Interne Netzwerke selbst auditieren

Mit den richtigen Werkzeugen lässt sich im eigenen Netzwerk ein Selbstaudit durchführen. So kann man im Active Directory, bei Passwörtern oder Netzwerkfreigaben mit wenig Aufwand Sicherheitsrisiken aufspüren und sie anschließend beheben.

Von Georg Bube

Bevor Administratoren oder Sicherheitsverantwortliche sich daranmachen, das interne Netzwerk auf eigene Faust zu untersuchen, sollten sie sich vergewissern, dass ein Offline-Backup aller relevanten Daten existiert. Zu einem Backup gehören auch Wiederherstellungstests, um sicherzustellen, dass das Wiedereinspielen im Fall der Fälle funktioniert. Denn der Verlust von Daten ist für Unternehmen ein Desaster und meistens mit hohem wirtschaftlichen Schaden verbunden – egal, ob es sich nun um eine Folge eines Angriffs oder eine unerwünschte Nebenwirkung eines Sicherheitstests handelt.

Beim Hacken des eigenen Netzwerks zum Zweck eines Selbstaudits steht man früher oder später vor der Entscheidung, ob man ein Tool oder Code aus dem Internet verwenden möchte oder nicht. Bei der Quelle Internet sollte man generell misstrauisch sein.

iX-TRACT

- Die meisten Unternehmen nutzen Windows und damit auch Active Directory. Bei dem Verzeichnisdienst können sich leicht Fehlkonfigurationen einschleichen, die die Sicherheit des Netzwerks gefährden. Auch schwache Passwörter erleichtern Angreifern das Leben.
- Mit teils automatisierten Werkzeugen lassen sich mit wenig Aufwand einige dieser Fehlkonfigurationen erkennen und anschließend beheben oder schwache Passwörter aufspüren.
- Der Artikel stellt einige Tools vor, mit denen sich sicherheitsrelevante Quick Wins in der eigenen Umgebung identifizieren lassen. So können die Verantwortlichen das Sicherheitsniveau des Netzwerks spürbar erhöhen.

Vertrauen ist gut, Kontrolle ist besser

Vor dem Einsatz von Tools oder Code aus dem Internet ist zu prüfen, wie vertrauenswürdig sie sind – und man sollte sich nicht darauf verlassen, ob andere Benutzer eine Quelle als verlässlich erachten. Falls der Quelltext verfügbar ist, kann man ihn gegebenenfalls prüfen. Das bedeutet je nach dessen Umfang einiges an investierter Zeit, ist jedoch bedeutend weniger zeitintensiv und nervenaufreibend als ein kompromittiertes internes Netzwerk.

Code aus dem Internet bedarf häufig einiger Anpassungen auf Nutzerseite, beispielsweise bei IP-Adressen, Ports und Shellcode. Sofern Base64-codierte Zeichenketten im Code enthalten sind, sollte der encodierte Inhalt überprüft werden, beispielsweise mit dem Onlinewerkzeug CyberChef (dieses und alle weiteren im Artikel angesprochenen Werkzeuge und Beiträge sind über ix.de/z7qk zu finden). Ist Shellcode enthalten, sollte man eine eigene Payload generieren und codieren.

Troubleshooting – Tools zum Laufen bringen

Sofern Windows Defender in der AD-Umgebung aktiv ist und eine Ausführungsrichtlinie die Skriptausführung in PowerShell einschränkt, kann es beim Ausführen der vorgestellten Tools zu Schwierigkeiten kommen. Um Probleme zu vermeiden, helfen die nachfolgend beschriebenen Maßnahmen. Sie sollten jedoch nur in Absprache mit den zuständigen Administratoren umgesetzt werden – was ebenso für den Einsatz der Tools gilt.

Ist Windows Defender aktiv, kann eine Ausnahme unter „Start/Einstellungen/Update & Sicherheit/Windows-Sicherheit/Viren- und Bedrohungsschutz“ definiert werden. Dazu auf „Einstellungen verwalten“ klicken und unter „Ausschlüsse/Ausschlüsse hinzufügen oder entfernen“ per „Ausschluss hinzufügen“ eine Ausnahme für eine Datei, einen Ordner, einen Dateityp oder einen Prozess hinzufügen.

Alternativ geht das auch per Gruppenrichtlinie (Group Policy Object, GPO): Computerkonfiguration/Richtlinien/Administrative Vorlagen/Windows-Komponenten/Microsoft Defender Antivirus/

iX-Workshop: Sich selbst hacken – Pentesting mit Open-Source-Werkzeugen

In diesem von Tim Mittermeier abgehaltenen iX-Workshop lernen die Teilnehmenden, wie sie Angriffsvektoren in ihrer eigenen Unternehmens-IT durch die Anwendung von Hacking-Techniken aufdecken und beseitigen können. Im Mittelpunkt stehen Themen wie das Sammeln und Auswerten öffentlich verfügbarer Informationen (OSINT), die Untersuchung auf Netzwerkebene und die Überprüfung von Webanwendungen mit Open-Source- und Auditing-Tools sowie die Privilegieneskalation unter Windows und Linux. Ein besonderer Schwerpunkt liegt auf dem zentralen Verzeichnisdienst Active Directory.

Der Workshop findet online statt, weitere Informationen gibt es unter https://heise.de/s/g1E2.

Listing 1: Auflistung der Ausführungsrichtlinien (Execution Policies)

```
PS > Get-ExecutionPolicy -List
        Scope ExecutionPolicy
        ----- ---------------
MachinePolicy       AllSigned
   UserPolicy       Undefined
      Process       Undefined
  CurrentUser       Undefined
 LocalMachine    Unrestricted
```

Ausschlüsse (Pfad beim Editieren einer Gruppenrichtlinie). In verwalteten Umgebungen kann es vorkommen, dass Systeme über einen externen Schutz durch die Verwaltungssoftware verfügen (Tamper Protection). Dadurch wird die Manipulation gewisser Windows-Defender-Einstellungen verhindert. In einem solchen Fall bleibt nur, die zuständigen Administratoren zu kontaktieren.

Die aktuell angewandte Ausführungsrichtlinie (Execution Policy) lässt sich in PowerShell per `Get-ExecutionPolicy` abfragen (siehe Listing 1).

Die Scopes `MachinePolicy`, `User Policy`, `Process`, `CurrentUser` und `Local Machine` listet die PowerShell nach Priorität geordnet auf. Der erste Scope in der Liste, der nicht den Wert „Undefined“ hat, wird selbst dann angewandt, wenn niedriger priorisierte Scopes eine restriktivere Ausführungsrichtlinie definieren.

Die Scopes `MachinePolicy` und `User Policy` werden per GPO vergeben. Ist keiner dieser beiden Scopes gesetzt, lässt sich die Execution Policy auf Ebene des laufenden Prozesses folgendermaßen aushebeln, ohne permanente Änderungen vorzunehmen:

```
PS > Set-ExecutionPolicy Bypass ↵
                 -Scope Process -Force
```

Falls `MachinePolicy` oder `UserPolicy` gesetzt sind, funktioniert das Aushebeln über den Befehl `Set-ExecutionPolicy` nicht. Sofern der verwendete Benutzer die Berechtigung besitzt, die entsprechenden Einträge in der Windows-Registrierung zu bearbeiten, kann er den Scope von `MachinePolicy` oder `UserPolicy` durch einen der folgenden Befehle auf „Bypass“ setzen:

```
PS > Set-ItemProperty -Path ↵
  HKLM:\Software\Policies\Microsoft\↵
                  Windows\PowerShell ↵
  -Name ExecutionPolicy -Value Bypass
PS > Set-ItemProperty -Path ↵
  HKCU:\Software\Policies\Microsoft\↵
                  Windows\PowerShell ↵
  -Name ExecutionPolicy -Value Bypass
```

Sobald das Aushebeln der Execution Policy nicht mehr nötig ist, stellt man den ursprünglichen Zustand durch das Wiedereinspielen der Gruppenrichtlinie via `gpupdate /force` wieder her (nach einer gewissen Zeit würde dies auch automatisch passieren). Hat die ausführende Person keine Berechtigung zum Anpassen des Registrierungseintrags, kann sie einen Administrator um Hilfe bitten.

