

Alex Smirnoff,
Oleg Makarov,
Glanc, Ltd



Addressing user authentication challenges for cryptocurrency exchange:

NEW THREAT MODEL AND IMPLEMENTATION CONSIDERATIONS

Objective

The threat model for cryptocurrency exchange is somewhat unique even for finance applications; two essential techniques that make anti-fraud efforts fruitful in traditional fintech, are **KYC** and the **capability to dispute a suspicious operation**. In the cryptocurrency exchange, these may turn totally non-reliable or even nonexistent since crypto currency operations are generally **anonymous and non-reversible**. Thus, we need more rigorous (and preventive) security as compared to traditional banks and payment systems¹. The traditional instruments we typically have at hand are obviously insufficient. We do not discard KYC completely, and we try to work around non-reversibility wherever possible. However, the purpose of this paper is to explore ways to surpass the intrinsic limitations of traditional methods, because the opposite approach of making case by case improvements is already getting enough public attention. A mindless combination of different authentication and recovery methods may merge into a cascade of failures¹ instead of increasing redundancy and reliability; thus we need a systemic approach.

Executive summary

There is no reliable way to achieve a reasonable level of security with **passwords** only¹¹ in the context of protecting high-value resources, yet passwords are the necessary first layer of protection (given the server-side credential storage is implemented in a secure manner¹¹¹).

To maintain a proper balance of security and usability, it is necessary to be flexible in protection requirements, in accordance with the monetary value at risk.

IF THERE IS NO SUBSTANTIAL VALUE, THERE IS NO NEED TO COMPLICATE THE USER'S EXPERIENCE WITH UNNECESSARY VERIFICATION STEPS.

Two-factor authentication should be mandatory, preferably in the form of **hardware OTP tokens** or hardware certificate tokens. The latter are notably more efficient if equipped with an on-device screen and a pin pad for digital signing of individual operations by the user within the device. For the period before these tokens are available for the user, we propose a transitional solution in the form of a one-time code sheet.

SMS and **email** are generally not considered as means of secure communication, mostly due to account ownership issues and social engineering problems; yet they could be used to deliver additional information about current operations to the customer. Email security could be significantly improved if end to end encryption is available (which usually is not due to the low adoption rate of the technology). **SMS as a second factor** may also be used as a **temporary low-security option** for newly registered accounts with a relatively small amount of funds which certainly do not exceed the cost of even most low-tech attacks and should be appropriately discarded afterward.

1 It would be unwise to completely ignore the NIST 800-63 Digital Identity Guidelines, yet those are even less relevant for our case as a high-level approach; however, it is still a valuable reference material for many technical aspects.

It is advised to implement **additional authorization** to perform high-risk operations (substantial funds transfer or changing access credentials), and in a case when a risky method is used for credentials management (like, changing the primary email address tied to an account while no secondary secure authorization method is defined).

IT IS ADVISABLE TO LIMIT POSSIBLE OPERATIONS WITH NEW CREDENTIALS FOR A GUARD PERIOD WHEN ALL CHANGES COULD BE REVERSED, AND OPERATIONS THAT CANNOT BE REVERSED ARE TEMPORARILY PROHIBITED.

Scope and prerequisites

We consider **3 types of authentication/authorization** to be within the scope of this memo:

1. **“MAIN”** user authentication, as at the beginning of user session;
2. **“RECOVERY”** methods to reinstate access if main credentials are lost or unavailable;
3. **“ADDITIONAL”** authorization for the pre-authenticated user to perform high-risk operations (like sending substantial funds out of the system or changing access credentials);

also, **one special scenario** which does not fall into one of these categories: **initial user registration process** (see the corresponding section for details).

We intentionally leave out of scope:

- In-depth analysis of user behavior to identify suspicious and automated activity;
- Endpoint and communication protection details (beyond common considerations);
- Session management, persistent cookies and related questions (those matters are well-covered by OWASP recommendations^{IV});
- Mobile applications and associated app-only security features like using a protected key store provided by mobile OS, device pinning, etc.;
- Out of band user verification specifics (it is likely to be an external service).

