UNIVERZA V LJUBLJANI
EKONOMSKA FAKULTETA

MAGISTRSKO DELO

ANALIZA USTREZNOSTI DVO-FAKTORSKE AVTENTIKACIJE NA PAMETNIH TELEFONIH

Ljubljana, marec 2019 JOŽE BEZEK

IZJAVA O AVTORSTVU

Podpisani Jože Bezek, študent Ekonomske fakultete Univerze v Ljubljani, avtor predloženega dela z naslovom Analiza ustreznosti dvo-faktorske avtentikacije na pametnih telefonih, pripravljenega v sodelovanju s svetovalcem prof. dr. Alešem Groznikom

IZJAVLJAM

1. da sem predloženo delo pripravil samostojno;

2. da je tiskana oblika predloženega dela istovetna njegovi elektronski obliki;

3. da je besedilo predloženega dela jezikovno korektno in tehnično pripravljeno v skladu z Navodili za izdelavo zaključnih nalog Ekonomske fakultete Univerze v Ljubljani, kar pomeni, da sem poskrbel, da so dela in mnenja drugih avtorjev oziroma avtoric, ki jih uporabljam oziroma navajam v besedilu, citirana oziroma povzeta v skladu z Navodili za izdelavo zaključnih nalog Ekonomske fakultete Univerze v Ljubljani;

4. da se zavedam, da je plagiatorstvo – predstavljanje tujih del (v pisni ali grafični obliki) kot mojih lastnih – kaznivo po Kazenskem zakoniku Republike Slovenije;

5. da se zavedam posledic, ki bi jih na osnovi predloženega dela dokazano plagiatorstvo lahko predstavljalo za moj status na Ekonomski fakulteti Univerze v Ljubljani v skladu z relevantnim pravilnikom;

6. da sem pridobil vsa potrebna dovoljenja za uporabo podatkov in avtorskih del v predloženem delu in jih v njem jasno označil;

7. da sem pri pripravi predloženega dela ravnal v skladu z etičnimi načeli in, kjer je to potrebno, za raziskavo pridobil soglasje etične komisije;

8. da soglašam, da se elektronska oblika predloženega dela uporabi za preverjanje podobnosti vsebine z drugimi deli s programsko opremo za preverjanje podobnosti vsebine, ki je povezana s študijskim informacijskim sistemom članice;

9. da na Univerzo v Ljubljani neodplačno, neizključno, prostorsko in časovno neomejeno prenašam pravico shranitve predloženega dela v elektronski obliki, pravico reproduciranja ter pravico dajanja predloženega dela na voljo javnosti na svetovnem spletu preko Repozitorija Univerze v Ljubljani;

10. da hkrati z objavo predloženega dela dovoljujem objavo svojih osebnih podatkov, ki so navedeni v njem in v tej izjavi.

V Ljubljani, dne _____ Podpis študenta:_____

KAZALO

KAZALO TABEL

KAZALO SLIK

SEZNAM KRATIC

angl. – angleško
$ – ameriški dolar
ADB – (angl. Android Debug Bridge)
AES – (angl. Advanced Encryption Standard); Napredni standard za šifriranje
AOSP – (angl. Android Open Source Project)
API – (angl. Application Programming Interface); Programski vmesnik
APK – (angl. Android Package); Android namestitveni paket
APT – (angl. Advanced Persistent Threat)
AV – (angl. Anti virus); Antivirusni program
BYOD - (angl. Bring Your Own Device); Prinesi svojo napravo
C&C – (angl. Command and Control servers)
CA – (angl. Certificate Authorities); Certifikatska agencija
CBC – (angl. Cipher-Block Chaining)
CDD – (angl. Compatibility Definition Document)
CTF – (angl. Compressed Token Format)
CT-KIP – (angl. Cryptographic Token Key Initialization Protocol)
CTS – (angl. Compatibility Test Suite); Združljivostni test
DDMS – (angl. Dalvik Debug Monitor Server); Dalvik strežnik za odpravljanje napak
DLL – (angl. Dynamic Link Library); Knjižnica dinamičnih povezav
DPC – (angl. Device Policy Client)
DRM – (angl. Digital Rights Management); Upravljanje digitalnih pravic
DSA – (angl. Digital Signature Algorithm); Digitalni podpisni algoritem

DVM – (angl. Dalvik Virtual Machine); Dalvikova bajtna koda
EAP – (angl. Extensible Authentication Protocols)
GA – (angl. Google Authenticator); Goolge avtentikator
GPS – (angl. Global Positioning System); Globalni sistem pozicioniranja
GSM – (angl. Global System for Mobile communications); Globalni mobilni sistem
HMAC – (angl. Hash-Based Message Authentication Code)
HOTP – (angl. Hash-based message authentication code OTP)
HTTPS – (angl. HyperText Transfer Protocol Secure)
ICO – (angl. Initial Coin Offering); Začetna ponudba kovancev
IDE – (angl. Integrated Development Environment); Celostno razvojno okolje
IMEI – (angl. International Mobile Equipment Identity); Edinstvena številka mobilnega aparata
IT – (angl. Information technology); Informacijska tehnologija
JVM – (angl. Java Virtual Machine); Java navidezna naprava
MAC – (angl. Mandatory Access Control)
MD5 – (angl. Message-Digest Algorithm)
MDM – (angl. Mobile Device Management); Upravljanje mobilnih naprav
MiTB – (angl. Man in the browser); Mož v brskalniku
MiTM – (angl. Man in the middle); Mož v sredini
mTAN – (angl. Mobile Transaction Authentication Number)
NTLM - (angl. NT LAN Manager)
OATH – (angl. HMAC-based One-time Password algorithm);
oAuth – (angl. open-Standard Authorization);
OEM – (angl. Original Equipment Manufacturer); Proizvajalec originalne opreme
OOB – (angl. Out Of Band)
OS – (angl. Operating System); Operacijski sistem
OTP – (angl. One Time Password); Enkratno geslo
PAM – (angl. Pluggable Authentication Modules)
PEAP – (angl. Protected Extensible Authentication Protocol)
PIN – (angl. Personal Identification Number); Osebna identifikacijska številka
PKI – (angl. Public Key Infrastructre); Infrastruktura javnih ključev
QR – (angl. Quick Response)
RADIUS – (angl. Remote Authentication Dial-In User Service)
RSA – (angl. Rivest-Shamir-Adleman)
SAML – (angl. Security Assertion Markup Language)
SD – (angl. Secure Digital); Oznaka za spominsko kartico
SDK – (angl. Software Development Kit); Orodje za razvoj aplikacij
SDTID – (angl. File-Based Provisioning)
SHA – (angl. Secure Hash algoritem)
SHA – (angl. Secure Hash Algorithm)
SIM – (angl. Subscriber Identity Module)

SMS – (angl. Short Message Service); Kratko sporočilo
SSH – (angl. Secure Shell)
SSL – (Secure Socket Layer)
TAN – (angl. Transaction Authentication Number); Transakcijska avtentikacijska številka
TET – (angl. Trusted Execution Environment's)
TLS – (angl. Transport Layer Security)
TOTP – (angl. Timebased One Time Password Algorithm)
TTLS – (angl. Tunneled Transport Layer Security)
UID – (angl. Unique identifier); Edinstven identifikator
UMTS – (angl. Universal Mobile Telecommunications System); Univerzalni mobilni telekomunikacijski sistem
URL – (angl. Uniform Resource Locator); Enolični krajevnik vira
VPN – (angl. Virtual private network); Navidezno zasebno omrežje
Wi-Fi – (angl. Wireless Fidelity); Brezžično omrežje
WPA2 – (angl. Wireless Protected Access 2)

UVOD

Povečanje rasti priljubljenosti pametnih telefonov in zelo priročne uporabe telefona za vse namene, kot so uporaba za namen fotoaparata, brskanja po internetu pa vse do elektronskega bančništva in uporabe namenskih aplikacij, je privedlo do potrebe po vse večji varnosti v medsebojni izmenjavi podatkov kot vsakodnevni nujni ozaveščenosti vsakega posameznika. V svetovnem merilu se ogromno sredstev namenja za mobilno varnost in do leta 2020 bodo sredstva dosegla 34,8 $ milijarde, kar pomeni 40,8 % letno rast (Research and Markets, 2015).

V zadnjih letih se je dramatično povečala tudi zlonamerna koda, napisana za mobilne platforme. Oktobra 2016 je zlonamerna koda prizadela 1,35 % vseh mobilnih naprav, kar je največ doslej. Na pametnih telefonih je zabeleženo skoraj 400 % povečanje zlonamerne programske opreme in predstavlja 85 % vseh okužb na mobilnih napravah (Nokia, 2017). Nekatere od zlonamernih aplikacij se obnašajo in so videti kot legalne, vendar so lahko razvite tako, da beležijo in posredujejo uporabnikove poverilnice z namenom pridobitve nezakonitega premoženja ali povzročitve kakršne koli druge škode.

Zaradi vse več internetnih strani, mobilnih naprav in uporabniških računov je posledica povečano število gesel in dostopov. Še vedno se za preverjanje pristnosti večinoma uporablja samo uporabniško ime in geslo. Tradicionalna strategija gesel pa ni v koraku s hitro razvijajočim se digitalnim okoljem, ker je identiteta vedno bolj kritičen vektor za grožnje. Izziv varnosti je kompleksen, saj imamo na eni strani hitro spreminjajoče se uporabnikove potrebe, na drugi strani pa informacije, do katerih morajo dostopati, in mobilne naprave, ki jih uporabljajo (EMC, 2014).

Kadar je geslo razkrito, za napadalca ni nobenega mehanizma, ki bi preprečil njegovo uporabo iz katerekoli mobilne naprave. Čeprav naj bi bila gesla zasebna in varna, pa ta predpostavka v praksi ni realna. Da preprečimo oziroma omejimo prijavo neželenim osebam na storitve z ukradenimi poverilnicami, moramo uporabiti dodaten faktor za dokaz uporabnikove identitete. Dodaten faktor dokazovanja identitete je lahko namenska strojna oprema, aplikacija za pametni telefon ali tekstovno sporočilo, prejeto s strani ponudnika storitve, ki jih imenujemo generatorji enkratnih gesel.

Dvofaktorska avtentikacija (v nadaljevanju 2FA) je način oziroma metoda, sestavljena iz dveh avtentikacijskih faktorjev, z namenom povečati varnost pri istovetnosti, avtentikaciji uporabnika. Avtentikacijski faktor je definiran kot nekaj, kar vemo, kar imamo in kar smo. Uporabniško ime in geslo največkrat sodi med tisto, kar vemo, fizično posedovanje je tisto, kar imamo in na primer biometrika je tisto, kar smo (SearchSecurity, 2015). Nivo varnosti se pri 2FA v primerjavi s prijavo samo z geslom, drastično poveča. Hkrati pa se je s prihodom

pametnih telefonov izkoristila njihova vsestranskost, zato so v ta namen razvili priročne programske generatorje enkratnih gesel. Programski generatorji enkratnih gesel (angl. software token), so aplikacije, ki ustvarijo gesla za prijavo z enkratno uporabo in se jih lahko namesti na namizne in prenosne računalnike ali na pametne telefone (RSA, 2017a).

Problem je, kako in na kakšen način izbrati med razpoložljivimi metodami primerno 2FA in jo s pravilnim pristopom implementirati v podjetje, ki bo napadalcem onemogočil izrabiti uporabnikove poverilnice v svoj namen. Na trgu je veliko različnih metod 2FA, vendar nekatere samo minimalno izpolnjujejo zakonske in varnostne standarde ter niso zasnovane tako, da izpolnjujejo precej strožje zahteve. Pri tem nastopi tudi izbira storitve v oblaku ali na lokaciji podjetja ter problem, ali je storitvi 2FA v oblaku zaupati, da izpolnimo vse varnostne zahteve, ki jih ima podjetje predpisane.

Namen magistrskega dela je analizirati in primerjati tri načine za programsko generiranje enkratnih gesel na pametnih telefonih z vidika varnosti in enostavnosti uporabe. Kot najstarejšo izbrano metodo za preverjanje pristnosti bom uporabil prejetje enkratnih gesel prek SMS, mTAN (angl. mobile Transaction Authentication Number). Druga metoda in najbolj priljubljena je generiranje enkratnih gesel na pametnih telefonih, TOTP (angl. Timebased One Time Password Algorithm). Zadnja metoda je avtentikacija iz pametnega telefona s pomočjo certifikata, PKI (angl. Public Key Infrastructre). Študija bo narejena za operacijski sistem Android zato, ker si lasti 85 % svetovnega obsega, hkrati pa tudi zaradi največje zaznave zlonamerne kode, napisane za ta operacijski sistem (IDC, 2017).

Cilj magistrskega dela je raziskati in analizirati, ugotoviti in argumentirati, kateri način 2FA na pametnih telefonih je z varnostnega in uporabnega vidika najbolj primeren. Pri uresničitvi zadanega glavnega cilja imam predpogojne cilje.

Za lažje razumevanje tematike je cilj predstaviti splošen pregled 2FA, kot so generatorji enkratnih gesel, strojni in programski žetoni, se seznaniti z varnostjo na Android operacijskem sistemu ter prikazati pristope in orodja za odkrivanje zlonamerne programske kode. Pri primerjavi bom izbral produkte podjetij RSA, HID Global in Google avtentikator.

Predvidena vprašanja:

- Ali lahko večja varnost pri uporabi in implementaciji programskega generatorja gesel vpliva na zmanjšanje tveganja ranljivosti 2FA na pametnih telefonih?
- Ali lahko zlonamerna programska koda ogrozi aplikacijo za 2FA na pametnih telefonih?
- Ali lahko s povratnim inženiringom OS Android pridobimo geslo, ki ga generira programski generator gesel?

To vprašanje je v svoji študiji za OS Android (angl. Hacking Soft Tokens) raziskoval avtor Bernhard Mueller (Vantage Point Secuirty, 2016) za produkt RSA Secure ID. Pomembno ga je raziskati tudi v magistrski nalogi.

Pri svojem delu bom na začetku magistrskega dela poglobljeno teoretično-analitično pregledal strokovno literaturo, znanstvene raziskave ter članke domačih in tujih strokovnjakov. V tem delu bom analiziral z opisno metodo. V teoretičnem delu magistrske naloge bom tako proučil strokovno domačo in tujo literaturo s področja operacijskega sistema Android, predstavil različne tipe zlonamerne programske opreme ter orodja, s katerimi jih zaznamo. Predstavil bom 2FA za mobilne naprave. V praktičnem delu bom uporabil študijo primera izbire in uvedbe 2FA na pametnih telefonih. Kot vire podatkov bom uporabil strokovno dokumentacijo ponudnikov 2FA na pametnih telefonih in dokumentacijo na projektu. V tretjem delu bom s pomočjo SWOT analize predstavil prednosti in slabosti sistemov za 2FA. V nadaljevanju magistrske naloge bom ugotovitve iz predhodnih analiz apliciral na primeru smotrne izbire 2FA in s tehničnimi in ekonomskimi dognanji podkrepil pravilno izbiro 2FA na pametnih telefonih. Pri magistrski nalogi bom uporabil znanja in izkušnje, pridobljene z delom na področju 2FA.

Znanstveni prispevek magistrskega dela bo temeljil na teoretičnem delu, na podlagi katerega bo možno hitrejše predvsem pa imeti pregled katere faktorje je potrebno upoštevati pri izbiri programske 2FA za mobilne naprave.

Pričakujem, da bo uporabna vrednost magistrskega dela na podlagi študije primera podlaga za lažje odločanje odgovornih za izbiro ustrezne avtentikacije na pametnih telefonih.

1 DVOFAKTORSKA AVTENTIKACIJA

Dvofaktorska avtentikacija se pogosto omenja kot dvostopenjsko preverjanje pristnosti in je varnostni postopek, v katerem uporabnik uporabi dva avtentikacijska faktorja za preverjanje pristnosti, da sistem preveri, če gre za osebo, za katero pravi, da je. S tem 2FA zagotavlja dodatno stopnjo varnosti, saj z enofaktorskim preverjanjem pristnosti uporabnik uporabi le en avtentikacijski faktor, po navadi samo geslo. Kot posledica uporabe 2FA je zato napadalcem težje pridobiti dostop do uporabnikovih naprav, kot so računalniki in mobilni telefoni ter spletni računi, saj poznavanje samo uporabnikovega gesla ni dovolj za preverjanje pristnosti.

2FA se že dolgo uporablja za preverjanje pristnosti, nadzora dostopa do občutljivih sistemov, podatkov in ponudniki spletnih storitev vse bolj promovirajo 2FA z namenom preprečiti hekerjem dostop do uporabnikovih podatkov, ker so ti že lahko ukradli podatkovno bazo z

gesli ali pa so uporabili lažno predstavljanje (angl. phishing) za pridobitev uporabniških gesel.

Načine, s katerimi se lahko uporabnik avtenticira, razdelimo na tri kategorije, poznane kot faktorji avtentikacije:

- faktor kaj vemo (angl. knowledge factors) je nekaj, kar uporabnik ve, kot so geslo, PIN[1] (angl. personal identification number) ali skupna skrivnost (angl. shared secret).
- faktor kaj imamo (angl. possession factors) je nekaj, kar uporabnik poseduje, na primer osebna izkaznica, varnostni žeton ali pametni telefon.
- faktor kaj smo (angl. inherence factors) oziroma bolj pogosto imenovano biometrija je, kaj uporabnik je. To so lahko osebni atributi, ki so narejeni iz fizičnih značilnosti osebe, kot so prstni odtisi, obraza in glasu. Biometrija vključuje tudi vedenjske biometrične podatke, kot so dinamika pritiska tipk, poteze ali vzorci govora.

Sistemi z zahtevnejšimi zahtevami glede varnosti lahko uporabijo lokacijo in čas kot četrti in peti faktor. Od uporabnikov se lahko zahteva, da se avtenticirajo na določenih lokacijah ali v določenih časovnih okvirih. 2FA in večfaktorska avtentifikacija vključujeta dve ali več neodvisnih poverilnic za prijavo. Uporaba dveh faktorjev iz iste kategorije ne predstavlja 2FA. Zahteva za geslo in skupna skrivnost še vedno veljata za enostopenjsko preverjanje pristnosti, saj oba pripadata istemu dejavniku, kaj vemo (TechTarget, 2016).

1.1 Generator enkratnih gesel

Generator enkratnih gesel je vključen v mnogih shemah 2FA in pri tem se uporablja kar nekaj metod, s katerimi se generira OTP (angl. One Time Password). Najbolj enostavna metoda je uporaba generatorja naključnega števila, da se s tem ustvari kompleksno enkratno geslo, ki ga je težko uganiti. Tako je OTP generator algoritem, ki ustvari novo naključno geslo. Vhodni podatek vnese uporabnik in sistem ustvari geslo, ki se vedno razlikuje od predhodno ustvarjenega gesla, da se s tem omogoči varno avtentikacijo z novim geslom, pa čeprav je lahko bilo predhodno geslo ukradeno, bodisi smo ga napačno vnesli pri prijavi. Druga metoda je vprašanje – odgovor avtentikacija (angl. Challenge-response authentication), pri kateri ponudnik storitve pošlje uporabniku vprašanje, uporabnik pa poda pravilen odgovor, da se avtenticira (Acharya, Polawar & Pawar, 2013) .

Začetni generatorji so bili izvedeni z algoritmi HOTP (angl. Hash-based message authentication code OTP), ki so temeljili na SHA-1 (angl. Secure Hash Algorithm) in MD5 (angl. Message-Digest Algorithm) algoritmih, ki danes ne veljajo več za zanesljive pred napadi. Ostali generatorji OTP so osnovani na Ping Pong-128 tokovni šifri, v kateri se

[1] PIN – od štiri- do šestmestna koda, sestavljena iz številk in/ali črk

uporablja algoritem Ping Pong-128 za generiranje naključnih števil. Med bolj varne algoritme sodijo genetski algoritem (ang. Genetic Algorithem), mravljični kolonijski algoritem (angl. Ant Colony Algorithm), algoritem eliptične krivulje (angl. Elliptic Curve Algorithm) in algoritem riba napihovalka (angl. Blowfish). Saini (2014) v svoji raziskavi predlaga metodo generiranja OTP z uporabo genetskega algoritma s kriptografijo eliptične krivulje (Saini, 2014).

Kljub vsemu je pri tem enkratno geslo varno, ker ga ni mogoče uporabiti dvakrat in ni reverzibilno, da bi ga dobili po viru. Distribucija gesla se uporabniku izvede prek storitve SMS-a na mobilni telefon.

1.2 Fizični žetoni

Varnostni žeton je fizična naprava, ki jo pooblaščeni uporabnik računalniških storitev dobi za preverjanje svoje pristnosti. Imenuje se tudi žeton za preverjanje pristnosti ali kriptografski žeton. Fizični žetoni so majhne naprave in so primerni za enostavno prenašanje, saj so posamezni narejeni kot obeski za ključe (angl. key fob). Nekatere od naprav shranjujejo kriptografske ključe ali biometrične podatke, druge pa prikazujejo PIN, ki se spreminja v časovnem intervalu. Kadar se želi uporabnik avtenticirati, uporabi PIN, ki je prikazan na zaslonu žetona, v kombinaciji z običajnim geslom za prijavo (Certic, 2013).

1.3 Programski žetoni

Programski žeton je aplikacija pri 2FA za preverjanje pristnosti, ki se uporablja za preverjanje pristnosti uporabnika za odobritev uporabe računalniških storitev. Programski žetoni so shranjeni na elektronski napravi, kot so računalniki in pametni telefoni in se lahko namestijo na večje število naprav oziroma se lahko podvojijo. Programski žetoni imajo prednost pred fizičnimi žetoni, ker so cenejši, predvsem pa jih lahko enostavno distribuiramo.

1.4 Biometrija

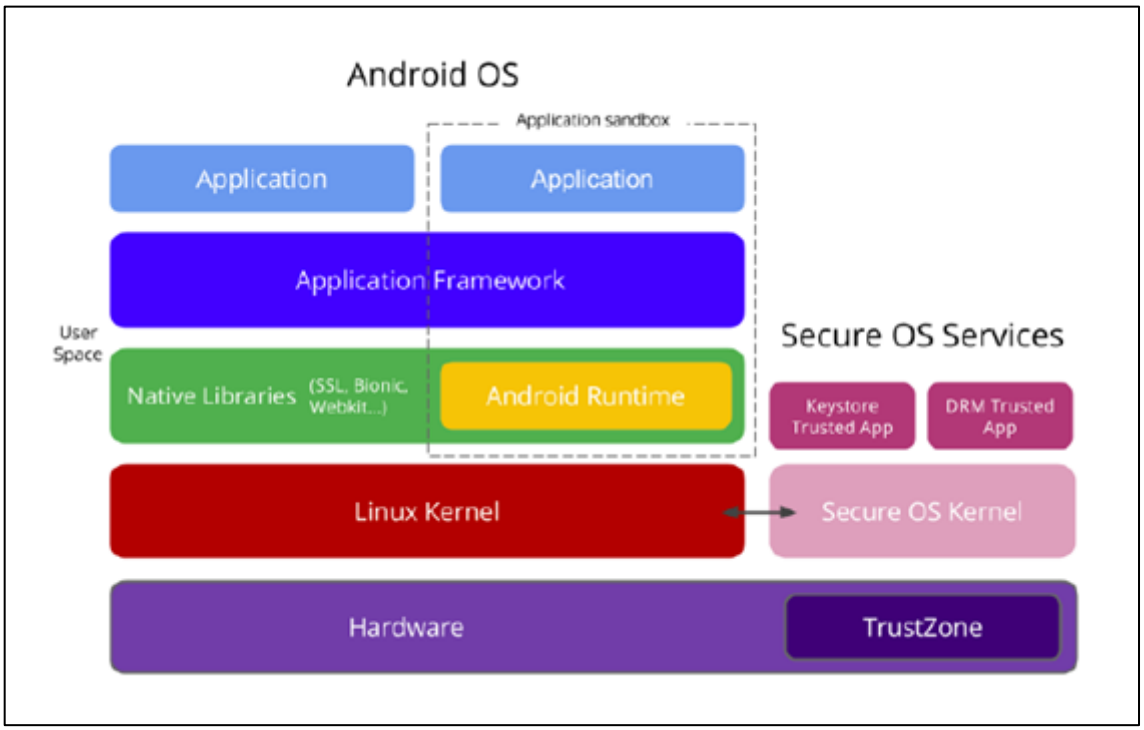
Biometrija je znanost o ugotavljanju identitete osebe, ki temelji na fizičnih lastnostih posameznika, kot so prstni odtisi, obraz, geometrija rok in oko, ali vedenjske lastnosti, kot so poteze podpisov, povezane s posameznikom. Tipični biometrični sistem uporablja ustrezno zasnovane senzorje za zajemanje biometričnih lastnosti osebe in ga primerja z informacijami, shranjenimi v podatkovni bazi, da se ugotovi identiteta. Biometrični sistem lahko deluje v dveh različnih načinih. V načinu preverjanja sistem potrdi ali zavrne zahtevano identiteto, medtem ko v identifikacijskem načinu določi identiteto posameznika (Ross, 2009).