PowerShell-Skripte sollten immer in neu gestarteten PowerShell-Sitzungen ausgeführt werden, damit es unter den eingebundenen Modulen nicht zu Konflikten kommt.

Subnetze, Webserver und Log-in-Oberflächen

Um sich mithilfe von Werkzeugen einen Überblick über das eigene Netzwerk zu verschaffen, kann man zunächst automatisiert die Subnetze ermitteln. Dazu dient ein auf GitHub verfügbares PowerShell-Skript:

```
PS > curl https://github.com/↵
     RogueValleyInformationSecurity/↵
                rvis-tools/blob/main/↵
           Get-InternalSubnets.ps1 ↵
          -O Get-InternalSubnets.ps1
PS > .\Get-InternalSubnets.ps1 ↵
                        > subnetze.txt
```

Mit dem Portscanner Nmap können darin offene Ports identifiziert werden. Die Resultate kann das Werkzeug EyeWitness nutzen, um Screenshots von Webservern und darauf laufenden Anwendungen zu erzeugen. Die dazu erforderlichen Tools sind in Kali Linux enthalten. Genauere Informationen zur Installation und Nutzung von Kali Linux sowie zu EyeWitness finden sich im ersten Artikel des Tutorials [1].

```
$ nmap -iL subnetze.txt -sV -oX ↵
                    nmap-ergebnisse.xml
$ eyewitness -x nmap-ergebnisse.xml
```

Mit den Screenshots von EyeWitness lässt sich beispielsweise herausfinden, ob es administrative Anmeldeoberflächen gibt, die nicht von überall aus dem internen Netzwerk erreichbar sein sollten. Möglicherweise finden sich so auch in Vergessenheit geratene Webanwendungen, die Verantwortliche aus dem Netz nehmen sollten.

AD-Sicherheit mit PingCastle

Das Programm PingCastle wird von Vincent Le Toux entwickelt und ist seit 2017 verfügbar. Die Software ist sowohl unter einer proprietären Lizenz als auch unter der Non-Profit Open Source License 3.0 lizenziert. Somit können Unternehmen PingCastle kostenlos gegen ihre eigene Umgebung einsetzen. Seit Kurzem kann PingCastle auch Azure AD auditieren, den Identitätsdienst der Microsoft-Cloud [2].

Die aktuelle Version von PingCastle lädt man auf der Webseite herunter, auf der sich auch eine übersichtliche Dokumentation der verschiedenen Funktionen findet (siehe ix.de/z7qk). Der Standard-Healthcheck läuft selbst in großen, komplexen AD-Umgebungen relativ schnell durch:

```
PS > .\PingCastle.exe --healthcheck ↵
                        --no-enum-limit
```

Die aufbereiteten Resultate legt die Software als interaktive HTML-Datei für die Betrachtung im Webbrowser ab. Es beginnt mit einer kondensierten Bewertung

Tutorialinhalt

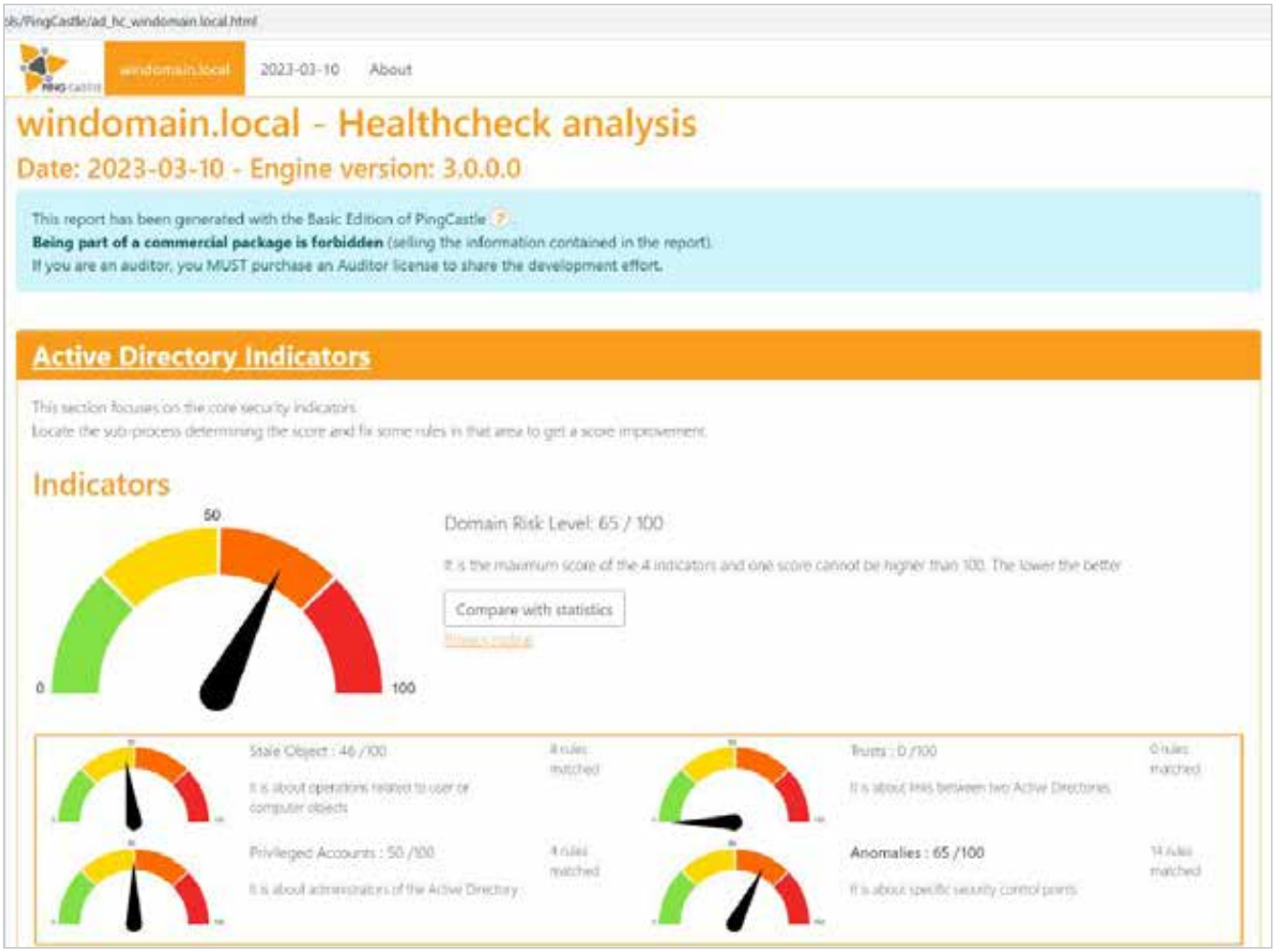
Teil 1: Scannen und Verifizieren der eigenen Systeme

Teil 2: Webapplikationen angreifen

Teil 3: Auditieren interner Netzwerke, Domänen und Systeme

Teil 4: Sammeln öffentlich verfügbarer Informationen und Analysieren der Angriffsfläche