Authentication methods comparison

1	2	Threats and attack characteristics	Mitigation	Notes
Password	Main	<p>Online brute force attacks.</p> <p>COMPLEXITY: LOW</p> <div style="display: flex; align-items: center;"> <div style="width: 20px; height: 10px; background-color: black; margin-right: 5px;"></div> <div style="width: 20px; height: 10px; background-color: #ccc; margin-right: 5px; border: 1px solid #ccc;"></div> <div style="width: 20px; height: 10px; background-color: #fff; border: 1px solid #ccc;"></div> </div> <p>COST: LOW</p> <div style="display: flex; align-items: center;"> <div style="width: 20px; height: 10px; background-color: black; margin-right: 5px;"></div> <div style="width: 20px; height: 10px; background-color: #ccc; margin-right: 5px; border: 1px solid #ccc;"></div> <div style="width: 20px; height: 10px; background-color: #fff; border: 1px solid #ccc;"></div> </div> <p>RISK: HIGH</p> <div style="display: flex; align-items: center;"> <div style="width: 20px; height: 10px; background-color: red; margin-right: 5px;"></div> <div style="width: 20px; height: 10px; background-color: red; margin-right: 5px;"></div> <div style="width: 20px; height: 10px; background-color: red;"></div> </div>	Behavior analysis, rate limiters, password strength meters.	Some users would always be susceptible to this kind of attack. Password strength meters are unreliable and contradictory because there is no objective metric. Also, need to consider availability issues to make it sure legit users won't be blocked out of the system.
		<p>Offline brute force attacks.</p> <p>COMPLEXITY: HIGH</p> <div style="display: flex; align-items: center;"> <div style="width: 20px; height: 10px; background-color: black; margin-right: 5px;"></div> <div style="width: 20px; height: 10px; background-color: black; margin-right: 5px;"></div> <div style="width: 20px; height: 10px; background-color: black;"></div> </div> <p>COST: VARIES</p> <div style="display: flex; align-items: center;"> <div style="width: 20px; height: 10px; background-color: #ccc; margin-right: 5px; border: 1px solid #ccc;"></div> <div style="width: 20px; height: 10px; background-color: #ccc; margin-right: 5px; border: 1px solid #ccc;"></div> <div style="width: 20px; height: 10px; background-color: #ccc; border: 1px solid #ccc;"></div> </div> <p>RISK: MEDIUM</p> <div style="display: flex; align-items: center;"> <div style="width: 20px; height: 10px; background-color: yellow; margin-right: 5px;"></div> <div style="width: 20px; height: 10px; background-color: yellow; margin-right: 5px;"></div> <div style="width: 20px; height: 10px; background-color: #fff; border: 1px solid #ccc;"></div> </div>	Protecting credential storage, "hard" password-based key derivation functions.	Though "hashes leak" is a low probability situation which also indicates some serious breach, consequences may be catastrophic at scale, so it is necessary to implement adequate protection.
		<p>Compromised endpoints (offline attacks) note: this part applies to saved passwords only, everything else is related to the session management (cookies and cache data).</p> <p>COMPLEXITY: MEDIUM</p> <div style="display: flex; align-items: center;"> <div style="width: 20px; height: 10px; background-color: black; margin-right: 5px;"></div> <div style="width: 20px; height: 10px; background-color: black; margin-right: 5px;"></div> <div style="width: 20px; height: 10px; background-color: #fff; border: 1px solid #ccc;"></div> </div> <p>COST: VARIES</p> <div style="display: flex; align-items: center;"> <div style="width: 20px; height: 10px; background-color: #ccc; margin-right: 5px; border: 1px solid #ccc;"></div> <div style="width: 20px; height: 10px; background-color: #ccc; margin-right: 5px; border: 1px solid #ccc;"></div> <div style="width: 20px; height: 10px; background-color: #ccc; border: 1px solid #ccc;"></div> </div> <p>RISK: HIGH</p> <div style="display: flex; align-items: center;"> <div style="width: 20px; height: 10px; background-color: red; margin-right: 5px;"></div> <div style="width: 20px; height: 10px; background-color: red; margin-right: 5px;"></div> <div style="width: 20px; height: 10px; background-color: red;"></div> </div>	Out of scope – user's responsibility.	We cannot deny users the possibility to save passwords, and attempts to do so would do more harm than good.
		<p>Compromised endpoints (online, a.k.a. synchronous attacks: keyloggers and direct session intervention).</p> <p>COMPLEXITY: MEDIUM</p> <div style="display: flex; align-items: center;"> <div style="width: 20px; height: 10px; background-color: black; margin-right: 5px;"></div> <div style="width: 20px; height: 10px; background-color: black; margin-right: 5px;"></div> <div style="width: 20px; height: 10px; background-color: #fff; border: 1px solid #ccc;"></div> </div> <p>COST: VARIES</p> <div style="display: flex; align-items: center;"> <div style="width: 20px; height: 10px; background-color: #ccc; margin-right: 5px; border: 1px solid #ccc;"></div> <div style="width: 20px; height: 10px; background-color: #ccc; margin-right: 5px; border: 1px solid #ccc;"></div> <div style="width: 20px; height: 10px; background-color: #ccc; border: 1px solid #ccc;"></div> </div> <p>RISK: MEDIUM</p> <div style="display: flex; align-items: center;"> <div style="width: 20px; height: 10px; background-color: yellow; margin-right: 5px;"></div> <div style="width: 20px; height: 10px; background-color: yellow; margin-right: 5px;"></div> <div style="width: 20px; height: 10px; background-color: #fff; border: 1px solid #ccc;"></div> </div>	Out of scope – user's responsibility.	


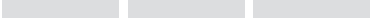


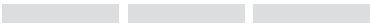
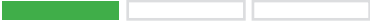



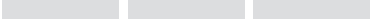
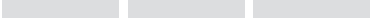