Na mobilnih napravah biometrija vključuje uporabo katerekoli izvedljive biometrične modalnosti v mobilnih okoljih, vključno s pametnimi telefoni, tabličnimi računalniki in prenosnimi računalniki. Cilj je doseči enako biometrično funkcionalnost kot pri tradicionalnih sistemih, vendar z dodatkom prenosljivosti. To pomeni, da morajo biti biometrične mobilne rešitve enakovredne tradicionalnim po stopnji napak in prepustnosti, vendar z nekaterimi dodatnimi omejitvami, kot so nižja procesorska moč, manjša poraba energije in novi uporabniški vmesniki za pridobivanje podatkov interakcije med uporabnikom in napravo. Te omejitve so vodile do celovite modifikacije biometričnega podsistema, vključno z optimizacijo algoritmov, upoštevanjem novih scenarijev in primerov uporabe ter prilagojenimi biometričnimi senzorji predvsem zato, ker so pametni telefoni najbolj razširjene mobilne naprave in njihove značilnosti procesorska moč, spomin in zmožnosti povezljivosti v stalnem izboljševanju (Blanco-Gonzalo & Sachez-Reillo, 2014).

2 VARNOST NA OS ANDROID

Po avtoritativnem varnostnem poročilu (Android, 2016), katerega vsebina se v celotnem poglavju Varnost na OS Android navezuje, je Android odprtokodni operacijski sistem, ki je zgrajen na Linux jedru, v katerem je lahko istočasno zagnanih več aplikacij. OS Android je sestavljen iz šestih glavnih komponent aplikacije (glej Slika 1), aplikacijsko ogrodje, knjižnice, izvajalno okolje, Linux jedro in zagonsko okolje. Aplikacije delujejo tako, da so podpisane in izolirane v svojem aplikacijskem peskovniku (angl. application sandbox) ter povezane s svojimi aplikacijskimi podpisi. Aplikacijski peskovnik definira privilegije, ki so na voljo aplikaciji. Aplikacije so običajno zgrajene z uporabo orodja Android Runtime in medsebojna komunikacija z OS poteka prek aplikacijskega ogrodja (angl. application framework), ki opisuje sistemske storitve, programske vmesnike (angl. application programming interfaces) in obliko sporočil. Ostali programski jeziki višje ravni, kot so JavaScript in jeziki nižje ravni kot je ARM assembly, so dovoljeni in delujejo znotraj istega aplikacijskega peskovnika. Sistemske storitve se izvajajo kot aplikacije in so omejene na aplikacijski peskovnik. Nad jedrom ne obstaja koncept super ali korenskega (angl. root) uporabnika z neomejenim dostopom do sistema.

Slika 1: Shema operacijskega sistema Android



Vir: Android (2016).

Android je večnamenski operacijski sistem, vendar veliko naprav z OS Android zagotavlja sekundarno, izolirano okolje za izvajanje privilegiranih ali varnostno občutljivih operacij, ki ne potrebujejo funkcionalnosti večnamenskega operacijskega sistema. Takšno okolje se imenuje **Secure OS**. Te varnostne zahteve so implementirane na ločenem procesorju kot samostojni varnostni element (angl. Secure Element) ali zaupanja vredni modulni platformi (angl. Trusted Platform Module), lahko pa so izolirane pod jedrom na skupnem procesorju.

Proizvajalci originalne opreme (angl. original equipment manufacturer OEM) uporabljajo Secure OS za zagotavljanje varnih storitev in aplikacij, kot sta Widevine DRM-protected za predvajanje video posnetka in kriptografske storitve, prek vmesnika KeyChain API, kjer se zagotovi, da aplikacije ustvarijo ključe, ki jih ni mogoče izvoziti tudi v primeru ogroženega OS Androida.

2.1 Kriptografija in zaščita podatkov

Na celotni ravni Android uporablja kriptografijo, da se zagotovi zaupnost in integriteto. Pri tem je podprtih večino industrijskih standardnih algoritmov, v glavnem pa se uporablja kriptografija za:

- šifriranje naprav;
- podpisovanje aplikacij;

- mrežne povezljivosti in šifriranje z SSL, Wi-Fi in VPN.

2.1.1 Šifriranje naprav

Šifriranje je proces kodiranja uporabniških podatkov v napravi Android s šifrirnim ključem. Ko je naprava enkrat šifrirana, se vsi podatki, ki jih je ustvaril uporabnik, samodejno šifrirajo, preden se shranijo na disk, pri branju pa se podatki samodejno dešifrirajo.

Šifriranje diska temelji na dm-crypt, ki je funkcija jedra in deluje na nivoju bloka naprave. Šifrirni algoritem je 128 AES (angl. Advanced Encryption Standard) s šifrirnim veriženjem (angl. cipher-block chaining, CBC) in ESSIV: SHA256 (angl. Encrypted salt-sector initialization vector). Glavni ključ je šifriran z 128-bitnim AES prek klicev v Android OpenSSL knjižnico, pri tem pa proizvajalci opreme za šifriranje glavnega ključa lahko uporabijo 128-bitov ali več.

Med novejšimi funkcijami šifriranja so: hitro šifriranje, ki šifrira samo uporabljene bloke na podatkovni particiji, da se prepreči dolgotrajen prvi zagon, funkcija forceencrypt za šifriranje ob prvem zagonu, funkcija podpore za vzorce in šifriranje brez gesla in funkcija strojno podprte shrambe šifrirnega ključa.

V OS Android obstajajo štiri vrste šifrirnih stanj:

- privzeto;
- PIN;
- geslo;
- vzorec.

Če je v napravi omogočeno **privzeto** šifriranje, bo naprava ob prvem zagonu generirala 128-bitni ključ, ki je nato šifriran s privzetim geslom ter shranjen v šifrirnih metapodatkih. Podpora strojni opremi je implementirana z uporabo TET (angl.: Trusted Execution Environment's) podpisnih kapacitet. Ustvarjen 128-bitni ključ je veljaven do naslednje tovarniške ponastavitve oziroma dokler se ne izbriše podatkovna particija. Ob tovarniškem ponastavljanju se ponovno generira nov 128-bitni ključ.

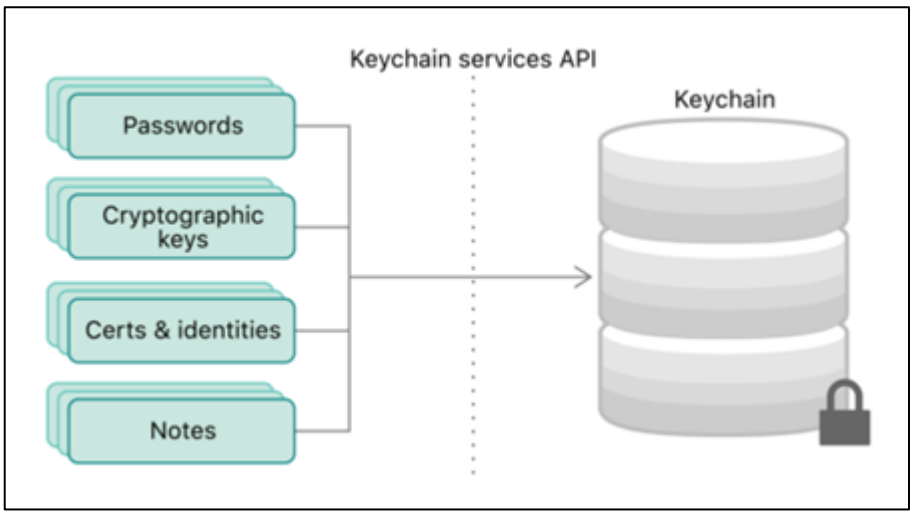
Ko uporabnik nastavi **PIN** ali **geslo** na napravi, je ponovno šifriran in shranjen le 128-bitni ključ, vendar pa sprememba poverilnic ne povzroči ponovnega šifriranja uporabniških podatkov. Dokument za definicijo združljivosti Android 5.0 (angl. Compatibility Definition Document-CDD) zahteva, da mora naprava v primeru nastavljenega zaklepanja zaslona podpirati šifriranje celotnega diska za zasebne podatke, to je /data in particijo SD kartice, če je stalni in neodstranljivi del naprave.

Pri tem je dobro vedeti, da šifrirni ključ ne sme biti nikoli shranjen nešifriran. Izjema je v aktivni uporabi, ko mora biti šifrirni ključ šifriran z AES enkripcijo s pomočjo zaslonskega gesla, pri katerem se uporabi počasni raztezni algoritem (angl. slow stretching algorithm). Če uporabnik ni navedel gesla za zaklepanje zaslona ali onemogočil uporabe gesla za šifriranje, bo sistem uporabil privzeto geslo za šifriranje šifrirnega ključa. Če naprava nudi strojno podprti ključ, mora biti algoritem za raztegovanje gesla kriptografsko povezan s tem ključem. Naprave, ki so šifrirane ob prvem zagonu, se po tovarniški ponastavitvi ne morejo vrniti v nešifrirano stanje.

Android ponuja nabor kriptografskih vmesnikov za namensko programiranje (angl. application programming interface, API), ki jih uporabljajo aplikacije. Vmesniki vključujejo implementacije standardnih in pogosto uporabljenih primarnih kriptografij, kot so AES, Rivest-Shamir-Adleman (RSA), digitalni podpisni algoritem (DSA) in Secure Hash algoritem (SHA). Poleg tega so vmesniki na voljo za protokole na višji stopnji, kot so zaščitni sloj (angl. secure socket layer, SSL) in HTTPS.

Razred **KeyChain** (glej Slika 2), omogoča aplikacijam, da uporabljajo sistemski prostor za shranjevanje poverilnic za zasebne ključe in certifikacijske verige. API KeyChain se uporablja za Wi-Fi in Virtual Private Network (VPN) certifikate.

Slika 2: KeyChain shema



Vir: Android (2016).

Razred **KeyStore** omogoča shranjevanje zasebnih ključev v zabojnik, da jih je težje potegniti iz naprave. Osredotoča se na aplikacije za shranjevanje poverilnic, ki se uporabljajo za preverjanje pristnosti, šifriranje ali podpisovanje.

Aplikacije lahko kličejo isBoundKeyAlgorithm metodo v KeyChain preden se uvozijo ali ustvarijo zasebni ključi danega algoritma, da se ugotovi, ali strojna oprema podpira shrambo ključev na način, da poveže ključe z napravo tako, da jih ni mogoče izvoziti.

2.2 Varnost aplikacij

Aplikacije so sestavni del katerekoli mobilne platforme in uporabniki jih na svoje naprave nameščajo vedno več. Android podpira večplastno zaščito aplikacij, ki omogoča, da uporabnik brezskrbno namesti najljubše aplikacije na svoj telefon, ker ima visoko raven zaščite pred zlonamerno programsko opremo, varnostnim izkoriščanjem in napadi.

2.2.1 Aplikacijski peskovnik in pravice

Android aplikacije so zagnane v aplikacijskem peskovniku. Tako kot stene peskovnika zadržujejo pesek pred razsipom, je vsaka aplikacija nameščena v navideznem peskovniku, s katerim se prepreči dostop ostalim zagnanim aplikacijam. Toda nekatere privzete aplikacije morajo uporabljati funkcionalnost na napravi, čeprav aplikacija ni v peskovniku, kot na primer dostop do kontaktnih informacij. Pred namestitvijo aplikacije se določi, ali ima lahko uporabnik dostop do aplikacije, kot je na primer vzpostavljanje telefonskih klicev. Aplikaciji za telefonski klic se tako seveda omogoči opravljanje telefonskih klicev. Na drugi strani pa moramo racionalno razsoditi, če dovolimo aplikaciji kot npr. igri Puzzle dostop do kontaktnih informacij, saj nam je to lahko sumljivo in ne dovolimo dostopa.

Veliko prednost Android platforme predstavlja uporabniško zasnovana Linux zaščita kot sredstvo za identifikacijo in izolacijo aplikacijskih virov. Sistem Android dodeli edinstven uporabniški ID (UID) vsaki Android aplikaciji in jo zažene kot uporabnik le-te v ločenem procesu. Ta pristop se razlikuje od drugih operacijskih sistemov vključno s tradicionalno konfiguracijo Linuxa, kjer je več aplikacij zagnanih z istim uporabnikom.

S tem se vzpostavi aplikacijski peskovnik na ravni jedra. Jedro zagotavlja varnost med aplikacijami in sistemom na ravni procesa s standardnimi objekti Linux-a, kot so uporabniški in skupinski ID-ji, ki se dodelijo aplikacijam. Aplikacije privzeto ne morejo medsebojno komunicirati, ker imajo omejen dostop do operacijskega sistema. Če na primer aplikacija A skuša narediti nekaj zlonamernega, kot je prebrati podatke aplikacije B ali vzpostaviti telefonski klic brez dovoljenja (kar je ločena aplikacija), potem operacijski sistem to prepreči, ker aplikacija A nima ustreznega uporabniškega dovoljenja. Peskovnik je preprost, preverljiv in temelji na desetletja starem UNIX uporabniškem ločevanju procesov in datotečnih dovoljenj.

Ker je aplikacijski peskovnik v jedru, se varnostni model razširi na prvotno kodo in OS aplikacije. Vsa programska oprema nad jedrom, vključno s knjižnicami OS, aplikacijskim

ogrodjem, izvajalnim okoljem in vsemi aplikacijami, se izvaja v aplikacijskem peskovniku. Na nekaterih platformah so razvijalci omejeni na poseben razvojni okvir, nabor API-jev ali jezik za uveljavljanje varnosti. V sistemu Android ni nobenih omejitev, kako naj bo aplikacija napisana za potrebe po uveljavljanju varnosti. Prvotna koda je prav tako varna kot prevedena koda. V nekaterih operacijskih sistemih napake pri pomnilniškem pomnilniku na splošno privedejo do popolnega ogrožanja varnosti naprave. To ne velja za Android zaradi vseh aplikacij, ki so dane v peskovnik na ravni OS. Napaka pomnjenja pomnilnika omogoča samo poljubno izvedbo kode v kontekstu določene aplikacije z določenimi dovoljenji s strani OS.

2.2.2 Varnostno izboljšani Linux

Kot del Androidovega varnostnega modela Android peskovnik uporablja varnostno izboljšani Linux (angl. Security Enhanced Linux, SELinux) za uveljavljanje obveznega nadzora dostopa (angl. Mandatory Access Control, MAC) do vseh procesov, tudi procesov, ki se izvajajo s privilegiji root in superuser. SELinux zagotavlja centralizirano analizno politiko in med seboj strogo ločene procese.

Android vključuje SELinux v načinu uveljavljanja, kot na primer varnostna politika je uveljavljena in prijavljena, ter ustrezen varnostni pravilnik, ki deluje privzeto skozi Android Open Source Project (AOSP). V načinu uveljavljanja se preprečijo nezakonite dejavnosti, ki kršijo politiko, in vse kršitve (zavrnitve) jedro prijavi na dmesg in logcat.

Android CDD zahteva, da naprave implementirajo politiko SELinux, ki omogoča, da se način SELinux nastavi za vsako domeno in se vse domene konfigurira v načinu uveljavljanja. Odprte domene niso dovoljene. Program za združljivostni test (CTS) za SELinux zagotavlja združljivost varnostne politike in uveljavlja najboljše prakse varnosti.

2.2.3 Podpisovanje aplikacij

Android zahteva, da so vse aplikacije digitalno podpisane s certifikatom, preden se lahko namestijo. Za certifikat ni potrebno, da je podpisan s strani overitelja certifikata. Android uporabi ta certifikat za identifikacijo aplikacijskega avtorja. Aplikacije pogosto uporabljajo samopodpisane certifikate, pri čemer ima razvijalec aplikacije zasebni ključ certifikata. Ko sistem namesti posodobitev aplikacije, primerja certifikat v novi različici s tistimi v obstoječi različici in če se certifikat ujema, sistem dovoli posodobitev.

Android dovoljuje aplikacijam, ki so podpisane z istim certifikatom, da se izvajajo v istem procesu, če aplikacije tako zahtevajo in jih sistem obravnava kot eno aplikacijo. Android podpira izvrševanje dovoljenj na podlagi podpisa, tako da lahko aplikacija razširi funkcionalnost na ostale aplikacije s specifičnim certifikatom. S podpisovanjem več

aplikacij z istim certifikatom in z uporabo dovoljenj na podlagi podpisa, lahko aplikacija deli kodo in podatke v varnem načinu.

Ključ mora imeti veljavnost, ki presega pričakovano življenjsko dobo aplikacije. Priporočeno obdobje veljavnosti je 25 let ali več. Ko poteče veljavnost ključa, uporabniki ne morejo več nadgraditi aplikacije na nove različice. Aplikacije, objavljene v Google Play, morajo biti podpisane s ključi z veljavnostjo vsaj do 22. oktobra 2033. Google Play uveljavlja to zahtevo, da zagotovi, da uporabniki lahko brez težav nadgradijo aplikacije, kadar je na voljo nova verzija.

2.2.4 Pregled aplikacije Google Play

Google Play je platforma za distribucijo aplikacij za Android, ki ščiti uporabnike pred potencialno škodljivimi aplikacijami. Google Play ima politiko za zaščito uporabnikov pred napadalci, ki poskušajo razširiti potencialno škodljive aplikacije ter validacijo razvijalcev v dveh fazah. Razvijalce se najprej preveri, ko ustvarijo svoj račun razvijalca za Google Play na podlagi svojega profila in kreditnih kartic. V drugi fazi se razvijalce preveri z dodatnimi signali pri predložitvi aplikacije. Google redno preverja aplikacije Play za zlonamerno programsko opremo in druge ranljivosti ter tudi ukine račune razvijalcev, ki kršijo politiko razvijalcev programov.

Google Play ima tudi ocenjevanje in pregled informacij o aplikaciji, preden se jo namesti. Če aplikacija poskuša zavesti uporabnike, bo verjetno imela nizko oceno in slabe komentarje.

Naprave s sistemom Android, v katerih je nameščen Google Play, imajo možnost, da uporabijo funkcijo za **preverjanje aplikacij**, ki preveri aplikacije ob namestitvi ter periodično pregleduje za potencialno škodljive aplikacije. Privzeto je preverjanje aplikacije vklopljeno, s čimer se zagotavlja, da so vse aplikacije v varnem načinu tudi po namestitvi, vendar se Googlu ne pošlje nobenega podatka, razen če se uporabnik ob namestitvi s tem strinja, preden namesti prvo aplikacijo iz vira, ki ni Google Play.

2.3 Varnost omrežja

Poleg zaščite podatkov, shranjenih v napravi, Android zagotavlja varnost omrežja za podatke v tranzitu, prejetih in poslanih. Android zagotavlja varno komunikacijo prek interneta za spletno brskanje, e-pošto, takojšnje sporočanje in druge internetne aplikacije, tako da podpira varnostni sloj prometa (angl. Transport Layer Security, TLS) in SSL.

2.3.1 Wi-fi

Android podpira protokol WPA2-Enterprise (802.11i), ki je posebej zasnovan za velika omrežja in se lahko integrira v široko paleto strežnikov za preverjanje pristnosti na klic (angl. Remote Authentication Dial-In User Service, RADIUS). V sistemu Android protokol WPA2-Enterprise podpira uporabo AES-128 šifriranja, s čimer uporabnikom zagotavlja visoko stopnjo zaščite pri pošiljanju in prejemanju podatkov prek omrežja Wi-Fi.

Android podpira 802.1x (angl. Extensible Authentication Protocols, v nadaljevanju EAP), vključujoč EAP-TLS, EAP-TTLS, PEAPv0, PEAPv1, in EAP-SIM.

2.3.2 VPN

Android zagotavlja omrežno varnost z uporabo VPN:

- **Always-on VPN** Vedno vključen VPN – VPN je mogoče konfigurirati tako, da aplikacije nimajo dostopa do omrežja, dokler se ne vzpostavi povezava VPN, kar preprečuje aplikacijam pošiljanje podatkov v druga omrežja.
- **Per User VPN** Na uporabniški VPN – Na večuporabniških napravah se VPN uporablja na uporabnika in je ves omrežni promet usmerjen prek VPN, ne da bi to vplivalo na druge uporabnike.
- **Per Profile VPN** Na profil VPN – VPN se uporablja na delovni profil, kar omogoča skrbniku IT zagotoviti, da gre samo omrežni promet skozi delovni profil VPN, ne pa omrežni promet uporabnika.
- **Per Application VPN** Na aplikacijo VPN – Nudi podporo za enostavne VPN povezave za dovoljene aplikacije ali prepreči VPN povezavo za prepovedane aplikacije.

2.3.3 Aplikacije tretjih oseb

Google je povečal uporabo TLS/SSL v vseh aplikacijah in storitvah. Ker aplikacije postajajo bolj kompleksne in povezane na več naprav, je enostavnejše aplikacijam predstaviti omrežne napake s pravilno uporabo TLS/SSL.

V ta namen je bilo razvito orodje, imenovano NoGoToFail, ki omogoča preprost način za potrditev, da so naprave ali aplikacije varne pred znanimi ranljivostmi in napačnimi konfiguracijami TLS/SSL.

Orodje NoGoToFail deluje na Android in drugih operacijskih sistemih kot preprost uporabniški odjemalec, kjer nastavimo nastavitve in dobimo obvestila. Orodje NoGoToFail

je bilo izdano kot odprtokodni projekt, da lahko razvijalci aplikacij testirajo svoje aplikacije, razvijejo nove funkcije za projekt in pomagajo izboljšati varnost omrežja v sistemu Android.

2.4 Upravljanje naprav in profilov

Android je vpeljal koncept lastnika naprave in profil lastnika v podporo napravam v službeni lasti in v primerih, ko je naprava last uporabnika. Koncept upravljanja profilov temelji na več uporabniškem konceptu Android, ki je bil prvič predstavljen v verziji Androida 4.2.

2.4.1 Uporabniki Android

Uporabnik na sistemu Android je namenjen za uporabo različnim osebam in njihovim aplikacijskim podatkom, edinstvenim nastavitvam in uporabniškem vmesniku za preklapljanje med njimi. Kadar je uporabnik aktiven, drugi uporabnik ostane zagnan v ozadju in podatki uporabnika so vedno izolirani od drugih uporabnikov. Podprta sta dva tipa uporabnikov:

- **Primarni uporabnik** je prvi uporabnik, dodan v napravo. Ni ga mogoče odstraniti, razen s tovarniško ponastavitvijo. Ta uporabnik ima tudi posebne pravice in nastavitve, ki jih je določil samo ta uporabnik. Primarni uporabnik se vedno izvaja, tudi če so v ospredju zagnani drugi uporabniki.
- **Sekundarni uporabnik** je kateri koli uporabnik, dodan v napravo, ki ni primarni uporabnik. Sekundni uporabnik se lahko odstrani sam ali s primarnim uporabnikom, vendar ne more vplivati na druge uporabnike v napravi. Sekundarni uporabnik lahko deluje v ozadju in ima še vedno povezavo z omrežjem, ko jo potrebuje. Vendar pa obstajajo nekatere omejitve, in sicer da ne more prikazati uporabniškega vmesnika ali v ozadju aktivno storitev Bluetooth™. Sistemski proces lahko ustavi izvajanje sekundarnega uporabnika v ozadju, če naprava potrebuje dodaten pomnilnik za operacije uporabnika v ospredju.

2.4.2 Upravljani profil

Odjemalec pravilnika za naprave (angl. Device Policy Client, v nadaljevanju DPC) je aplikacija, ki se uporablja za upravljanje korporacijskega prostora v napravi. DPC ima dostop do API-ja za upravljanje naprav, ki so na voljo v razredu DevicePolicyManager in prejema povratne klice iz sistema prek razreda DeviceAdminReceiver.

Delovni profil (angl. Work Profile) je upravljan profil, ustvarjen, ko DPC sproži upravljani tok podpore. V tem primeru deluje delovni profil kot navaden uporabnik, vendar je povezan s primarnim uporabnikom tako, da so obvestila in nedavni seznam opravil v skupni rabi.