Teil 5: Überprüfen von Cloud-Umgebungen

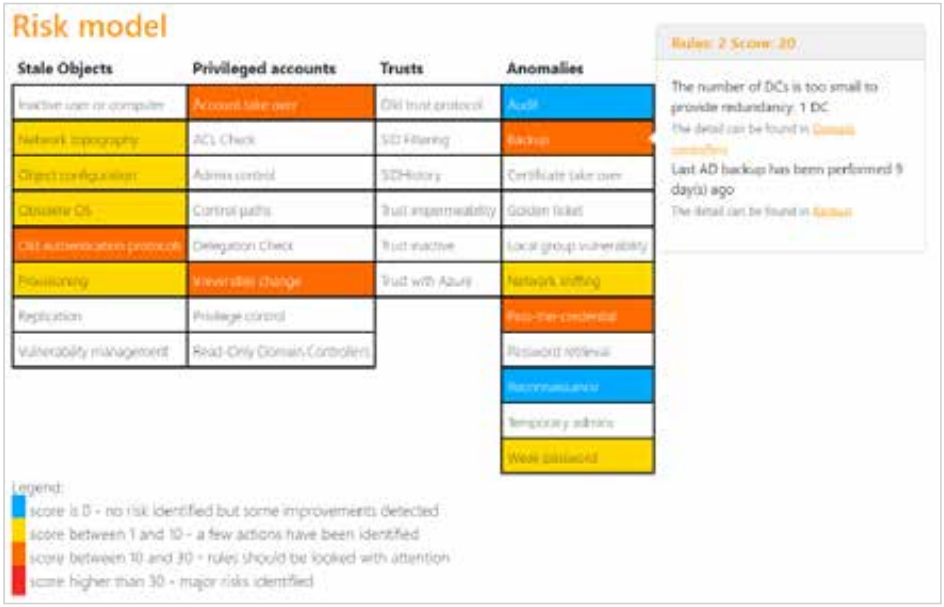


Der Healthcheck mit PingCastle liefert eine erste grobe Einschätzung des Risikolevels anhand einiger wichtiger Kategorien. Die anfälligste Kategorie gibt das Gesamtlevel vor (Abb. 1).

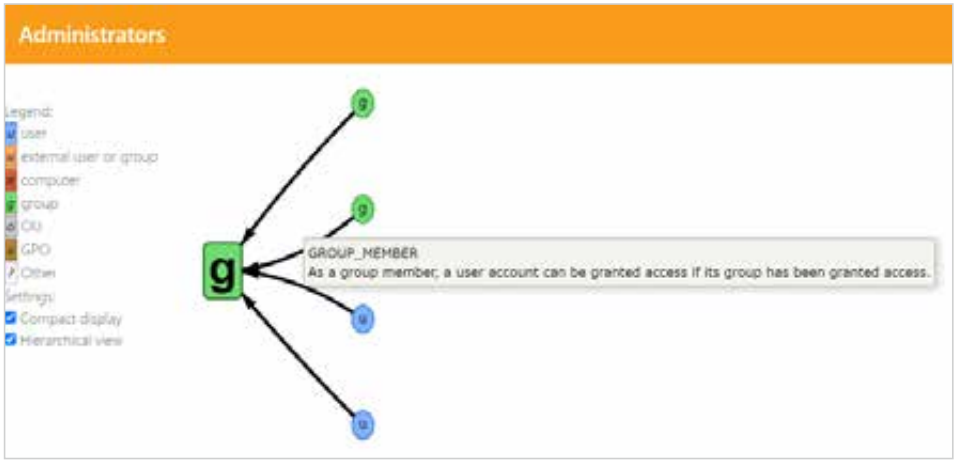
der Sicherheit der untersuchten AD-Umgebung anhand von vier Indikatoren: Vorgänge im Zusammenhang mit Benutzer- und Computerobjekten (Stale Objects), Vertrauensbeziehungen zwischen zwei Active Directories (Trusts), privilegierte Konten (Privileged Accounts) und alle Sicherheitsaspekte, die nicht in eine der vorigen Kategorien passen (Anomalies). Die höchste Bewertung eines einzelnen Indikators ergibt das Risikolevel für die gesamte Domäne (Abbildung 1).

Risiken auf einen Blick

Darunter zeigt das Risikomodell (Risk model, Abbildung 2) eine tabellarische Übersicht über die gefundenen Risiken für jeden der vier Indikatoren. In der Übersicht kann man auf einzelne Einträge klicken und erhält bereits einige detailliertere Informationen zum jeweiligen Risiko. Das Modell ist hilfreich, um sich einen schnellen Überblick über die gefundenen Risiken zu verschaffen. Die Farbcodierung unterstützt bei der Priorisierung der Risiken – trotzdem sollten Sicherheitsverantwortliche die Kritikalität der Risiken im Kontext des eigenen Active Directory (AD) evaluieren.



Das Risikomodell in PingCastle liefert einen Farbcode zur Priorisierung sowie detaillierte Informationen zu den gefundenen Risiken (Abb. 2).



Die Kontrollpfadanalyse offenbart Wege zu interessanten Benutzern und damit potenzielle Angriffspfade (Abb. 3).

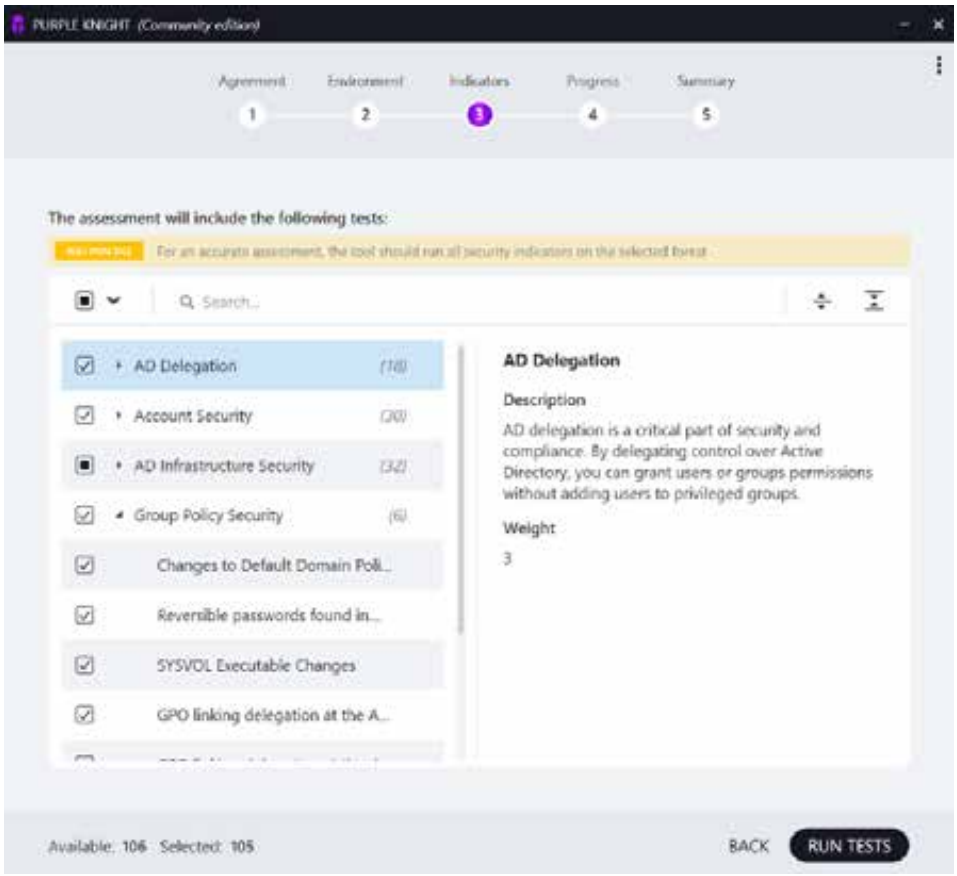
Die Bewertung von Risiken lässt sich anpassen, wenn dies aufgrund von Beobachtungen bei Angriffen in der freien Wildbahn angemessen erscheint. In der Regel ist diese Anpassung mit einer höheren Punktzahl in der Risikobewertung verbunden (siehe Blogartikel zu PingCastle und PurpleKnight; ix.de/z7qk).