Authentication methods comparison

	1	2	Threats and attack characteristics	Mitigation	Notes
Password	Main	<p>Lost/stolen credentials on physical media.</p> <p>COMPLEXITY: LOW</p> <div style="display: flex; align-items: center;"> <div style="width: 20px; height: 10px; background-color: black; margin-right: 5px;"></div> <div style="width: 40px; height: 10px; border: 1px solid gray; margin-right: 5px;"></div> <div style="width: 40px; height: 10px; border: 1px solid gray;"></div> </div> <p>COST: VARIES</p> <div style="display: flex; align-items: center;"> <div style="width: 20px; height: 10px; background-color: gray; margin-right: 5px;"></div> <div style="width: 40px; height: 10px; background-color: gray; margin-right: 5px;"></div> <div style="width: 40px; height: 10px; background-color: gray;"></div> </div> <p>RISK: MEDIUM</p> <div style="display: flex; align-items: center;"> <div style="width: 20px; height: 10px; background-color: yellow; margin-right: 5px;"></div> <div style="width: 40px; height: 10px; background-color: yellow; margin-right: 5px;"></div> <div style="width: 40px; height: 10px; border: 1px solid gray;"></div> </div>	Out of scope – user’s responsibility.	Stronger passwords are typically harder to memorize, and certainly, some of those are to be written down or saved in text files on random places.	
		<p>Phishing attacks.</p> <p>COMPLEXITY: LOW</p> <div style="display: flex; align-items: center;"> <div style="width: 20px; height: 10px; background-color: black; margin-right: 5px;"></div> <div style="width: 40px; height: 10px; border: 1px solid gray; margin-right: 5px;"></div> <div style="width: 40px; height: 10px; border: 1px solid gray;"></div> </div> <p>COST: VARIES</p> <div style="display: flex; align-items: center;"> <div style="width: 20px; height: 10px; background-color: gray; margin-right: 5px;"></div> <div style="width: 40px; height: 10px; background-color: gray; margin-right: 5px;"></div> <div style="width: 40px; height: 10px; background-color: gray;"></div> </div> <p>RISK: HIGH</p> <div style="display: flex; align-items: center;"> <div style="width: 20px; height: 10px; background-color: red; margin-right: 5px;"></div> <div style="width: 40px; height: 10px; background-color: red; margin-right: 5px;"></div> <div style="width: 40px; height: 10px; background-color: red;"></div> </div>	Limit legitimate interactions that would involve following links sent by email; easily recognizable URLs, EV certificates. ^{1v}	If there is some action required from the user side, we should never ask to “follow this link and log in to complete the operation”. All messages of this type should be delivered in-system (like, log in from the previously bookmarked link or a manually typed URL and then do whatever is needed).	
Recovery codes	Recovery	<p>Compromised endpoints (offline attacks).</p> <p>COMPLEXITY: MEDIUM</p> <div style="display: flex; align-items: center;"> <div style="width: 20px; height: 10px; background-color: black; margin-right: 5px;"></div> <div style="width: 40px; height: 10px; background-color: black; margin-right: 5px;"></div> <div style="width: 40px; height: 10px; border: 1px solid gray;"></div> </div> <p>COST: VARIES</p> <div style="display: flex; align-items: center;"> <div style="width: 20px; height: 10px; background-color: gray; margin-right: 5px;"></div> <div style="width: 40px; height: 10px; background-color: gray; margin-right: 5px;"></div> <div style="width: 40px; height: 10px; background-color: gray;"></div> </div> <p>RISK: MEDIUM</p> <div style="display: flex; align-items: center;"> <div style="width: 20px; height: 10px; background-color: yellow; margin-right: 5px;"></div> <div style="width: 40px; height: 10px; background-color: yellow; margin-right: 5px;"></div> <div style="width: 40px; height: 10px; border: 1px solid gray;"></div> </div>	Make sure recovery codes page would not be stored in browser cache; download should be on demand and not unsolicited.	Once information is sent out to the user, you cannot control it.	
		<p>Compromised endpoints (online attacks).</p> <p>COMPLEXITY: MEDIUM</p> <div style="display: flex; align-items: center;"> <div style="width: 20px; height: 10px; background-color: black; margin-right: 5px;"></div> <div style="width: 40px; height: 10px; background-color: black; margin-right: 5px;"></div> <div style="width: 40px; height: 10px; border: 1px solid gray;"></div> </div> <p>COST: VARIES</p> <div style="display: flex; align-items: center;"> <div style="width: 20px; height: 10px; background-color: gray; margin-right: 5px;"></div> <div style="width: 40px; height: 10px; background-color: gray; margin-right: 5px;"></div> <div style="width: 40px; height: 10px; background-color: gray;"></div> </div> <p>RISK: MEDIUM</p> <div style="display: flex; align-items: center;"> <div style="width: 20px; height: 10px; background-color: yellow; margin-right: 5px;"></div> <div style="width: 40px; height: 10px; background-color: yellow; margin-right: 5px;"></div> <div style="width: 40px; height: 10px; border: 1px solid gray;"></div> </div>	Reduce attack window by making it clear and visible which recovery codes user has enabled and when.	Also, see “credentials management” section below.	

1 Unfortunately, email marketing tools like MailChimp made “nonsense URLs” a new norm (the HTML letters make it even easier, and do not require a 3rd-party – in HTML you may plainly hide a URL behind any word (looks like everything is designed to confuse a user)). And EV certificates are obviously dying out. Anyway, there is no systemic solution to the phishing problem and never was.


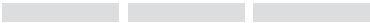


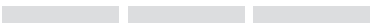







Authentication methods comparison

1 – Type and description; 2 – Application

1	2	Threats and attack characteristics	Mitigation	Notes
Recovery codes	Recovery	Lost/stolen credentials on physical media. COMPLEXITY: LOW  COST: VARIES  RISK: MEDIUM 	Out of scope – user’s responsibility.	More likely lost than stolen.
	Recovery	Cryptographically insecure (predictable) recovery codes. COMPLEXITY: MEDIUM  COST: VARIES  RISK: LOW 		
Control questions	Recovery	Online brute force attacks, social engineering attacks, data mining attacks. COMPLEXITY: LOW  COST: LOW  RISK: HIGH 	Recovery questions are evil, don’t use them.	Truly random huge answers would be not any worse than passwords, but the existence of recovery questions as we know it certainly encourages users to dangerous behavior.
Physical identification (local)	Recovery	Identity theft, third party risks. COMPLEXITY: VARIES  COST: VARIES  RISK: MEDIUM 	If done properly, identifying a user by “real world” credentials is reliable enough, but generally requires trusted third party like a notary.	Might be very undesirable for many users because of privacy issues (and availability too, people travel). Also, it is a “slow” method and requires a trusted global partner.