Aplikacije, obvestila in pripomočki iz upravljanega profila so vedno označeni. Ker je delovni profil ločen Android uporabnik, obstaja močna ločitev med službenim in osebnim profilom, vsi podatki v delovnem profilu pa se upravljajo ločeno.

Lastnik profila (angl. Profile Owner) je primer, v katerem upravitelj naprave lahko upravlja samo korporativen prostor na osebni napravi uporabnika, kadar uporabnik uporablja svojo napravo za službene namene (angl. bring your own device, BYOD). Lastniki profilov so podani v delovni profil in jih je mogoče opredeliti le kot del pripravljalnega postopka. V praksi je uporabniku tako omogočen enostaven dostop do osebnih in delovnih profilov hkrati. Uporabnik ne more deaktivirati profila lastnika, vendar lahko vedno pogleda in potrdi nastavitve, ki se izvajajo v delovnem profilu, in lahko izbere, da kadarkoli po želji odstrani delovni profil in profil lastnika.

Lastnik naprave je kot profil lastnika, vendar je namenjen celotni napravi. Lastnik naprave je skrbnik naprave v primeru uporabe naprave v lasti podjetja.

V primeru BYOD so podatki v delovnem profilu ločeni od osebnih podatkov uporabnika. Vendar pa obstajajo primeri, ko je uporaba drugih namenov v enem profilu lahko uporabna in izboljšuje produktivnost podjetja.

V Android ima več varnostnih politik in nastavitev za upravljanje naprave in profilov. IT skrbniki lahko politike posredno nastavijo prek aplikacije za upravljanje mobilnih naprav (angl. mobile device management, MDM) za zaščito podatkov na napravah.

3 ORODJA ZA ANALIZO ZLONAMERNE KODE NA OS ANDROID

Varnostni model OS Android je podoben Linuxu, saj je tudi osnovan na Linux jedru. Glavni varnostni deli Androida vključujejo predvsem peskovnik, podpis aplikacij in mehanizem za dovoljenja. Mehanizem za dovoljenje omejuje aplikacijam dostop do zasebnih podatkov uporabnika (npr. telefonske številke, stike itd.), virov (tj. datoteke dnevnika) in sistemskega vmesnika (npr. internet, GPS itd.). V mehanizmu dovoljenj so viri telefona organizirani z drugačnimi kategorije in vsaka kategorija ustreza eni vrsti dostopnega vira.

Če aplikacija zahteva dostop do določena vira, mora imeti ustrezna dovoljenja. Čeprav je ta mehanizem preprost, ima tudi nekaj napak in ne more ustrezno zaščititi osebnih podatkov uporabnika. Nekateri raziskovalci, Singh, Tiwari & Singh (2016), so pregledali model varnosti Android in poudarili, da trenutni model dovoljenj za Android ne more izpolniti določenih varnostnih zahtev.

Zahtevana dovoljenja za aplikacijo morajo biti pred namestitvijo odobrena in jih ni mogoče spremeniti po namestitvi. Ta model dovoljenj vodi do nekaterih možnih varnostnih groženj. Po eni strani so dovoljenja za dostop do zasebnih podatkov določena s strani uporabnika. Za uporabnike, ki ne vedo, kaj pomeni varnost, je postopek podeljevanja dovoljenj nič posebnega in jih dodeljujejo na slepo. Če med fazo namestitve program pridobi dovoljenja za dostop do podatkov o zasebnosti, potem lahko samovoljno zlorablja zasebnost in zelo občutljive podatke uporabnika. Drugače povedano, varnostni mehanizem ne more učinkovito preprečiti stopnjevalnih napadov za pravice, aplikacije lahko tako izkoristijo kombinacijo dovoljenj za krajo občutljivih podatkov uporabnika. Da bi razkrili uhajanje informacij o zasebnosti uporabnikov v aplikacijah Android v skladu z varnostnim mehanizmom Android OS, sta za ta namen uporabljeni dinamična in statična metoda, ki ju bomo podrobneje obravnavali v nadaljevanju (Singh, Tiwari & Singh, 2016).

3.1 Zlonamerna koda na OS Android

Da bi lahko razvili zaščito pred zlonamerno programsko opremo, kot so virusi, črvi in trojanski konji, je potrebno pridobiti poglobljeno razumevanje delovanja le-teh. Načini, s katerimi lahko vzpostavimo obrambo, so protivirusni programi, ki lahko pregledajo izvedljive datoteke za sumljive vzorce kod ali reagirajo na obnašanje programa z uporabo znanj, kako zlonamerna programska oprema deluje. Način pridobivanja teh znanj je analiza zlonamerne programske opreme, ki temelji na tehnikah povratnega inženirstva in študiju vedenja programa. V praksi se to imenuje statična analiza in dinamična analiza (Hell & Altman, 2015).

3.2 Statična analiza

Največja prednost statične analize je, ker aplikacija ni zagnana in sistem ni izpostavljen morebitni zlonamerni programski kodi. Statična analiza temelji na preiskovanju izvorne programske kode na splošno ali podrobno. Hitra in osnovna statična analiza programa je izvleči potencialne nize, vdelane v binarno datoteko, da se poišče specifične vzorce ali pregleda, katere programske knjižnice program poskuša uvoziti ob zagonu. V primeru, da ne moremo dobiti izvorne kode s ponovnim prevajanje (angl. decompiling) ali s povratnim inženirstvom, je aplikacije težko natančno analizirati, še posebej, če je ciljni program zlonamerna datoteka. Pri podrobni analizi se uporabi razstavljalska (angl. deassembly) programska oprema IDA pro ali Radare2, da pridobimo sestavljeno (angl. assembly) kodo za program (Hell & Altman, 2015).

Pred filtriranjem pravic in dinamičnim spremljanjem moramo razširiti Android aplikacijsko datoteko AndroidManifest.xml in smali datoteke ekvivalentno za APK. Android aplikacija je inštalacijski paket s končnico .apk, kar je okrajšava za Android Package. APK je enak kot .exe datoteka v računalniku in se jo lahko po inštalaciji v OS Android takoj zažene. APK je

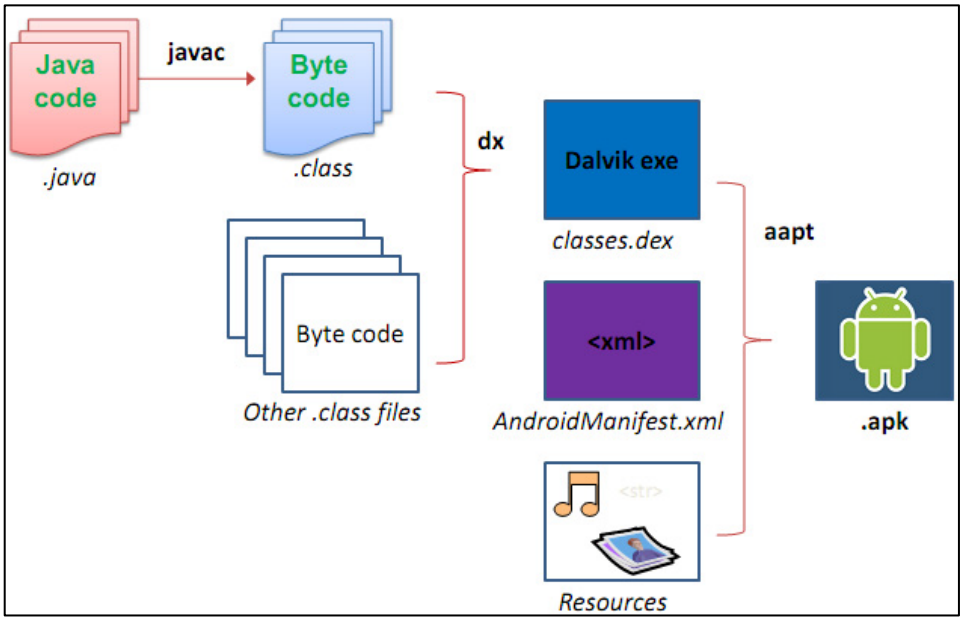
pravzaprav stisnjena datoteka, skladna z ZIP formatom, ki se lahko razširi s popularnim .zip kompatibilnim dekompresijskim orodjem. Poleg tega je treba opozoriti, da je večina aplikacij kodiranih in datotek se ne more direktno analizirati. Struktura .APK datoteke je po razstavljanju prikazana v spodnji tabeli in sliki (Singh, 2016).

Tabela 1: Struktura datotek po razstavitvi APK

Mapa/Datoteka	**Opis**
res	Datoteka virov aplikacije, vključno s slikami, zvokom, videoposnetki itd.
smali	Dalvik register bytecode APK datoteke
AndroidManifest.xml	Globalna konfiguracijska datoteka APK, vključno z imenom paketa, dovoljenji, referenčnimi knjižnicami in drugimi povezanimi informacijami aplikacije.
Apktool.yml	Konfiguracijska datoteka Apktoola

Vir: Singh, Tiwari & Singh (2016).

Slika 3: Koraki za pretvorbo v APK



Vir: Singh, Tiwari & Singh (2016).

V modulu za dovoljenja obstaja nekaj dovoljenj, ki ne predstavljajo tveganja sama po sebi, vendar v različnih kombinacijah lahko pomenijo varnostno tveganje. Kot na primer aplikacija ima pravice branja stanja telefona in pošiljanja sporočil, pri čemer lahko obstaja možnost posredovanja telefonske številke ali IMEI-a naprave (angl. International Mobile Equipment Identity). Modul dovoljenj temelji na nizu varnostnih pravil, s katerimi ugotovi, ali ima aplikacija kaj posebnih kombinacij pravic, izpostavljenih tveganju. Za vsa dovoljenja

za Android obstajajo štiri vrste varnostnih ravni. To so normalna, nevarna, podpisi ter podpisi in sistem (Singh, Tiwari & Singh, 2016).

3.2.1 Orodja za statično analizo

VirusTotal je spletna storitev, ki s pomočjo velikega izbora protivirusne programske opreme pregleda potencialno zlonamerne binarne datoteke in URL-je. Poleg rezultatov odkrivanja različnih proizvajalcev AV se izvede tudi avtomatizirana statična analiza in omejena dinamična analiza. Uporabimo ga, da bi dobili natančno poročilo za statično analizo. Za Android aplikacije pridobljene informacije vključujejo potrebna dovoljenja, povezana dovoljenja s klici API, obstoječe dejavnosti, storitve in sprejemnike, opazovanje kod, informacije v zvezi s certifikati itd. (VirusTotal, 2017).

Keytool je orodje za upravljanje ključev in certifikatov. Uporabnikom omogoča, da upravljajo svoje lastne pare ključev (javni/zasebni) in s tem povezana potrdila, ki se uporabljajo za avtentikacijo ali integriteto podatkov in avtentikacijo servisov z digitalnimi podpisi. Prav tako omogoča uporabnikom, da shranijo javne ključe komunikacijskih partnerjev v obliki certifikatov. Najkoristnejša uporaba tega orodja je za pridobivanje podatkov o certifikatu iz aplikacije, saj morajo biti vse aplikacije podpisane, preden se lahko namestijo (Oracle, 2017a).

Jarsigner je orodje za podpisovanje in preverjanje Java .jar arhivov z javnim in zasebnim ključem. Uporabi se za podpis modificirane .apk datoteke z novima ključema, da se omogoči namestitev aplikacije v napravo (Oracle, 2017b).

Strings je orodje za pridobivanje potencialnih nizov iz binarnih datotek s skeniranjem za bajtne sekvence, ki jih je mogoče razlagati kot vgrajeni niz. Uporablja se za pridobitev dodatnih informacij, kakšen namen ima aplikacija med statično analizo (The Open Group, 2017).

Dex2jar je orodje za pretvorbo Android izvajalskih razredov .dex v Javo .jar arhiv. Odkar je za .jar arhive potrebna uporaba prevajalskega orodja Java JD-GUI, se mora uporabiti dex2jar orodje (Github, 2017).

JD-GUI je orodje za prevajanje Java .jar arhivov v Java kodo, vendar pa lahko koda vsebuje napake, ker je rekonstruirana iz bajtne kode. Orodje ponuja tudi grafični vmesnik za enostavnejšo brskanje po konstruirani kodi (Java Decompiler, 2017).

APKTool je orodje za razstavljanje in obnovo .apk arhivov in se uporablja po spremembi aplikacijske .smali sestavljene kode z namenom preveritve pravilne obnove (Ibotpeaches, 2017).

3.3 Dinamična analiza

Dinamična analiza testira uspešnost delovanja, kadar je aplikacija zagnana. Metoda je zelo natančna pri iskanju zlonamerne kode in njenega obnašanja, vendar ima pomanjkljivost, ker ne more zagotoviti, da so bile med testiranjem preiskane vse poti, ki naj bi jih sprožila zlonamerna koda. Za uspešno analizo je potrebna kombinacija statičnega in dinamičnega modela varnostne analize, ki se med seboj dopolnjujeta zaradi svojih pomanjkljivosti, da dosežemo uspešnejšo in pravilno opravljeno analizo zlonamerne kode. Dinamična analiza poteka tako, da je pred analizo Android aplikacije potrebno ponovno prevajanje APK (angl. android application package), da dobimo ustrezno konfiguracijo in Smali datoteke. Znotraj konfiguracijskih datotek se nahaja datoteka v formatu AndroidManifest.xml, ki se v glavnem in predvsem uporablja za fazo filtriranja dovoljenj, smali datoteke pa se uporabljajo za dinamično spremljanje. Za analizo najprej izberemo sumljive aplikacije, za katere menimo, da so potencialne zaradi uhajanja uporabnikovih podatkov. Če se nam zdi aplikacija sumljiva, izvedemo dinamično spremljanje modula, kjer vnesemo želeno smali datoteko in kodo za sledenje ter prepakiramo in ponovno podpišemo APK. Ko je enkrat APK zagnan, lahko dinamično spremljamo obnašanje za uhajanje zasebnosti, pri čemer se v primeru okužbe nemudoma sproži alarm. Opozorila ali dnevnike lahko pozneje uporabimo za nadaljnjo podrobno analizo (Singh, Tiwari & Singh, 2016).

Ko pri dinamičnem modulu vstavimo v razširjene APK datoteke željeno kodo, spremljamo v trenutnem, realnem času. Android razvijalci napišejo aplikacijo v Javi, jo prevedejo v Java bajtno kodo in jo na koncu prenesejo v Dalvikovo bajtno kodo, ki se jo lahko izvede v DVM (angl. Dalvik Virtual Machine). Pri tem je enostavno narediti obratno inženirstvo s pretvorbo Dalvikove bajtne kode v Java bajtno kodo in nato ponovno napisati Java bajtno kodo in na koncu pretvoriti Java bajtno kodo nazaj v Dalvik bajtno kodo. Vendar ta metoda ne deluje vedno, ker obstaja kar nekaj pomembnih razlik med Java Virtual Machine in Dalvik Virtual Machine. JVM temelji na skladu, medtem ko DVM temelji na registru. Na voljo so številna orodja, kot je dex2jar in ded, ki pretvorijo Dalvikovo bajtno kodo v Java bajtno kodo. Še vedno pa to ni pretvorba brez izgub, ker se nekatere informacije iz Java byte kode izgubijo ob pretvorbi v Dalvik. Ta orodja poskušajo zbrati izgubljene podatke, ki temeljijo na kontekstu, toda včasih je sklepanje pomanjkljivo. Medtem ko te napake ne prepreči statična analiza pretvorjene Java bajtne kode, je izkušnja taka, da vodi do neveljavne bajtne Java kode ali kasneje neveljavne Dalvik bajtne kode. Ko spremenimo aplikacijsko Dalvik bajtno kodo v Java bajtno kodo z dex2jar in nato nazaj na Dalvik bajtno kodo, rezultira, da aplikacija ne deluje pravilno. Možna pot je, da neposredno uporabimo Dalvikovo bajtno kodo. Smali in baksmali sta assembler in disassembler za dex format, uporabljen v DVM. Smali je posredni prikaz Dalvikove bajtne kode in lahko v celoti prikaže vse značilnosti dexovega formata, kot so opombe, informacije za odpravljanje napak in niti. Poleg tega lahko dex in Smali med seboj pretvarjata brez izgub (Singh, Tiwari & Singh, 2016).

3.3.1 Orodja za dinamično analizo

Android Emulator – Android SDK (angl. software development kit) vključuje Android AVD (angl. Virtual Device Manager) za kreiranje in izvajanje emuliranih naprav v računalniku z namenom testiranja aplikacij. Analiza aplikacij se lahko izvede tudi na fizični napravi, vendar pa virtualno okolje omogoča hitro in enostavno ponastavitev slik OS (Developer Android, 2017a).

Wireshark – to orodje lahko zajame ves omrežni promet na katerem koli mrežnem vmesniku in ga prikaže na berljiv način. Uporabi se za spremljanje in analiziranje vhodnega in odhodnega omrežnega prometa iz emulirane naprave (Wireshark, 2017).

Eclipse – Java IDE (angl. integrated development environment) z vgrajenim vtičnikom Android SDK in ki je zagnan z vidika Dalvik Debug Monitor Server (DDMS). Ta orodja se bodo uporabljala za spremljanje emulirane Android naprave in za pošiljanje lažnih SMS-ov ter za izvedbo lažnih telefonskih klicev (Eclipse, 2017).

ADB – Android Debug Bridge se uporablja za namestitev aplikacij in izvrševanje ukazov na emulirani napravi. Uporabi se za ustvarjanje log datotek in namestitev aplikacij (Developer Android, 2017b).

4 DVOKAFTORSKA AVTENTIKACIJA NA PAMETNIH TELEFONIH

V tem poglavju se bomo seznanili z štirimi načini 2FA za pametnih telefonih. Najstarejši način je preverjanje pristnosti s prejemanjem SMS-a, naslovljen kot mTAN. Drugi način je zelo priljubljen in je standardizirana shema avtentikacije generiranja gesel na pametnih telefonih. To je metoda TOTP. Tretji način je GA, ki temelji na običajno uporabljenih oblikah 2FA, ki jih uporabljajo javne spletne storitve. Zadnji način temelji na PKI oziroma na infrastrukturi javnega ključa.

4.1 mTAN

mTAN (angl. Mobile Transaction Authentication Number) je način preverjanja pristnosti, ki temelji na dostavi OTP preko komunikacijskega kanala uporabniku. Komunikacijski kanal v primeru mTAN je telefonsko omrežje v obliki SMS-a. Ko se začne preverjanje pristnosti in uporabnik vnese svoje poverilnice, ponudnik storitve generira OTP in ga pošlje na predregistrirano telefonsko številko uporabnika. OTP bo nato veljal za določen čas, ki bo odvisen od ponudnika storitev (Hell & Altman, 2015).

mTAN temelji na enostavnem načelu, ko uporabnik enkrat na varen način dokaže lastništvo svojega mobilnega telefona in se le-tega lahko uporabi kot drugi faktor za prijavo. mTAN uporablja enkratno kodo SMS, ki se generira na strežniku in se jo posreduje na telefon uporabnika med vsakim poskusom prijave. Nekateri pristopi uporabljajo načelo ključne kontinuitete (angl. principle of key continuity) za dosego dogovora o žetonu, ki je uporabljen kot drugi faktor za prijavo. Ta model je že bil uspešno implementiran v protokolih, kot sta SSH ali zFone avtorja Phila Zimmermana. Zamisel tega pristopa je, da če lahko zanesljivo potrdimo eno sejo (običajno prvo) in dobimo skupno skrivnost, se lahko ta skrivnost ponovno uporabi za preverjanje poznejših sej – tj. vsaka skrivnost S_n je neposredno odvisna od prejšnje skrivnosti S_{n-1} (DSwiss Ltd., 2010).

Ob predpostavki, da je skupna skrivnost S_{n-1} varno shranjena v mobilni napravi, lahko drugi faktor za prijavo generira sama mobilna naprava, tj. naslednjo skupno skrivnost S_n, to pa zato, ker je odvisna od nečesa, kar uporabnik ima. Z varnostnega vidika je to zelo podobno kot mTAN, vendar ponuja veliko boljšo uporabnost (DSwiss Ltd., 2010).

Podrobnejši pristop deluje na naslednji način: Pred začetkom nove seje n je mobilna naprava shranila skupno skrivnost S_{n-1} iz prejšnje seje n-1. S_{n-1} se nato uporabi za izračun skupne skrivnosti za preverjanje pristnosti nove seje z uporabo formule: S_n = f (<skupna skrivnost>, S_{n-1}). Dokler je na mobilni napravi ustrezno zaščiten žeton S_{n-1}, dokazovanje poznavanja o njem zadostuje za drugi faktor za prijavo, ki ga odjemalec opravi z izračunom in zagotavljanjem Sn (DSwiss Ltd., 2010).

Tabela 2: Delovanje načela ključne kontinuitete

Client		**Server**
(looks up S_{n-1})		(looks up S_{n-1})
(computes S_n = f(<shared secret>, S_{n-1})		(computes S_n = f(<shared secret>, S_{n-1})
	S_n→	
		(checks S_n == S_n)
	←OK	
(stores S_n)		(stores S_n)

Vir: DSwiss Ltd. (2010).

4.2 TOTP

TOTP (angl. Time-based One-time Password Algorithm) ali časovni algoritem enkratnega gesla, v katerega sodita RSA SecureID in HID ActiveID (glej Slika 4) je standardiziran način uporabe kriptografske funkcije hash za izračunavanje OTP na osnovi preddefinirane

skrivnosti (angl. preshered secret) in trenutnega časa. Gre za spremembo standarda HOTP, ki namesto trenutnega časa uporablja števec na osnovi vprašanja. Prednost uporabe trenutnega časa namesto števca je, da se stalno spreminja in samodejno zagotavlja nove vrednosti. OTP običajno velja za določeno časovno obdobje. Strežnik za preverjanje pristnosti opravi enake izračune, kot jih opravi pri uporabniku na napravi, in ker imata isto skrivnost in trenutni čas, bodo kode enake.

Moč TOTP je, da ne potrebuje nobene oblike omrežne povezave, da bi lahko generiral nove kode in dokler je ura naprave sinhronizirana s strežnikom, bo še naprej ustvarjala veljavne OTP.

TOTP je opredeljen kot razširitev HOTP, ki ustvarja svoje vrednosti s pomočjo HMAC-SHA-1 (angl. hash-based message authentication code) na naslednji način v enačbi (1):

$$HOTP(K;C) = Truncate(HMAC\text{-}SHA\text{-}1(K,C)) \quad (1)$$

Kjer je funkcija Truncate način pretvorbe izhoda HMAC-SHA-1 v vrednost HOTP. K in C predstavlja skrivnost in števec.
V TOTP se uporablja enaka metoda kot zgoraj, števec pa se nadomesti z vrednostjo T, ki izhaja iz časovne reference in časovnega koraka. TOTP je zato opredeljen v enačbi (2) kot:

$$TOTP(K, T) = HOTP(K; T) \quad (2)$$

Kjer je T definiran z enačbo (3):

$$T = (Current\ Unix\ time - T0) / X \quad (3)$$

Kjer je referenca časa T0, privzeto opredeljena kot Unixova epoka, tj. 0. Trenutni čas Unixa (angl. Current Unix time) je število sekund, ki so pretekle od Unix epoke, tj. od 1. januarja 1970. Časovni korak X je privzeto določen kot 30 sekund. Celoštevilčna delitev se uporablja pri izračunu T, kar pomeni, da npr. če je T0 = 0, X = 30 in trenutni Unix čas je 59 sekund, T = 1. Če bi bil trenutni Unix čas, bi bil 60 sekund, T = 2 (Hell & Altman, 2015).

4.3 Google avtentikator

Google Authenticator (glej Slika 4) je programski generator gesel za izvajanje 2FA z uporabo časovnega algoritma (TOTP) in HMAC algoritma za preverjanje pristnosti uporabnikov.

Generator ponuja šest do osemmestno enkratno geslo, ki ga morajo uporabniki poleg svojega uporabniškega imena in gesla vnesti za prijavo v Googlove storitve ali druga spletna mesta.

