system vendor can patch the vulnerability and affected users can recover their phones.

Note that if the exploit was delivered by downloading a malicious app which used privilege escalation from an app store, the app store provider is in a position to collaborate and provide a list of all users who have downloaded both the authentication application and the malicious application, giving the potential for a very targeted security warning to be sent out, which should yield very few false positives. Malware infecting from website drive-by will not be enumerable in the same easy manner, but should be less frequent anyway in the second stage as it requires two exploits together – one to seize control through the web browser, and a second to escalate privileges to root.

Again, note that the challenge is to modify the functioning of the application to frustrate effectiveness of the current attack only for the duration until the vulnerability can be patched. Hopefully this will be as little as two to three months, depending on the severity of the compromise.

To this end phase 2 proposes use of the following techniques to render the malware temporarily ineffective.

3.2.1 Data Obfuscation

The phase 1 design already includes obfuscation of stored data in file on disk which is designed to deter users from trivially extracting the key via analysing the stored files when running the app within an emulator or on a jailbroken/rooted phone. This obfuscation can be upgraded to require engineering effort for the attacker to re-engineer the key recovery algorithm. We use several core techniques here:

- Prevent attacker from harvesting required data. A not-so-smart attacker would reverse engineer the authentication app, implement key recovery in the malware, and then upload just the key to the malware controller. Therefore simply changing the algorithm will require the attacker to re-engineer this type of malware, buying a couple of weeks of safety. A smarter attacker might have the malware upload all stored files made by the phone application, so that if the recovery algorithm is changed, the malware need not be updated and the recovery can be performed offline at a central location. Therefore, one must
 - Increase the amount of storage used for the key. Make at least 10MB of stored data which might contain the key.
 - Increase the complexity of stored file data – make files in a complex directory structure which appear and disappear in an evolving process over multiple days. Make sure that any missing or extra files contribute to the recovery algorithm to stop it working.
 - Integrate incidental phone setting data into the storage format and recovery algorithm. This might include phone IMEI, UDID, current date, volume and locality settings and so forth, which the attacker may have forgotten to harvest



when the malware uploads the files from the app to the malware operator. This might make the harvested data worthless.

Together these confounding and obfuscation techniques can drive up the time it takes an attacker to upgrade their malware to harvest registration data from new infections.

3.2.2 Code Obfuscation

Together with data obfuscation, code obfuscation can be used to slow down the attacker in determining the algorithm that the app uses to recover the key, and forcing the malware to work rather than by interpreting the static application long-term data, to observe the application and then steal the key at the moment of use. For this reason a well-designed resistant application will use the minimum of system calls (especially avoiding crypto system calls) as these are easy hooks for malware with root control to access the key despite various obfuscation mechanisms. Such code obfuscation is easier to do for C code than it is for Java (favouring the iPhone platform) but it still has value on both platforms. Any obfuscation which requires the attacker to deploy a monitoring rootkit which watches a running process rather than just reading files from storage has already elevated the game and won a significant battle in the overall war.

3.2.3 Limits of Secure Coprocessors

Secure coprocessors used to hold crypto keys such as SIM cards, Secure Elements and TPMs may become available during the second phase of deployment, and the barriers to use of such coprocessors have already been discussed in this paper. But assuming they are available, an important aspect to consider before utilising the coprocessor is whether the algorithm used for authentication has predictable inputs or not.

For example, time-based, counter-based OTPs and transaction signing all have predictable inputs. Therefore the attacker can simply use temporary access to the security coprocessor from having infected the main processor OS with malware to harvest a codebook or dictionary of inputs and outputs which represents everything he will need in future to pretend to have possession of this key.

It is only where a truly random challenge number is provided by the authentication service (such as in a challenge response protocol) that the attacker has no idea in advance of an authentication attempt what data he will need to process using the stored key. Unfortunately moving to challenge/response authentication incurs significant usability penalties as the user must transfer the challenge onto the mobile device to be signed (either by entering it on the keypad or using some automated transfer such as 2D barcodes or flicker patterns).

An alternative is to implement clever logic in the secure coprocessor which limits the type of message which can be processed using the key, for instance if the key is used to make an HMAC of an authentication input, it might use an internal monotonic counter (which can only ever increase) to implement an HOTP password generator. Here the attacker can use his compromised access to the coprocessor to harvest many hundreds or thousands of OTPs, but he will be unable to reset the secure processor to its original state and therefore is likely to be detected straight away.