Authentication methods comparison

1 – Type and description; 2 – Application








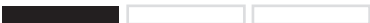




1	2	Threats and attack characteristics	Mitigation	Notes
Physical identification (remote)	Recovery	Identity theft, third party risks. COMPLEXITY: VARIES  COST: VARIES  RISK: MEDIUM 	Works better if a user has “deposited” his credentials (say, live photo with ID document) in advance.	If no “deposited” identity exists, making “web camera quality” fake documents is trivial. Also, the recent shift in the video processing technology makes “online video” even less reliable.
	Recovery	Compromised endpoints. COMPLEXITY: MEDIUM  COST: VARIES  RISK: HIGH 		
	Recovery	In-transit interception (transit server or communication channel compromised). COMPLEXITY: HIGH  COST: HIGH  RISK: MEDIUM 		
Recovery links sent via email	Recovery	Mail server compromised. COMPLEXITY: HIGH  COST: HIGH  RISK: MEDIUM 	End to end email encryption, make sure recovery links a) expire after a brief period; b) cannot be reused.	Today most users rely on cloud email providers which are usually reasonably secure, yet we should include this risk in the threat model because environment may vary.

Authentication methods comparison

1 – Type and description; 2 – Application


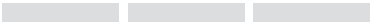





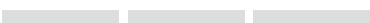

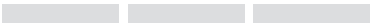
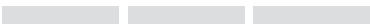




1	2	Threats and attack characteristics	Mitigation	Notes
Recovery links sent via email	Recovery	<p>Mail account compromised (scenarios include: hacked accounts; accounts on expired domains; loss of ownership when using work account; cloud account takeover via social engineering).</p> <p>COMPLEXITY: LOW </p> <p>COST: LOW </p> <p>RISK: HIGH </p>	End to end email encryption, make sure recovery links a) expire after a brief period; b) cannot be reused.	End to end email encryption is not widely adopted.
		<p>Mail account ownership expiration (expiring domains, work accounts, etc.).</p> <p>COMPLEXITY: LOW </p> <p>COST: LOW </p> <p>RISK: MEDIUM </p>	End to end email encryption.	End to end email encryption is not widely adopted.
		<p>Cryptographically insecure (predictable) recovery links.</p> <p>COMPLEXITY: MEDIUM </p> <p>COST: VARIES </p> <p>RISK: LOW </p>	Implementation review.	Set to low risk because it is easy to mitigate.
		<p>Vulnerabilities in the server-side recovery application.</p> <p>COMPLEXITY: VARIES </p> <p>COST: LOW </p> <p>RISK: MEDIUM </p>	Implementation review, source code audit.	

Authentication methods comparison

1	2	Threats and attack characteristics	Mitigation	Notes
SMS codes and links	Main, Recovery, Additional	Compromised mobile devices. COMPLEXITY: VARIES  COST: MEDIUM  RISK: MEDIUM 	Out of scope – user’s responsibility.	This type of risk could be easily mitigated by user.
		Phone number ownership loss (expiring contracts). COMPLEXITY: LOW  COST: LOW  RISK: MEDIUM 	Could be partially mitigated if a partner MNO supports ICCID or IMSI fingerprinting.	SIM card fingerprinting is available only if local MNO supports it and is unsuitable for worldwide operations.
		Phone number hijacking (social engineering or access to the MNO facilities). COMPLEXITY: MEDIUM  COST: LOW  RISK: HIGH 	May be a partially mitigated if a partner MNO supports ICCID or IMSI fingerprinting.	SIM card fingerprinting is available only if local MNO supports it and is unsuitable for worldwide operations.
		SMS interception via SS7 signaling attacks. COMPLEXITY: MEDIUM  COST: MEDIUM  RISK: MEDIUM 	Out of scope.	

Authentication methods comparison

1 – Type and description; 2 – Application

1	2	Threats and attack characteristics	Mitigation	Notes
SMS codes and links	Main, Recovery, Additional	SMS interception via the sending provider. COMPLEXITY: HIGH  COST: VARIES  RISK: MEDIUM 	Limiting lifespan of links and codes and proper deactivation is vital to make sure past links cannot be used or re-used.	Could be “low”, but actually happens.
		SMS interception via radio networks (IMSI catchers). COMPLEXITY: HIGH  COST: HIGH  RISK: LOW 		
Hardware OTP tokens	Main, Additional	Compromised endpoints (online attacks). COMPLEXITY: MEDIUM  COST: VARIES  RISK: MEDIUM 	Out of scope.	Hardware OTP tokens almost completely mitigate the possibility of offline attacks.
		Lost or stolen tokens. COMPLEXITY: VARIES  COST: VARIES  RISK: MEDIUM 		
		Key seed lifecycle management issues. COMPLEXITY: HIGH  COST: HIGH  RISK: LOW 		

