Az ügyfélkapu plusz technológiája

és biztonsága

Ez a szolgáltatás – amely a célja szerint biztonságos bejelentkezést valósít meg – 2022-ben indult, és két-tényezős azonosítást (2FA) tesz lehetővé TOTP[1] segítségével. Az írás célja, hogy a technológia iránt érdeklődőknek áttekintést adjon, valamint olyan hivatkozásokat, melyek alapján az érdeklődők elindulhatnak és elmélyülhetnek a témában.

Konklúzió röviden

A 2FA jó és hasznos megfelelő megvalósítás esetén a biztonságot növelő dolog! Jelen megvalósítás, szinte bármelyik TOTP-t megvalósító alkalmazással használható, ami **előny lehetne[2].** Technológiai szempontból **kissé avíttas**[3] –amin könnyen lehet majd javítani[4]– de legalább van és működik.[5] – bővebben a végén.

1. A technológia

A két, a témában sokat emlegetett rövidítés a 2FA, illetve a TOTP sokaknak ismerős lehet, ám a legtöbb embernek vélhetően nem sokat jelent. Mélységében olykor még a szakemberek sem ismerik, csak mint eszközt és „szakszót” használják. Ezen próbálunk most változtatni!

1.2 Mi az a 2FA / MFA

Two- / mutli-factor authentication, azaz magyarul két-, vagy többfaktoros hitelesítés, azt jelenti, hogy a sikeres azonosításhoz nem elegendő első és egyetlen faktorként a megfelelő felhasználónév, illetve jelszó párost ismerni, hanem egy második, esetleg harmadik faktorként valamit tudni kell, esetleg valamit birtokolni (igazoltan!) kell.

Miért jó a 2FA?

Hatékony védelmet jelent még akkor is, ha valaki megszerzi a jelszavadat. A 2FA megakadályozza ez illetéktelen belépést, lévén a támadónak szüksége van a második – vagy akár további – azonosítási tényezőkre is a sikeres azonosításhoz, amit általában nem sikerül a jelszóval egy időben kompromittálni). Egy birtok alapú megoldás – mint amilyen például egy hardveres kulcs – nem

1 TOTP: Time-Based One-Time Password Algorithm (2011 május) (nem keverendő össze a TOTP.APP nevű alkalmazással)
2 Ha nem lenne néhány alkalmazás elavult (és emiatt kevéssé biztonságos) hash, vagy titok méret, esetleg mind kettő miatt! Ilyenkor nem szerencsés a kompatibilitáshoz ragaszkodni, még akkor sem ha problémás alkalmazás nagy gyártó elterjedt eszköze.
3 Az avíttas (melléknév) jelentése: 1- kissé régi, kopottas, színehagyott ⟨dolog, holmi⟩ 2- kissé elavult, idejétmúlt, korszerűtlen (forrás: A magyar nyelv nagyszótára azaz https://nagyszotar.nytud.hu/dictsearch.html?entryid=4750)
4 A titok méretét kell megnövelni és az SHA–1 korszerűbbre cserélni. – vélhetően ezek mint alapértelmezett beállítások a használt eszköznél, és „úgy maradtak”.
5 Mire leírtam, már beugrott fiatalkorom szamizdat filmje – a Tanú – egy híres, sokat idézett mondata: „Az új magyarnarancs. Kicsit sárgább, kicsit savanyú, de a mienk.”

hozzáférhető egy támadó számára, hiába sikerült is a jelszót megszereznie.

A megoldás gyenge jelszavak esetén is plusz biztonságot jelent: az emberek általában nem használnak elegendően erős jelszavakat, így nem csak azok ellopásától kell tartani, de attól is, hogy a támadó képes azt kitalálni (például a személyes adataink birtokában, ha az a jelszavunk része), ám a második faktor (2FA) ilyenkor is jelentősen növeli a fiók biztonságát, pont úgy, mint ahogy növeli, ha a jelszó lopták el. – hiszen pont úgy nem leget csak a jelszó birtokában belépni.