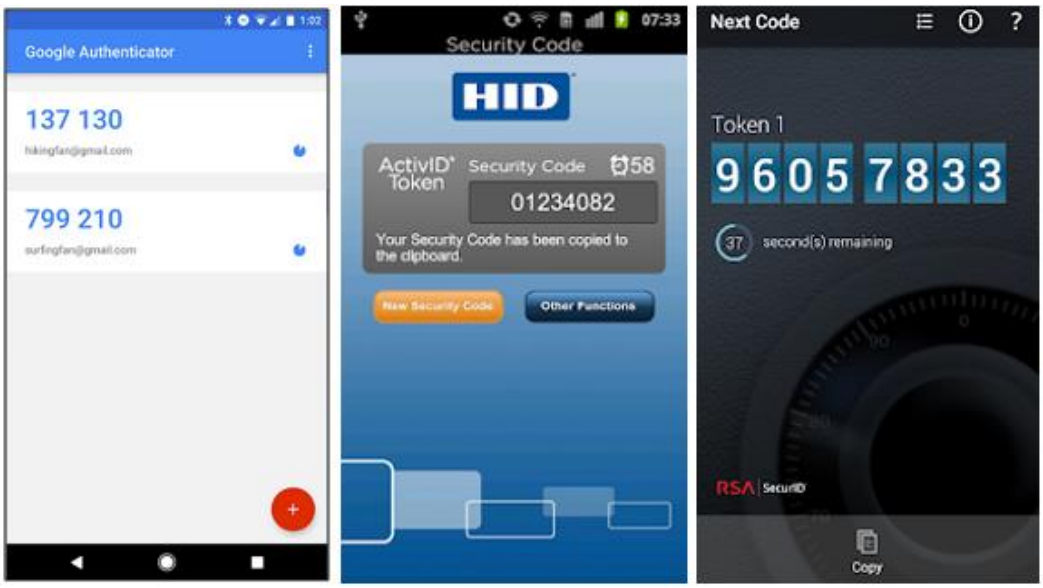
Generator lahko ustvari tudi kodo za aplikacije tretjih oseb, kot so upravitelji gesel ali storitve v oblaku.

Običajno uporabnik namesti aplikacijo Goolge Authenticator na pametni telefon. Če se želi prijaviti na spletno mesto ali storitev, ki uporablja 2FA preverjanja pristnosti, uporabnik za spletno mesto vnese uporabniško ime in geslo ter dodatno šestmestno enkratno geslo, ki ga generira aplikacija Google Authenticator. Isto geslo je neodvisno ustvarilo spletno mesto, ki ga od uporabnika zahteva. Uporabnik ga vnese in s tem potrdi svojo identiteto.

Da bi to delovalo, je potrebno opraviti inicializacijo skupnega skritega ključa med aplikacijo Google Authenticator in spletnim mestom prek varnega kanala, ki se shrani v aplikacijo in bo uporabljen za vse prihodnje prijave.

S to vrsto dvostopenjskega preverjanja pristnosti, poznavanje uporabniškega imena in gesla ne zadostuje napadalcu za prijavo v uporabniški račun. Napadalec potrebuje tudi poznavanje skupnega skrivnega ključa ali fizičnega dostopa do mobilne naprave. Alternativna pot napada je lahko napad mož na sredini (angl. man-in-the-middle attack) v primeru, če je računalnik okužen s škodljivo programsko kodo, potem lahko koda prestreže uporabniško ime in geslo ter lahko začne lastno sejo za prijavo na spletno mesto ali spremlja in spreminja komunikacijo med uporabnikom in spletno stranjo (Willis, 2014).

Slika 4: Google, HID ActiveID, RSA SecureID avtentikatorji



Vir: Google Play (2018).

4.4 PKI dvofaktorska prijava

Infrastruktura javnega ključa (glej Slika 5) je sistem za vzdrževanje varnega omrežnega okolja z uporabo šifriranja javnih ključev. Elementi PKI (angl. Public Key Infrastructure) se

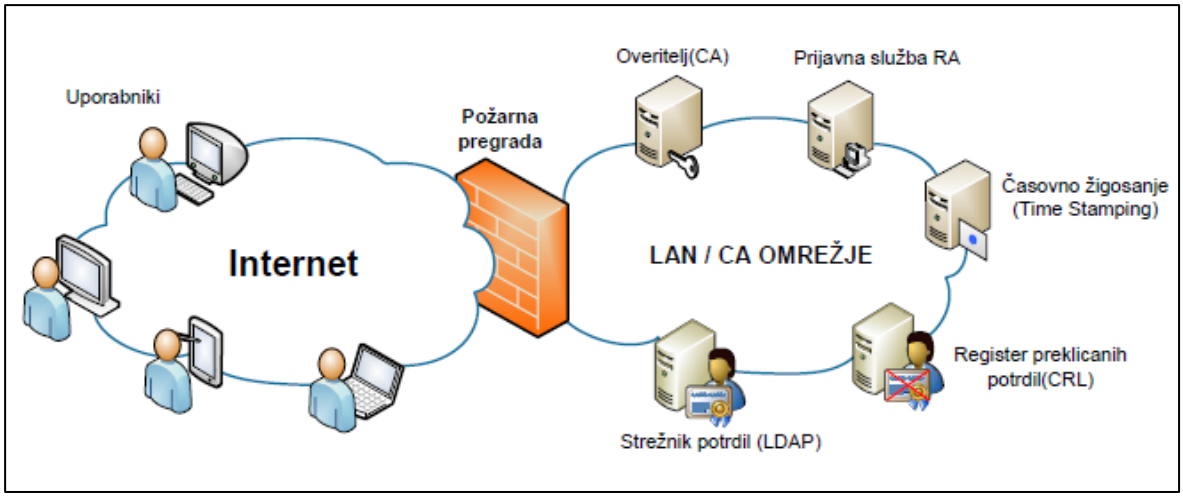
uporabljajo za upravljanje identitet, ki jih potrjuje CA (angl. Certificate Authorities) in delujejo kot zaupanja vredna tretja oseba. Rešitve za 2FA se lahko s certifikatom gradijo na obstoječi PKI (Karanjgaokar, 2016).

Izbira tehnologije PKI za 2FA je odvisna od grožnje za funkcijo, ki jo poskušamo zaščititi. Dojemanje tveganja je precej visoko, kadar organizacije želijo biti zagotovo prepričane o uporabniški identiteti ali kadar se identiteta in podatki delijo z več aplikacijami. Prenos sredstev tretje osebe z uporabnikovega bančnega računa z uporabo mrežnega bančništva je kot primer uporabe 2FA za preverjanje pristnosti na podlagi PKI (Karanjgaokar, 2016).

V primerih, ko je potrebno isto preverjanje pristnosti razširjati v več različnih aplikacij, je lahko tehnologija PKI zelo priročna, saj gre za mehanizem, ki temelji na standardih, in je sprejet po vsem svetu. To pomaga rešiti vprašanja interoperabilnosti[2], ki jih lahko predstavljajo drugi mehanizmi za preverjanje pristnosti, kot je OTP (Karanjgaokar, 2016).

Standard X.509 certifikatov odjemalca za preverjanje pristnosti ni v razširjeni distribuciji, razen v namensko izdelanih produktih na zahtevo stranke. Večinoma jih najdemo na področjih z omejenim dostopom, običajno v obliki pametnih kartic (Zink & Waldvogel, 2017).

Slika 5: PKI infrastruktura



Vir: Karanjgaokar (2016).

Prijava s pametno kartico je izbirna funkcija nekaterih sistemov, ki uporabnikom omogoča, da se v operacijski sistem prijavijo s pametno kartico in PIN-om. Nadomešča privzeto uporabniško ime in geslo za prijavo. Ker ima pametna kartica nameščen zasebni ključ, nekaj kar imaš, in zahteva identifikacijo PIN-a, nekaj kar veš, je to pravilen dvofaktorski

[2]interoperabilnost je definirana kot zmožnost informacijskih sistemov in poslovnih procesov, ki jih ti sistemi podpirajo, da izmenjujejo podatke, informacije ter znanja.

mehanizem za preverjanje pristnosti. Zato lahko s pametno kartico zmanjšamo tveganje kraje identitete uporabnika, ker bi potencialni slepar moral ukrasti oba faktorja, da bi se predstavil kot legitimen uporabnik (Be'ery, 2014).

Vendar je to le lažen občutek varnosti. Težava se pojavi, kako zagotoviti zaščito v primerih, kot so napadi z zlonamerno kodo. Pametna kartica je samostojna fizična naprava, ki ščiti svoje certifikate in mora biti imuna na zlonamerno programsko opremo, ki je nameščena na samem računalniku. To drži, ker so operacijski sistemi ločeni od kriptografskih operacij. Pametne kartice niso dovzetne za napade na operacijski sistem, kot so napadi na pomnilnik, ki lahko razkrijejo zasebne ključe ali druge kriptografske skrivnosti. Vendar oba protokola za preverjanje pristnosti, kot sta NTLM (angl. NT LAN Manager) in Kerberos, ustvarita v pomnilniku NTLM hash in Kerberos žeton, ki podpira preverjanje pristnosti z enim samim prijavljanjem (angl. Single sign-on, SSO). V pomnilniku se ti žetoni generirajo med postopkom vsakega prijavljanja, vključno s prijavo s pametno kartico. Posledično je ob prijavi uporabnika s pametno kartico v vsakem trenutku žeton izpostavljen grožnji, kot je to uporabnikovo geslo za prijavo, za napade Pass-the-Hash in Pass-the-Ticket, ki napadalcu omogočajo zlorabo poverilnic žrtve in krajo njihove identitete po odstranitvi pametne kartice z okuženega računalnika (Be'ery, 2014).

Če pogledamo naprej, se nevarnost za Pass-the-Hash pametne kartice ne ustavi pri lažnem občutku varnosti. Za podporo sistemom, ki zahtevajo preverjanje pristnosti z NTLM, mora operacijski sistem ustvariti NTLM hash. Ker pametne kartice nimajo gesla, iz katerega bi lahko dobili hash, so se inženirji operacijskega sistema odločili, da umetno ustvarijo NTLM hash za uporabo pametnih kartic. Težava je, da ta žeton, ki je enakovreden geslu, nikoli ne poteče. Če je bil račun konfiguriran z atributom pametne kartice (angl. Smart Card), ki je potrebna za interaktivno prijavo, je NT hash naključna vrednost, izračunana, ko je ta atribut omogočen za račun. Ta hash gesla je v odjemalskem računalniku med procesom prijave pametnih kartic zagotovljen s strani domenskega kontrolerja. To geslo, ki se samodejno ustvari, ko je atribut nastavljen, se ne spremeni. Potemtakem zlonamerna programska oprema, ki lahko prestreže uporabnikov NTLM hash, za vedno ukrade njegovo identiteto (Be'ery, 2014).

Vendar ta oblika potrjevanja pristnosti ni več veljavna oblika dvofaktorske avtentifikacije, ker identiteta uporabnika ni dodatno preverjena z identiteto, določeno s strani odjemalca certifikata. Z varnostno ekipo bi morali skrbno pretehtati preverjanje pristnosti Windows Smart Card in jo nadomestiti z dodatnimi varnostnimi kontrolami za ublažitev napadov Pass-the-Hash in Pass-the-Ticket. (Be'ery, 2014).

5 NAPADI NA DVOFAKTORSKO AVTENTIKACIJO

Poleg vseh prednosti, ki jih prinaša 2FA, je z njeno vse večjo uporabo prišlo tudi do porasta zlorab. Namen tega poglavja je raziskati praktične in teoretične napade na 2FA. Začeli bomo s pregledom najpogostejših napadov na 2FA, kjer se moramo paziti na lažne spletne strani, prek katerih zlikovci ribarijo dragocene podatke, beleženja tipkanja in vse do socialnega inženiringa. Nadaljevali bomo s primeri napadov enojne okužbe, ko je okužen samo računalnik ali pametni telefon, kjer se uporablja 2FA za namene preverjanja pristnosti ter končali s pregledom napadov dvojne okužbe.

5.1 Najpogostejši napadi na 2FA

Kadar razmišljamo o preverjanju pristnosti z 2FA moramo razmisliti o nekaterih najpogostejših napadih pri preverjanju pristnosti 2FA. V poglavju so raziskani najpogostejši napadi na 2FA in ponazarjajo pomembnost širokega razmišljanja o tem, kako lahko je dvomljivo preverjanje pristnosti.

5.1.1 Mož v sredini

Napadi na omrežje, ki temeljijo na »mož v sredini« (angl. Man in the middle, MiTM), so povezani s kriptografskima omrežnima protokoloma SSL in TLS. Ponarejanje z lažnimi kriptografskimi **certifikati** je sorazmerno redko. Napad se izvrši z vrinjenjem lažnih korenskih certifikatov v zaupanja vredno bazo certifikatov v brskalniku ali s spremembo kateregakoli avtoritativnega korenskega certifikata v bazi. Tako napadalec postane posrednik in lahko izvede posredovanje prijavnih podatkov ali prilagaja vsebino komunikacije (Fenton, 2016).

Če nas napadalec v MitM lahko preslepi, da obiščemo njegovo lažno spletno stran, in zahteva, da vnesemo svoje 2FA poverilnice, je praktično že konec igre. Napadalec v MitM lahko ponaredi spletno mesto, za katero verjamemo, da je pristno, kjer nato uporabimo svoje generirane 2FA poverilnice in kadar jih na njegov poziv vpisujemo, jih lahko ukrade. Še bolj pogosto je, da po uspešni potrditvi z uporabo 2FA lahko posledično ukradejo piškotke seje (angl. session cookie).

Večina ljudi ne razume, da ko enkrat uporabimo 2FA za prijavo ne glede na to, kako to storimo – biometrično, s strojnim žetonom ali pametno kartico – je operacijski sistem tisti, ki sedaj obravnava naša pooblastila za dostop do objektov z uporabo sekundarno generiranega programskega žetona. Ta žeton je lahko ukraden in ponovno uporabljen. V primeru, ko naš operacijski sistem potrebuje prstni odtis za overjanje in prijavo, se v večini primerov v ozadju pogosto uporablja NT Lan Manager (NTLM) ali Kerberos žetone. Kako

se avtenticiramo, je ponavadi malo vpliva na to, kako se potem avtoriziramo za dostop do objektov, zato je pomembno, da razumemo ta koncept varnosti in njegove posledice, ki so lahko ogromne (Grimes, 2018).

5.1.2 Mož v brskalniku

Naprednejša zlonamerna programska oprema, kot je Zeus, je napad, »mož v brskalniku« (angl. Man in the browser, MiTB), ki omogoča napadalcu ponarejanje uporabnikovega prikazovanja v brskalniku, zaradi česar uporabnik misli, da na spletni strani počne tisto, kar vidi, medtem ko dejansko napadalec v ozadju počne nekaj povsem drugega. Najboljši protiukrep za to je uporaba dvofaktorske avtentikacije, ki je neodvisna na pametnem telefonu (Fenton, 2016).

Podobno kot napad MitM lahko heker namesti zlonamerno programsko opremo v naš računalnik in spremeni aplikacijo, ki se uporablja v procesu 2FA ter ukrade podatke, zaščitene z 2FA ali pa uporabi že odobreno avtentikacijo za dostop.

Bančni trojanci[3] to počnejo že od začetka leta 2000. Delujejo tako, da počakajo na našo uspešno prijavo in nato zaženejo skrite, lažne seje v ozadju. Mislimo, da preprosto preverjamo svoje stanje na bančnem računu, vendar se v ozadju prenese ves naš denar na zunanji bančni račun.

Banke so mislile, da so te vrste trojancev zaustavile z ustvarjanjem sekundarne kode 2FA, ki je pridobljena iz številke transakcije in je edinstvena za transakcijo. Bančni trojanski ustvarjalci so se odzvali tako, da so prestregli prvotno zahtevano transakcijo, ustvarili in oddali lastno veliko večjo transakcijo in to poslali banki. Banka, ki ni seznanjena s tem, da je nova transakcija lažna, ustvari sekundarno transakcijo 2FA z uporabo številke goljufov in jo pošlje legitimnemu uporabniku. Uporabnik potem vtipka sekundarno kodo 2FA, ne da bi vedel, da je koda, ki so jo poslali, veljala samo za skrivno prevarantovo transakcijo, ki tako ukrade ves naš denar.

Banke so se odzvale na napade tako, da so uporabniku poslale znesek transakcije in druge s tem povezane podatke, da jih potrdi z 2FA kodo. Banke so bile presenečene, ker so videle, da mnogi komitenti niso pozorni na podrobnosti o transakcijah in so jih vsebinsko vnesli v kodo. Tako so hekerji s pomočjo trojancev v mnogih primerih še vedno lahko ukradli denar.

Ne glede na to, kako se prijavljate v vaš računalnik, z ali brez 2FA, lahko po preverjanju pristnosti skrita zlonamerna programska oprema naredi karkoli. Lahko samo počaka, da gre računalnik v stanje hibernacije ali počaka, da se zaklene zaslon, ker so tudi ko zaklenemo zaslon, naši avtentikacijski in avtorizacijski žetoni aktivni in jih je mogoče ponovno uporabiti (Grimes, 2018).

[3] trojanci – ali trojanski konji so programi, ki se predstavljajo uporabniku kot uporaben program

5.1.3 Okužba programske opreme za 2FA

Specializirani napad man-in-the-browser je okužba programske opreme za 2FA. Kot na primer, če želimo uporabljati pametno kartico v računalniku, moramo imeti nameščeno programsko opremo za pametno kartico, ki bere in piše s pametne kartice. Ponudnik pametne kartice nam dodeli programsko opremo za namestitev ali pa se uporabi vnaprej nameščen splošni gonilnik v operacijskem sistemu ali napravi, ki jo uporabljamo.

Če dovolimo hekerju namestiti njegovo škodljivo programsko opremo, lahko manipulira ali nadomesti legitimno 2FA programsko opremo. V primeru pametne kartice bi lahko zlonamerna programska oprema ob naslednji prijavi s pametno kartico zahtevala, da ji zaupa svoje shranjene skrivnosti, ali da obdrži žeton, ki označuje uspešno preverjanje pristnosti dejavnega v pomnilniku za daljši čas, kot bi legalna programska oprema to običajno dovolila, kar omogoča, da heker ukrade ali ponovno uporabi poverilnice. V nekaterih scenarijih se lahko zloraba programske opreme uporablja za popolno krajo in zamenjavo pametne kartice na napravi, ki jo ima heker (Grimes, 2018).

5.1.4 Beleženja tipkanja in preusmeritev

Pri napadu z beleženjem tipk (angl. keylogger) se spremlja uporabnikovo tipkanje, da se pridobi prijavne poverilnice, uporabniško ime in geslo. Zlonamerna programska oprema potem preusmeri podatke odtipkanih tipk k napadalcu v trenutku, ko nam je bilo omogočeno, da smo se prijavili. Vendar je 2FA proti tem pasivnim napadom učinkovita, saj vključuje enkratno geslo, ki je pridobljeno iz naprave, kot sta strojni žeton ali programski žeton na pametnem telefonu (Fenton, 2016).

5.1.5 Kraja in ponovna uporaba generatorja gesel

Veliko strojnih in programskih žetonov generira enkratno geslo, ki je unikatno za uporabnika in njegovo napravo. Avtentikacijska programska oprema in uporabnikova naprava lahko generirata enkratno geslo istočasno. Potem se primerja vneseno uporabnikovo geslo z geslom avtentikacijskega sistema, če sta identična.

V večini primerov so enkratna gesla generirana vedno iz skupne naključne vrednosti semena, ki je edinstveno za vsako 2FA napravo in uporabnika. Nato se vsa nadaljnja gesla generirajo iz semena v vnaprej določenih časovnih intervalih z uporabo skupnega algoritma. To je tip žetona 2FA, kjer je uporabnik pozvan k enkratnem geslu in ima le 30 sekund do nekaj minut, da se prijavi, preden se generira nova vrednost. RSA-jevi SecureID žetoni popularizirajo te vrste naprav 2FA, čeprav obstaja na desetine, če ne na stotine podobnih strojnih žetonov danes, in na stotine, če ne na tisoče različic na osnovi programske opreme.

Na splošno različice, ki temeljijo samo na programski opremi, niso tako varne kot strojne različice, saj je programsko opremo veliko lažje zlorabiti. Strojni žetoni običajno zahtevajo fizičen dostop za zlorabo.

Hekerji že dolgo vedo, da če zajamejo **prvotno vrednost** semena in poznajo sinhronizacijski algoritem za sinhronizacijo, lahko ustvarjajo enkratno geslo, kot legalen sistem in 2FA naprava. Nekatere 2FA naprave so uporabljale takšne šibke enkratne generatorje gesel in napadalci so lahko zajeli katerokoli enkratno vrednost in nato generirali vse prihodnje vrednosti. Če se to lahko zgodi brez poznavanja prvotne naključne vrednosti semena, je uporabljeni kriptografski algoritem prešibak. Ne bi smeli imeti možnosti, da se zajame eno naključno generirano vrednost in se jo nato uporabi za poenostavljeno iskanje naslednje naključno generirane vrednosti.

Pogosto uporabljana, vsakodnevna hekerska orodja so vključevala povezano funkcionalnost. Če heker dobi vrednost semena, lahko ustvari ponarejeno 2FA napravo. APT napadalci (angl. advanced persistent threat) so te vrste napadov uporabili tudi v svojo korist. Najbolj znan primer je, ko so kitajski hekerji kompromitirali [4]RSA, da bi dobili semenske vrednosti enega od računov uporabnika podjetja Lockheed Martina, ki so jih nato uporabili, da so vdrli v sistem podjetja Lockheed Martin (Grimes, 2018).

5.1.6 2FA ni potrebna

Veliko storitev, vključno s priljubljenimi spletnimi mesti, ki nam omogočajo uporabo 2FA, tega ne zahtevajo, kar ni dobro. Večina uporabnikov meni, da morajo 2FA potem, ko so jo omogočili, vedno uporabljati. To pogosto ni res. Večina spletnih mest omogoča uporabnikom, da vnesejo geslo ali odgovor na vprašanja o ponastavitvi gesla ali celo pokličejo tehnično podporo, da bi zaobšli 2FA.

Na spletnih mestih, ki uporabnikom omogočajo, da se prijavijo z uporabo več metod, vključno z 2FA, od uporabniku ne zahtevajo 2FA in hekerji lahko s pomočjo socialnega inženiringa teh spletnih mest od tehnične podpore pridejo s ponastavitvijo gesla do uporabnikovih poverilnic ali pa preprosto odgovorijo na vprašanja o ponastavitvi gesla.

Veliko lažje in mnogo načinov je, da se geslo lahko ugane kot zaščiti. Tako so vprašanja za ponastavitev gesla nadloga industrije za preverjanje pristnosti. Hekerji lahko s socialnim inženirstvom pridejo do gesla uporabnika in ga uporabijo namesto 2FA avtentikacije. Seveda v primeru, kadar spletna stran omogoča uporabo 2FA, a je ne zahteva. Če podjetje uporablja 2FA, a ne omogoča uporabe na vseh spletnih mestih in storitvah podjetja, temveč se uporablja uporabniško ime in geslo za prijavo, se v veliki meri poraja vprašanje namena imeti 2FA (Grimes, 2018).

[4] kompromitirati – kateri koli računalniški vir, katerega zaupnost, celovitost ali razpoložljivost je bil namerno ali nenamerno prizadel nezaupen vir

5.1.7 Ponarejena identiteta

Pri pametnih karticah imamo prav posebno **skrivnost,** za katero proizvajalci nočejo slišati. Vsaka 2FA naprava oziroma 2FA programska oprema je povezana z identiteto uporabnika in naprave, kjer mora biti identiteta edinstvena v sistemu overjanja. V mnogih sistemih 2FA, še posebej pametnih karticah, lahko spremenimo uporabnikovo identiteto tako, da tudi začasno uporabimo katero koli napravo 2FA in jo povežemo z drugo osebo ter nato uporabimo za preverjanje pristnosti.

Recimo da je pametna kartica povezana z identiteto, imenovano uporabnik1@primer.com. Heker, ki dobi katero koli drugo pametno kartico in PIN, recimo od uporabnika2, lahko vstopi v sistem za preverjanje pristnosti in spremeni identiteto uporabnika1 na uporabnika2 in obratno. Nato se lahko prijavi kot uporabnik2 s pametno kartico in PIN-om uporabnika Sistem bo to zabeležil, kot če bi se prijavil uporabnik1. Ko bo heker opravil svoje goljufive dejavnosti, lahko potem preprosto preklopi identiteti nazaj, ne da bi kdaj poznal PIN uporabnika1 ali posedoval pametno kartico uporabnika1 in uporabnik1 ne bi o tem nič vedel. Zato so pametne kartice zrele za notranje napade 2FA. To velja za številne naprave 2FA. Če ima nekdo dovoljenje za spremembo identitete nekoga, lahko dobesedno preklopi identiteto tega uporabnika, naprave na katero koli drugo napravo 2FA, na kateri ima nadzor (Grimes, 2018).