Authentication methods comparison

1 – Type and description; 2 – Application


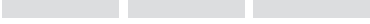




	1	2			
Software OTP tokens	Hardware OTP tokens	Main, Additional	<p>Server-side application vulnerabilities.</p> <p>COMPLEXITY: VARIES</p> <p>■■■■■ ■■■■■ ■■■■■</p> <p>COST: LOW</p> <p>■■■■■ ■■■■■ ■■■■■</p> <p>RISK: MEDIUM</p> <p>■■■■■ ■■■■■ ■■■■■</p>	Implementation review, source code audit.	Need to pay attention to server-side credentials storage, because current OTP technologies require key material that could be reversed to token seeds to be stored on the authenticating side.
	Software OTP tokens	Main, Additional	<p>Compromised endpoints (online attacks).</p> <p>COMPLEXITY: MEDIUM</p> <p>■■■■■ ■■■■■ ■■■■■</p> <p>COST: VARIES</p> <p>■■■■■ ■■■■■ ■■■■■</p> <p>RISK: MEDIUM</p> <p>■■■■■ ■■■■■ ■■■■■</p>	Out of scope.	Software OTP tokens transfer the offline attack risks to the authenticator device.
			<p>Compromised authenticator devices (offline attacks).</p> <p>COMPLEXITY: VARIES</p> <p>■■■■■ ■■■■■ ■■■■■</p> <p>COST: VARIES</p> <p>■■■■■ ■■■■■ ■■■■■</p> <p>RISK: HIGH</p> <p>■■■■■ ■■■■■ ■■■■■</p>	User education about mobile security (a software token is likely to reside on a mobile phone).	
			<p>Lost or stolen authenticator devices.</p> <p>COMPLEXITY: VARIES</p> <p>■■■■■ ■■■■■ ■■■■■</p> <p>COST: VARIES</p> <p>■■■■■ ■■■■■ ■■■■■</p> <p>RISK: HIGH</p> <p>■■■■■ ■■■■■ ■■■■■</p>	On-device authentication and encryption. Even if OTP is set, a password should still be required; set a guard period after the password recovery; use PIN-protected tokens.	Phones are more likely to be lost or stolen than dedicated authentication tokens.
			<p>Server-side application vulnerabilities.</p> <p>COMPLEXITY: VARIES</p> <p>■■■■■ ■■■■■ ■■■■■</p> <p>COST: LOW</p> <p>■■■■■ ■■■■■ ■■■■■</p> <p>RISK: MEDIUM</p> <p>■■■■■ ■■■■■ ■■■■■</p>	Implementation review, source code audit.	Need to pay attention to server-side credentials storage, because current OTP technologies require key material that could be reversed to token seeds to be stored on the authenticating side.

Authentication methods comparison

		1	2	Notes
X.509 certificates (hardware storage)	Main, Additional	<p>Compromised endpoints (online attacks).</p> <p>COMPLEXITY: MEDIUM</p> <p>COST: VARIES</p> <p>RISK: MEDIUM</p>	<p>Even if certificate is used, a password should still be required; set a guard period after the password recovery; use PIN-protected tokens and tokens with on-device screens.</p> <p>If a token device does not have hardware buttons to authorize its operation, it is essential to keep it unplugged when not in use.</p>	<p>Using a certificate token itself to sign operations is the only reliable method when the user's computer cannot be trusted. However, it is costly and complicated on the user's side (it requires token hardware with a display).</p>
		<p>Lost or stolen tokens.</p> <p>COMPLEXITY: VARIES</p> <p>COST: VARIES</p> <p>RISK: MEDIUM</p>	<p>Even if certificate is used, a password should still be required; set a guard period after the password recovery; use PIN-protected tokens.</p>	
		<p>Server-side application vulnerabilities.</p> <p>COMPLEXITY: VARIES</p> <p>COST: LOW</p> <p>RISK: MEDIUM</p>	<p>Implementation review, source code audit.</p>	
X.509 certificates (software storage)	Main, Additional	<p>Compromised endpoints (online attacks).</p> <p>COMPLEXITY: MEDIUM</p> <p>COST: VARIES</p> <p>RISK: MEDIUM</p>		

Authentication methods comparison

1 – Type and description; 2 – Application

1	2	Threats and attack characteristics	Mitigation	Notes
X.509 certificates (software storage)	Main, Additional	Compromised endpoints (offline attacks). COMPLEXITY: MEDIUM  COST: VARIES  RISK: MEDIUM 	Passphrase protection of software certificate storage.	This risk is generally higher than a risk of hardware tokens being stolen.
		Server-side application vulnerabilities. COMPLEXITY: VARIES  COST: LOW  RISK: MEDIUM 	Implementation review, source code audit.	

Secure credentials storage

A thorough architecture review is required to determine that even if the main application is compromised, the risk to credential storage is minimal (that applies not just to passwords, for which the impact, given OTP use is to be mandatory, is mostly reputational, but to other key material as well). For passwords it is always recommended to use **yescrypt** or similar PBKDF implementation that addresses the attack path in multiple points, simultaneously making it hard to retrieve hash data and to attack it once it is retrieved.

High-risk operations

It is advised to implement additional authorization to perform high-risk operations (substantial funds transfer or changing access credentials), and in case a risky method is used for credentials management (like changing the primary email address tied to account while no secondary secure authorization method is defined).

Additional authorization may include methods mentioned in this document or anything out of its scope, yet deemed feasible (obtaining a biometric voiceprint via a phone call, for example).

Guard period and funds lock-in

In the context of resisting credit card fraud the following practice is very successful. Funds that were deposited to the account via a credit card payment remain unavailable for withdrawal / external transfers for a specific period, typically 5-10 days. We advise extending this type of lock-in to all cases of account access when the account access authenticity cannot be immediately verified, as described below in “**Credentials management scenarios**” section.

Event notifications

Message delivery methods that were deemed insufficiently secure to be used to send one-time codes for authentication or authorization purposes, like SMS and email, could still be useful to keep the user aware of operations being performed on his account. However, the best known way is to sign individual operations with hardware electronic signature token that is also capable of displaying detailed information about what exactly the signed operation is. When altering notification methods, the change should not be in effect immediately: the old method should remain active (and clearly marked for retention in all notifications) at least for 5 days, giving the user an option to contact the customer support in case of suspected fraud.