Thus for proper benefit a secure coprocessor needs to have specific security features available, and coprocessors which simply store a key securely but grant access to use it for any purpose have little value. Thus in our long-term strategy we expect to see limits on the



effectiveness of secure coprocessors for defence and their likely value may mainly be in obscurity and creating a new barrier for the attacker to reverse engineer against. Therefore they may only be employed in phase 2 if the cost analysis is effective.

In the long term, especially considering the move away from OTPs toward transaction authentication, the secure coprocessor will need *trusted path* to the user – to display the data which is about to be authenticated and to seek approval or rejection in a way which cannot be interfered with by the malware. However such a trusted UI is unlikely to be graphically very pretty and will grate considerably with the value proposition of modern smartphones such as the iPhone (and the new Android 4.0 Ice Cream Sandwich) where visuals and aesthetics and usability is everything. For that reason it is unlikely that trusted UI proposals will garner a lot of support from the handset manufacturers and it is crucial to have them on board.

3.2.4 Dependence upon HTTPS

The increasing suspicion being levelled at the global SSL/TLS infrastructure due to the proliferation of root certificate authorities represents a barrier to using TLS libraries provided by the phone manufacturer. These might not have a configurable set of trusted certificates and so make the authentication application open to attack by powerful adversaries who compromise CAs.

For this reason in the second stage of the plan we can abolish the outer layer of TLS authentication and fall back to the inner RSA/AES proprietary encryption algorithms. This move would mainly be done to remove TLS from the protocol in order to mitigate reputational and white-hat hacker attacks – in terms of malware resistance, even if it is ineffective there is no harm in leaving it in. Yet to protect against this broader range of threats to an authentication app one must have a plan in place to remove and replace it.

3.2.5 Clock Tampering and File Migration

Cryptomathic's security core already has mechanisms in place to mitigate the risks from a layperson gaining temporary access to an individual's mobile phone and trying to steal some authentication credentials. Of course while due diligence may be satisfied simply by providing a PIN it is likely that a family member could discover this by shoulder surfing.

We have a mechanism prepared to detect clock tampering, where in the case of a time-based OTP token, the attacker might borrow the phone, set the clock forward by several days, then run the app and harvest future TOTP values for use in authentication, before returning the time to normal. The clock tampering detector will spot this change in the time sequence and initiate a warning message which can be displayed to the user, at an unknown time in the future (so that the attacker cannot set the time to just when this message is about to appear and then dismiss it). Likewise we have a mechanism to include an IMEI/UDID check when recovering obfuscated data from storage to prevent an attacker who tries to take a full backup of the phone from being able to restore data from this backup to a different handset and thus access the authentication credentials. Here the general decision taken in phase 1 to prevent back-up of key material to the user's PC or to the cloud will hopefully remain in force anyway, rendering this control redundant.



3.3 Phase 3 Plans and Beyond

In an evolutionary strategy, there is no certainty for what the third phase of defensive measures might contain. But there are ideas and tentative plans.

3.3.1 Breaking criminal economic models

Crucially one should focus on “breaking the criminal economic model”, rather than trying to *prevent* the possibility of attack, one should try to reduce the profitability of these attacks. This is actually a much easier task, as it is a form of attack, rather than defence. When you defend against technical breaches, you have a large code base to protect and the attacker may look anywhere, so the app developer is disadvantaged. When you change your strategy to *attack* the criminal business model, their business model must be resilient in all places – here is where one gets the advantage.

Criminals have managed to achieve the scale of attack they currently do through creating a (black market) economy and marketplace which allows different types of crooks to specialise. Nowadays exploits are bought on the market, botnets are rented, and so forth. But while a closed group can be policed with ‘heavies’ torturing or killing rivals and injecting fear to create honour among thieves, this is much harder to achieve in a distance selling marketplace. So the black market economy does not have access to the same quality of dispute resolution and fair dealing frameworks that are available to us.

Applying this to mobile apps: any design features which can cause uncertainty about the value of a criminal product of “a compromise on app X”, or “1000 harvested login sets for app X”, will dramatically reduce value. The more unpredictable these changes, the better.

- For example, a probabilistic upgrade strategy for a mobile app might mean that 75% of users receive new version A, and 25% version B. When the crooks harvest data, they don’t realise that they need to support both types of app, so 25% of the data comes out as invalid. If they don’t anticipate this, the friction involved in selling data which turns out to be partly duff can inhibit cooperation, until the crooks develop a



way of amortising data harvests into large enough sets that it won't matter if some of the credentials are invalid.