5.1.8 Kraja biometrije

Biometrične atribute identitete, kot so prstni odtis ali mrežnica, lahko ukradejo in uporabijo za prijavo, težko pa je onemogočiti napadalčevo uporabo le-teh. Obstaja veliko vprašanj z biometričnimi identitetami, kot je velika stopnja lažnih negativov in lažnih pozitivnih rezultatov. Dejstvo je, če bodo biometrične identitete ukradene, bodo za vedno kompromitirane. Ukradeno geslo lahko spremenimo, ne moremo pa preprosto spremeniti prstnih odtisov ali mrežnice (Grimes, 2018).

5.1.9 Skupna, integrirana avtentikacija

Kadar uporabljamo skupne integrirane sheme za preverjanje pristnosti, kot je na primer oAuth (angl. open-standard authorization), le-te uporabniku omogočajo, da se prijavi enkrat in se nato njegove poverilnice ponovno uporabijo za prijavo na več storitvah in spletnih mestih. Kadar se uporablja integrirana skupna avtentikacija, pogosto komponenta 2FA, ki je bila zahtevana v izvirni avtentifikaciji, ni potrebna za drugo in ostale naknadne prijave, četudi bi za to naknadno prijavo običajno sicer potrebovali prijavo 2FA. Integrirane prijavne sheme pogosto uporabljajo že uporabljen avtentikacijski prijavni žeton za dodatne prijave (Grimes, 2018).

5.1.10 Socialni inženiring, obnova računa OTP

Ker vse več spletnih strani uporablja 2FA, so se hekerji naučili, kako s socialnim inženirstvom priti do podatkov. Ti napadi so lahko kot napada MITM ali man-in-the-endpoint napad, vendar so lahko bolj izpopolnjeni in vključujejo prodajalca, ki nenamerno zahteva uporabo 2FA. Skratka samo zato, ker uporabljamo 2FA, še ne pomeni, da ne moremo biti prevarani (Grimes, 2018).

V primeru, ko izgubimo strojni ali programski žeton za preverjanje pristnosti, moramo imeti izdelan pravilen postopek. Če je v postopku, da začasno onemogočimo dvofaktorsko preverjanje pristnosti, lahko napadalec s socialnim inženiringom prek obnovitvenega postopka računa pridobi dostop do računa. Velika napaka je, če se uporablja preverjanje pristnosti, ki temelji na vprašanju, kot je: »Kakšno je bilo ime vašega prvega ljubljenčka?« Za obnovitev računa so ti odgovori pogosto zelo enostavni in napadalec jih lahko ugane, zato zagotavljamo veliko slabšo varnost. V mislih moramo imeti, da bo napadalec izbral najšibkejšo točko našega sistema za preverjanje pristnosti za napad. To je kot uničujoč napad izpostavil Mat Honan (Wired Magazine) pri obnovi računa s pomanjkanjem dvofaktorske avtentikacije (Fenton, 2016).

5.1.11 Uporaba 2FA tretje osebe

Nekateri dvofaktorski sistemi za preverjanje pristnosti se zanašajo na izdajo, preverjanje ali komunikacijo s preverjanjem fizičnih žetonov. Podedovano ranljivost tretje osebe najbolje kažejo kršitve RSA sistema za preverjanje pristnosti SecurID leta 2011. Čeprav obseg kršitve RSA ni povsem znan, se domneva, da je napadalec lahko dobil dostop do informacij za ponarejanje žetonov.

Preverjanje pristnosti z besedilnimi sporočili SMS je odvisna od operaterja, kako dodeljuje telefonske številke. Če lahko napadalec prepriča operaterja, da je uporabnik in da je izgubil svoj telefon ter potrebuje novega, bi lahko prestregel besedilna sporočila in telefonske klice ter si tako zagotovil drugi faktor za preverjanje pristnosti. To je pripeljalo, da so že operaterji svetovali nekaterim bankam, da prepovedo uporabo SMS-a pri dvofaktorski avtentikaciji (Fenton, 2016).

5.1.12 Neposreden napad na 2FA

Za izgubljene 2FA žetone ni znano, da bi jih hekerji lahko obnovili. Če pa spletno mesto ali storitev, ki uporablja za prijavo 2FA, nima politike zaklepanja računov pri nepravilnem vnosu podatkov 2FA, lahko napadalci ugibajo 2FA PIN kodo in jo vnašajo znova in znova (angl. brute force), dokler ne uganejo pravilne. Večina spletnih mest 2FA ima blokade, vendar ne vse (Grimes, 2018).

5.1.13 Hroščna implementacija

Zanesljivo lahko trdimo, da je veliko prijavnih strani z 2FA in programske opreme s hrošči, ki omogočajo, da se zaobide 2FA kot popolnoma varno stran (Grimes, 2018).

5.2 Napad enojne okužbe

V primeru, da je okužen samo računalnik, kjer se uporablja 2FA za namene preverjanja pristnosti, menimo, da so mobilne aplikacije, ki generirajo OTP za to prijavo, varne. Napadalec, ki je okužil samo računalnik uporabnika, a nima kontrole nad mobilno napravo, se ne more prijaviti v imenu uporabnika. Takšna arhitektura je smotrna, saj domnevno nezaupanja vreden računalnik zahteva potrebo po uporabi ločene naprave za generiranje drugega faktorja poverilnic za preverjanje pristnosti (Dmitrienko, Liebchen, Rossow & Sadeghi, 2014).

5.2.1 OTP-ji z nizko entropijo

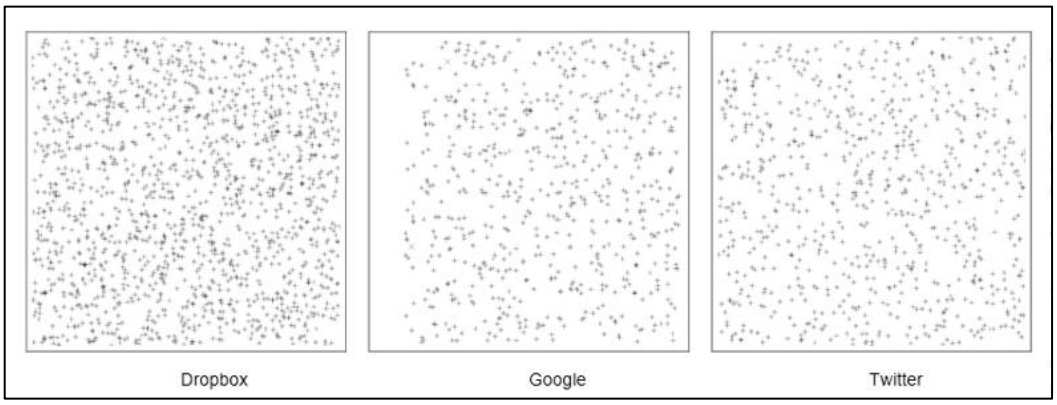
Na splošno so gesla z nizko entropijo oziroma kako nepredvidljivo je geslo, občutljiva na napade številnih poizkusov (angl. brute-force). Generirani OTP mora izpolnjevati osnovna merila naključnosti. V podjetju CASED so izvedli postopek za samodejno zbiranje OTP-jev iz Twitterja, Dropbox-a in Googla. Avtomatizacijo postopka zbiranja OTP-jev so izvedli s pomočjo programske opreme, ki je sprožila prijavo s poverilnicami, ter iz dohodnega SMS sporočila izluščili OTP in shranili v bazo podatkov. Skupaj so zbrali 1564 (Dropbox), 659 (Google) in 775 (Twitter) OTP-jev. Podatki so prikazani v tabeli (Tabela 3) in grafični prikaz zbranih OTP-jev na sliki (Slika 6). Pri OTP-jih, ki sta jih ustvarila Dropbox in Twitter, so opravili standardne preskuse naključnosti in opazili, da se pri Google OTP nikoli ne začne s številko 0, kar pomeni, da se izpusti desetino vseh možnih vrednosti OTP-ja in se posledično zmanjša entropija ustvarjenih gesel, saj se število možnih gesel zmanjša za 10 % iz 10^6 do 10^6 - $10^{5.}$ (Dmitrienko, Liebchen, Rossow & Sadeghi, 2014).

Tabela 3: Zapis rezultatov entropije OTP gesel

Service Provider	Nr of collected OTPs	Nr of unique OTPs	Collect. interval min.	Avg. OTP value
Dropbox	1564	1561	15	507809
Google	659	654	30	559851
Twitter	775	772	15	505883

Vir: Dmitrienko, Liebchen, Rossow & Sadeghi (2014).

Slika 6: Slikovni prikaz entropije OTP gesel



Vir: Dmitrienko, Liebchen, Rossow & Sadeghi (2014).

5.2.2 Pomanjkljivost OTP razveljavitve

Kadar gre za razveljavitev OTP, ostane v idealnem primeru generirani OTP veljaven samo za določeno časovno obdobje in se ne ponovi več, tudi če uporabnik ne opravi prijave. Pri Googlu, kadar ne dokončaš postopka 2FA, je Google večkrat ustvaril isti OTP za zaporedne prijave preverjanja pristnosti. Google razveljavi generiran OTP po eni uri ali po tem, ko je uporabnik uspešno končal prijavo z 2FA. OTP se ponovi tudi, kadar se zamenjajo uporabnikov IP naslov, brskalnik in verzija OS. Napadalec bi lahko izkoristil to pomanjkljivost in zajel OTP, hkrati pa preprečil uporabniku, da se prijavi z OTP na storitev, s tem pa ostane zajet OTP veljaven. Napadalec lahko ponovno uporabi OTP v ločeni prijavni seji, saj bo Google še vedno pričakoval isti OTP tudi za drugo sejo (Dmitrienko, Liebchen, Rossow & Sadeghi, 2014).

5.2.3 Deaktivacija 2FA

Če se za preverjanje prijave uporablja 2FA, se uporabniki običajno odločijo za uporabo funkcije 2FA. Pri odjavi uporabe 2FA je popolno, če se zahteva nadaljnjo varnostno preverjanje. V nasprotnem primeru tvegamo, da bo zlonamerna programska oprema zajela obstoječo sejo z namenom onemogočiti 2FA (Dmitrienko, Liebchen, Rossow & Sadeghi, 2014).

Pri CASED so ustvarili račune pri ponudnikih storitve, se prijavili s temi računi, aktivirali 2FA, izbrisali vse podatke o seji, se odjavili in ponovno prijavili. Opazili so, da lahko uporabniki storitev Google in Facebook deaktivirajo 2FA brez dodatnih preverjanj. Twitter in Dropbox dodatno zahtevata uporabniško ime in geslo. Nobeden od testiranih ponudnikov storitev ni zahteval OTP, da odobri to dejanje. Ugotovitve kažejo, da se lahko 2FA sheme ponudnikov zaobide brez potrebe po okužbi mobilne naprave. Zlonamerna programska oprema v računalniku lahko počaka, dokler se uporabnik ne prijavi, nato pa zajame sejo in

onemogoči 2FA v nastavitvah uporabniškega računa. Če so potrebna dodatna pooblastila za prijavo za izvajanje te operacije kot zahtevata Twitter in Dropbox, lahko zlonamerna programska oprema v računalniku ponovno uporabi poverilnice, ki jih je mogoče ukrasti z beleženjem tipkanja ali z napadom mož v brskalniku (angl. man-in-the-browser) (Dmitrienko, Liebchen, Rossow & Sadeghi, 2014).

5.2.4 Mehanizem za obnovo 2FA

Medtem ko sistem 2FA zagotavlja večjo varnost, morajo uporabniki imeti svoje mobilne naprave vedno s seboj za overjanje. To lahko vpliva na uporabnost, saj lahko uporabnik izgubi upravljanje nad svojim računom, če ne upravlja svoje naprave, kot na primer, če napravo izgubi, je ukradena ali začasno ni na voljo zaradi prazne baterije. Ponudniki storitev za reševanje teh težav omogočajo mehanizem za obnovitev, ki uporabniku omogoča, da obdržijo upravljanje nad svojim računom, kadar mobilna naprava ni na voljo. Slabost je, da lahko napadalci zlorabijo obnovitveni mehanizem in si pridobijo nadzor nad uporabniškim računom, ne da bi okužili uporabnikovo mobilno napravo (Dmitrienko, Liebchen, Rossow & Sadeghi, 2014).

Ponudnik Twitter nima mehanizma za obnovitev. Dropbox uporablja tako imenovano obnovitveno geslo, naključni niz s 16 znaki v berljivi obliki, ki se prikaže v nastavitvah računa in se ustvari, ko uporabnik omogoči 2FA. Facebook in Google uporabljata mehanizem za obnovitev, ki uporabniku omogoči, da ustvari seznam desetih OTP-jev za obnovitev, ki jih je mogoče uporabiti, kadar nima dostopa do svoje mobilne naprave. Seznam je shranjen v nastavitvah računa, podobno kot gesla za obnovitev v Dropboxu. Dropbox in Google ne zahtevata dodatnih preverjanj pristnosti, preden omogočijo dostop do obnovitvenih podatkov, medtem ko Facebook dodatno zahteva prijavne poverilnice (Dmitrienko, Liebchen, Rossow & Sadeghi, 2014).

Ker so nastavitve računa na voljo uporabnikom po prijavi, lahko do teh obnovitvenih poverilnic (OTP in gesel) dostopa zlonamerna programska oprema, kadar zajame uporabniško sejo. Zlonamerna programska oprema, ki se nahaja v računalniku, pridobi podatke tako, da čaka, dokler se uporabnik ne prijavi v svoj račun. Z zajetjem seje zlonamerna programska oprema pridobi gesla za obnovitev s spletne strani v nastavitvah računa, mimo vseh dodatnih preverjanj prijavnih poverilnic kot v Facebooku (Dmitrienko, Liebchen, Rossow & Sadeghi, 2014).

5.2.5 Slabost inicializacije OTP generatorja

Rešitve 2FA z generiranimi OTP stranmi odjemalca, kot je Google Authenticator (GA), temeljijo na vnaprej določenih skupnih skrivnostih, kar je primerno za vektorske napade.

Inicializacija GA se začne, ko uporabnik v svojih nastavitvah računa omogoči avtentifikacijo na podlagi GA. Ponudnik storitev ustvari kodo QR, ki se prikaže uporabniku na osebnem računalniku in jo mora uporabnik skenirati s telefonom. Koda QR vsebuje vse informacije, potrebne za inicializacijo GA s podrobnostmi o računu za uporabnika in pred določeno skupno skrivnostjo. Inicializacijska koda QR vključuje podrobnosti, kot so vrsta sheme na osnovi števca ali časa, identifikacijo storitve in računa, števec, dolžino generiranega OTP in skupne skrivnosti. Vsi ti podatki so predstavljeni v jasnem besedilu. GA ne podpira nobenega alternativnega inicializacijskega postopka.

Žal zlonamerna programska oprema lahko prestreže inicializacijsko sporočilo ali jasno besedilo, kodirano kot QR-kodo. Napadalec lahko nato inicializira svojo različico GA in lahko ustvari veljavne OTP za ciljne račune (Dmitrienko, Liebchen, Rossow & Sadeghi, 2014).

5.3 Napad dvojne okužbe

Kadar sta ogroženi obe napravi 2FA, računalnik in pametni telefon, lahko napadalec ukrade oba žetona za preverjanje pristnosti in se predstavi kot legitimni uporabnik, ne glede na to, kakšna posebna instanca mobilne 2FA je uporabljena. Taki napadi so izvedljivi in jih zato ni smiselno izključiti (Dmitrienko, Liebchen, Rossow & Sadeghi, 2014).

5.3.1 Kraja SMS-a

Dokazano je, da metoda 2FA, ki uporablja SMS sporočila za pošiljanje OTP uporabnikom, ni varna, zaradi potencialno nameščene zlonamerne programske opreme na pametnih telefonih. Na OS Android se lahko namesti zlonamerna aplikacija kot prejemnik SMS-a z višjo prioriteto kot program za upravljanje SMS-a. Te pravice omogočajo, da ima na vsakem prejetem besedilnem sporočilu prvi vpogled zlonamerna aplikacija.

Z dovoljenji, kot je dostop do interneta, zlonamerna programska oprema prejme oddaljen ukaz iz SMS-a, poslan z določene telefonske številke, brez da bi uporabnik to vedel, ker zlonamerna programska oprema filtrira prejeto sporočilo za prikaz. Vsebina prejetega sporočila se pošlje na C&C strežnik (angl. Command and Control servers) po preprosti HTTP POST zahtevi.

Takšen napad bo uspešen, če bodo lahko avtorji zlonamerne programske opreme privabili žrtev v namestitev zlonamerne aplikacije bodisi iz neznanega vira ali iz trgovine Google Play in potrdili dovoljenja, ki jih zahteva aplikacija. Aplikacija, ki bi dejansko potrebovala dovoljenja za upravljanje SMS v nastavitvah, lahko celo zavestnega in bol izkušenega uporabnika preslepi. Prepričati uporabnika, da namesti zlonamerno aplikacijo s strani

neuradnega vira, bi lahko dosegli tako, da bi vir deloval kot dobro poznan legalen ponudnik storitve, kot je npr. Google (Hell & Altman, 2015).

Neposreden kanal OOB med oddaljenim strežnikom in pametnim telefonom se lahko realizira bodisi na podlagi HTTPS ali prek sporočil SMS. Rešitev, ki temelji na SMS-jih, prevladuje in se široko uporablja za sheme TAN (angl. Transaction Authentication Number) v spletnem bančništvu. OTP sistemi za prijavo v sistem na osnovi SMS-a so postali zelo popularni in jih uporabljajo ponudniki spletnih storitev, kot so Dropbox, Facebook, Microsoft, Google in Apple (Dmitrienko, Liebchen, Rossow & Sadeghi, 2014).

Če želimo zaobiti to rešitev, mora zlonamerna programska oprema pridobiti prijavne poverilnice, PIN ali geslo iz računalnika, še preden se prenesejo na spletni strežnik ponudnika storitve. Zlonamerna programska oprema mora nato pridobiti tudi sekundarne poverilnice, OTP ali TAN, tako da prestreže SMS sporočila na mobilni napravi (Dmitrienko, Liebchen, Rossow & Sadeghi, 2014).

Z napadom mož med brskalnikom se ukrade prijavne poverilnice iz računalnika. Uporabi se DLL vrivanje, da se vrine knjižnico v naslovni prostor brskalnika in poveže funkcijo, da preusmeri legalne funkcije, da kličejo zlonamerno funkcijo podtaknjeno DLL. Na ta način se prestreže klicno funkcijo, ki vsebuje uporabniške poverilnice kot parametre navadnega besedila, preden se jih pošlje prek šifrirane komunikacije HTTPS (Dmitrienko, Liebchen, Rossow & Sadeghi, 2014).

Za prestrezanje SMS sporočil zlonamerna programska oprema deluje kot MitM med GSM modemom in Android telefonskim sistemom ter prestreže vsa povezana SMS sporočila z OPT tako, da jih uporabnik ne prejme, posreduje pa vsa druga SMS sporočila za običajno uporabo. Poleg tega pa se izvaja protokol za ukaze in nadzor (angl. command & control protocol), ki temelji na SMS-u med napadalcem in pametnim telefonom. Protokol se uporablja za (de)aktiviranje prestrezanja OTP ali TAN ali za določitev naslova za posredovanje podatkov (Dmitrienko, Liebchen, Rossow & Sadeghi, 2014).

5.4 Napadi na TOTP

Kadar napadalec načrtuje poosebljanje identitete, mora za dostop do storitve, zaščitene z 2FA, v času prijave ustvariti pravilen OTP. V idealnem primeru mora dobiti popolnoma funkcionalno kopijo ali klon žrtvinega OTP generatorja, tako da lahko proizvaja veljavne OTP v zahtevanem času. Učinkovitejša je lahko tudi preprostejša metoda, kot je upravljanje generatorja žetonov neposredno na žrtvini napravi (Mueller, 2016).

Programski žetoni na OS Android so razviti, da podatke uporabnikov in skrivne ključe privzeto varuje operacijski sistem. To pomeni, da napadalec potrebuje fizični ali oddaljeni

korenski dostop do žrtvine naprave kot predpogoj za uspešen napad. Ko se enkrat pridobi korenski dostop, stoji pred napadalcem samo proizvajalčeva zaščita aplikacije. Dejstvo je, da je najpomembnejša točka te obrambe preprečiti kloniranje primerka žetona.
Ob predpostavki, da je napadalec že prevzel kontrolo (angl. pwned) nad žrtvinim pametnim Telefonom, je nekaj metod, ki jih lahko napadalec uporabi za reprodukcijo žrtvinega OTP generatorja (Mueller, 2016).

5.4.1 Pridobitev iz trajnega pomnilnika

Skupno skrivnost je potrebno shraniti nekje na napravi in jo je zato vedno mogoče kopirati iz trajnega pomnilnika. Napadalcu je kot predpogoj nujno vedeti, kaj sestavlja skrivnost, kateri algoritem je uporabljen in kakšni so pričakovani skrivni vložki, kje je shranjena in kako je zaščitena (Mueller, 2016).

Žrtvin OTP generator lahko na primer uporablja algoritem OATH-TOTP. V tem primeru je skrivnost, ki jo je treba kopirati, 32-bitni skrivni ključ. Ta vrednost je lahko shranjena v bazi podatkov SQLite ali v skupnih nastavitvah. Mogoče je uporabljeno dodatno šifriranje z dinamično ustvarjenim ključem naprave. Napadalec mora najprej preoblikovati mehanizme za shranjevanje in šifriranje, ki jih uporablja aplikacija. Ko je to storjeno, lahko zasnuje orodje za napad, ki kopira podatke za programski žeton (Mueller, 2016).

Sedaj prideta na vrsto prikrivanja (angl. obfuscation) in protiposeganje (angl. anti-tampering). Proizvajalci programskih žetonov uporabljajo lastne algoritme, prikrivanje kriptografskih funkcij, kodiranje semen in šifriranje ključe v izvajanju aplikacije, tako da ni mogoče dobiti nobenih šifrirnih ključev in napadalcu onemogoča razumevanje, kako se OTP generirajo. Bolj nejasen je izračun OTP, večji napor je potreben za ponovitev računanja veljavnih OTP glede na vhodne podatke iz žrtvine naprave (Mueller, 2016).

5.4.2 Pridobitev iz začasnega spomina

Če so uporabljeni običajni kriptografski sestavi, morajo biti skrivne vhodne vrednosti v določeni točki med izvajanjem shranjene v pomnilniku. Med tem časovnim oknom lahko napadalec izloči podatke iz delovnega procesa. To je mogoče storiti na različne načine, odvisno je od tega, kako aplikacija programskega žetona upravlja s pomnilnikom (Mueller, 2016).

V preprostih primerih zadostuje posnetek Java kopice. Običajno pa skrivnost ni v pomnilniku v navadnem besedilu, ki čaka, da jo posnamemo, potrebno je izvesti zahtevnejši postopek. Uporabi se injektiranje kode za določanje šifrirnih razredov aplikacije, ki se uporabljajo za nalaganje, dešifriranje in posnetek šifrirane skrivnosti (Mueller, 2016).

Da bi preprečili takšen napad, se kriptografsko računanje in funkcionalnost obdelave ključa prikrije tako, da nasprotniku postane manj razumljivo. Skrivni ključ se lahko skriva v izvedbi, namesto da se eksplicitno uporablja (bela-škatla-kriptografija) ali celotno računanje lahko izvaja tolmač, skrit za enim ali celo več slojev virtualizacije. V teh primerih mora nasprotnik, ki želi reproducirati računanje, porabiti nekaj časa za reševanje dodatne kompleksnosti ali uporabo kodiranja (Mueller, 2016).

5.4.3 Dvigovanje kode in instrumentacija

Za večino praktičnih namenov, kot je ponovitev zaporedja OTP, ni potrebno da napadalec ve vse podrobnosti o izračunu OTP. Namesto povratne inženiring kode se lahko preprosto uporabi kopiranje in ponovno uporabi celotno izvedbo. Ta vrsta napada je znana kot dvigovanje kode in se običajno uporablja za razbijanje DRM (angl. Digital rights management) in kriptografskih implementacij v beli škatli (Mueller, 2016).

Da bi preprečili takšne napade, so prodajalci programskih žetonov izvedli ukrep, da je izračun OTP povezan z uporabnikovim pametnim telefonoma. OTP generator se pravilno izvaja le v specifičnem, predpisanem mobilnem okolju, tako so nekateri kriptografski ključi ustvarjeni z uporabo podatkov, zbranih v napravi. Tehnika, ki povezuje funkcionalnost aplikacije z določenim okoljem, je znana kot povezovanje naprave (angl. device binding) (Mueller, 2016).