Neben der detaillierten (technischen) Beschreibung des Risikos empfiehlt PingCastle Maßnahmen und Links zu weiterführenden Informationen und der Umsetzung der Maßnahmen. Die Befunde werden ebenfalls gruppiert nach Themen angezeigt, beispielsweise Domänen-, Benutzer- und Computerinformationen, Admingruppen, Kontrollpfadanalyse, Infrastruktur, Passwort-Policy und Gruppenrichtlinien.

Außerdem findet sich unter der Kontrollpfadanalyse versteckt in den Tabellen (in der Detailspalte ganz rechts) ein Link auf eine Analyse in Form eines Graphen (Abbildung 3). Er stellt die Zusammenhänge verschiedener Objekte des Active Directory dar und blendet Details zu den Objekten und Beziehungen beim Überfahren mit der Maus ein. Solche Graphen machen mögliche Pfade sichtbar, die ein Angreifer ausnutzen könnte, um seine Rechte in der AD-Umgebung auszuweiten. Diese Graphen ähneln jenen, die das Tool BloodHound (ix.de/z7qk und [3]) erzeugen kann, um nach potenziellen Angriffspfaden zu suchen. Beispielsweise kann man damit nach Wegen zum Domänenadmin oder anderen privilegierten Benutzern [4] suchen.

Purple Knight für hybride AD-Umgebungen

Purple Knight ist eine von Semperis im März 2021 auf den Markt gebrachte Closed-Source-Software (siehe ix.de/z7qk), die ähnlich wie PingCastle zum Bewerten der Sicherheit von AD und Azure AD dient. Sie wird von einem Team von Microsoft-Identity-Experten entwickelt und gepflegt und liefert wie PingCastle wertvolle Einsichten in das Sicherheitsniveau der eigenen Domänenlandschaft. Purple Knight verwendet dabei sogenannte Indicators of Exposure (IOE) zur Bewertung und Kategorisierung. Diese Indikatoren sind derzeit „Hybrid“, „Account Security“, „AD Delegation“, „Kerberos Security“, „AD Infrastructure Security“ und „Group Policy Security“.



Die auszuführenden Indikatoren in Purple Knight lassen sich genau konfigurieren. Anschließend kann der Scan für die vorher festgelegte Umgebung starten (Abb. 4).

Die Community-Version von Purple Knight ist kostenfrei verfügbar. Um einen Download-Link zu erhalten, ist eine Registrierung notwendig – der Link wird im Anschluss zusammen mit einigen Informationen zum Einstieg per E-Mail zugestellt.

Um Purple Knight zu starten, entpackt man die heruntergeladene ZIP-Datei und führt PurpleKnight.exe aus. Nach Akzeptieren der Lizenzbestimmungen wählt man die zu untersuchende Umgebung (Forest, Domäne) aus. Im folgenden Schritt lassen sich die auszuführenden Indikatoren granular konfigurieren (Abbildung 4). Ist das erledigt, kann der Scan ausgeführt und sein Fortschritt verfolgt werden. Das Ergebnis lässt sich in zwei Formaten exportieren: zum einen in Form eines vollständigen Berichts als PDF-Datei, zum anderen als CSV-Dateien mit den Ergebnissen verschiedener Indikatoren. Die Resultate finden sich außerdem als interaktive HTML-Datei im Output-Ordner, der im selben Verzeichnis wie PurpleKnight.exe liegt. Die Software speichert hier für jeden Scan einen mit Datum und Uhrzeit benannten Unterordner, in dem sich HTML- und JavaScript-Dateien befinden.

Purple Knight bewertet die untersuchte Umgebung mit einer Prozentzahl

und einem Buchstaben. Die Indikatoren fließen darin in verschiedener Gewichtung ein. Darunter findet sich eine tabellarische Übersicht mit Informationen zum Scan – ausgeführte Indikatoren, gefundene IOEs (Abbildung 5), fehlgeschlagene Indikatoren et cetera – sowie eine Bewertung der einzelnen Indikatoren. Detaillierte Informationen zur Bewertungsmethode stehen am Ende des Berichts im Appendix.

Außerdem findet sich dort eine Liste der gefundenen Indikatoren unterteilt in kritische und zusätzliche Ergebnisse, jeweils mit einem Link zum detaillierten Befund. Dieser enthält eine Bewertung des Schweregrads (Informational, Warning, Critical), eine Beschreibung des Indikators, die Wahrscheinlichkeit der Ausnutzung, die Feststellung des Scans sowie Schritte, um das Risiko zu beseitigen.

Der Abschnitt „AD Results" enthält neben den gefundenen IOEs alle durchgeführten Indikatoren, auch solche ohne Befund.

PingCastle und Purple Knight – zwei Tools im Vergleich

PingCastle liefert tiefe Einsicht in eine AD-Umgebung inklusive vieler Details und hat einen hohen Reifegrad erreicht. Besonders wenn es um die Beziehungen zwischen Objekten und Einstellungen geht, ermöglicht PingCastle es, Angriffspfade ausfindig zu machen. Die vielen Details machen es allerdings schwierig, gleichzeitig die Resultate zu überprüfen und darin zu navigieren – insbesondere in großen Forests mit mehreren Domänen.

Purple Knight erzeugt schnelle To-do-Listen, die mit ihrer Bewertung gut bei der Priorisierung unterstützen. Dies ist insbesondere dann hilfreich, wenn in die Sicherheit der AD-Umgebung noch nicht viel investiert wurde. Sobald das Active Directory grundlegend gehärtet ist, ist es sinnvoll, auch die zusätzlich angebotenen Einsichten und Details von PingCastle zu nutzen, etwa die Kontrollpfadanalysen. Ein Blogbeitrag von James Shakespear vergleicht ebenfalls beide Werkzeuge (siehe ix.de/z7qk).