- Sudden gluts in the availability of easily harvested data followed by droughts can cause criminals to over-reach and if they don't project the amount of data they will be able to steal correctly, then they over commit to an order and have to let the other party down.
- Artificial diversity – creating multiple versions of apps and logon routines and credentials and passwords can increase the effort to attack at a greater rate than it does to defend. Similar versions with subtle differences are much harder to work with than totally different version.

So the long-term aim is to convince crooks not to attack application X but applications Y and Z instead, but not on the basis that X is more secure (this is an expensive race between X, Y and Z) but because doing continued business selling credentials from app X is just too risky.

This type of business model attack was commonplace in the Pay TV industry where a well-funded and organised group of attackers would reverse engineer and clone Pay TV subscription cards in order to sell subscriptions at a cheaper price. After initial shortcomings of the industry, one core strategy was to roll out new security measures iteratively with the next release just before a big sports event, meaning that pirated cards then stopped working. This caused the customers to complain to their black market provider, and would leave enough time for the customer to have a change of heart and buy a genuine subscription before the start of the match.

The nature of an arms-race on a locked platform such as the X-Box or a mobile phone rather than on a PC is such that these business model defences are often very effective.

3.3.2 Non-standard crypto

At a technical level, it may pay long term to use these high-level principles of driving up cost of attack at a deep technical level as well as at a strategic level. For example, non-standard crypto can be included for which there are no easy reference implementations. This is a common strategy for military security, to keep both the key and the algorithm secret. *Non-standard crypto* is not the same as *home grown crypto*, the differencing being that the former consists of conservative modifications made to existing crypto algorithms, such as AES with extra rounds, or DES with different S-boxes, whereas the latter usually is a brand new algorithm thought up on the back of an envelope by a developer with no crypto experience, and are usually variants of the Viginère or Caesar ciphers whose designs are hundreds of years old and trivial to break for an expert. Cryptomathic, having specialist symmetric and asymmetric crypto cipher designers on its payroll is well placed to safely develop non-standard algorithm variations. In short, non-standard crypto is really easy to write, but hard to reverse-engineer and debug, especially when moving implementations between different languages.

The criminal community is actually rather short of good cryptography know-how (although it is growing), evidenced by the number of brand new viruses which still use very outdated algorithms such as the RC4 stream cipher. Malware authors seem reluctant to use code that they cannot copy/paste from somewhere else, and here non-standard crypto is the ultimate in obscurity.



3.3.3 Threat Monitoring and Intelligence

Phase three plans should consider scaling up and outsourcing of mobile security threat monitoring and intelligence. Services here are offered both by anti-virus companies who use their wide deployment base to monitor and collate statistics, and from smaller intelligence-focussed companies who put most of their stock in human intelligence (HUMINT in military speak). Cryptomathic proposes that the latter will remain a long-term viable solution, to infiltrate and keep watch upon the criminal economy and marketplace in order to gain a couple of weeks to a couple of months advance notice on how the capability of the market to provide services for new attacks develops.

In the case of payments card security, Cryptomathic has been involved in collaboratively monitoring and assessing intelligence output from human intelligence operatives to assess the rate at which known payment card security vulnerabilities such as *SDA card cloning* have permeated into the criminal economy, as opposed to these techniques being confined to experimenters and hobbyists. This way one can detect the difference between a future attack not seen deployed at scale, and an attack that is just about to scale up.

Such intelligence is invaluable in eking out a longer lifespan from security measures based on obscurity and economic attack, as the defences can be deployed at the right moment, and more importantly it can keep an eye on the economy for evidence of an unexpected surge of innovation which might threaten to push fraud above an acceptable level – buying extra time to deploy a larger set of counter measures.

So while high level statistical data on malware and threat may be useful to set overall departmental budgets for certain types of security at a high level in corporations, human intelligence and infiltration can and does yield genuinely useful information for iterative development of secure applications.



4 Cryptomathic Offerings

As a leading security vendor specialising in the financial sector, Cryptomathic has helped to secure Internet banking solutions for more than a decade. Having obtained a unique insight into banking requirements for authentication systems, Cryptomathic is able to offer a complete range of 2FA solutions.

Central to these are the Cryptomathic Authenticator and our complete range of PKI products. These proven products are available today.