Második faktor gyanánt számos eszköz és módszer kínálkozik, melyek között vannak vegytiszta példák tudás / birtok alapú, valamint biometrikus azonosításra

- tudás alapú: amit tudsz
  - a jelszavad (és a bejelentkező neved)
- birtok alapú: amit birtokolsz
  - belépéshez használt kártya, vagy token
  - banktól kapott fizikai jelszógenerátorod
- biometrikus: ami vagy
  - az ujjlenyomatod

illetve vegyes megoldások

- a mobil telefonod rajta az autentikációs applikációval, ami csak ujjlenyomattal használható
- a gépeden a böngésződben a kliensoldali kulcs, ami jelszóval védett

1.2.1. 2FA módszerek

Példakánt álljon itt néhány módszer – a teljesség igénye nélkül – a második faktor megvalósítására, azzal a megjegyzéssel, hogy mint semmi a biztonságtechnikában, a második faktor sem tekinthető csodaszernek, olyan megoldásnak mellyel elérhető a teljes biztonság[6].

- SMS[7]-ben, vagy e-mailben küldött kódok
  A rendszer egy egyszer használatos – rendszerint korlátozott ideig érvényes – kódot küld SMS-ben, vagy az e-mailben, amit a weboldalon, vagy az alkalmazásban kell beírni. Biztonsági szempontból kevésbé szerencsés, mint az ügyfélnél levő TOTP megvalósítás, mivel az SMS-eket[8] és az e-maileket továbbítás közben el lehet fogni[9].
- Időalapú egyszeri jelszavak ügyfélnél (TOTP)
  Ez a leginkább elterjedtebb módszer. Egy hitelesítő alkalmazás (pl.: FreeOTP[10], Google Authenticator[11], Authy[12]), vagy egy hardveres eszköz (pl.: Yubico[13]) generál egy rövid ideig (általában 30 másodperc) érvényes kódot, amit az előző esethez hasonlóan kell megadni. Biztonsági szempontból előnyösebb, mivel a kódot nem kell kiküldeni, az az eszközön generálódik. Hardveres megoldás esetén még a telefonunk feltörése (lásd Pegasus-kémszoftver[14]) sem jelent kockázatot.

6 Miért ne becsüljük le a kisbetűs jelszavakat? 2. rész (Bitport) (https://bitport.hu/miert-ne-becsuljuk-le-a-kisbetus-jelszavakat-password-security-2-resz)
7 Short Message Service Wikipédia https://hu.wikipedia.org/wiki/SMS
8 A klasszikus SMS nem titkosított, tehát például lehallgatható, erre a célra szoktak akár hamis bázis állomásokat (IMSI-catcherek) is használni. (Az IMSI-catcherek (ezekkel minden lehallgatható) használata, sok országban illegális.). Lehet a SIM kártya klónozásával vagy cseréjével is megszerezni az üzeneteket (SMS), ez ellen bár tehetnék, de a költségessége miatt, nem védenek a mobil hálózzatok. Telefonra feljuttatott rosszindulatú szoftverrel, mely előled elrejti, de a támadónak tovább küldi az SMS-t.
9 Mára (2025. január 15) elkészült az e-mail alapú megvalósítás.
10 Szerencsésebb a FreeOTPPlus https://play.google.com/store/search?q=freeotp%2B&c=apps használata, mert a FreeOTP mentése jelenleg nem visszatölthető.
11 Google Authenticator (Wikipedia)
12 https://www.authy.com
13 https://www.yubico.com
14 Mit tud a hírhedt Pegasus program, ha rákerül valaki telefonjára? Szó szerint mindent (HVG) https://hvg.hu/tudomany/20210720_nso_pegasus_lehallgatas_megfigyeles_kemszoftver_telepitese_mit_tud