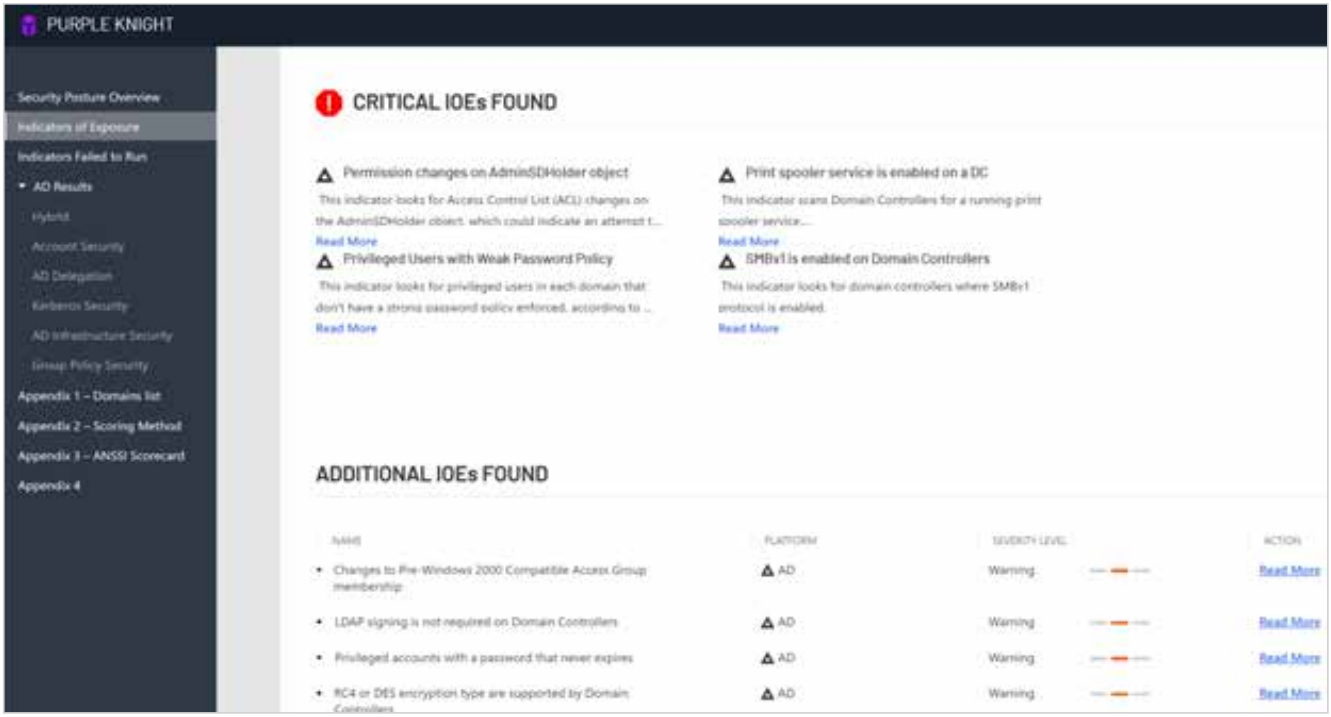
Beide Tools sind aus OPSEC-Sicht sehr „laut". OPSEC steht für Operations Security und bezieht sich in Hackerkreisen darauf, einen kleinen Fußabdruck zu hinterlassen und den Verteidigern nicht ins Netz zu gehen. Sowohl PingCastle als auch Purple Knight versuchen in kürzester Zeit möglichst viele Informationen zu sammeln und hinterlassen dadurch zahlreiche Spuren. Sollten bei einem Selbst-Hacking nicht alle Verantwortlichen informiert sein, kann es passieren, dass Monitoring- oder andere Tools beim SIEM Alarm auslösen und die Verantwortlichen diesen nicht zuordnen können.

Sichere Passwörter und Passwort-Hashes aus Datenlecks

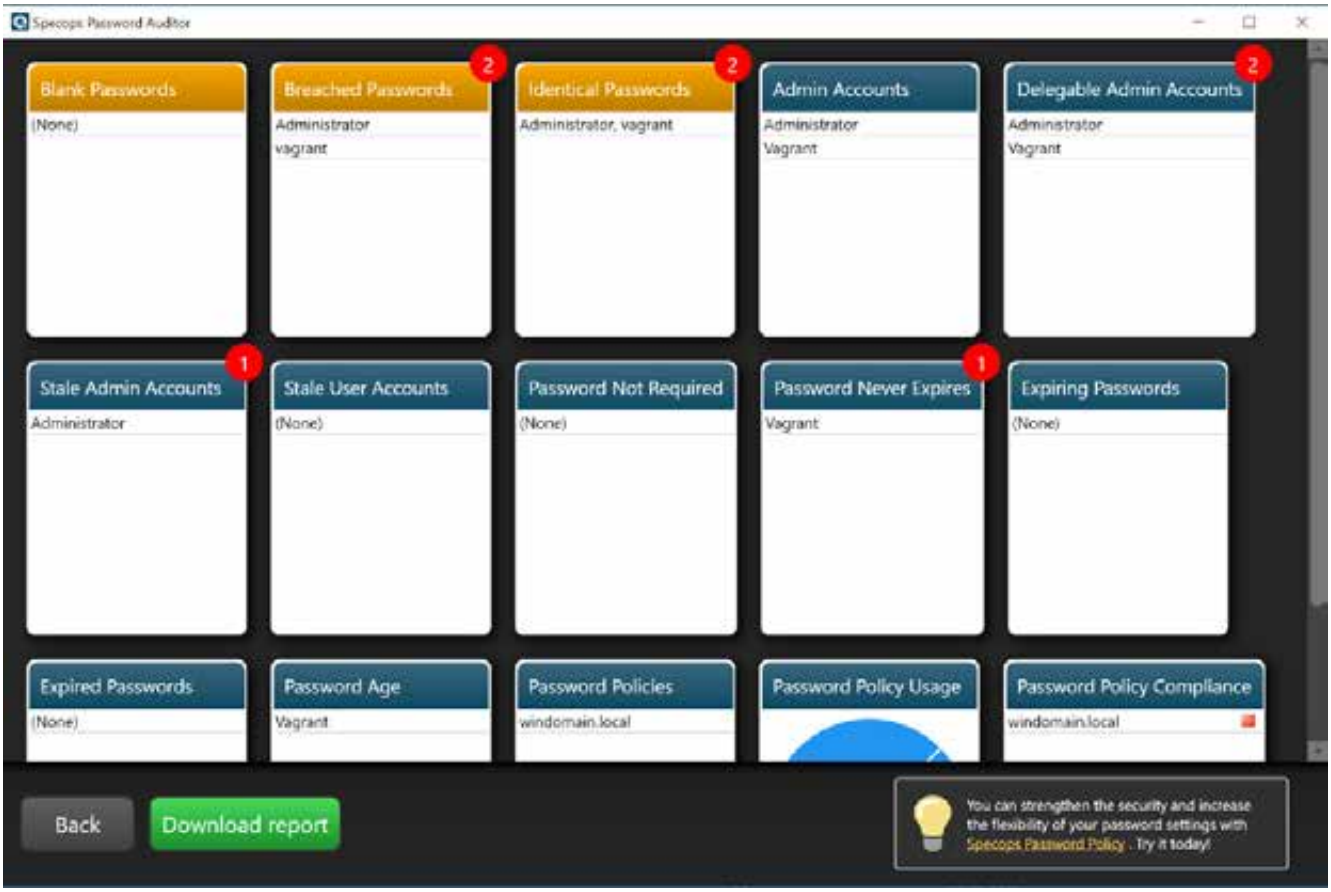
Ein grundlegender Baustein für die Sicherheit in einer Domäne sind starke Passwörter [5]. Diese sollten sich nicht auf hochprivilegierte Konten beschränken, denn jedes mit dem AD verbundene Gerät kann als Einfallstor für Angreifer dienen. Starke Passwörter sind daher fundamental, um eine Umgebung abzusichern. Sie schützen auch gegen Angriffe wie Kerberoasting, AS-REP-Roasting [6], Password Spraying und abgefangene Net-NTLM-Hashes, bei denen versucht wird, das Passwort eines Kontos zu knacken.

Ein wichtiger Aspekt für die Stärke eines Passworts ist seine Länge, denn mit ihr steigt die Anzahl möglicher Kennwörter exponentiell. Daher sind reine Brute-Force-Angriffe gegen sehr lange Passwörter selbst mit modernster Hardware unpraktikabel. Um ihre Erfolgschancen zu verbessern, wenden Angreifer Techniken und Hilfsmittel wie Wörterbuchangriffe, Passwortmasken oder Regenbogentabellen an.