6 PRIMERJAVA DVOFAKTORSKIH AVTENTIKACIJSKIH METOD

V nadaljevanju bomo predstavili produkte RSA SecureID, HID ActiveID in Google avtentikator, kjer bomo obravnavali njihove prednosti in slabosti ter jih primerjali med seboj iz več različnih vidikov, vključno z uporabnostjo in varnostjo.

6.1 Uporabnost

Po Hell in Altman (2015, str. 31) sta uporabnost in varnost dva atributa, ki se pogosto nanašata drug na drugega, kadar načrtujemo izdelke in sisteme. Če se osredotočimo na varnostni vidik sistema, običajno trpi uporabnost, in obratno. Uporabnost je lahko definirana kot: v kolikšni meri lahko določen uporabnik uporablja izdelek, da doseže določene cilje z učinkovitostjo, zmogljivostjo in zadovoljstvom uporabe. Klasičen primer, ko gre za informacijsko varnost, je dilema uporabnikovega gesla. Geslo mora biti dolgo in dovolj zapleteno, da bi napadalcem otežili ugibanje, a še vedno kratko in preprosto uporabniku, ki si ga mora zapomniti. Podobno razmišljanje bi lahko uporabili pri OTP, vendar tu lahko potrdimo, da so vsi trije produkti RSA SecureID, HID ActiveID in Google avtentikator zelo

enostavni za uporabo. Razlika nastane pri izbiri dolžine OTP-ja, ker bo z večjim številom znakov bolj varen, vendar bo »naporno«, da ga uporabnik vnese v prijavni obrazec.

6.1.1 Razpoložljivost

Rešitev 2FA mora biti robustna, da je na voljo in uporabnikom omogoča dostop do zahtevane storitve ves čas. Jasno je, da se bodo vse rešitve za preverjanje pristnosti vedno zanašale na same strežnike za preverjanje pristnosti in da bodo na voljo. Razpoložljivostni vidik ni povezan z metodo 2FA, temveč z infrastrukturo in podjetja jih pogosto izključijo iz primerjave. V takem primeru se podjetja zavarujejo s postavitvijo v visoko razpoložljivem načinu (angl. High Availability Architecture), kar pomeni, da je strojna oprema podvojena v primeru fizične napake. Takšna postavitev se naredi vedno za RSA in HID, medtem ko je GA gledano s strani uporabnika v tako imenovanem oblaku in se pričakuje, da ima takšno postavitev (Hell & Altman, 2015).

Rešitve 2FA z uporabo metode mTAN zahtevajo določeno obliko **omrežne povezave**, da bi nasploh delovale. Uporabnik, ki je nastavil 2FA z uporabo metode mTAN, mora imeti zagotovljeno, da njegov pametni telefon lahko v vsakem trenutku prejme SMS sporočila, da se lahko prijavi v storitev. Nekatere pritožbe, ki temeljijo na SMS-u, so bile nezmožnost, da jih sprejemamo v določenih časovnih obdobjih, na primer, ko smo v tujini. To se lahko zgodi tudi v primeru, če so težave s telefonskim omrežjem ali če se uporabnik znajde na lokaciji s slabim signalom. Če so uporabniki podvrženi takšnim težavam, moramo zagotoviti ostale kanale za prejetje OTP-ja z elektronsko pošto ali možnost poklicati posebno brezplačno številko, če so v tujini. To dodaja še en dejavnik tveganja za rešitve mTAN poleg prvotno navedenega faktorja strežnika za preverjanje pristnosti (Hell & Altman, 2015).

Izvedbe, ki zahtevajo internetno povezljivost, so po navadi povezane z bančnimi rešitvami oziroma so jih izdelale banke. Povezave s telefonskim omrežjem ne potrebujejo, aplikacija zahteva le dostop do interneta, da bi jo lahko uporabljali. Lahko bi trdili, da je prenos podatkov na pametnem telefonu bolj nujna zahteva kot povezljivost s telefonskim omrežjem. Nekateri paketi zagotavljajo le omejeno količino podatkovnega prometa vsak mesec in kadar je izpolnjena kvota, se promet podatkov popolnoma izključi. Za primerjavo, telefonskega omrežja ponudnik ne bo izključil, ko uporabnik doseže določeno količino poslanih SMS-ov ali klicev in uporabnik bo še vedno lahko prejemal OTP prek SMS-a. Nasprotno pa bi lahko tudi trdili, da kjer ni dostopa do interneta, mora biti na voljo uporabniku telefonsko omrežje, da se lahko prijavi v spletno storitev (Hell & Altman, 2015).

TOTP je shema, katere predstavniki so RSA, HID in GA, za katere ni potrebe po povezljivosti v času preverjanja pristnosti, ker vse svoje OTP-je izpelje, dobi iz prednastavljene skrivnosti in trenutnega časa. Možnost za napako je le, če se naprave ne more več povezati s časovno sinhronizacijsko storitvijo.

V nekaterih okoliščinah se pojavijo **napake** v sistemih 2FA. Če ena od teh napak na nek način ovira uporabnikovo sposobnost, da se avtenticira, postane vprašanje razpoložljivosti. Napake, za katere mislimo, niso napake v programski opremi ampak v implementaciji same izvedbe.

Ker rešitev mTAN ne zahteva, da je dejanska aplikacija nameščena na pametni telefon uporabnika, ni nagnjen k napakam na enak način kot TOTP. Napaka, ki bi se lahko pojavila, je, da so OTP-ji poslani na pametni telefon napačni. Pri rešitvi TOTP bo prišlo do napačnih OTP-jev, če bo notranja ura naprave nehala sinhronizirati. Vendar je rešitev te težave precej hitra in enostavna, saj se sinhronizacija lahko izvede po zahtevi takoj, ko naprava dobi dostop do interneta.

TOTP s predstavniki RSA, HID in GA postaja vse bolj priljubljen s številnimi storitvami in zagotovo pomaga pri standardizirani rešitvi. Rešitve zagotavljajo uradno aplikacijo za Android, iOS in Blackberry, vendar obstaja še veliko drugih izvedb za druge **platforme**.

6.1.2 Enostavnost uporabe

Hell in Altman (2015, str 34) sta mnenja tudi, če so izpolnjeni vsi pogoji za delovanje programa 2FA in ni zabeleženih napak, bi morala biti metoda hitra, preprosta in enostavna za uporabo. Če rešitev vzame preveč časa uporabniku ali zahteva preveč napora, to lahko postane utrujajoče za uporabo in potem uporabnik preprosto izklopi 2FA. Postopek namestitve mora biti dovolj preprost, da uporabnik celo razmišlja, kako ga najprej aktivirati.

Metoda mTAN je verjetno ena najboljših glede enostavnosti uporabe v zvezi z namestitvijo in dejansko uporabo. Postopek namestitve navadno preprosto sestavlja registracija uporabnikove telefonske številke in potrditve kode, ki je poslana prek sporočila SMS na isto telefonsko številko. Dolžina gesla ali OTP mora biti dovolj kompleksna, da je varna, vendar dovolj kratka, da si jo zapomnimo in ne utrudljiva za vnašanje. Število znakov, ki jih izberemo za rešitev mTAN, je 6 do 8.

Če pa pogledamo fazo nastavitve za RSA SecureID, Global HID in Google Authenticator s svojo izvedbo TOTP, so potrebna dodatna prizadevanja uporabnika v primerjavi z rešitvijo mTAN. Dejansko aplikacijo za pametne telefone je potrebno najprej prenesti iz trgovine Google Play Store, potem pa mora kamera pametnega telefona preslikati QR kodo ali s prenosom semena prek datoteke pri RSA in HID, kar zahteva, da ima uporabnik dostop do pametnega telefona v času namestitve. Pravzaprav je uporaba programa 2FA po namestitvi podobna kot mTAN z enakim številom znakov, ki se generirajo za uporabo OTP, in dokler je napravi zagotovljena internetna povezava, ni dodatnega vzdrževanja.

6.1.3 Komunikacija

Varno **komuniciranje** z omrežjem je pomemben del vsega, kar je povezano z avtorizacijo ali overjanjem. Če se s tem ne ravna pravilno, so občutljive informacije, povezane z uporabniškimi računi ali prijavnimi sejami, ranljive za prisluškovanje.

Shema, ki ni občutljiva za varno komunikacijo, saj ni zahtevana v času overjanja, je TOTP. Čeprav je potrebno skupno skrivnost prenesti na uporabniški pametni telefon, ne uporablja komunikacijskega kanala, ki mu je mogoče prisluškovati.

Metoda mTAN prenese svoj OTP preko telefonskega omrežja (komunikacijski kanal), ki mu s praktičnega vidika ni enostavno učinkovito prisluškovati. Komunikaciji s pametnim telefonom je mogoče prisluškovati z ustrezno opremo in UMTS (angl. Universal Mobile Telecommunications System) je mogoče izkoristiti z napadom Man-In-The-Middle. V primeru GSM (angl. Global System for Mobile communications) je potrebno, da se napadalec nahaja v istem geografskem območju, saj je potrebno promet, ki potuje z bazne postaje GSM na uporabniški telefon, ujeti (Hell & Altman, 2015).

6.1.4 Človeški faktor

Hell in Altman (2015, str. 36) sta mnenja, če skupna strategija pri napadih na sisteme, ki zahtevajo interakcijo z ljudmi, je, da napadajo osebo s tem, da jo prisilijo v dejanja, ki bi lahko potencialno škodovala sistemu ali uhajanju občutljivih podatkov. Lažni napad (angl. phishing attack) je nezakonit poskus pridobitve občutljivih informacij, kot so uporabniška imena, gesla in podrobni podatki o kreditni kartici, včasih posredno tudi denar, zaradi predstavitve kot zaupanja vredno podjetje v elektronski komunikaciji.

2FA rešitev RSA, HID in GA te napade preprečujejo, tako da je napadalcem nemogoče izvesti napad na uporabnike. Čeprav ni zabeležen noben napad lažnega predstavljanja zoper TOTP metodo, je teoretično mogoče, da se tak napad izpelje. Napadalec bi lahko uporabil orodje Webinject[5], da bi uporabnika preslepil, da vnese veljaven OTP, ki bi ga nato uporabil napadalec.

Nasprotno pa je pri rešitvi mTAN, kjer celoten napad kraje OTP-jev iz prejetega SMS-a temelji na neprevidnosti uporabnika. To je zelo močan napad, ki ga napadalci uporabijo za izogibanje 2FA v tem smislu, da ne izkoriščajo nobene ranljivosti v operacijskem sistemu ali sami programski opremi.

[5] Webinject – orodje za avtomatsko testiranje spletnih aplikacij in spletnih storitev

6.1.5 Izgubljen ali ukraden pametni telefon

V primeru ukradenega ali izgubljenega pametnega telefona morajo biti vzpostavljeni varnostni mehanizmi, da se prepreči neželeno uporabo 2FA. To je še posebej pomembno, če uporabnikov brskalnik shrani poverilnice ali piškotke za prijavo na spletne storitve.

Vsi operacijski sistemi pametnih telefonov imajo funkcijo odklepanja, ki zahteva določeno obliko vnosa od uporabnika. To je lahko PIN, vzorec ali v nekaterih primerih le navaden poteg po zaslonu. V podjetjih, kjer se uporablja RSA ali HID, se to lahko naredi prek politike obveznega PIN-a na pametnem telefonu s pomočjo aplikacije za elektronsko pošto oziroma sporočilnega sistema. Google ponuja na Androidu metodo prepoznavanja obraza, ki jo Google označuje kot zelo varno, in je lahko hkrati priročna ter zabavna za uporabo. Obstaja tudi druga metoda biometričnega odklepanja, ki obstaja na prenosnih računalnikih že nekaj časa, skeniranje prstnih odtisov (Hell & Altman, 2015).

Če se faza odklenitve telefona zaobide ali preprosto ni omogočena, je mTAN metoda izpostavljena napadalcu, če aplikacija za SMS ne potrebuje gesla za uporabo. RSA, HID in GA so izpostavljeni isti težavi in napadalec lahko preprosto odpre aplikacijo za OTP. Pri tem je potrebno še dodati, da imata RSA in HID možnost nastavitve PIN-a pri zagonu aplikacije programskega žetona.

Učinkovit način za zaščito pred napadi za ukraden telefon je, da ga lahko na daljavo zaklenemo. Android ima možnost, da to storimo prek brskalnika s prijavo z Google računom. Vendar če je za račun omogočena 2FA in potreben izgubljeni pametni telefon, obstajajo še vedno načini, kako dostopiti do računa. Google predlaga, da uporabnik uporablja nadomestne kode, se prijavi iz zaupanja vrednega računalnika ali izpolni tako imenovan obrazec za obnovitev računa. Pri rešitvi RSA in HID je to dokaj enostavno, saj mora uporabnik samo sporočiti službi za pomoč uporabnikom, kjer onemogočijo nadaljnjo avtentikacijo s programskim ključkom, nameščenim na ukradenem ali izgubljenem pametnem telefonu.

6.2 Primerjava proizvajalcev 2FA

2FA za preverjanje pristnosti lahko podjetju prinese pomembno prednost, vendar je tehnologija zapletena, orodja pa se lahko močno razlikujejo od proizvajalca do proizvajalca.

Zelo priporočljivo je preučiti uporabo posebnih orodij, ki prikazujejo, kako lahko izdelek proizvajalca izpolnjuje večfaktorske potrebe in zahteve podjetja. Predvsem bomo skušali

primerjati RSA in HID rešitev, ker je postavitev on premises [6] in jo upravljamo, medtem ko je rešitev GA upravljana s strani Google.

HID ima dva ločena izdelka – ActiveID, ki je nameščen lokalno, in HID Approve kot potisnjeno preverjanje pristnosti (angl. push notification authentication). RSA je namenjen samo lokalnim namenom (angl. on-premises), čeprav imajo podjetja lahko nameščena navidezno upravljanje gostiteljske različice. GA se obravnava le kot storitev v oblaku, čeprav ima dodatek Google Authenticator Pam Module. Vse to deluje zaradi knjižnice, imenovane PAM (angl. Pluggable Authentication Modules), ki omogoča komunikacijo prek FreeRADIUS. Za primerjavo bomo nadaljevali z lokalnimi namestitvenimi različicami ActiveID in SecureID (Strom, 2014).

Oba 2FA izdelka nudita možnosti za preverjanje pristnosti, povezavo z aktivnim imenikom, preverjanje spletnih storitev in razširjene prijave spletnega strežnika – skupaj v enem izdelku. Vendar vsaka možnost preverjanja zahteva dodatne module za podporo SAML (angl. Security Assertion Markup Language) ali aktivnemu imeniku (angl. Active Directory). Na primer, Authentication Manager RSA je aplikacija rešitve SecureID, ki sodeluje s Adaptive Federation Manager za zagotavljanje integracije spletnih storitev SAML (Strom, 2014).

Različna paleta dodatkov je nasploh značilna za proizvodni prostor 2FA, zato je zelo pomembno, da razumemo, katere aplikacije in pod katerimi okoliščinami podjetje želi uporabiti dodatne faktorje.

Če govorimo o dodatkih, preden izberemo rešitev 2FA, ki temelji na svoji aplikacijski podpori, je pomembno razumeti, kako se ta podpora zagotavlja. Oba vrhunska ponudnika 2FA za preverjanje pristnosti vsebujeta več strežniških komponent programske opreme ali agentov, ki jih je treba namestiti, da bi okrepili prijavo prek zunanjega dostopa, strežnikov ali spletnih storitev.

Medtem ko to pomaga povečati njihov doseg, povečuje tudi stopnjo **kompleksnosti** namestitve in delovanja, saj obstaja več elementov za konfiguriranje in sledenje. Del postopka ocenjevanja izdelkov 2FA je, kaj se zgodi, ko uporabljate običajne vsakodnevne dejavnosti teh orodij, kot je registriranje novih žetonov in novih uporabnikov, nastavitev zaščite za novo aplikacijo, spreminjanje varnostnih pravil in iskanje rešitve, zakaj se uporabnik ne more prijaviti v poslovne aplikacije. Izdelki, kot sta RSA in HID, ponujajo veliko prožnost pri procesih prijave z žetonom (angl. token-workflow). To se odraža sorazmerno s količino časa, ko se podjetja ukvarjajo z 2FA. Rešitvi imata tudi samopostrežni

[6]on premises – programska oprema je nameščen in deluje na računalnikih v prostorih podjetja

portal, kamor se uporabnike napoti, in kjer lahko nastavijo svoje podatke, ponastavijo PIN, zahtevajo nov žeton ali podaljšajo veljavnost obstoječega (Strom, 2014).

Oba proizvajalca 2FA imata možnost vključiti veliko različnih poročil in različnih možnosti za izvoze poročil ter možnost načrtovanja določenih poročil in spremljanja opozoril ter drugih dejavnosti v realnem času. Pri tem je RSA naprednejši, saj omogoča spremljanje prijave v trenutnem času, kar nam zelo olajša odpravljanje napak pri preverjanju pristnosti.

Orodji povečata svoja preverjanja pristnosti na podlagi tveganja (angl. risk-based authentication). Ta funkcionalnost je dodana, da se okrepijo metode preverjanja pristnosti s sorazmerno novim mehanizmom, ki se različno imenuje. Pri RSA je to preverjanje pristnosti na podlagi tveganja (angl. risk based-authentication), pri HID grožnje in odkrivanja goljufij (angl. threat and fraud-detection). Ta mehanizem omogoča beleženje prijav in jih oceni na podlagi določenega vedenjskega vzorca (Strom, 2014).

RBA deluje tako, da dostop do določene poslovne aplikacije poteka skozi vrsto zaupanja. Večje kot je tveganje, večja varnost je potrebna, zato uporabniki ne vedo, da so njihove prijave bolj skrbno preverjene. Poleg tega se to zgodi v realnem času, tako kot pri tipičnih večfaktorskih metodah. To je podobno požarnim zidovom naslednje generacije, ki delujejo z lastnimi orodji za ocenjevanje tveganj obnašanja notranjega omrežnega paketa (Strom, 2014).

Preverjanje pristnosti na podlagi tveganja uporablja elemente, kot so (Strom, 2014):

- Vloga: Ali je uporabnik član privilegiranega razreda, kot so skrbniki omrežja ali nadzorniki računa? Če je tako, morajo opraviti strožje dialoge za preverjanje pristnosti;
- Lokacija: bodisi z zaznavanjem fizične lokacije ali določene geografske lokacije. Na primer, če se je uporabnik prijavil pred desetimi minutami iz Kanade in se zdaj poskuša prijaviti s Kitajske, se to zagotovo šteje za transakcijo z višjim tveganjem. Lahko imamo tudi druge atribute v splošni oceni tveganja;
- Na podlagi dejavnosti: na primer, prenosi velikih vrednosti imajo večje tveganje;
- Spremembe običajnih vzorcev transakcij: če uporabnik počne nekaj, kar se ne ujema z njegovo zgodovino, postane bolj tvegana transakcija, zahteve za preverjanje pristnosti in prijave pa se izpodbijajo z dodatnimi ukrepi za preverjanje pristnosti. Zaradi dvoma nenavadnih vzorcev uporabe se ustvari ovira, ki se je heker ali goljuf ne more zlahka izogniti.

Vsak od izdelkov podpira osnovne štiri mobilne operacijske sisteme: Windows Phone, Apple iOS, Android in BlackBerry, zato ne bi smelo biti težav, razen če je v uporabi mobilna naprava, ki jo izbrani ponudnik ne pokriva, kot npr. nekaj starih telefonov operacijskega sistema Windows ali Android.

Vsak produkt ima široko zbirko strojnih in programskih žetonov, ki se lahko uporabijo kot dodatni dejavniki preverjanja pristnosti. To jim daje največjo fleksibilnost pri zagotavljanju določenih prijav in storitev, ki se lahko srečajo skoraj v vsakem primeru. Poleg mobilnih aplikacij ponujajo tudi namizno programsko opremo za zagon enkratnih generatorjev gesel. Čeprav je to lepa funkcija, razen če večina uporabnikov podjetja izključno uporablja namizne računalnike, verjetno ni razlog za izbiro enega od teh izdelkov pred drugim izdelkom 2FA.

Vsak od teh izdelkov dobro opravlja delo pri zagotavljanju z 2FA zaščito. Podpirajo metode mobilnega žetona, imajo prožne metode preverjanja pristnosti in metodo, ki temelji na tveganju. Njihove razlike so bolj stvar pakiranja, določanja cen in ali lahko osebje organizacije razume in dela z različnimi poročili, ki jih vsak proizvaja, kar je bolj stvar sloga kot vsebine.

7 ŠTUDIJA PRIMERA

Marsikatero zanimivo spoznanje pa lahko pridobimo na osnovi analize primerov. V vseh obravnavanih primerih so oz. so bili vključeni študija primera RSA, študija primera najboljše prakse ter študija primera GA. Študije primera nam lahko pomaga pri odgovorih na vprašanja, ki so bila izpostavljena v začetku naloge.

7.1 Študija primera RSA

Študija primera (Evolver, 2018) se navezuje na finančni sektor v ZDA, kjer so želeli dodatno varnost za uporabnike, predvsem pa izboljšati zaščito do podatkov in sistemov z 2FA avtentikacijo z uporabo RSA SecurID-ja.

Finančni sektor je bil v središču pozornosti zelo pozno, šele ko je bila sprejeta nova kibernetska ureditev za finančno industrijo. Razlog za dodatni poudarek na kibernetski varnosti so podatki potrošnikov, ki so občutljivi, in seveda denar. Vse finančne institucije se soočajo s predpisi in grožnjami v tem sektorju, prednost in korist pa imajo le od uvedbe 2FA.

Nemogoče je imeti eno rešitev, ki ustreza vsem programom kibernetske varnosti, ker nekateri elementi veljajo samo za organizacijo ali za celoten finančni sektor. Mnogo ljudi govori o pomembnosti 2FA, ker znatno izboljša varnost, vendar se pogosto obravnava kot ovira za uporabnike zaradi dodanega koraka. Sistem, kot je RSA SecurID, ki se integrira s tem, kar je v podjetju, je ključnega pomena za usposabljanje, sprejetje in uspešno implementacijo.

Izziv za podjetje na področju kibernetske varnosti je, da dodatno zavaruje dostop do virov podjetja ter:

- prepreči uporabo skupnih poverilnic, še posebej administratorskih poverilnic, ki jih uporablja prodajno osebje;
- ustvariti protiukrep za preprečevanje vpliva ukradenih poverilnic zaradi zlonamernih napadov;
- zagotoviti način varnega dostopa do vseh notranjih aplikacij preko spletnega portala, kjer mora biti dostopnost od kjerkoli na svetu.

Ponudnik je priporočal rešitev in implementirana je bila uporaba RSA SecurID. Ta dvofaktorska aplikacija za preverjanje pristnosti je zagotovila varen in priročen dostop za vsakega uporabnika, od kjerkoli.

Izdelek RSA SecurID zagotavlja prednosti in funkcionalnost RSA Authentication Manager-ja in storitve za preverjanje pristnosti v oblaku, združene v en izdelek.

Nekatere funkcije vključujejo:

- možnost izbire med različnimi metodami preverjanja pristnosti, kot je preverjanje s programskimi žetoni, preverjanje prstnih odtisov, identifikator EyePrint ID ali standardnimi žetoni;
- več načinov za povezovanje, na primer z varnim dostopom na podlagi politike, z enojno prijavo do vodilnih spletnih in Saas aplikacij skozi SAML, reverznega proxy-ja ali trezorja za gesla;
- fleksibilnost uporabe obstoječih žetonov RSA SecurID za zaščito oblaka, uporaba aplikacije RSA SecurID s tradicionalnimi lokalnimi viri, kot so VPN-ji, ali s kombinacijo, da bi izpolnili edinstvene zahteve.

Prednosti RSA SecureID dostopa za podjetje so bile sledeče:

- podjetje lahko svojim uporabnikom zagotovi varno in priročno dvofaktorsko preverjanje pristnosti;
- podjetje lahko zaščiti občutljive podatke in kritične sisteme;
- okrepljen postopek prijave tako, da ga ni le otežil, temveč tudi prepreči uporabo kompromitiranih poverilnic brez dodatnega varnostnega koraka z varnostnim žetonom;
- dodan je dodaten varnostni sloj za zaščito podjetja pred napadom lažnega predstavljanja, socialnega inženirstva ter množičnega ugibanja gesla. Zaščitene prijave pred napadalci, ki izkoriščajo šibke ali ukradene poverilnice;
- integrira se z aplikacijami v podjetju, spletnem mestu in v oblaku.