4.1 The Cryptomathic Authenticator

The Cryptomathic Authenticator forms the cornerstone of our authentication offerings, as an authentication server designed specifically for banking applications. In contrast with other authentication systems, the Authenticator offers a unique combination of advantages:

- A **token-vendor independent**, modular architecture
- **Best-in-class security**, through use of **Hardware Security Modules** with custom firmware
- **Strong administrative controls**, especially for critical key management tasks
- **Simple integration** into existing systems
- A flexible, scalable **token management** system designed for consumer deployments
- **High performance and availability**, through redundancy and clustering
- **Tamper-evident** audit logging

By being token-vendor independent, Cryptomathic is able to offer the broadest range of options to our banking customers and enable them to negotiate the best possible prices on authentication devices. The flexibility to change authentication methods also ensures long-term value from the investment in our infrastructure, regardless of future attack and authentication trends.



For a banking authentication server, it is crucial that the management of secret information is secure, be it a token key, static password, or similar, that is shared with the customer but which must not be shared with any other individual. Clearly, a compromise of these authentication secrets would be disastrous for the bank. In addition, to ensure that customer actions cannot be repudiated, the bank requires the ability to demonstrate, to a high level of certainty, that such a compromise could not have occurred.

Such assurance can be achieved only with strong technical and procedural controls, both of which are fully supported. Central to this is our use of Hardware Security Modules (HSMs)—secure, tamper-proof hardware devices which have exclusive control of the server-side authentication secrets.

Managing the secure delivery and registration of authentication devices is a critical step in any 2FA deployment. The Cryptomathic Authenticator supports this process either through our own Token Manager server, or via third-party or in-house systems.

The product offers straightforward integration with most banking infrastructures. A remote administration client offers easy management of the system, which is usually clustered to ensure continuous availability.

Cryptomathic has unrivalled experience in developing HSM-based systems, and our R&D facility includes a world-class HSM programming team with unrivalled experience. Cryptomathic works with a wide range of HSM vendors, again offering our customers vendor independence.



4.2 PKI Solutions

PKI-based authentication solutions differ from OTP-based schemes in that a PC-connected (or software) device is used to manage the user's private key. Each approach has its own advantages and disadvantages, and either may be appropriate depending on the requirements of the business application. A PKI offering is therefore an essential part of any complete authentication package.

Cryptomathic has been delivering PKI solutions over 10 years. Our offering includes a full suite of PKI products:

- **Certificate Authority**, for issuing and managing certificates and revocation lists
- **OCSP Responder**, for real-time, on-line certificate status checks
- **Time-Stamping Authority**, for independent audit-keeping and non-repudiation
- **The Signer**, a central, server-based alternative to device-based user private keys
- **Cryptographic toolkits**, for certificate handling and application integration

Each of our PKI servers shares the operational and security advantages of the Authenticator: high security (including HSM support), tight procedural controls, scalability and redundancy through clustering, and tamper-evident audit logging. And since all our PKI products are standards-based, they can integrate out-of-the box with other PKI systems from other vendors.

4.3 Benefits of Working with Cryptomathic

Working with Cryptomathic on securing Web banking you will experience a number of technical and business benefits:

- **Comprehensive offerings in 2FA**—from tokens to PKI to consultancy services.
- **The most innovative, skilled and experienced company in banking 2FA**—with over 10 years experience of large-scale banking authentication projects.
- **The highest degree of flexibility on the market**—a complete range of authentication methods, not tied up with any token or HSM provider, and able to match any requirement for peak performance and availability.
- **The most secure solution**—designed for banks and other financial-sector institutions, using specialised Hardware Security Module code and the highest security standards including tamper-evident auditing and tight access control.
- **Prime contractor capability**—taking full responsibility for the coordination and integration of other suppliers, including device vendors and distribution bureaux.
- **A trusted technology provider**—we have successfully led comprehensive 2FA projects with high-street banks including Lloyds TSB (UK) and LuxTrust (Luxembourg).



5 About Cryptomathic

5.1 Company Background

With 25 years of experience, Cryptomathic is one of the world's leading providers of electronic security solutions. We specialise in commercial cryptography, and assist our customers in securing their businesses by providing best-of-breed security software products and technologies together with consultancy and education.

Our extensive expertise in the financial services industry has been achieved through investment in research and development and by providing customers worldwide with both product-based and tailor-made solutions. Our product portfolio ranges from cryptographic tools to large-scale server applications, such as the Authenticator for banking authentication, and CardInk, a data preparation product for card issuing.

Our customers include banking organisations, central banks, commercial banks, card bureaux and transaction processors, as well as other large corporations outside the financial sector. They are served through our offices in Canada, Denmark, Germany, the UK and USA.

25
YEARS



CRYPTOMATHIC
25th Anniversary 1986-2011