- Hardveres biztonsági kulcsok
  Ezek kézzelfogható tárgyak (eszközök), melyeket a számítógéphez – például az USB porthoz – kell csatlakoztatni, a bejelentkezéskor megérintve, vagy megnyomva a kulcsot. Ma ezt tartjuk a legbiztonságosabbnak, ugyanakkor itt már a kényelmi szint csökken, mivel az eszközt midig magunknál kell tartani, amikor csak be akarunk jelentkezni. Ha a rendszer kínál alternatív megoldást arra az esetre, ha nincs nálunk az eszköz, akkor a biztonsági szint valójában az alternatíva biztonsági szintje.
- Biometrikus azonosítás
  Van, ahol lehetővé teszik az ujjlenyomat, arcfelismerés, vagy más biometrikus adatok használatát második faktorként. Kényelmi szempontból roppant előnyös, hiszen a biometrikus azonosítók velünk vannak, ugyanakkor nem biztos, hogy rendelkezésre is állnak (megégetett, vagy olajos ujj), illetve az alkalmazott technikai eszköz is hordozhat biztonsági kockázatot[15], illetve arról sem szabad megfeledkezni, hogy jelszavaink cserélhetők egy, vagy több biztonsági incidens után, ujjlenyomatunk csak korlátozott számban. Ezen túlmenően itt is igaz, hogy a helyettesítő megoldás (pl.: ujjlenyomat, arcfelismerés helyett PIN-kód) elérhetősége itt is lecsökkenti biztonság szintjét a helyettesítő megoldás szintjére.

1.2.3. Az ügyfélkapuplusz.hu[16] által a TOTP-hez ajánlott szoftverek

Térjünk vissza az idő alapú egyszer használatos jelszavakhoz, illetve a Ügyfélkapu oldal által ajánlott megoldásokhoz.

- Google Authenticator: Egyszerűen használható alkalmazás, amely egy meghatározott idő elteltével új kódot generál.
- Microsoft Authenticator: Hasonló funkcionalitással rendelkezik, és támogatja a felhőalapú biztonsági mentést is[17].
- NISZ Hitelesítő: Kifejezetten az Ügyfélkapu+ számára fejlesztett alkalmazás, amely Android és iOS platformokon is elérhető.

Ezeken kívül is működnie kell(ene) minden más TOTP megvalósításnak[18] is – *bár mindet – érthető okoknál fogva –sem mi, sem a fejlesztő vagy ép üzemeltető nem tesztelte le.*

1.2.4 A FreeOTP alkalmazás

A FreeOTP alkalmazást – ami egy szabad szoftver licencű Google Authenticator alternatíva – tesztelve a rendszer működését komoly meglepetés ért. Az Ügyfélkapu+ beállításakor az alábbi figyelmeztetés jelenik meg:

> Token is unsafe!
>
> The token you are attempting to add contains weak cryptographic parameters. Use of this token is strongly discouraged! Please alert your token provider.

15 Miért ne becsüljük le a kisbetűs jelszavakat? 2. rész (Bitport) (https://bitport.hu/miert-ne-becsuljuk-le-a-kisbetus-jelszavakat-password-security-2-resz)

16 Az alkalmazások listája eredetileg (2024) itt (https://ugyfelkapuplusz.hu/authenticator/) volt elérhető ami 2025 januárjában már átvisz ide (https://www.iksznet.hu/ugyfelkapu-plusz-authenticator/)

17 2022, 2023 hibajegyek szerint elavult, és az adott időszakban nem javították. pl.: https://answers.microsoft.com/en-us/msoffice/forum/all/authenticator-app-not-working-with-sha-256-and-sha/f0023746-2d4b-499e-aee5-2463d96a8144, https://answers.microsoft.com/en-us/msoffice/forum/all/sha-256-token-with-ms-authenticator-returns-wrong/c5dcf183-da18-4409-93b5-30a4784d1419

18 Az AEGIS-t elégedetten használják egyesek (https://getaegis.app/#), a KeePAss, KePassxc, valamint az Enpass eredményes használatáról is írnak,

amit természetesen figyelmen kívül lehet hagyni és továbblépni,

Cancel
Add anyway (ezzel lehet továbblépni)

viszont az üzenet elgondolkodtató, aminek érdemes utánanézni.

JAVASLAT
A FreeOTP helyett, a FreeOTP+[19] használata javasolt – egy elágazása, forkja a FreeOTP-nek –, mivel kódok mentése és másik mobilon való visszatöltése az előbi esetén hibásan, az utóbbi esetén hibátlanul működik.

1.3. Mi a OTP / HTC és mi a HOTP?

Az OTP / HTC (One Time Password / HMAC-based one-time password) egy egyszer használható jelszó. Bár lehetne ezt a megoldást akár önállóan is alkalmazni, de a gyakorlatban név és jelszó párossal kombinálva használják, mint második – kiegészítő – hitelesítő eljárást. A módszer azáltal nehezíti az illetéktelen hozzáférést, hogy egy dinamikusan változó adatot is meg kell szereznie a támadónak, a statikus belépési név és jelszó pároson kívül, hogy sikeres támadást hajthasson végre.