Uspešna vpeljava RSA SecudeID za dvofaktorsko avtentikacijo je dodala sledeče izboljšave:

- zaščita s programskim žetonom RSA SecureID za dvofaktorsksko avtentikacijo. Žetoni omogočajo uporabnikom, da imajo možnost preverjanja pristnosti s pomočjo aplikacije pametnega telefona RSA ali tradicionalnega ključka strojne opreme;
- nastavljene politike dostopa, s katerimi lahko vsa preverjanja pristnosti potrjujemo glede na posamezni dodeljeni ključek. Dostop brez validiranega ključa ni mogoč, tudi če je geslo pravilno;
- upravljani dostop s politikami občutljivosti na kontekst, ki gledajo na atribute aplikacije in atribute uporabnika;
- avtomatizirano evidentiranje, spremljanje in beleženje nepooblaščenega dostopa ter neuspešnih poskusov preverjanja pristnosti;
- vgrajen RSA SecureID dostop do spletnega portala stranke. Ta integracija omogoča uporabnikom avtentikacijo in povezovanje z notranjimi aplikacijami od koderkoli na svetu.

7.2 Študija primera najboljše prakse

Druga študija (Mulliner, 2018) je primer najboljše prakse prehoda investicijske banke s strojnih na programske žetone. Banka se je v veliki meri osredotočala na inovacije in tehnologijo, da zagotovi strankam enostavno uporabo spletne in mobilne varnosti. Leta 2016 je banka začela projekt, ki strankam omogoča izboljšano avtentikacijo, vključno s tehnologijo varne avtentifikacije na osnovi pametnih telefonov za mobilne transakcije. Z integriranjem najboljše avtentikacije neposredno v aplikacijo za mobilne naprave in spletno stran je banka izboljšala varnost, da je izpolnila nove zakonske zahteve in zmanjšala stroške, povezane z izdajo in podporo strojnim ključkom.

Banka se je morala soočiti z **izzivom**, ker je bila primorana nadgraditi svojo tehnologijo za preverjanje pristnosti. Medtem ko so bančne stranke že uporabljale OTP drugega ponudnika, je banka morala implementirati močnejšo avtentikacijo, da bi izpolnila nove varnostne oziroma regulativne zahteve. Skladnost z regulativo ni bila edini izziv. Za nadgradnjo so bili še trije dodatni razlogi:

- **kmalu potečeni strojni žetoni:** Ena tretjina vseh OTP generatorjev, ki jih uporabljajo stranke, je le nekaj mesecev od izteka;
- **izkušnje kupcev:** Banka je še vedno uporabljala sistem z več gesli. Stranke so se avtenticirale s svojo OTP napravo v kombinaciji s tremi gesli, vendar si je bilo težko zapomniti tri različna gesla. Postalo je vse bolj pomembno poenostaviti avtentikacijo, ker uporabniki, ki so pozabili gesla, niso mogli plačevati ali opravljati poslov;
- **gesla za ponovno izdajo so bila draga:** Metoda z več gesli je ustvarila veliko delovno obremenitev za službo za pomoč uporabnikom. Pozabljena gesla so sprožila ročni

postopek ponastavitve gesla, ki je porabil dragocen čas službe za pomoč. Odprava tega bi lahko povzročila znatne prihranke.

Banka je že več let stranke oskrbovala s tradicionalnimi strojnimi žetoni. Vendar so inovacije v svetu mobilne varnosti bankam dale nove možnosti, in sicer preverjanje pristnosti s programsko opremo za spletne transakcije.

Banka je naredila **analizo stroškov** za primerjavo dveh metod avtentifikacije. Analiza ostaja zaupna, je pa jasno, da s stroškovnega vidika uporaba samo strojne opreme ni bila možnost. Uvedba avtentifikacije s programsko opreme bi zagotovila močnejšo zaščito za mobilne uporabnike, hkrati pa znižala stroške.

Vendar je banka imela pomisleke glede sprejetja pri strankah, zato je raziskala oziroma **anketirala** svojo bazo strank, da bi potrdila pripravljenost sprejetja avtentikacije za pametne telefone. Podatki so pokazali mešane rezultate. Nekateri komitenti so bile pripravljeni, drugi niso. Banka je ugotovila in odločila, da je hibridno izvajanje najboljša strategija. Dejansko so njihove raziskave potrdile, da večina kupcev dejansko želi oboje. Kupci želijo uporabiti svoje mobilne naprave, vendar če gre kaj narobe, kot je izguba telefona ali izpraznjena baterija, še vedno želijo imeti strojni ključek kot rezervno opcijo.

Medtem ko bi preverjanje pristnosti programske opreme omogočilo prihranke, se je odprla druga plat. Nekateri komitenti niso imeli pametnega telefona. Med tistimi, ki so imeli pametni telefon, so rezultati raziskav pokazali odpor na spremembe. Medtem ko je segment imetnikov pametnih telefonov zanimalo preverjanje pristnosti s programsko opremo, niso vsi hoteli uporabljati svojega pametnega telefona kot faktorja, kaj imaš za preverjanje pristnosti.

Banka je morala tako premagati tri ovire za sprejetje:

- pomanjkanje poznavanja in zaupanja v mobilno preverjanje pristnosti;
- zaskrbljenost, ker stranke že imajo preveč aplikacij in ne želijo izgubljati prostora na telefonu;
- skrbi za izgubo ali krajo telefona.

Banka se je odločila za uvedbo hibridnega sistema strojnega in programskega preverjanja pristnosti na promociji programske opcije.

Da bi podprla hibridni pristop, je banka želela enega samega ponudnika, ki bi lahko ponudil obe metodi avtentifikacije. Vse druge ključne **zahteve** so bile razdeljene v tri kategorije:

- **varnostne izkušnje:** Banka je zahtevala ponudnika in zaupanja vrednega partnerja z globokim strokovnim znanjem;

- **izkušnje uporabnikov:** Banka je potrebovala ponudnika z dokazno zgodovino, ki kaže na brezteţavno a zelo dobro izkušnjo z uporabniki.
- **lokalni distribucijski center:** Banka je imela zahtevo za dokazano varnostno prakso glede fizične distribucije novih strojnih ţetonov, saj bi moral ponudnik delati z osebnimi podatki uporabnika. Banka je zahtevala, da je PII (angl. personally identifiable information) še naprej prisoten.

Rešitev je poleg tega, da svoje strojne naprave preklopijo na izbranega ponudnika, banka integrirala tudi programsko opremo neposredno v svoje mobilne in spletne aplikacije.

Opazili so, da imajo uporabniki na začetku odpor proti spremembam, toda ko poskusijo mobilno preverjanje pristnosti, so zelo zadovoljni in ostanejo pri tem. Zato je komunikacija tako pomembna. Stranke so morali prepričati, da ga preizkusijo.

Za izvedbo rešitve je banka sestavila skupino, sestavljeno predvsem iz notranjih virov. Ti so vključevali:

- člani varnostnega oddelka;
- osnovna ekipa šestih zunanjih razvijalcev;
- skupina za upravljanje sprememb v banki.

Pri **izdaji in sprejetju** je od leta 2016 do leta 2017 banka dala prednost namestitvi dvema skupinama uporabnikov: tistim, ki bi se jim strojni ţetoni kmalu iztekli, in uporabnikom mobilnih aplikacij.

Banka je uporabnikom posredovala obvestila za spremembo po elektronski pošti, z objavo na spletni strani, kot posebno stran z informacijami, videoposnetki ter pogosto zastavljenimi vprašanji in službo za pomoč uporabnikom.

Banka se je odločila za postopno razporeditev iz dveh razlogov:

- pritisk na oddelku za pomoč uporabnikom: Migracija vseh ljudi hkrati je težavna zaradi velike količine klicev, tudi z zunanjo podporo;
- proračun: banka ima veliko strank, katerih življenjska doba baterije strojnega ključka je še vedno veljavna. Ti uporabniki lahko še naprej uporabljajo strojni ključek.

Od izkušenj banke je sprejemanje uporabnikov odvisno od:

- kako sporočiti dodano vrednost uporabniku?
- kakšne možnosti ponujate?

Način, kako banka pojasni nove metode preverjanja pristnosti uporabnikom, neposredno vpliva na sprejem, zato najprej promovira programsko metodo. Če stranka nima pametnega telefona, bo banka predstavila možnost strojne opreme.

Bili so mnenja, če jih prosite, da izbirajo med A in B, je verjetno, da bo sprejetje programske opreme nižje, ker kupci niso vedno pripravljeni na spremembo. Vzrok za sprejetje je tako visok, ker v njihovih e-poštnih sporočilih strankam promovirajo le programski žeton.

To je bila ena od izkušenj z začetnega uvajanja leta 2016, ko je banka kupcem ponudila izbiro avtentikacije strojni ali programski žeton. Naslednje leto je banka spremenila svoj pristop. S spodbujanjem mobilnega preverjanja pristnosti so banke v aktivacijah zabeležile pomemben dvig, pri čemer je 62 % uporabnikov aktiviralo programski ključ.

Uspešna uvedba je bila zaradi ponudnika, ker ponuja celoten spekter procesov za uvajanje, zagotavljanje in aktiviranje rešitev za preverjanje pristnosti. Ti postopki zagotavljajo varno generiranje, shranjevanje in dostavo osebnih poverilnic uporabnikov, kar preprečuje napade na poverilnice. V večini implementacij so banke sposobne znova uporabiti obstoječe prakse in komunikacijske metode s svojimi strankami, da odpravijo preobremenitev službe za pomoč uporabnikom in zmanjšajo varnostna tveganja. To zagotavlja bankam, da pred goljufi v prehodnih obdobjih preprečijo napad socialnega inženiringa.

Med izvedbeno fazo projekta so uporabniki z izkušnjami in varnostni svetovalci sodelovali z bančnim osebjem za razvoj procesov upravljanja s poverilnicami. Na primer:

- rešitev ponuja več načinov za aktiviranje uporabniških naprav za preverjanje pristnosti. To omogoča banki ustvariti uporabniku prijazne delovne tokove za selitev iz starih žetonov na novo metodo;
- rešitev zagotavlja edinstveno varno vizualno tehnologijo, ki strankam omogoča, da aktivirajo svoje poverilnice v nekaj sekundah;
- ker je izvorno integriran znotraj mobilne aplikacije banke, rešitev omogoča samodejno povezovanje računa stranke s svojimi varnostnimi poverilnicami in podatki, specifičnimi za naprave.

Ena izmed najbolj opaznih **koristi** je bila raven zadovoljstva strank med tistimi, ki so poskušali preveriti avtentikacijo s programskim ključem. Povratne informacije oseb, ki so aktivirale programski ključ, so bile zelo pozitivne, ker je veliko lažji za uporabo in so vedno pri njih. Avtentikacija poteka veliko hitreje, ker uporablja PIN namesto gesel, ki jih porabniki pozabljajo.

Udeleženci projekta so bili zadovoljni z:

- enostavnostjo integracije v bančno aplikacijo;

- banka ni bilo treba usmeriti uporabnikov na ločeno aplikacijo le za preverjanje pristnosti;
- celovito pokrivanje varnostnih groženj, ki strankam zagotavlja zaupanja vredno rešitev za preverjanje pristnosti prek svojih mobilnih naprav;
- kakovost podpore po prodaji, ki so jo prejeli od ekipe ponudnika.

Na splošno večina uporabnikov ni imela težav z **razumevanjem** mobilne avtentikacije. Informacije so našli na spletnem mestu, jih prebrali ter aktivirali in začeli uporabljati avtentikacijo brez pomoči centra za pomoč uporabnikom.

Eno od ključnih **učenj** je bila pomembnost ustrezne priprave za uporabnike, ki potrebujejo podporo. Majhen odstotek lahko povzroči veliko obremenitev službe za pomoč uporabnikom. Za to niso bili popolnoma pripravljeni. Morali so zelo hitro povečati skupino za pomoč uporabnikom. Druga lekcija je bila pomembnost prilagajanja komuniciranja z uporabniki. Najboljša praksa je razčlenjevanje in prilagajanje komunikacij različnim uporabniškim skupinam, saj vsi uporabniki ne vedo, kaj je koda QR (angl. Quick Response Code). Niso vsi strokovno podkovani, nekateri pa ne zaupajo novim metodam preverjanja pristnosti.

Velik uspeh za banko je bilo izobraževanje in videi z navodili. Vsi uporabniki si ne bodo vzeli časa za temeljito branje informacij na spletni strani ali e-poštnih sporočil banke.

Kot je poudarjeno v tej študiji primera, uspešna migracija zahteva ravnovesje vrhunske tehnologije in dokazane prakse za upravljanje sprememb in pospeševanje sprejetja. Preverjanje pristnosti s programsko opremo zagotavlja prepričljive prednosti, večjo varnost, preprostejšo uporabniško izkušnjo in znatne prihranke pri stroških.

7.3 Študija primera Google avtenticator

Študija primera (VarniNaInternetu, 2017) za Google Avtenticator je vezana na borze kripto valut. Zahteva je zavarovati težko prislužene prihranke, da ne bi pristali v rokah oziroma denarnicah kriminalcev, ki jim kripto kovanci zelo dišijo, in upoštevanje dobre varnostne prakse, s katero bi precej zmanjšali možnost zlorabe in posledično krajo kripto kovancev in žetonov. Vse spletne borze in storitve omogočajo nastavitev 2FA, pri katerih moramo pri prijavi poleg običajnega gesla vpisati tudi unikatno, časovno spremenljivo kodo, ki jo generira aplikacija na telefonu. S tem izredno povečamo varnost računa, saj napadalec vanj ne bo mogel vdreti, tudi če jim nekako uspe ukrasti geslo.

S porastom števila uporabnikov kripto valut so se v zadnjem času na tem področju močno povečale tudi dejavnosti kriminalcev. Število phishing napadov na storitve, povezane s kripto valutami, se je izredno povečalo. Po nekaterih izračunih naj bi v letu 2017 preko takih napadov prišlo do več 100-milijonskih izgub. Napadalci postavijo zelo prepričljive kopije spletnih storitev in nato na različne načine nanje preusmerijo žrtve. Če uporabniki na lažni

strani vpišejo svoje podatke, jih s tem pravzaprav sporočijo napadalcem. Zaščita pred temi napadi je preprosta: do storitve nikoli ne dostopajte prek ponujenih povezav, ampak vedno prek zaznamka v brskalniku. Na udaru so tudi t. i. ICO (angl. Initial Coin Offering) oblike množičnega zbiranja kapitala**,** pri katerih morajo investitorji sredstva nakazati v točno določeno denarnico. Napadalci lahko na različne načine uporabnikom podtaknejo svoj naslov denarnice. Sredstva, ki jih investitorji nakažejo v tako denarnico, so za vedno izgubljena. Naslov denarnice, kamor nakazujete sredstva, vedno preverite preko različnih kanalov uradne spletne strani, družabnih omrežij ipd.

Prejemanje kod prek SMS sporočil z varnostnega stališča ni najbolj priporočljivo, to je tudi možnost za Googole Avtenticator na spletni borzi, zato raje uporabimo namensko aplikacijo Google Authenticator na pametnem telefonu. Pri nastavitvi avtentikacije obvezno preverimo možnost, kako si lahko povrnemo dostop do računa, če izgubimo aplikacijo za generiranje kod. Pri nekaterih ponudnikih si moramo zapisati dodatno kodo, drugi pa v teh primerih zahtevajo, da kontaktirate pomoč uporabnikom. Osnova za varno uporabo spletnih storitev je uporaba 2FA na našem predalu za elektronsko pošto, na katerega imamo vezane te storitve.

8 SWOT ANALIZA

Spodnja SWOT analiza predstavlja prednosti, slabosti, priložnosti in nevarnosti 2FA na pametnih telefonih.

Tabela 4: SWOT analiza 2FA na pametnih telefonih

Notranji dejavniki	
PREDNOSTI (+)	**STABOSTI (-)**
➢ ni potrebe po nošenju dodatne strojne opreme ➢ stroškovna učinkovitost ➢ enostavna distribucija ➢ podpira vse platforme OS pametnih telefonov	➢ dodeljevanje ključkov potrebuje kontrolirano okolje ➢ dovzetnost za razkritje skupne skrivnosti pri inicializaciji
Zunanji dejavniki	
PRILOŽNOSTI (+)	**NEVARNOSTI (-)**
➢ pridobivanje novih uporabnikov v korelaciji z varnostnim mehanizmom ➢ integracija v različne produkte	➢ izguba, kraja pametnega telefona ➢ izguba obnovitvenih gesel ➢ nedostopnost do spletnih virov

Vir: Lastno delo.

Programski žetoni imajo kar nekaj prednosti, saj nadomeščajo fizične žetone kot programska aplikacija, ki se izvaja na različnih mobilnih napravah oziroma pametnih telefonih. Z enostavnostjo se integrira s strojno opremo in napravami, ki jih imajo uporabniki, kot je pametni telefon. Programski žetoni so preprosti za uporabo in jih je enostavno upravljati ter ne zahtevajo namenske strojne opreme. Rešitev je stroškovno učinkovita in se s svojim potencialom brezmejno lahko razširi, saj podjetjem ni treba distribuirati in upravljati svojih naprav. Z zagotavljanjem, da vsi v organizaciji uporabljajo isti varnostni protokol, programski žetoni poenostavljajo upravljanje varnosti. Ponujajo bolj fleksibilno, dinamično, varno in enostavno upravljanje v današnjem vse bolj mobilnem okolju, kakor tudi rešitev v oblaku oziroma integracijo v različne produkte. Zagotavljajo večjo hitrost dostopa in širok razpon razporeditve. Programski žetoni prav tako povečujejo zadovoljstvo uporabnikov tako zaposlenih in strank, ker naprave že imajo in uporabljajo kot pametne telefone. Pravzaprav večina uporabnikov pametnih telefonov ima telefon s seboj v vsakem trenutku, kar zagotavlja, da ne bodo pozabili svojega žetona doma. Ker se programski žetoni vključujejo v obstoječo tehnologijo, usposabljanje v večji meri ni potrebno, kar olajša potrebo po namestitvi in pomoči IT. Za slabost, razen za GA, ker je kot storitev v oblaku, dodeljevanje ključkov potrebuje kontrolirano okolje in v vseh primerih izvedb programskega ključka ob neprimernih postopkih, dovzetnost za razkritje skupne skrivnosti pri inicializaciji, kadar je seme najbolj izpostavljeno. Izguba, kraja pametnega telefona, izguba obnovitvenih gesel in nedostopnost do spletnih virov so nevarnosti, ki lahko povzročijo popolno nedostopnost do svojih virov, ki imamo zaščitene z 2FA.

9 NAJBOLJŠA PRAKSA

Da bi lahko z večjo varnostjo vplivali na zmanjšanje tveganja, ranljivosti uporabe programskih generatorjev gesel in preprečili morebitno okužbo programske opreme z zlonamerno kodo ali prišli do OTP gesla s povratnim inženirstvom, je v veliki meri odvisno od pravilne uporabe 2FA kot celote. V nadaljevanju se bomo dotaknili zelo dobrih priporočil uporabe, ki smo jih skozi nalogo opisovali ter sedaj sestavili v mozaik, s katerim skušamo poskrbeti za zlonamerno nedotakljivost programskih generatorjev.

Če pogledamo vse študije primera, imajo pri uvedbi programskih žetonov kar nekaj skupnih ugotovitev, in sicer vsak si želi doseči sledeče:

- izboljšati varnost za svoje stranke;
- poskrbeti za hitrejšo in lažjo uporabniško izkušnjo;
- zmanjšati stroške;
- izboljšati dostop do storitev za stranke.

Namen magistrskega dela je tudi poiskati najboljše prakse, ki so zasnovane za zagotavljanje varnega delovanja izdelkov programskega žetona in jih v tem poglavju tudi navajamo. Odgovornost je, da se zagotovi pravilna uporaba, skrbno spremljanje ter vzdrževanje programskih žetonov.

Produkti programskih ključkov so izdelani tako, da varno naključno izberejo seme programskega žetona ob izdaji tako, da prejšnje seme ni več veljavno. Hkrati pa so platforme zasnovane tako, da naključno generirajo novo seme za vsak ključek posebej, ko se jih kot nove, uvaža v sistem (RSA, 2017b).

Distribucija programskih žetonov je najpomembnejša prvina, saj se lahko že na samem začetku kompromitira žeton zaradi izpostavljenosti semena in v ta namen obstaja kar nekaj zelo varnih možnosti pri RSA in HID, in sicer (RSA, 2017b):

- dinamična distribucija semena (angl. Dynamic seed provisioning): protokol s štirimi prehodi, ki uporablja protokol za inicializacijo ključa za kriptografski žeton (angl. Cryptographic Token Key Initialization Protocol, CT-KIP), da se odpravi potreba po distribucijski datoteki žetonov;
- distribucija datoteke (angl. File-based provisioning, SDTID): datoteka programskega žetona, ki se distribuira kot priloga v e-pošti;
- format stisnjenega žetona (angl. Compressed Token Format, CTF): alfa numerični ali numerični format za pošiljanje programskih žetonov na pametne telefone.

Priporoča se uporaba funkcije dinamične distribucije semena, ker postopek CT-KIP preprečuje morebitno prestrezanje semena žetona. SDTID ali CTF pa se naj uporabi samo, če politika podjetja določa, da se aplikacija žetonov ne more povezati z internetom ali da strežnika CT-KIP ni mogoče nastaviti. Distribucija programskih žetonov se naj izvrši po naslednjih korakih (RSA, 2017b):

- dodelitvi kompleksnega gesla vsem žetonom programske opreme, ki se distribuirajo s pomočjo datoteke SDTID. Gesla naj ustrezajo najboljšim praksam in ga uporabniku pošljemo ločeno od datoteke SDTID žetona po varnem kanalu;
- določitvi kompleksnega gesla za naslove CTF po meri. Z uporabo varnostne konzole konfiguriramo žeton programske opreme, izberemo možnost distribucije CTF in izberimo možnost za zaščito gesla.
- povezovanje programskih žetonov z ID-ji pametnih telefonov.

Prav pri slednjem, povezovanju programskega žetona na točno določen pametni telefon, dosežemo namestitev le na napravo ali razred naprav z ID-jem prilagojene naprave. S tem dosežemo, da se programski žeton ne more namestiti na noben drug pametni telefon in tako postane edinstven faktor, kaj imamo. Kot smo že poudarili, se moramo izogniti težavam pri

preverjanju pristnosti in ne dovoliti uporabnikom, da namestijo programski žeton, ki ga identificira edinstvena serijska številka na več naprav. Namestitev žetona z isto serijsko številko na več napravah z različnimi časovnimi viri lahko povzroči napake pri preverjanju pristnosti na strežniku za preverjanje pristnosti. Primerna rešitev je uporaba vezave žetona in naprave, da uporabniku omogočite namestitev žetona samo na eni napravi (RSA, 2017b).

Sodelovati pa morajo tudi uporabniki, saj jih je potrebno poučiti, da ID naprave obravnavajo kot občutljive podatke in naj uporabijo varen komunikacijski kanal, da ga pošljejo skrbniku. Ko uporabnik pošlje skrbniku odgovarjajoč ID pametnega telefona, mora skrbnik uporabiti ločen, varen kanal za posredovanje informacij, ki jih potrebuje aplikacija programskega žetona, da se dokonča postopek povezovanja. Povezovalni ID (angl. binding ID), zagotavlja, da je distribuiran žeton namenjen določeni napravi in bi moral biti del celovite strategije vzdrževanja varnosti med postopkom distribucije. Med potekom distribucijskega procesa je potrebno zagotoviti, da je vsak element, potreben za povezovanje, vključen v ločene, varne komunikacije in ni združen v eno samo komunikacijo. Za CTF se priporoča uporabo gesla in da se CTF ne sporoči skupaj z povezovalnim ID-jem (RSA, 2017b).

Kot najboljšo varnostno prakso bi morali zahtevati PIN kot del pri avtentikaciji. Če se odločimo za uporabo žetonov brez PIN-a, se moramo prepričati, da sistem omogoča drugi faktor preverjanja pristnosti, geslo za Windows (RSA, 2017b).

Če sistem nima drugega faktorja, Windows gesla, in ga ni mogoče implementirati, je potrebno znova nastaviti PIN za žetone. Če ne moremo nastaviti vseh žetonov, redno preverjamo agente[7] na sistemih, ki ne potrebujejo drugega faktorja za preverjanje pristnosti (RSA, 2017b).