Credentials management scenarios

A user account may be in one of the following states:

- 1. INSECURE** – after initial registration, before the user has any **secure credentials** associated with account: there is no substantial amount of funds at user’s disposal (less than \$100-200, as for current estimation we derived from the probable corresponding attack cost) and no strong authentication (see **secure credentials** below) methods are defined. This state allows “relaxed” access recovery procedure if there is nothing of value to protect. Before any funds are transferred to the system, it is required for the user to define one or more type of **secure credentials** to maintain further account access (NB: scenarios may vary depending on business requirements, and this part requires special attention to ensure there are no loopholes or race conditions).
- 2. SECURE** – the user has **secure credentials** associated with the account. It is mandatory for all accounts exceeding the threshold limit mentioned above (Google Authenticator, Authly and alike on mobile phones, most probably). For accounts with the balance exceeding some higher limit, use of a hardware OTP token should be encouraged.
- 3. INACTIVE/DISABLED.**

An account can also be optionally (if it does not contradict the privacy requirements) **verified** if the user provides a government issued id, which could later be used for access recovery (see corresponding table entries for caveats). **Guard period** could also be imposed on certain operations (either funds transfer or both funds transfer and credentials change).

We consider credentials (and user contact methods that may be relevant for access recovery) to be one of three types:

1. **SECURE** – any type of credentials that could be reasonably presumed to be secure if appropriately handled on the client side. Secure credentials include any type of a second factor, such as software or hardware tokens, or one-time codes used for access recovery; PGP-encrypted email; it does not include reusable passwords, because passwords are susceptible to numerous threats and once compromised could be used for further access indefinitely. Caveat: the term “secure” should not be taken literally as an absolute characteristic; there are attack scenarios (see table above) that are still possible when using this type of credentials, just the residual risk is lower.
2. **INSECURE** – any type of credentials that either likely to be partially outside of user’s control (plaintext email, phone numbers) or could be easily reused once compromised (passwords).
3. **INTRINSIC** – user identification which is presumed to be protected from arbitrary manipulation by “real world” constraints; this includes government-issued ID’s, biometric data and so on. Using this type of identification may have privacy drawbacks, and requires a trusted, established process which is likely to be slow. This method requires human support intervention, and thus we do not cover it in details. It is recommended, however, that the guard period would be in effect after using this method as well.

GENERAL RULE: YOU NEED ONE “REGULAR” (“INSECURE”) AND AT LEAST ONE OF “SECURE” CREDENTIALS SIMULTANEOUSLY TO CHANGE A THIRD ONE (REGARDLESS IF IT IS “SECURE” OR NOT) INSTANTLY BYPASSING THE “GUARD PERIOD” IMPOSED OTHERWISE.

Some kind of “secure” credentials is required to be set up to get full account access.

WE STRONGLY ADVISE AGAINST TREATING A COMBINATION OF INSECURE CREDENTIALS AS A SECURE METHOD.

Say, two-part verification code consisting of a link sent via e-mail and a PIN sent to a phone is not reliable, because if the phone is compromised, it is very likely that email would be taken over as well. The security is not cumulative! Pay attention to the real world dependencies between the auth methods (outside of the secure system in question).

For example:

If you have presented a password and an OTP token (the same combination which is enough to log in), you may change email or phone.

If you have email access and OTP (or backup codes), you may request instant password recovery (that’s why you probably do not want to keep both in a single place, like authenticator app and mobile email account being on the same smartphone);

If you have email access but no OTP nor backup codes, then the password recovery should be burdened with a guard period.

If you have lost both access to your email and your password, but have your OTP code, you need to contact the customer support first, and the account should be, after appropriate identity verification, limited with guard period.

All those operations should be accompanied by email notification and SMS notification if a phone number is set.

Adding recovery methods, and changing authentication methods should be possible with additional authorization only. Customers are encouraged to routinely review authentication and recovery methods that are enabled for their accounts. (it is important to note that typical user interfaces are not review-friendly)

Transitional one-time codes

This document defines several occasions when it is mandatory for the customer to use **secure credentials**. It might be reasonable to provide hardware OTP tokens to high-value customers for free; and the rest is encouraged to use their own hardware or software tokens, the latter being supported by all major mobile platforms. It is possible, however, that a user does not have the mobile device he wishes to use as OTP token at hand at the moment of the registration, and delivering the hardware token takes time, so that won't be available right away. We need to provide temporary access to those users until they get their permanent authentication device, and we cannot rely on **insecure credentials** even for a brief period.

As a **transitional** solution, we suggest using a **code sheet**, consisting of 10 numbered 6-digit one-time codes that could be used before a permanent method is established, along with the recommendation to print it or write it down (not to save on a computer! Though we probably should expect it to happen). Codes are relatively short because they are to be typed frequently and possibly to be written down by hand (see scenario examples below for details about backup access codes); however, they are totally unsuitable for long-term use for obvious reasons; one of those is that the sheet is likely to reside in a wallet where it wears out quickly. Making a photo might be a not that bad idea too, but the user should be warned about risks associated with cloud photo backups. While using this code sheet customer gets continuously warned that he should switch to authenticator app (or a hardware token if he was issued with one, depending on account balance); and that after using the 10th code it won't be possible to log in anymore. The sessions after the 5th code should only provide the ability to add a new OTP token.

