DOCUMENT Artikel RPA en security
DATE 17.07.2017
DISTRIBUTION OK
PETER RIETVELD Peter.Rietveld@Traxion.com
JAN KOOT Jan.Koot@Traxion.com
MELVIS HADZIC Melvis@Hadzic.com

Identity Management in het tijdperk van de robots

In de wereld van cyberinsecurity voelt Identity en Access Management (IAM) vaak als een reis terug in de tijd. Gericht op de interne bedreiging en bezig met de interne populatie. De tijd dat alles nog simpel en veilig was. In 2009 leek IAM klaar: standaardproducten, gevestigde namen en een volwassen aanpak. Kortom; een commodity. Een tikkeltje saai, zeg maar. Nou, daar klopt anno 2017 helemaal niks meer van. IAM is in een stroomversnelling terechtgekomen die alle zekerheden heeft weggevaagd.

Vroeger was het simpel: IAM beveiligde de voordeur en hackers kwamen door een raam of een achterdeur. De voordeur was voor het personeel en IAM regelde dat medewerkers een sleutel van de voordeur kregen – een gebruikersnaam en een wachtwoord. IAM-systemen, en dan vooral de provisioningsystemen zelf, zijn zelden hackerbestendig. Ze staan immers in het interne netwerk, in een veilige zone.

IAM in de frontlinie
Nu komen de hackers aan de voordeur rammelen. Volgens het Verizon DBIR 2017 is er bij 81% van alle daadwerkelijke breaches sprake van gekaapte accounts. Blijkbaar zijn die achterdeuren door de mensen van cybersecurity gefixt of zodanig aangepakt dat IAM nu de zwakste schakel is. Dat wachtwoord is dus niet meer voldoende nu we echt hackers buiten moeten houden. Daarom zijn we massaal met Multifactor Authenticatie bezig en met het extra beveiligen van de accounts van de beheerders. Zelfs Privileged Access Management staat overal op de agenda, want die hackers willen natuurlijk vooral de credentials van de beheerders. Ook dit kunnen we nog wel aan. Maar het begint wel echt krap te worden op die arbeidsmarkt.

En dat is nog niet alles. De dogma's leren ons dat IAM preventieve beveiliging is: het voorkomen van ongelukken. Dat heeft geleid tot een afstandelijke houding tot wat IAM als hardcore security beschouwt: optreden bij breaches. Dat weten die anderen maar al te goed: IAM gaat gewoon op tijd naar huis.

Maar als het doel van de aanvaller het domain administrator wachtwoord is – of de hash daarvan – dan is IAM de partij die dit wachtwoord van alle administrators kan resetten. Als een hack wat langer geduurd heeft, is dat een veilige aanname. Zoals Notpetya liet zien, is Mimikatz met een paar muisklikken erin gezet. Mimikatz vangt niet alleen de wachtwoorden van de domain administrators, maar ook van alles wat ooit inlogt; inclusief die van applicaties en devices. Dan krijg je de vraag of IAM die ook kan resetten.

Bij dit soort aanvallen moet je aannemen dat de root certificaten van de Active Directory en de web authenticatiesystemen gekopieerd zijn. Vervangen dus. Als de aanvaller ietwat slimmer is, moet je ook alle applicatieaccounts, SSH keys en de API's resetten – die je meestal helemaal niet in scope hebt. Oftewel, IAM is onmisbaar in incident response en daar niet klaar voor.

In de praktijk moet IAM de capabilities dus nog ontwikkelen, als in een soort IAM Resilience. In 2017 zal IAM moeten blijven tot de hack gestopt is. Hier moet een culturele kloof overbrugd worden voordat we het gapende gat in de beveiliging kunnen dichten. Want we willen toch graag alleen op kantoortijden werken, nietwaar?

Dit is nog maar het begin. We hebben ook nog een leuk aapje dat op onze schouder is gaan zitten.

DOCUMENT Artikel RPA en security
DATE 17.07.2017
DISTRIBUTION OK
PETER RIETVELD Peter.Rietveld@Traxion.com
JAN KOOT Jan.Koot@Traxion.com
MELVIS HADZIC Melvis@Hadzic.com

Non-Person Accounts

Van oudsher is Identity Management een extensie om HRM-gegevens te vertalen naar IT-voorzieningen. Het is procesautomatisering van uitgifte van accounts en rechten aan wat tegenwoordig de workforce heet. Workforce, omdat de externen en de leveranciers zo onderhand wel geland zijn, al dan niet via een federatieve koppeling. Maar nu zien we een explosie van accounts waar geen mens achter zit: Non-Person Accounts (NPA).