A HOTP szabvány[20] egy egyszer használatos jelszó generálási módszert definiál, ami egy számlálón alapul, amely számláló minden jelszó generálás után növekszik Ez praktikusan azt jelenti, hogy minden alkalommal új kódot generálódik.

1.3.1. Mi a HMAC?

Hash-based Message Authentication Code, vagy magyarul kivonat-alapú üzenet hitelesítő kód. Ez az eljárás egy üzenet (bájt- / karaktersorozat) hitelességének és integritásának ellenőrzésére szolgál. Egy – mindkét fél által ismert – titkos kulcsot és egy kriptográfiai hasítófüggvényt (hash function) használva egy kivonatot képez, amit az üzenethez mellékel, melynek segítségével a fogadó fél ellenőrizheti az üzenet sértetlenségét.

1.3.1.1. Mi hash függvény?

Egy kriptográfiai eljárás, amely tetszőleges hosszúságú bemeneti adatból – legyen az néhány kilóbájt, vagy néhány terrabájt – egy kötött hosszúságú – néhány 10 bájtos – kimeneti értéket állít elő. Ezt hívják hash értéknek, kivonatnak, vagy ujjlenyomatnak (fingerprint). A hash érték úgy áll elő, hogy a bemeneti adatot valamilyen matematikai műveletsorozatnak vetik alá (pl.: bitműveletek, XOR, moduláris aritmetika stb.), ami egy fix méretű kimenetet eredményez, ami a gyakorlatban minden bemenet esetén jelentősen különbözik. Ennek praktikusan az a jelentősége, hogy hash értékek lényegesen könnyebben hasonlítható össze – nem csak géppel, de szemmel is – mint két nagy méretű állomány. Például ha letöltünk egy telepítő szoftvert, amihez mellékeltek hash értéket, azt egy programmal könnyen és gyorsan kiszámolhatjuk a letöltött állományra és ha egyezik, akkor tudhatjuk, hogy a letöltése során nem keletkezett hiba.

19 FreeOTP+ (Google Play) https://play.google.com/store/apps/details?id=org.liberty.android.freeotpplus
20 RFC 4226 – HOTP: An HMAC-Based One-Time Password Algorithm (2005. december) https://datatracker.ietf.org/doc/html/rfc4226

1.3.1.1.1. Hash függvénytől elvárt tulajdonságok

- Egyértelmű, meghatározott (determinisztikus): ugyanaz a bemenet mindig ugyanazt a hash értéket eredményezze.
- Gyorsan számolható azaz hatékony: nagy mennyiségű adatot is rövid idő alatt feldolgozható legyen.
- Egyirányú (gyakorlatilag): a hash értékből ne lehessen visszaállítani az eredeti bemeneti adatot.
- Ütközésállóság: nehéz (ideális esetben lehetetlen) két különböző bemeneti adatot találni, amelyek ugyanazt a hash értéket eredményezik (ütközés). Minél kisebb az ütközés valószínűsége, annál jobb a függvény.

Ha egy függvényről igazolják[21], hogy előállítható ütközés azaz, belátható időn belül lehet találni olyan bemeneti adatot, amely ugyan azt a hash értéket eredményezi, mint egy másik bemeneti adat, akkor azt a hash függvényt (pl.: SHA–1) indokolt nem használni, hiszen ez visszaélésekre adhat lehetőséget. Maradva a telepítő példájánál, valaki úgy módosíthatja kódot és ezzel működést, hogy hash érték változatlan marad, vagyis a felhasználónak nem tűnik fel a módosítás ami egy dokumentum (pl.: szerződés) esetén hasonlóan problémás lehet.
Meg kell ugyanakkor jegyezni, hogy HMAC – a hash kétszeri futtatása és a titkos kulccsal való kombinálás miatt – kevéssé érzékeny hasítófüggvény elleni ütközéses támadásokra.