Alle Passwörter sollten eine Länge von mindestens 14 Zeichen haben, jene von hochprivilegierten Konten mindestens 20 Zeichen. Je höher die Komplexität ist (mehr Zeichenarten), desto kürzere Passwörter sind akzeptabel (Empfehlungen des BSI siehe ix.de/z7qk). Passwörter für besonders gefährdete Konten wie solche mit einem Service Principal Name (SPN) sollten deutlich länger sein, also mindestens 32 Zeichen (Stichwort Kerberoasting). Falls bestimmte Anwendungen diese Längen nicht unterstützen, sollte man die zulässige Länge



Neben den kritischen Funden listet PurpleKnight auch Indikatoren ohne Befund auf (Abb. 5).



Der Specops Password Auditor markiert gefährliche Funde mit Rot und macht Vorschläge, wie sich die Passwortsicherheit verbessern lässt (Abb. 6).

zumindest ausschöpfen. Solche Maßnahmen können durch Gruppenrichtlinien mit Fine-grained Password Policies (siehe ix.de/z7qk) umgesetzt werden. Außerdem sollten Nutzer dazu angehalten werden, ganze Passphrasen zu verwenden, da diese ohne Anstrengung längere Passwörter ergeben und leichter zu merken sind.

Neben Anforderungen an die Länge sollten zusätzlich schlechte Passwörter aus der AD-Umgebung beseitigt werden. Dazu zählen einerseits schwache Kennwörter wie „FirmenName2023“, aber auch solche, die in Listen von Passwort-Hashes aus Datenlecks enthalten sind. Des Weiteren sollten sich zwei Konten niemals dasselbe Passwort teilen.

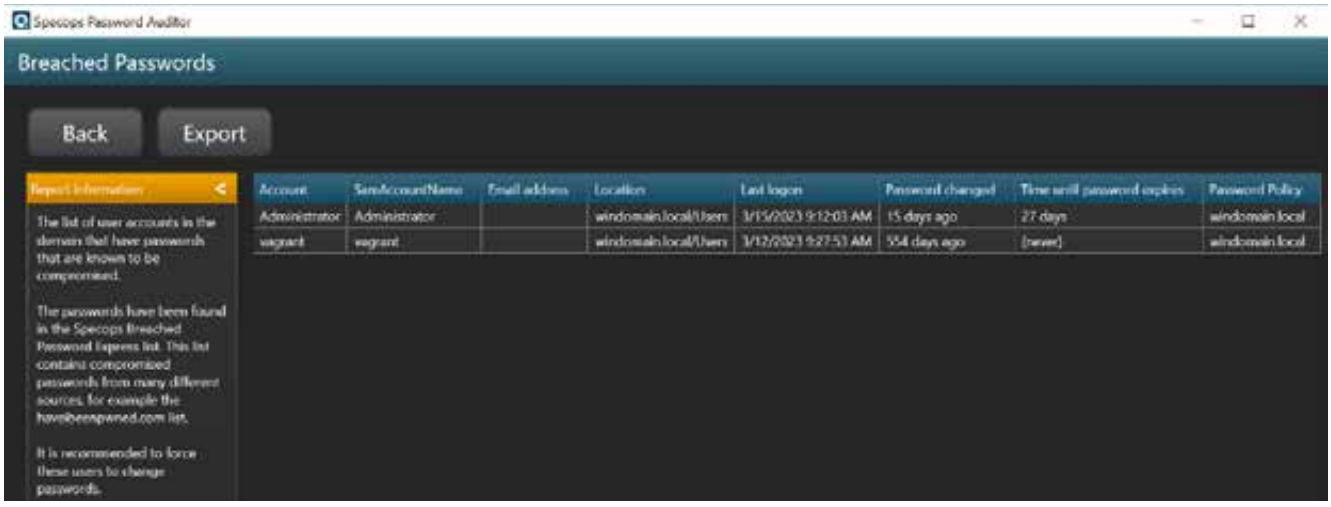
Geklaute Passwörter überprüfen

Eine große Anzahl Passwort-Hashes aus Datenlecks kann man von der bekannten Seite „Have I Been Pwned“ für die Offlinenutzung herunterladen. So lässt sich unter anderem überprüfen, ob in einem Netzwerk kompromittierte Passwörter in Benutzung sind. Zu diesem Zweck wurde das .NET-Programm haveibeenpwned-downloader entwickelt (siehe ix.de/z7qk). Es setzt .NET 6 voraus. Ist es eingerichtet, installiert man den Downloader einfach per PowerShell durch folgenden Befehl:

```
PS > dotnet tool install --global ↵
          haveibeenpwned-downloader
```

Falls das Paket nicht gefunden wird, sollte das Hinzufügen folgender Paketquelle Abhilfe schaffen:

```
PS > dotnet nuget add source ↵
          https://api.nuget.org/v3/↵
            index.json -n nuget.org
```



Details erfährt man, wenn man in die Kacheln klickt – hier entdeckte der Specops Password Auditor beispielsweise ein Passwort, das bekanntermaßen schon kompromittiert wurde (Abb. 7).

Abhilfe, so rät das Tool, lässt sich schaffen, indem man den Benutzer des kompromittierten Passworts zum Wechseln zwingt (Abb. 8).

Nach dem Installieren lädt der folgende Befehl alle SHA1- beziehungsweise NTLM-Passwort-Hashes herunter, wobei das zuletzt stehende Argument den Namen der Textdatei darstellt:

```
PS > haveibeenpwned-downloader.exe ↵
                                pwnedpasswords
PS > haveibeenpwned-downloader.exe ↵
                             -n pwnedpasswords_ntlm
```

Der Download der Passwort-Hashes nimmt einige Zeit in Anspruch. Im Sommer 2023 ist die Textdatei mit den SHA1-Hashes etwa 37 GByte groß, die mit den NTLM-Hashes kommt auf knapp 31 GByte.

Hashes überprüfen mit dem Specops Password Auditor

Im Folgenden werden eine proprietäre und eine freie Software vorgestellt, mit deren Hilfe sich die Konten in der eigenen AD-Umgebung auf Passwort-Hashes aus Datenlecks überprüfen lassen.

Der Specops Password Auditor (siehe ix.de/z7qk) ist gegen Abgabe von Kontaktdaten kostenlos verfügbar. Der Downloadlink für die Software sowie eine ein Jahr gültige Lizenzdatei werden im Anschluss an die E-Mail-Verifikation zugestellt. Das Ausführen des Specops Password Auditors erfordert einen Domänenadministrator, da dieses Konto über das Recht zum Auslesen der Passwort-Hashes verfügt.

In der Beschreibung heißt es, dass der Password Auditor ausschließlich Informationen aus der AD-Umgebung ausliest und keine Änderungen durchführt. Im Code überprüfen lässt sich dies jedoch nicht, da der Quelltext nicht öffentlich verfügbar ist. Daher sollte man den Specops Password Auditor nur nach Absprache mit Geschäftsführung und Betriebsrat nutzen.

Für die Installation ist .NET Framework 4.7 oder höher nötig. Nach dem Start des Programms kann man die Lizenz links unten importieren sowie den Ausgangspunkt für den Scan definieren. Optional kann man die Benutzerdaten anonymisieren. Mit Klick auf „Start“ konfiguriert man im nächsten Schritt, ob der Password Auditor die Passwort-Hashes auf bekannte Hashes aus Datenlecks überprüfen soll. Dazu gibt man einen lokalen Ordner an, in den er die Hashes aus den Datenlecks für die Prüfung herunterlädt (etwa 6 GByte im Frühjahr 2023). Mit Klick auf „Start Scanning“ beginnt die Analyse. Ist sie abgeschlossen, präsentiert das Tool die Befunde als einzelne Kacheln – mit roten Zahlen an der Kachel, falls es Verbesserungspotenzial gibt (Abbildung 6).