NPA is een probleem geworden met de opmars van API's, en dan in het bijzonder vanwege REST. REST is de razend populaire opvolger van SOAP – lichtgewicht applicaties die lekker snel in elkaar geschroefd kunnen worden. Voor een deel komt die snelheid doordat REST geen enkele ingebouwde beveiliging heeft. Geen gedoe dus zoals bij SOAP, waar je door topzware standaarden als WS-Security en WS-Trust moet ploeteren.

REST gebruikt http en is daarmee aangewezen op de standaard login forms of basic authentication om met een andere machine te mogen praten. Dat betekent een interface waar mensen ook gewoon bij kunnen komen.

Opletten met de term NPA
Er zijn creatieve mensen die er Non-Personal van maken en dat zijn dan gedeelde accounts die door hele afdelingen gebruikt kunnen worden. Dan willen ze ook de wachtwoordpolicy eraf, zodat het wachtwoord jaren hetzelfde blijft en je een reset niet hoeft te melden bij alle collega's. Als je thuiswerkt, kun je immers niet op het prikbord kijken. Dit is niet waar NPA over gaat. Het is wél heel erg fout.

Natuurlijk zou je de authenticatievraag voor een REST-interface op kunnen lossen met dual-side TLS en dus met certificaten, maar die dingen verlopen en dat is een vreselijk gedoe. Daarbij zal niet iedere client een veilige opslag van een geïnstalleerd certificaat bieden – veel meer dan een plekje op de schijf is het meestal niet. Dat wordt hem dus niet zonder meer: het is een stukje van de puzzel, maar niet het hele plaatje.

Bovendien kunnen certificaten de authenticatie niet vervangen, alleen versterken. Er blijft sprake van een account waarop aangelogd wordt. En daar zitten eisen aan, zoals expiry. REST is echter niet in staat wachtwoorden te veranderen, dus ook hier moet die wachtwoordpolicy eraf – of de programmeur moet dat zelf gaan bakken en dan is REST opeens niet simpel en goedkoop meer.

Ook moet die API dat wachtwoord ergens opslaan, en het is gebruikelijk dat dan maar ergens on disk in een script te doen; de gevreesde hardcoded oplossing. Nu kunnen sommige PAM-systemen daar iets aan doen, maar dat is nieuw en bewerkelijk - de meeste organisaties hebben nog maar net de folder gelezen. Nog een stukje van de puzzel.

Ja maar, er zijn toch ook allerlei 'API Keys'-oplossingen waarbij je met een secret key inlogt? Vanuit authenticatie bezien, zijn dat langere wachtwoorden: een stuk lastiger te bruteforcen, dus beter, maar zonder roll-over niet goed. Ze staan ergens op de applicatieserver en zijn dus goed te kopiëren en te misbruiken. Bruikbaar puzzelstukje, maar wederom niet hele plaatje.

Oauth wordt ook nog als oplossing gepresenteerd. Het lastige is dat Oauth geen authenticatie doet, maar pas daarna komt. Sterke authenticatie dan? Een computer heeft geen vingers, dus een sterke authenticatie kan meestal niet op een API – TOTP via een ander apparaat of biometrie is wat we zouden kunnen, maar dus niet voor Non-Person Accounts. Wellicht is er iets te doen met omgekeerde behavioral biometrie – zeg maar een omgekeerde turingtest. Als een API opeens typt als een mens weet je zeker dat er iets aan de hand is. Mogelijk nog een stukje van de oplossing.

Is API wel in scope?

Moet IAM de puzzel van API-accounts leggen? API's worden door applicaties gebruikt, dus wordt dit vaak nog buiten IAM om geregeld. Het gaat immers over accounts, niet over identiteiten. Vanuit het thema Moeilijke Dingen Buiten De Deur Houden is dit correct

PETER RIETVELD Peter.Rietveld@Traxion.com
JAN KOOT Jan.Koot@Traxion.com
MELVIS HADZIC Melvis@Hadzic.com

geredeneerd. Maar feitelijk is dit niet te verantwoorden, want de accounts leven in de door IAM gemanagede account stores. Oftewel: in de praktijk zijn het dan unmanaged accounts en daar weet de gemiddelde bad guy wel raad mee. Als IAM serieus met security bezig wil zijn, is er dus maar één mogelijk antwoord op de vraag of dit ook IAM is: ja.