The new user registration process

Current recommendations:

- If immediate email access verification is not possible in a direct way on early stage (though should be completed afterward), the email address is to be entered twice to avoid typos (as it is usually done with passwords).
- Upon the completion of the registration, a user is advised to enable full account access by defining a secure second factor. If he has an OTP token, he may wish to use it as the second factor. If he wishes to use his present mobile device the necessary software

should be provided. Otherwise, he must write down a recovery code and keep it in a safe place until he acquires the hardware he needs. The current session is considered to be secure and provides full access to credential management until the user logs out or is logged out automatically.

- If a user makes an initial payment with a credit card, a guard period is imposed over the funds regardless of the amount.
- If a user made an initial payment with a credit card, the same credit card could be used to reinstate account access after a possible registration glitch if it has 3DSecure support or equivalent.

Acknowledgements

We would like to express our gratitude to Roman Akopov (Bixtrim), Eugene Panferov (independent expert and valuable author at ITHipster), Oleksandr Nikitin (Grand Central) and Andrey Kovalev (ThreatMetrix) for critical discussion and suggestions on this paper.

Annex A: Example scenarios

Let's consider 3 different threat models for accounts, depending on the funds on the account balance: 1) if peak total amount is less than \$150, we consider the overall situation to be of low risk, since any targeted attack would be impractical; such accounts may lack secure credentials, and that's fine. 2) if account value is in the range \$150-\$10000, then using secure software-based OTP is recommended. 3) for accounts exceeding \$10000 balance, it might be practical to provide a hardware OTP token. If we cannot immediately provide the user with an appropriate OTP token, a transitional one-time code system (see the corresponding section above) should be used as a temporary solution.

Scenarios that imply a change of primary notification/contact methods (phone, email) should not rely on the availability of old contact method since the motive for a change could be that user no longer has control over it because it has been deactivated, compromised or expired. We also removed email and phone reset scenarios if the password is not known: such a situation is exceptional enough to be handled by customer support on a per-case basis.

Assumptions:

- There is no user id, a user is identified by email. All scenarios may be easily adjusted to use a separate user id if needed, or to use social network profile as a source of contact data (email and phone).
- Phone number with SMS capability is a mandatory attribute (which could be somewhat disputable for a privacy-focused service).

1. NEW USER REGISTRATION

Initial state: a user is not logged in, an account does not exist;

Final state: an account is registered, a user is logged in;

Entry fields: phone number, email (two times), password (two times), captcha (optional), SMS code.

- a) Log in (see below "Insecure login" and "Secure login") or register – a user is asked to provide (as a minimum) phone number (verification code is sent and required to be entered), email address (twice to avoid typos) and password (twice to avoid typos). Email and phone should be verified later (by visiting a link sent by email and by entering a verification code sent via SMS), but this affects nothing except frequent reminders to do so.
- b) Optionally, add OTP token as defined below. A user may skip this step.

2. INSECURE LOGIN

Initial state: a user is not logged in, an account exists in the insecure state;

Final state: an account is insecure, a user is logged in;

Entry fields: email, password, SMS code, captcha (optional).

A user enters email, password and SMS code and is permitted to enter an insecure session. SMS and email notifications are being sent about login attempt (Unsuccessful, mandatory. Successful, optional. Mind possible flood situations, so event aggregation is required). If any other active user session exists, it is terminated forcibly upon successful login.

3. SECURE LOGIN

Initial state: a user is not logged in, an account exists in secure state;

Final state: an account is secure, a user is logged in;

Entry fields: email, password, second factor, captcha (optional).

A user enters login, email and second factor and gets full account access. SMS and email cannot be used as a second factor. SMS and email notifications are mandatory for unsuccessful logins and optional for successful ones (same caveats as previous case). If any other active user session exists, it is terminated forcibly on successful login.

4. PASSWORD CHANGE

Initial state: a user is logged in, an account exists and may be secure or not;

Final state: a user is logged in, the password is changed;

Entry fields: old password, a new password (2 times), second factor (optional).

For insecure accounts, SMS code may be used as a second factor, or the second factor could be omitted completely (depends on business requirements). If a second factor exists and the account is secure, it should be used for this operation. SMS and email notifications are issued in both cases.

5. PASSWORD RESET

Initial state: a user is not logged in, an account exists and may be secure or not, a user has email access, the password is not known to a user;

Entry fields: email, captcha; after the link is opened in the browser – a second factor and a new password (2 times);

Final state: a user is logged in, the password is changed.

For insecure accounts SMS code is used as a second factor; for secure accounts, it should be an OTP token or a recovery code. A link is sent by email (not SMS!) combined with the second factor is enough to perform the change.

NB: THE LINK SENT BY EMAIL SHOULD CONTAIN NO ENTRY FORMS FOR SENSITIVE INFORMATION, JUST A BUTTON TO CONFIRM ACTION AND, POSSIBLY, CAPTCHA!

6. EMAIL ADDRESS CHANGE/RESET

Initial state: a user is logged in, an account exists and may be secure or not;

Final state: a user is logged in, the email address is changed;

Entry fields: new email, password, second factor, SMS code.

If the account is not secure, an SMS code is enough to perform the change, if it is, both SMS code and second factor are required. Password should be re-entered as well to avoid session hijack situations. Email notifications are being sent to both new and old address for next 5 days, and a link sent to the old email address could be used to either confirm the operation or temporarily block account access (support intervention is required to resolve the dispute afterward). A link is also sent to the new email to verify it is correct.

Attacks to consider:

- a) An attacker in temporary possession of user's account credentials, possibly including OTP, tries to take over the account quickly.
- b) An attacker takes over the victim's email (and, possibly, phone), and it cannot be easily recovered; thus user needs to change the email address.