Den Bericht kann man als PDF herunterladen („Download report“). Die Informationen aus den einzelnen Kacheln lassen sich in der Detailansicht (Klick auf die Kachel) als CSV-Datei exportieren. Darin finden sich außerdem detailliertere Informationen zu Befunden (Abbildung 7). Der PDF-Bericht enthält ebenfalls Informationen dazu, warum ein Befund ein Risiko darstellt und wie es beseitigt werden kann (Abbildung 8).

DSInternals – ein weiterer Passwortwächter

Eine Open-Source-Software mit ähnlichem Funktionsumfang ist das Framework DSInternals, genauer gesagt ein PowerShell-Cmdlet aus dem Framework (siehe ix.de/z7qk). DSInternals stellt einige interne Funktionen des Active Directory zur Verfügung und kann von beliebigen .NET-Anwendungen verwendet werden. Die Installation erfolgt wie in Listing 2 beschrieben.

Das Cmdlet für das Untersuchen der AD-Passwörter auf schwache Einträge, Duplikate, Standardpasswörter und nicht gesetzte Kennwörter heißt „Test-PasswordQuality“ (siehe ix.de/z7qk). Die Analyse kann man online (Variante DCSync) oder offline (Variante ntds.dit) durchführen.

Für die Onlinevariante benötigt man die Rechte eines Domänenadministrators oder ein vergleichbares Konto. Als Liste von Passwort-Hashes aus Datenlecks lassen sich die heruntergeladenen Passwort-Hashes von „Have I been Pwned“ heranziehen. Die Überprüfung wird wie in Listing 3 durchgeführt.

Für die Offlinevariante ist zunächst eine Kopie der Datei ntds.dit zu erstellen. Da das Active Directory sie permanent verwendet, ist sie gesperrt und lässt sich nicht einfach kopieren. In Listing 4 wird dies über eine Schattenkopie umgangen. Es wird außerdem aus der Registry der Eintrag HKLM\SYSTEM

Listing 2: DSInternals installieren

```
# Enable TLS1.2 on older versions of Windows
PS > [Net.ServicePointManager]::SecurityProtocol ↵
                                        = [Net.SecurityProtocolType]::Tls12

# Download the NuGet package manager binary.
PS > Install-PackageProvider -Name NuGet -Force

# Install DSInternals
PS > Install-Module -Name DSInternals -Force
```

Listing 3: DSInternals ausführen

```
# Perform password quality test (DCSync)
PS > Get-ADReplAccount -All -Server DC -NamingContext "dc=windomain,dc=local" |
Test-PasswordQuality -WeakPasswordHashesFile E:\pwnedpasswords_ntlm.txt ↵
                                                    -IncludeDisabledAccounts
```

kopiert, da dieser den Boot-Key für die Entschlüsselung der kopierten ntds.dit enthält.

Nachdem man die Datei ntds.dit erfolgreich kopiert hat, wird der Boot-Key extrahiert und das Cmdlet Test-PasswordQuality gegen die Benutzerdatenbank des AD ausgeführt (siehe Listing 5). Gegebenenfalls muss ntds.dit zunächst repariert werden, der Befehl dazu ist am Ende von Listing 5 aufgeführt.

Seit Mai 2023 gibt es ebenfalls kostenfrei einen Wrapper für DSInternals namens „PasswordSolution“; Details zum Einsatz stehen in einem ausführlichen Blogbeitrag des Autors (zu finden über ix.de/z7qk).

Gefährlich: großzügige Netzwerkfreigaben

Ein häufig übersehener Angriffsvektor sind zu freizügig konfigurierte Netzwerkfreigaben [6]. Durch nicht autorisierten Zugriff auf Freigaben werden beispielsweise vertrauliche Daten wie Betriebsgeheimnisse oder Passwortlisten exponiert, er kann aber auch zu Rechteerweiterung führen oder für einen Ransomware-Angriff genutzt werden.

Listing 4: ntds.dit als Domänenadministrator (oder ähnlich) extrahieren

```
C:\Windows\system32>vssadmin create shadow /for=C:
vssadmin 1.1 - Volume Shadow Copy Service administrative command-line tool
(C) Copyright 2001-2013 Microsoft Corp.

Successfully created shadow copy for 'C:\'
    Shadow Copy ID: {e776fc8c-8b9d-424e-a558-11db177a56f4}
    Shadow Copy Volume Name: \\?\GLOBALROOT\Device\HarddiskVolumeShadowCopy1

C:\Windows\system32>copy \\?\GLOBALROOT\Device\HarddiskVolumeShadowCopy1\↵
                                                Windows\NTDS\ntds.dit C:\tmp\ntds.dit
        1 file(s) copied.

C:\Windows\system32>reg SAVE HKLM\SYSTEM C:\tmp\SYS
The operation completed successfully.

C:\Windows\system32>vssadmin delete shadows /shadow={e776fc8c-8b9d-424e-↵
                                                                    a558-11db177a56f4}
vssadmin 1.1 - Volume Shadow Copy Service administrative command-line tool
(C) Copyright 2001-2013 Microsoft Corp.

Do you really want to delete 1 shadow copies (Y/N): [N]? y

Successfully deleted 1 shadow copies.
```

Listing 5: DSInternals ausführen (Offlinevariante, administrative PowerShell)

```
PS > $key = Get-BootKey -SystemHivePath C:\tmp\SYS
PS > Get-ADDBAccount -All -BootKey $key -DBPath C:\tmp\ntds.dit | ↵
         Test-PasswordQuality -WeakPasswordHashesFile E:\pwnedpasswords_ntlm.txt

# Reparieren von ntds.dit, falls die Fehlermeldung "The database is not in ↵
                                                        a clean state." erscheint.
PS > ESENTUTL /p C:\tmp\ntds.dit /!10240 /8 /o
```

```
PS C:\Windows\system32> Get-ADReplAccount -All -Server DC -NamingContext "dc=windomain,dc=local" | Test-PasswordQuality
 -WeakPasswordHashesFile E:\pwnedpasswords_ntlm.txt -IncludeDisabledAccounts

Active Directory Password Quality Report
----------------------------------------

Passwords of these accounts are stored using reversible encryption:

LM hashes of passwords of these accounts are present:

These accounts have no password set:
  WINDOMAIN\DefaultAccount
  WINDOMAIN\Guest

Passwords of these accounts have been found in the dictionary:
  WINDOMAIN\Administrator
  WINDOMAIN\vagrant

These groups of accounts have the same passwords:
  Group 1:
    WINDOMAIN\Administrator
    WINDOMAIN\vagrant

These computer accounts have default passwords:

Kerberos AES keys are missing from these accounts:

Kerberos pre-authentication is not required for these accounts:

Only DES encryption is allowed to be used with these accounts:

These accounts are susceptible to the Kerberoasting attack:
  WINDOMAIN\krbtgt

These administrative accounts are allowed to be delegated to a service:
  WINDOMAIN\Administrator
  WINDOMAIN\krbtgt
  WINDOMAIN\vagrant

Passwords of these accounts will never expire:
  WINDOMAIN\DefaultAccount
  WINDOMAIN\Guest
  WINDOMAIN\vagrant

These accounts are not required to have a password:
  WINDOMAIN\DefaultAccount
  WINDOMAIN\Guest

These accounts that require smart card authentication have a password:
```

Das Test-PasswordQuality-Cmdlet entlarvt schwache, nicht gesetzte oder kompromittierte Passwörter (Abb. 9).