Aanpakken dus. En snel een beetje. Want de opmars van de 'Things' is begonnen.

Op dit podium is de laatste drie jaar al het nodige gepubliceerd over IOT en de gebrekkige security ervan. De meeste zorgen gaan over het kapen of misbruiken van de Things om iets anders aan te vallen. Zwaar onderbelicht is dat als Things interacteren met andere apparaten, de cloud of bedrijfssystemen, ze daarop zullen inloggen. Kortom, ze hebben accounts nodig. Met een knipoog noemen we deze puzzel de Identity Of Things: IDOT.

Lessen van een hype
Vorig jaar is de DAO van Ethereum blockchain gehackt. De DAO was een digital decentralized autonomous organization, oftewel een volledig geautomatiseerd (ook wel gerobotiseerd) venture capital fonds. De DAO was snel een doorslaand succes, en haalde binnen enkele weken $168 miljoen op via crowdfunding. Binnen twee maanden was de DAO gehackt waarbij 3,6 miljoen Ether ontvreemd werd - $80 miljoen tegen de koers van dat moment. Daarna verdween de DAO en hadden de investeerders het nakijken. De software bleek de meest basale beveiligings-fouten te bevatten, en was zo gebouwd dat patching onmogelijk was. Er was geen echt softwareontwerp, er was geen security test geweest – en het was open source. De aanval volgde twee weken nadat een onderzoeker op de gaten had gewezen. De hype verdrong het gezond verstand dat zegt dat er geen veilige software bestaat.

Vanuit IAM-perspectief komen in IDOT alle traditionele kwesties voorbij: wie mag een IDOT-account aanmaken, wat mag een Thing op je systeem, hoe doen we de account lifecycle, en – het moeilijkste – hoe weet je wel zeker of je het juiste Thing voor je hebt: de authenticatie.

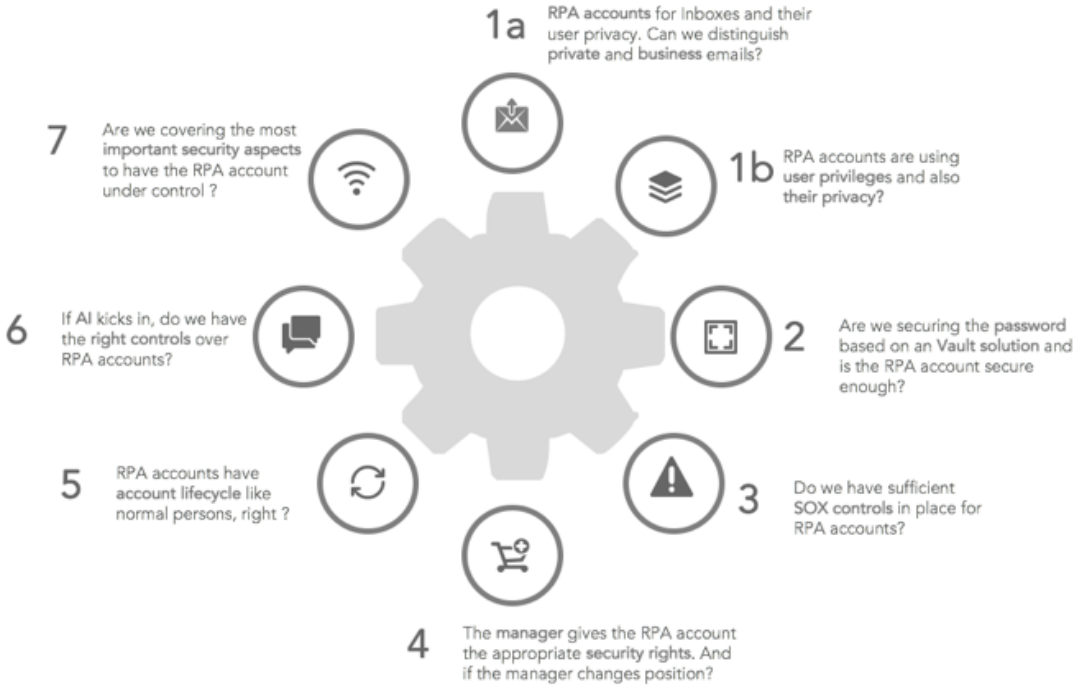
Net als bij een API is het een computer die inlogt en waarvan je zeker wilt weten dat het de goede is. Ook hier komen we niet veel verder dan username/password: de oplossing die we voor gewone gebruikers te zwak vinden. Mochten we bij IDOT iets met certificaten willen, dan lopen we heel hard tegen de muur van revocation en enrollment aan: hoe krijg je een goedkoop apparaat zo gek om nieuwerwetse protocollen als CMC/CMS of EST te spreken zodat we certificaten geautomatiseerd kunnen vervangen? Gegeven de stand der techniek moeten we vaststellen dat we óf de IOT-revolutie moeten vertragen, óf met z'n allen nog wat extra uurtjes per dag moeten klokken. Zo niet, dan weten we wat rond 2020 de eerste plaats in de DBIR krijgt.