1.3.2. A TOTP (Time-based One-Time Password)

Időalapú egyszeri jelszót jelent; az OTP / HOTP továbbgondolása. A módszer[22] lényege, hogy míg a HOTP esetén az üzenet szerepét – a HMAC algoritmus alkalmazása során – az idő – jellemzően 30 másodpercre – kerekített értéke tölti be, így egy korlátozott ideig (30 másodperc, de akár kevesebb) és korlátozott alkalommal (egyszer) használható jelszót kapunk, amit a bejelentkezésnél a sikeres név és jelszó beírást követően kell megadni. Ha az egyszer használatos jelszó megadása a megfelelő időkereten kívül történik, az ellenőrzés sikertelen lesz.

Ugyanakkor a fentiek még nem elegendőek, hiszen mint arról szó esett, a HMAC algoritmus használatához nem csak egy üzenetre (ez esetben az idő), de egy mindkét fél – a bejelentkezést végző felhasználó és az általa megadott kód ellenőrzését végző rendszer – által ismert titokra is szükség van. Ezt titkot tartalmazza az a QR-kód, amit a második faktorként használt TOTP élesítésekor kell beolvasunk a már említett alkalmazások valamelyikével. A titkon túlmenően a QR-kód a kibocsátót (pl.: egy weboldal) – amihez a TOTP tartozik –, illetve az alkalmazandó hash algoritmust – hiszen ebből is többféle létezik – is tartalmazza, pontosabban tartalmazhatja, lévén ezek opcionális elemek[23].

Érdekesség: A TOTP jelszó generálásához a jelszót generáló eszköznek nem kell vezetékes, Wifi, vagy mobil internetkapcsolattal rendelkeznie, hiszen az előállításhoz szükséges titok és a pontos idő egyaránt rendelkezésre áll. Ezzel együtt – például a felhős adattárolás miatt, vagy egyéb okból – a megvalósítás akár igényelhet internetelérést.

Az érvényességi időablak már csak azért is szükséges és praktikus (no meg kockázatos), mert nem garantálható a felhasználói eszközök és ez elérendő eszköz órájának szinkronban léte, no meg mi emberek is benne vagyunk és ugye még a gyors és gépírók is lassan gépelnek egy számítógép eseményeihez képest…

---

21 SHA–1 elméleti gyengeségeit már 2005-ben (Xiaoyun Wang, Yiqun Lisa Yin és Hongbo Yu) kimutatták, a 2017-es SHAttered támadás (Google és a CWI Amsterda kutatói demonstrálták) volt, amikor a gyakorlatban is bebizonyosodott, hogy az SHA–1 nem alkalmas biztonságos hash függvényként való használatra.
22 TOTP: Time-Based One-Time Password Algorithm (2011. május) https://datatracker.ietf.org/doc/html/rfc6238
23 Google Authenticator – Key Uri Format (2018. november) https://github.com/google/google-authenticator/wiki/Key-Uri-Format

1.4. Technikai anomáliák, zárványok

Az ominózus „*Token is unsafe!*” hibaüzenetet a FreeOTP akkor jelenti meg, ha megosztott titok hossza kisebb, mint 128 bit. Nos, az Ügyfélkapu+ esetén (a vizsgálatunk szerint) 80 bit, ami alig kétharmada a TOTP által elvártaknak.

1.4.1. Elég-e a 80 bit, ha tényleg 80 bit?

A kellő szintű biztonság eléréséhez megfelelő méretű megosztott titok használata szükséges. A megosztott titoknak legalább 128 bitesnek kell lennie – a vonatkozó szabvány (RFC4226[24]) értelmében. Ráadásul ez a dokumentum már ekkor is 160 bites megosztott titok használatát ajánlja, ezzel együtt is, hogy szabvány 2005-ben íródott. Az Ügyfélkapu+ ezen ajánlott értéknek csupán a felét használja, ami kevésnek tűnik!

Érdekesség: az említett 80 bitnyi megosztott titkos entrópiája[25] 17 véletlenszerűen választott kisbetűből álló jelszó elméleti maximális entrópiájának felel meg. Ha nagybetűk is lehetnek a jelszóban, akkor 14 karakter az ekvivalens jelszóméret. Tovább bővítve, mikor már a kis- és nagybetűk, számok és szimbólumok (a-z, A-Z, 0-9, és a 32 szimbólum, összesen 94 karakter) is szerepelhetnek, akkor kerekítve 12 karakter hosszú jelszóval lesz ez a 80 bittel egyenértékű. Ez olyan szempontból bír jelentőséggel, hogy ha jelszavakkal is elérhető a 80 bit entrópia, akkor ez gépek által generált kódok esetén még annyira kívánatos.