Ker je velika prednost imeti 2FA v podjetju in ne v oblaku, se lahko nastavijo še kako pomembne dodatne nastavitve, kot je postopek, da se uporabniku omogoči, da se avtenticira samo, ko uporabnik zahteva dostop do sistemov, ki uveljavljajo dodatni faktor za preverjanje pristnosti in postopek dovoljenja, da se uporabnik avtenticira samo, če je na vsakem sistemu, do katerega dostopa uporabnik, potreben drugi faktor za preverjanje pristnosti. Označi se lahko tudi skupine, ki vsebujejo uporabnike z žetonom Tokencode[8], da zagotovimo, da so te skupine omogočene le na agentih, ki ščitijo sisteme in potrebujejo drugi faktor za preverjanje pristnosti. Če uporabljamo žetone s Tokencode, je smotrno pregledovati dnevnike za naslednje dejavnosti, da zagotovimo pooblaščenim osebam pravilno izvajanje operacij overjanja (RSA, 2017b):

- ustvarjanje agentov;
- ustvarjanje in dodeljevanje skupine;

[7] agent – programska oprema, nameščena na strežniku, ki komunicira z aplikacijo za overjanje
[8] tokencode – od šest do osem mest dolga enkratna številka na programskem generatorju gesel

- spremembe članstva v skupini;
- prenos žetonov;
- omogočanje žetonov.

Kar se tiče **zaščite** pametnega telefona, je potrebno izobraziti uporabnike, da omogočijo PIN ali geslo za pametni telefon. Podjetja bi morala vzpostaviti pravilnike, ki zahtevajo uporabo PIN-a ali gesla naprave za dostop pri nameščanju programskih žetonov na pametne telefone. Ko so enkrat te varnostne funkcije omogočene, morajo uporabniki vnesti svoj PIN ali geslo za dostop do aplikacije programskega žetona, kar pa povečuje varnost (RSA, 2017b).

V veliki meri je najbolj pravilno, da se odločimo za sestavo enkratnega gesla (angl. passcode[9]) z uporabo PIN-a, faktorja, kaj vemo, ki je unikatna kombinacija številk in črk, dolžine vsaj štirih znakov, in TokenCode, šest do osemštevilčne kode, ki jo izpiše vsako minuto programski generator. Za popolnejšo zaščito je odlično upoštevati pravilno upravljanje žetonov in PIN-a z naslednjimi ukrepi (RSA, 2017b):

- vsi uporabniki morajo vedeti, da svoje PIN kode nikomur ne povedo. Skrbniki in osebje za pomoč uporabnikom ne smejo nikoli vedeti ali zahtevati PIN-a od uporabnika;
- če je mogoče, ne uporabljamo štirimestne kode PIN. Če moramo uporabiti kratek PIN (4-mestni PIN), zahtevajmo črkovno-številčne znake (a-z, A-Z, 0-9), če jih žeton podpira;
- konfigurirajmo PIN politiko, da zahteva, da uporabniki v rednih časovnih presledkih spreminjajo svoje PIN kode. Ti intervali naj ne presegajo 60 dni. Če uporabljamo štirimestne številčne kode PIN, interval ne sme preseči vsakih 30 dni;
- nastavimo pravilnike, ki omejujejo ponovno uporabo PIN-ov;
- konfigurirajmo uporabo slovarja, da preprečimo uporabo preprostih PIN-ov;
- če so žetoni za programsko opremo izdani kot žetoni v obliki PINPad[10], mora biti PIN enako dolg kot tokencode in številčen;
- kadar so žetoni za programsko opremo izdani kot žetoni fob-style[11], mora biti PIN alfa numeričen in dolžine osem znakov.

[9] passcode = PIN + tokencode
[10] PINPad – enkratno geslo se generira s PIN-om
[11] fobstyle – programski generator se obnaša kot strojni generator kod

Tabela 5: Prikaz, koliko ur je potrebnih za uspešen napad glede na dolžino kod

OTP nastavitve		čas v urah
PIN koda na generatorju	4-mestna številka 6-mestna številka	138
PIN koda na generatorju	6-mestna številka 6-mestna številka	13.859
PIN koda na generatorju	4-mestne črke 6-mestna številka	23.470
PIN koda na generatorju	6-mestne črke 6-mestna številka	30.419.781
PIN koda na generatorju	4-mestna številka 8-mestna številka	13.955
PIN koda na generatorju	4-mestne črke 8-mestna številka	2.373.371

Vir: Ackerman, Bowness, Brainard & Duane (2005).

Distribucija aplikacij in programske opreme se lahko razlikuje med različnimi platformami. V mnogih primerih je pametni telefon oziroma mobilna naprava v lasti uporabnika, zato lahko podjetje upravlja ali pa ga ne upravlja. Prepričati se moramo, da uporabniki pridobijo aplikacijsko programsko opremo za svojo napravo samo iz zaupnih virov. Pri tem je ključnega pomena, da produkti 2FA **ne podpirajo** nameščanja izdelkov programskega žetona na rooted[12] pametni telefon (RSA, 2017b).

Veliko pozornosti moramo nameniti izgubljenim mobilnim napravam ali prenosnim računalnikom, ki jih morajo uporabniki nemudoma prijaviti skrbniku storitve 2FA, saj vsebujejo programski žeton. Skrbnik mora preveriti identiteto uporabnika in zagotoviti, da je žeton onemogočen za uporabo, dokler ni najdena naprava, ali če je pridobljena nadomestna naprava in je opremljena z nadomestnim žetonom. Poleg tega je potrebno prositi uporabnika za informacije o tem, kdaj je bila naprava izgubljena ter zapisati datum in preverjati dnevnike za poskuse preverjanja pristnosti z izgubljenim žetonom (RSA, 2017b).

Če uporabnik zapusti podjetje, je potrebno čim prej izbrisati ali onemogočiti uporabo uporabniških računov. Z uporabo aplikacije 2FA, kjer so uporabniki shranjeni v notranji bazi podatkov aplikacije oziroma uporabniki, ki uporabljajo aktivni imenik AD, se lahko izbriše samo kot vir identitete, in sicer z uporabo lokalnega vmesnika za aktivni imenik. Če je uporabnik 2FA dlje časa odsoten, onemogočite uporabniški račun in omogočite račun, ko se uporabnik vrne na delo. Ko je onemogočen uporabniški račun, se prekine uporabnikovo dovoljenje za overjanje, ki prepoveduje dostop do zaščitenih virov (RSA, 2017b).

[12] rooted – imeti dostop kot super uporabnik

Uporabniki morajo biti poučeni o informacijah, ki jih lahko delijo s skrbniki 2FA sistema. Nikoli ne smejo razkriti serijske številke žetona nikomur, razen skrbniku sistema, vendar le v primeru, če ima težave z žetonom. Prepričani moramo biti, da so uporabniki seznanjeni z informacijami, ki jih skrbniki ne smejo zahtevati, vključno s PIN-on naprave ali geslom naprave, PIN-om, tokencode, passcode ali geslom za distribucijo žetonov. Z vsem zgoraj omenjenim pa se seveda želimo izogniti napadom socialnega inženiringa (RSA, 2017b).

Podpora uporabnikom zahteve posebno skrb, predvsem pa je potrebno vzpostaviti varne postopke pri dodeljevanju programskih žetonov. Prepričati se je potrebno, da operaterji med svojim postopkom dodeljevanja žetonov razumejo, da vključijo vsak element, ki je potreben za dodeljevanja v ločenih, varnih komunikacijah in ni kombiniran v eni komunikaciji. Prepričati se je potrebno, da operaterji razumejo pomen moči PIN-a in občutljivosti podatkov, kot so uporabniško ime za prijavo in serijska številka žetona. Za zmanjšanje tveganja za napade socialnega inženiringa je potrebno naučiti operaterje, da zahtevajo občutljive podatke le, kadar je to nujno potrebno ter da zahtevajo od uporabnikov najmanjšo količino informacij, potrebnih za vsako situacijo (RSA, 2017b).

Primerni so tudi nasveti za uporabnike, kjer jih je potrebno izobraziti z naslednjim (RSA, 2017b):

- nikoli ne povejte serijske številke žetona, PIN-a, tokencoda, žetona ali gesla drugemu uporabniku;
- da bi preprečili lažne napade, ne vnašajte tokencodes v povezave, ki jih lahko odprete iz e-pošte. Namesto tega vnesite URL pooblaščenega spletnega mesta, na katerem se avtenticirate;
- pred brskanjem po spletnem mestu, ki zahteva uporabo tokencode, popolnoma zaprite brskalnik in ga znova zaženite v čistem stanju;
- redno zaprite brskalnik in počistite predpomnilnik podatkov;
- ko končate delo, se vedno odjavite iz aplikacij;
- vedno zaklenite namizje, ko zapustite svoj delovni prostor;
- takoj prijavite izgubljene ali ukradene žetone.

Preprečevanje socialnih inženirskih napadov, kjer goljufi pogosto preslepijo nič hudega slutečega uporabnika, da bi razkril občutljive podatke, ki jih je mogoče uporabiti za dostop do zaščitenih sistemov, se da preprečiti s primernim izobraževanjem operaterjev 2FA v službi za pomoč uporabnikom in jim dati smernice za zmanjšanje verjetnosti uspešnega napada socialnega inženiringa, in sicer (RSA, 2017b):

- zahtevati samo uporabniško matično številko uporabnika v podjetju, ko uporabnik pokliče službo za pomoč. Nikoli ne prositi za kode, PIN, gesla in tako naprej;
- vsi uporabniki naj imajo telefonsko številko za pomoč uporabnikom;

- potrditev uporabniške identitete pred izvedbo opravil na žetonu uporabnika ali PIN-a. Na primer zastavite uporabniku vprašanja, na katerega zna pravilno odgovoriti za preverjanje njihove identitete:
- ko morate vzpostaviti stik z uporabnikom, ne zahtevajte nobenih podatkov o uporabniku. Namesto tega naročite uporabniku, da vas pokliče nazaj na telefonsko številko službe za pomoč uporabnikom;
- če želite potrditi spremembo PIN-a, da jo je naredil uporabnik, nastavite pravilnik obveščanja, da bo uporabnik obveščen o spremembi PIN-a. Na primer, pošljite e-poštno obvestilo na uporabniški e-poštni naslov podjetja ali pustite glasovno sporočilo. Uporabniki, ki sumijo, da je spremembo naredila nepooblaščena oseba, naj pokličejo službo za pomoč uporabnikom.

Da se izognemo napadu socialnega inženirstva, je nujna tudi potreba po potrditvi uporabniške identitete. Operaterji oziroma administratorji sistema 2FA v oddelku za pomoč uporabnikom morajo preveriti identiteto uporabnika preden izvedejo katerokoli operacijo, kot je dodeljevanja ključka, ponastavitev gesel, ponastavljanje PIN-a. Ukrepe za preveritev identitete naj vključujejo naslednje (RSA, 2017b):

- pokličejo uporabnika nazaj na telefon, ki je v lasti organizacije na številko, ki je shranjena v sistemu;
- pošljejo uporabniku e-pošto na naslov podjetja. Če je mogoče z uporabo šifrirane e-pošte;
- sodelovanje z vodjo zaposlenega, da preverijo identiteto uporabnika;
- osebno preverijo identiteto uporabnika.
- postavijo več odprtih vprašanja iz evidenc zaposlenih, na primer: »Imenujte eno osebo v svoji skupini«, »Kakšna je vaša matična številka v podjetju?«. Vendar pa ne postavljajte vprašanj, na katera je mogoče odgovoriti z da ali ne.

SKLEP

V raziskavi smo ugotovili, da lahko programe, ki ustvarjajo žetone za dvofaktorsko preverjanje pristnosti zlorabijo in jih klonirajo z zlonamerno programsko opremo. Napadalec s korenskim dostopom lahko kopira zaupne podatke žetona iz okuženih naprav in jih uporabi za pridobitev žetona žrtve. Vendar je to videti kot v filmu Mission: Impossible, kjer Ethan Hunt s svojimi sodelavci vdre v še tako varovan sistem. Pri tem moramo vedeti, da so napadi simulirani pri zvišanih privilegijih z rooted metodo in kot smo raziskali, je pri tem ključnega pomena, da produkti 2FA ne podpirajo nameščanja izdelkov programskega žetona na rooted pametni telefon. To pomeni, da je napadalec moral ukrasti pametni telefon, kar pa uporabnik opazi v današnjem času že v prvi sekundi. In če se upošteva primerna raba, morajo uporabniki nemudoma prijaviti odtujitev ali izgubo telefona skrbniku storitve, da se

onemogoči nadaljnjo uporabo programskega žetona. Poleg tega so programski žetoni dvofaktorski sistem za preverjanje pristnosti. Napadalec mora tudi ogroziti drugi faktor, običajno PIN ali geslo, da bi imel kakršenkoli vpliv na uporabnika. Najboljša praksa pri tem je uporaba žetona z dodatno kodo PIN ali geslom. Če vse to upoštevamo, mora napadalec priti do treh faktorjev v zelo kratkem času. In če bo poskušal ugibati PIN ali geslo, bo sistem samodejno povzročil, da se zaklene račun. Odgovorni za izbiro ustrezne avtentikacije na pametnih telefonih morajo pri produktih, kot prvo paziti na varen način distribucije programskih žetonov, zato ker lahko pride do kraje semena žetona že pred uporabo. Hkrati pa imeti v mislih, da se zagotovi faktor kaj imamo, kar pomeni vezati programske žetone samo na eno napravo, pametni telefon.

Skozi nalogo smo ugotovili, da so programski avtentikatorji na pametnih telefonih ob zavedanju, kaj pomeni varnost v informacijski dobi, s primerno uporabo varni. Z večjo varnostjo uporabe se vsekakor zmanjša tveganje in to dosežemo z dobrimi praksami pri načrtovanju, dodeljevanju in uporabi programskih žetonov. Po drugi strani pa se moramo zavedati, da je tveganje veliko večje, če ne uporabljamo za kritične sisteme vsaj minimalne dvofaktorske avtentikacije, kot je prejetje enkratnega gesla prek SMS sporočila.

LITERATURA IN VIRI

1. Acharya, S., Polawar, A. & Pawar, Y. P. (2013). Two Factor Authentication Using Smartphone Generated One Time Password. *IOSR Journal of Computer Engineering, 11*(2), 85-89.
2. Ackerman, K., Bowness, P., Brainard, J. & Duane, B. (2005). *Secure Offline Verification of One-Time Password.* Pridobljeno 5. decembra 2018 iz https://www.rsa.com/en-us/services/rsa-product-and-customer-support
3. Android. (2016). *Android security white paper.* Pridobljeno 10. julija 2017 iz https://static.googleusercontent.com/media/enterprise.google.com/en//android/static/files/android-for-work-security-white-paper.pdf
4. Android. (2017). *Android Security 2016 Year in Review.* Pridobljeno 10. julija 2017 iz https://source.android.com/security/reports/Google_Android_Security_2016_Report_Final.pdf
5. Be'ry, T. (2014). *Smart Card Logon: The Good, the Bad and the Ugly*. Pridobljeno 16. avgusta iz http://www.infosecisland.com/blogview/23657-Smart-Card-Logon-The-Good-the-Bad-and-the-Ugly.html
6. Bitstamp. (2015). *Two factor authentication.* Pridobljeno 29. junija 2017 iz https://www.bitstamp.net/s/documents/bitstamp_2_factor_authentication_guide.pdf
7. Blanco-Gonzalo, R. & Sachez- Reillo, R. (2014). *Biometric on Mobile Devices.* Pridobljeno 30. junija 2017 iz

https://link.springer.com/referenceworkentry/10.1007/978-3-642-27733-7_9228-2#page-1
8. Certic. S. (2013). *The Future of Mobile Security.* Pridobljeno 4. julija 2017 iz https://arxiv.org/ftp/arxiv/papers/1302/1302.4201.pdf
9. Developer Android. (2017a). *Android Emulator*. Pridobljeno 16. avgusta 2018 iz https://developer.android.com/studio/run/emulator
10. Developer Android. (2017b). *ABD.* Pridobljeno 16. avgusta 2018 iz https://developer.android.com/studio/command-line/adb
11. Dmitrienko, A., Liebchen, C., Rossow, C. & Sadeghi, A. R. (2014). One the (In) Security analysis of mobile two-factor Authentication Schemes. *Intel Technology Journal, 18*(4), 138-156.
12. DSwiss Ltd. (2010). *2-factor Authentication form obile applciations: Introducing DoubleSec.* Pridobljeno 14. julija 2017 iz https://pd.zhaw.ch/publikation/upload/206119.pdf
13. Du, Y., Wang, Y., Wang, J. (2015). A static Android malicious code detection method based on multi-source fusion. *Security and Communication Networks, 8*(17), (str. 3238–3246).
14. Eclipse. (2017). *Eclipse.* Pridobljeno 16. avgusta 2018 iz https://www.eclipse.org/
15. EMC. (2014). *Taking control of the digital and mobile user authentication challange.* Pridobljeno 12. junija 2017 iz https://www.emc.com/collateral/white-paper/taking-control-mobile-user-authentication-challenge.pdf
16. Evolver. (2018). *Case study: Multi-factor authentication using RSA SecurID access*. Pridobljeno 5. decembra 2018 iz https://evolverinc.com/case-study-multi-factor-authentication-using-rsa-securid/#respond
17. Fenton, A. (2016). *Five Most Common Security Attacks on Two-Factor Authentication.* Pridobljeno 16. avgusta 2018 iz https://www.itbusinessedge.com/slideshows/five-most-common-security-attacks-on-two-factor-authentication.html
18. Github. (2017). *Dex2jar.* Pridobljeno 16. avgusta 2018 iz https://github.com/pxb1988/dex2jar
19. Grimes, R. A. (2018). *11 ways to hack 2FA.* Pridobljeno 16. avgusta 2018 iz https://www.csoonline.com/article/3272425/authentication/11-ways-to-hack-2fa.html?nsdr=true
20. GSMA. (2016). *Mobile Connect: Mobile high-security authentication.* Pridobljeno 14. julija 2017 iz https://www.gsma.com/identity/wp-content/uploads/2016/10/MC_high-security-authentication_Sep-16.pdf
21. Gupta, S. (2012). *Strong Authentication for Physical Access using Mobile Devices.* Pridobljeno 4. julija 2017 iz http://www.electrosoft-inc.com/sites/default/files/2017-03/Strong%20Authentication%20for%20Physical%20Access%20using%20Mobile%20Devices%202012.pdf

22. Gupta, N. & Rani, R. (2015). *Implementing High Grade Security in Cloud Application using Multifactor Authentication and Cryptography.* Pridobljeno 5. septembra 2017 iz http://airccse.org/journal/ijwest/papers/6215ijwest02.pdf
23. Hell, M. & Altman, A. (2015). *Two-factor Authentication in Smartphones: Implementations and Attacks.* Pridobljeno 28. avgusta 2017 iz http://lup.lub.lu.se/luur/download?func=downloadFile&recordOId=7792889&fileOId=7792890
24. Hypersecu. (2014). *Hard vs. Soft Tokens.* Pridobljeno 4. julija 2017 iz https://hypersecu.com/downloads/files/whitepapers/HSTE-NB0012-RV1.0-Hard-Soft-Tokens.pdf
25. Ibotpeaches. (2017). *Apktool.* Pridobljeno 16. avgusta 2017 iz https://ibotpeaches.github.io/Apktool/
26. IDC. (2017). *Smartphone OS Market Share*. Pridobljeno 12. junija 2017 iz http://www.idc.com/promo/smartphone-market-share/os
27. Java Decompiler. (2017). *JD-GUI.* Pridobljeno 16. avgusta 2018 iz http://jd.benow.ca/
28. Karanjgaokar, R. (2016). *Demystifying PKI technology based two factor authentication.* Pridobljeno 16. avgusta 2018 iz https://www.computerweekly.com/tip/Demystifying-PKI-technology-based-two-factor-authentication
29. Kaspersky Lab. (2016). *Kaspersky security bulletin 2016.* Pridobljeno 12. junija 2017 iz https://kasperskycontenthub.com/securelist/files/2016/12/KASPERSKY_SECURITY_BULLETIN_2016.pdf
30. Lexle. (2014). *A comparison of two-factor authentication methods: which is best for you?* Pridobljeno 28. avgusta 2017 iz https://www.expressvpn.com/blog/best-two-factor-authentication
31. Mueller, B. (2016). *Hacking Soft Tokens.* Pridobljeno 16. avgusta 2018 iz https://gsec.hitb.org/materials/sg2016/whitepapers/Hacking%20Soft%20Tokens%20-%20Bernhard%20Mueller.pdf
32. Mulliner, J. (2018). *Best Practices for Switching from Hardware to Software Tokens.* Pridobljeno 5. decembra 2018 iz https://blog.vasco.com/authentication/hardware-to-software-token/
33. NIST. (2017). *Electronic Authentication Guideline.* Pridobljeno 28. junija 2017 iz http://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-63-2.pdf
34. Nokia. (2017). *Nokia malware report reveals new all-time high in mobile device infections and major IoT device security vulnerabilities.* Pridobljeno 12. junija 2017 iz https://www.nokia.com/en_int/news/releases/2017/03/27/nokia-malware-report-reveals-new-all-time-high-in-mobile-device-infections-and-major-iot-device-security-vulnerabilities
35. Oracle. (2017a). *Keytool.* Pridobljeno 16. avgusta 2018 iz https://docs.oracle.com/javase/6/docs/technotes/tools/solaris/keytool.html

36. Oracle. (2017b). *Jarsigner.* Pridobljeno 16. avgusta 2018 iz https://docs.oracle.com/javase/7/docs/technotes/tools/windows/jarsigner.html
37. Ross, A. & Jain, A. K. (2009). *Biometric, Overview.* Pridobljeno 10. julija 2017 iz https://link.springer.com/referenceworkentry/10.1007/978-0-387-73003-5_182
38. RSA. (2017a). *Two-factor authentication: RSA Securid® software tokens.* Pridobljeno 12. junija 2017 iz https://www.rsa.com/en-us/products/rsa-securid-suite/securid-software-tokens
39. RSA. (2017b). *RSA SecurID® Software Token Security Best Practices Guide for RSA Authentication Manager 8.x.* Pridobljeno 22. novembra 2018 iz https://community.rsa.com/docs/DOC-35128
40. Saini, T. (2014). One Time Password Generator System. *International Journal of Advanced Research in Computer Science and Software Engineering, 4*(3).
41. Schaad. J. (2005). *Internet X.509 Public Key Infrastructure.* Pridobljeno 16. avgusta 2018 iz http://www.ietf.org/rfc/rfc4211.txt
42. Sharad, R. D. & Patil, B. M. (2016). Effective Risk Analysis AND Risk Detection for Apps. *International Jourlan of Computer Applications, 147*(6), (str. 29–32).
43. Singh, P., Tiwari, P. & Singh, S. (2016). Analysis of Malicious Behavior of Android Apps. *Procedia Computer Science, 79,* 215–220e.
44. Soare, A. C. (2012). Internet Banking Two-Factor Authentication using Smartphones. Journal of Mobile, *Embedded and Distributed System,*. ISSN 2067-4074.
45. Strom, D. (2014). *Comparing the top multifactor authentication vendors.* Pridobljeno 17. oktobra 2018 iz https://searchsecurity.techtarget.com/feature/The-fundamentals-of-MFA-Comparing-the-top-multifactor-authentication-products
46. TechTarget. (2016). *Two-factor authentication (2FA).* Pridobljeno 12. junija 2017 iz http://searchsecurity.techtarget.com/definition/two-factor-authentication
47. The Open Group. (2017). *Strings.* Pridobljeno 16. avgusta 2018 iz http://pubs.opengroup.org/onlinepubs/9699919799/utilities/strings.html
48. Trewin, S., Swart, C., Koved, L., Martino, J., Singh, K. & Shay, B-D. (2012). *Biometric Authentication on a Mobile Devices: A Study of User Effort, Error and Task Disruption.* Pridobljeno 28. junija 2017 iz http://researcher.ibm.com/researcher/files/us-kapil/ACSAC12.pdf
49. VarniNaInternetu. (2017). *Varno v svet kripto valut.* Pridobljeno 5. decembra 2018 iz https://www.varninainternetu.si/varno-v-svet-kripto-valut/
50. VirusTotal. (2017). *How it works.* Pridobljeno 16. avgusta 2018 iz https://support.virustotal.com/hc/en-us/articles/115002126889-How-it-works
51. Willis, N. (2014). *FreeOTP.* Pridobljeno 16. avgusta 2018 iz https://lwn.net/Articles/581086/
52. Wireshark. (2017). *Wireshark.* Pridobljeno 16. avgusta 2018 iz https://www.wireshark.org/

53. Zink, T. & Waldvogel, M. (2017). *X.509 User Certificate-based Two-Factor Authentication For Web Applications.* Pridobljeno 16. avgusta 2018 iz https://pdfs.semanticscholar.org/ed87/5b008f6c7272dc6fdcc815778d31dc8b6022.pdf