De klap op de vuurpijl: RPA
Dat van die eerste plaats, is niet zeker. Als hacker heb je namelijk een nieuw doelwit dat nog veel aantrekkelijker is: RPA. RPA staat voor Robotic Process Automation. Het is een hype in vooral de financiële wereld, waar het streven de hoofdkantoren leeg te automatiseren nu de kantoormedewerkers heeft bereikt. Bij RPA vervangt een softwarerobot de administratieve kracht voor de meeste repetitieve taken: mail archiveren, orders inboeken, klanten te woord staan, eerstelijns helpdesk, dat soort werk. Het is een hype omdat er grote besparingen te realiseren zijn[1].

"In truth most of the automation today is not yet about intelligence, but what you might term 'automated stupidity' – it's the dull and mindless work that's been automated so that humans are freed up to leverage their intelligence.[2]"

[1] http://www.cio.com/article/3019287/robotics/how-robotic-process-automation-threatens-workers-today.html
[2] https://www.ssonetwork.com/robotic-process-automation/articles/how-to-manage-risk-and-ensure-control-%E2%80%93-what-to

PETER RIETVELD Peter.Rietveld@Traxion.com
JAN KOOT Jan.Koot@Traxion.com
MELVIS HADZIC Melvis@Hadzic.com



Figuur 1: IAM en RPA

RPA Sales vertelt ons dat RPA 100% foutloos werk levert, 24 uur per dag en ook nog veel veiliger is – omdat er geen mens zit die misleid kan worden door phishing of social engineering. "Robots lack curiosity (e.g., won't open phishing emails), cannot be tricked into divulging information or downloading unauthorized software, and have no motives which might tempt them to violate existing policies[3]". Nu is de mens inderdaad te beïnvloeden – maar computers net zozeer, alleen op een iets andere manier. Drive-by installers en SMB exploits breken in computers in, of er nu een mens achter zit of niet: als er onfeilbare code bestond, zouden er geen hacks en virussen zijn.

De eerste generatie RPA die nu her en der met grote urgentie wordt uitgerold is een technologie in de kinderschoenen en de beveiliging is navenant. Dat vraagt enige uitleg. RPA doet wat een gebruiker doet, oftewel er staat een Windows computer waarmee de robot op andere applicaties maar ook op shares en in Excelsheets werkt. RPA kan overweg met ongestructureerde data; dat maakt het – voor de business – van grote waarde. Voor een RPA installeer je software op een werkstation die op de active directory en allerlei applicaties, cloud of on premise, inlogt.

Als iets inlogt, dan is IAM betrokken. Laten we bij het begin beginnen. Een robot authentiseert zich tegen alle systemen waar hij bij moet. Dat zullen dezelfde systemen zijn als waar de gewone gebruiker inlogt. Net als een mens gebruikt de robot een username en een wachtwoord. Meer niet. Een robot 'doet' geen biometrie en zal ook niet met zijn smartphone even kijken op de Google Authenticator voor de TOTP-code. Oftewel: voor alle systemen waar RPA bij moet kunnen, kan je geen sterke authenticatie afdwingen. Nu kun je natuurlijk de accounts van de robots van sterke authenticatie vrijstellen, maar dan zet je een deur open die je nu net aan het sluiten was. Voor alle systemen waar de RPA op inlogt, zal de beveiliging zwakker worden. Daar zitten als je pech hebt ook systemen bij waar PCI-DSS sterke authenticatie voorschrijft.