1.4.2. Az SHA–1 [26]

Az Ügyfélkapu+ hash függvény gyanánt SHA–1 algoritmust használ (HMAC-en belül), ami nem elegáns, lévén egy elavult (deprecated) algoritmusról van szó. Az USA szabványügyi és technológiai intézete (NIST[27]), egy 2024 októberében megjelent anyaga[28] (4-es táblázat) az SHA–1 algoritmust 2030-ig ugyan engedélyezi, de elavultnak (deprecated) tekinti, 2030-tól pedig tiltja annak használatát – már ha a NIST elvárásainak meg akar valaki felelni.

1.4.2.1. Az SHA–1 használata nem ajánlott, de „szódával elmegy”

Az NIST ugyan nem ajánlja, de jelen környezetben az SHA–1 használata nem igazán jelent köztudott kockázatot/ regisztrált ismert sebezhetőséget, mivel a HMAC nem érzékeny arra a típusú hibára (ütközés), amire tudomásunk szerint az SHA–1 érzékeny, bár a megoldás összességében nem is nevezhető modernnek.

2. Végeredmény

A már olvasott konklúzió a következő: **kicsit avítt a megvalósítás, ám jelen állás szerint, ha nem is ele-**

24 RFC 4226 – HOTP: An HMAC-Based One-Time Password Algorithm (2005. december) https://www.ietf.org/rfc/rfc4226.txt
25 A kriptográfiában az entrópia egy véletlenszerű változó vagy egy titkos információ (például jelszó, titkos kulcs) kiszámíthatatlanságának, bizonytalanságának mértéke. Minél nagyobb az entrópia, annál nehezebb kitalálni vagy feltörni az adott információt. Az entrópiát bitekben mérjük.
26 Az SHA–1 (Secure Hash Algorithm 1) egykor széles körben használt kriptográfiai hash függvény volt. Az Amerikai Egyesült Államok Nemzetbiztonsági Ügynöksége (NSA) tervezte, és NIST tette közzé 1995-ben. Ma már Deprecated azaz nem ajánlott!
27 National Institute of Standards and Technology
28 NIST Special Publication 800 – Transitioning the Use of Cryptographic Algorithms and Key Lengths (2024. október) https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-131Ar3.ipd.pdf

gáns, de használható. A javítás sem jelent különösebben nehézséget, bár meg kell jegyezni, hogy mind a titok méretének növelése, mind az alkalmazott HMAC algoritmus cseréje a második faktor újbóli kiadását igényli. Ez a szerver oldalon fejlesztést, felhasználó oldalon kényelmetlenséget, esetleg költség nélküli eszköz cserét okozhat a felhasználónál. Az esetleges szoftver csere oka, hogy a modernebb (biztonságosabb) algoritmusok használata nem minden alkalmazás esetén támogatott (Írásunk idején problémás pl.: Microsoft Authenticator[29]). A túl rövid titok és az általában nem javasolt algoritmus már 2021,2022 években (mikor a fejlesztés feltehetően zajlott) is ismert kellet, hogy legyen, tehát az most felmerülő munka megfelelő tervezéssel, kivitelezéssel a vonatkozó szabványnak / ajánlásnak való megfeleléssel. vélhetően elkerülhető lett volna.

Terveink szerint az Authenticator szoftverekkel egy másik írásban részletesebben foglalkozunk majd.

29 2022, 2023 évben született hibajegyek szerint elavult (Nem támogatja az SHA–256 algoritmust), és írásunkig nem javították. pl.: https://answers.microsoft.com/en-us/msoffice/forum/all/authenticator-app-not-working-with-sha-256-and-sha/f0023746-2d4b-499e-aee5-2463d96a8144, https://answers.microsoft.com/en-us/msoffice/forum/all/sha-256-token-with-ms-authenticator-returns-wrong/c5dcf183-da18-4409-93b5-30a4784d1419