Disabling the account temporary to prevent abuse is an acceptable inconvenience in all those cases.

7. PHONE NUMBER CHANGE/RESET

Initial state: a user is logged in, an account exists and may be secure or not;

Final state: a user is logged in, the phone is changed;

Entry fields: new phone, second factor (if one exists), SMS code to confirm the change.

This scenario is almost similar to one for the email address change/reset. If a second factor is not defined and the account is not secure, a link sent by email and a confirmation code are enough to perform the change. If second factor exists and account is secure, it should be used for this operation. The password should be re-entered as well to avoid session hijack situations. Phone notifications are being sent to both new and old phone for next 5 days, and a link sent to the old phone could be used to either confirm the operation or temporarily block the account access (support intervention is required to resolve the dispute afterward). A link is also sent to the new email to verify it is correct.

Attacks to consider:

- a) An attacker in temporary possession of user's account credentials, possibly including OTP, tries to take over the account quickly.
- b) An attacker takes over the victim's phone (and, possibly, email) and it cannot be easily recovered; thus the user needs to change the phone number.

Disabling the account temporarily to prevent abuse is an acceptable inconvenience in all those cases.

8. ADDING AN OTP TOKEN

Initial state: a user is logged in, an account exists and may be secure or not;

Final state: a user is logged in, a second factor is added, the account status is assigned as secure;

Entry fields: password, second factor to test.

Notifications are sent via email and SMS. We have two types of backup access codes that could be used in absence of "proper" OTP authenticator:

- a) Temporary access codes are generated if the user is forced to use OTP, but has no authenticator at hand or does not wish one right now. Such codes are short (6-digits) because the user needs to type it in frequently, the total number is limited to 10, and after 5 are already used a notification should appear with a reminder to switch to "proper" OTP soon. When the user adds "proper" OTP token, all temporary codes are to be instantly disabled.
- b) Permanent backup access codes are used to restore account access if OTP token is lost, stolen, damaged, malfunctioning or compromised. Such codes may be considerably longer and alphanumeric, and creation of those codes is considered a mandatory part of OTP token provisioning.

There are some possible concerns if it is a good idea to allow more than one OTP token at a time. Since we cannot prevent a user from using one seed with multiple tokens, it is better to give him a possibility to do it more conveniently, having individual control over authenticator devices. Since notifications are being sent out if a user adds a new OTP device, it is unlikely that a new token could be added by a malicious actor without drawing the user's attention. Signing individual operations (using "interactive" tokens) is to be revised in future versions.

9. REMOVING AN OTP TOKEN

Initial state: a user is logged in, an account exists and is secure;

Final state: a user is logged in, the second factor is removed, a user is prompted to add a new OTP token as defined above;

Entry fields: password.

The user should be able to log in using a recovery code first. Notifications are being sent via email and SMS. If a user has no OTP tokens left (other than recovery codes), he is prompted to set up a new one (see “Adding OTP token”).

10. OTP TOKEN RESET IN THE ABSENCE OF BACKUP CODES

Initial state: a user is not logged in, an account exists and is secure;

Final state: a user is logged in, the second factor is removed, a user is prompted to add a new OTP token as defined above, the account is being put on a 5-day guard period;

Entry fields: email, password, SMS confirmation code.

A recovery link is sent by email and the second confirmation code is sent via SMS. The guard period is still mandatory.

11. HANDLING INCOMPLETE CREDENTIALS CHANGE SCENARIOS

Until changes are fully confirmed (confirmation sent, the new code was entered, etc) the new settings (if to be added) should not be in effect and old credentials (if to be removed) should be treated as valid. If a user makes a purchase, but fails to complete creation of secure credentials, the purchase (and the beginning of the corresponding guard period, if applicable) should be deferred until the user successfully enables secure credentials. During the guard period changing the phone number and the email address should not be permitted.

12. EXCEEDING THE BALANCE THRESHOLD FOR THE INSECURE ACCOUNT (AS MENTIONED ABOVE, \$100-200)

If account balance exceeds the threshold for any reason, the user is directed to the OTP token creation procedure which he cannot cancel (the account cannot be used normally until an OTP is provisioned).

-
- I Mohammad Ghasemisharif, Amrutha Ramesh, Stephen Checkoway, Chris Kanich, Jason Polakis
O Single Sign-Off, Where Art Thou? An Empirical Analysis of Single Sign-On Account Hijacking and Session Management on the Web
<https://www.cs.uic.edu/~polakis/papers/sso-usenix18.pdf>
 - II Denise Ranghetti Pilar, Antonio Jaeger, Carlos F. A. Gomes, Lilian Milnitsky Stein
Passwords Usage and Human Memory Limitations: A Survey across Age and Educational Background
<http://journals.plos.org/plosone/article?id=10.1371/journal.pone.0051067>
 - III Solar Designer
yescrypt: large-scale password hashing
<http://www.openwall.com/presentations/BSidesLjubljana2017-Yescript-Large-scale-Password-Hashing/>
 - IV OWASP Session Management Cheat Sheet
https://www.owasp.org/index.php/Session_Management_Cheat_Sheet
 - V Troy Hunt
Extended Validation Certificates are Dead
<https://www.troyhunt.com/extended-validation-certificates-are-dead/>
 - VI Alex Smirnov
A Hacker's Guide to Not Get Hacked
<http://ithipster.com/blog/72.html>



+359878830030



arkenoi@gmail.com



facebook.com/glancltd



Varna, Bulgaria