[3] https://inform.tmforum.org/research-reports/robotic-process-automation-rise-machines/

PETER RIETVELD Peter.Rietveld@Traxion.com
JAN KOOT Jan.Koot@Traxion.com
MELVIS HADZIC Melvis@Hadzic.com

De robot van de RPA die de mens nadoet, moet ook beschikken over een goede error handling. Een robot zal niet de helpdesk bellen of een KBA ('wat is het merk van je eerste auto') invullen. Oftewel het user account zal geblokkeerd raken en blijven. Een robot loopt snel vast in beveiligingssystemen die gemaakt zijn voor de mens. Als er zaken veranderen, zoals de inlogprocedure, kan een mens lezen en snappen wat er staat, of de helpdesk bellen als het er verdacht uitziet. De huidige generatie robots (alles is gescript) zal echter stuk gaan bij de minste verandering, want dat kunnen ze niet aan. RPA zal een reden zijn om zo min mogelijk te veranderen.

Een laatste woord over het misleiden van robots. Ja, ze zullen aan de Microsoftmedewerker aan de telefoon wellicht geen toestemming geven om de sessie over te nemen. Maar als de RPA mail moet verwerken en de inhoud op de juiste plekken moet parkeren, dan zal de RPA de bestanden moeten lezen. Dit betekent dat ze parsers voor van alles en nog wat hebben. De robot zal dus de PDF of DOCX inlezen die hij ontvangt. Als dat uit de mail is, dan moge duidelijk zijn dat de verificatie van de afzender zwak blijft en als het attachment een aanval bevat, deze net zo goed werkt als wanneer een mens dat attachment opent.

Op een hoger abstractieniveau hebben we een nog veel groter beveiligingsprobleem. We weten wellicht hoe één robot werkt, maar we weten niet hoe een combinatie van robots werkt. Denk aan het Black Swan-syndroom. Dit kunnen we met enige regelmaat zien bij de zogeheten Flash Crash op de beurs of bij Ethereum[4]: de robots zijn geprogrammeerd voor stop loss orders – als de beurskoers daalt worden aandelen snel van de hand gedaan om de verliezen te beperken. Als de robots niet op enig moment stoppen omdat de koers belachelijk laag is geworden, dan stort de hele boel volledig in. Binnen enkele minuten daalde de koers van $320 naar $0,10 en de schade bij de robots liep in de miljoenen. Menselijke kopers reageerden razendsnel toen de koers de bodem raakte – en verdienden miljoenen, tot de koers weer terug was op het oorspronkelijk peil.

RPA en ontwikkelen
Er wordt hard geroepen dat RPA makkelijk te implementeren is en dat je er snel geld mee bespaart. Maar neem een willekeurig project en je ziet dat:
1. De applicatie moet om kunnen gaan met onverwachte veranderingen en crashes, makkelijk te patchen zijn en het moet simpel zijn om problemen op te lossen.
2. De applicatie moet zowel met business logische fouten als technische fouten om kunnen gaan.
3. De applicatie moet goed beveiligd zijn door middel van encryptie van data.
4. De applicatie mag niet meer rechten hebben dan nodig.
5. De applicatie moet voldoende logging geven om aanvallen en problemen te detecteren, en om statistieken uit de logging te halen.
Dan roept ineens iedereen: dit is een vreselijk complex project.

Bovenstaande punten moeten opgelost worden voor elk RPA-project.

Nu kun je zeggen dat de Flash Crash geen cybersecurity is – er komt immers geen hacker aan te pas – maar de schade door ondoordachte robots is wel heel erg echt.

Samenvattend: met alle nieuwe ontwikkelingen is Identity en Access Management toe aan een harde reset. IAM gaat over alles wat inlogt, en is centraal onderdeel van iedere beveiligingsinspanning.

Peter Rietveld is senior security consultant bij Traxion. Hij werkt in de IT-security sinds midden jaren negentig en is op dit moment ingezet bij Ahold Delhaize als domain architect IAM.

Jan Koot is security engineer bij Traxion. Hij is vooral actief op gebied van penetration testing en security verbeteringen bij kleine en grote bedrijven.

Melvis Hadzic is 15 jaar actief in de wereld van IAM, waarvan de laatste 10 jaar zelfstandig opererend in diverse rollen, van architectuur tot projectmanagement. Hij werkt momenteel bij Ahold Delhaize als architect op het programma IAM.

[4] http://fortune.com/2017/06/22/ethereum-crash/