Das PowerHuntShares-Dashboard verrät, wo zu freizügige Rechte eingeräumt wurden und welche Geräte ungehindert zu erreichen sind (Abb. 10).

Bereits lesender Zugriff kann schwerwiegende Folgen haben, wenn Kriminelle die Informationen verkaufen oder die Bestohlenen damit erpressen. Im Extremfall kann lesender Zugriff sogar Remote-Code-Ausführung bedeuten, wenn Zugangsdaten für Datenbankserver oder gar einen Domänenadministrator im Klartext herumliegen. Schreibender Zugriff bietet einem Angreifer darüber hinaus weitere Möglichkeiten für Angriffe.

Es ist nicht einfach, im Active Directory einen Überblick über Netzwerkfreigaben zu behalten und das Prinzip der geringsten Privilegien umzusetzen. Die Gründe für Fehlkonfigurationen sind vielfältig. Als Beispiel sei ein Problem angeführt, das dazu führen kann, dass alle Domänenbenutzer Zugriff auf eine Netzwerkfreigabe haben, obwohl diese nur lokalen Nutzern gewährt werden sollte. Das hat seine Ursache darin, dass die Gruppe der lokalen Benutzer (Builtin\Users) die Gruppe der authentisierten Benutzer (Authenticated Users) enthält. Sobald das System Teil eines AD ist, gehören die Gruppen der Domänenbenutzer (Domain Users) und der Domänencomputer (Domain Computers) als Mitglieder der authentisierten Benutzer ebenfalls dazu – was dazu führt, dass jeder Domänenbenutzer Zugriff auf die Netzwerkfreigabe hat.

Risiken erschnüffeln

Einige Quick Wins lassen sich schnell mit dem Tool Snaffler finden (siehe ix.de/z7qk). Es wird mit `snaffler.exe -s -o snaffler.log` ausgeführt.

Snaffler enumeriert die Computer in der Active-Directory-Umgebung, stellt fest, ob sie Netzwerkfreigaben haben, und listet im Anschluss die auf den Freigaben lesbaren Dateien auf. Dabei ist die Ausgabe wie folgt formatiert, wobei einzelne Elemente durch | voneinander getrennt sind:

```
Timestamp | Log Entry Type, i.e. ↵
   File, Share, etc. | Triage Level ↵
   | Matched Rule Name | Permission ↵
 (R/RW) | Matched Regex | File Size ↵
   | Last Modified Time | Full File ↵
                Path | Match Context
```

Das Triage-Level umfasst folgende Optionen (von aufsteigendem Interesse für Angreifer): Green, Yellow, Red, Black. Das Triage-Level Red enthält beispielsweise viele reguläre Ausdrücke, die Passwörter, private Schlüssel oder Ähnliches zu erkennen versuchen. Snaffler kann somit nützlich sein, um unnötige Netzwerkfreigaben sowie Dateien mit potenziell vertraulichen Informationen zu identifizieren.

Große Netzwerke – unübersichtliche Freigaben

Je größer die Umgebung, desto schwerer ist es, die Übersicht über Netzwerkfreigaben zu bewahren. Um zu freizügige Rechte auf Netzwerkfreigaben zu beseitigen, müssen Fragen wie „Wo befinden sich Netzwerkfreigaben?“, „Welche Netzwerkfreigaben bringen ein hohes Risiko mit sich?“ und „Wer erstellte die Netzwerkfreigabe und wann geschah dies?“ beantwortet werden. Viele Antworten kann ein normaler Domänenbenutzer manuell per LDAP über PowerShell-Befehle finden. Einfache Suchen nach Netzwerkfreigaben mit hohem Risiko wie `C$` oder `ADMIN$` identifizieren bereits wichtige Freigaben. Danach ist es hilfreich, die Netzwerkfreigaben zu gruppieren und zu zählen – beispielsweise „Wie viele Netzwerkfreigaben beinhalten `Backup` in ihrem Namen?“ oder „Wie viele Netzwerkfreigaben befinden sich in den verschiedenen Subnetzen?“.

Ein Werkzeug, das diese Aufgabe automatisiert und die gesammelten Informationen ansprechend aufbereitet und gruppiert, ist PowerHuntShares (siehe ix.de/z7qk). Der zugehörige Blogpost und die auf YouTube verfügbare Präsentation sind sehr zu empfehlen, beide verlinkt vom GitHub-Repository.

Das Dashboard von PowerHuntShares (Abbildung 10) gibt unter Reports einen ersten Überblick über den Anteil an Computern und Netzwerkfreigaben mit zu freizügigen Rechten sowie den Rechten auf den Netzwerkfreigaben und eine Verteilung der häufigsten Namen. Die Computer-, Netzwerkfreigaben- und ACL-Übersicht schlüsselt alles detaillierter auf (ACL steht für Access Control Lists, Zugriffskontrolllisten [6]). Aus der Computerübersicht lässt sich herauslesen, ob alle Geräte erreichbar sind oder vielleicht eine Firewall die Verbindung verhindert – gegebenenfalls muss man das Ganze aus einem anderen Netzwerksegment wiederholen.

Unter „Data Insights“ verbergen sich Übersichten, welche Gruppen Zugriff auf Netzwerkfreigaben haben, Statistiken über Namen und Besitzer von Netzwerkfreigaben und betroffene Subnetze. Die Übersicht „Top Folder Groups“ soll helfen, die Ursache für die zu freizügigen Berechtigungen auf Netzwerkfreigaben zu finden – beispielsweise, wenn die gleichen Dateien wegen einer Anwendung auf diversen Freigaben liegen.

Als langfristige Lösung können Desktop-Suchmaschinen dienen, mit denen man das lokale System regelmäßig nach vertraulichen Informationen durchforstet. Einige (zum Teil kommerzielle) Tools für diesen Zweck vergleicht ein Blogartikel des SANS-Instituts (siehe ix.de/z7qk). Sobald man die Netzwerkfreigaben einigermaßen im Griff hat, ist es außerdem ratsam, kritische Freigaben zu überwachen sowie verfügbare Netzwerkfreigaben regelmäßig zu überprüfen, beispielsweise vierteljährlich.

Lokale Analyse von Clients und Servern

Auch zum Überprüfen einzelner Systeme gibt es sowohl für Windows als auch für Linux Werkzeuge, die Systeme auf Angriffsvektoren zur Rechteerweiterung oder Härtungsoptionen untersuchen.

Für Windows steht beispielsweise das Werkzeug PrivescCheck (siehe ix.de/z7qk) bereit, das auf einem lokalen System nach Möglichkeiten der Rechteerweiterung sucht (etwa die als „Unquoted Service Paths“ bekannte Fehlkonfiguration):

```
PS > .\PrivescCheck.ps1
PS > Invoke-PrivescCheck -Extended ↵
             -Report "PrivescCheck_$↵
                 ($env:COMPUTERNAME)"
```

Am Ende der Ausgabe liefert das Werkzeug eine Übersicht, in der sich erkennen lässt, in welchen Bereichen Probleme bestehen.

Ein Tool, das Windows-Systeme hinsichtlich Härtungsmaßnahmen nach verschiedenen Standards überprüft, ist HardeningKitty (siehe ix.de/z7qk). Es sollte als lokaler Administrator ausgeführt werden, um die Systemeinstellungen lesen zu können:

```
PS > Import-Module ↵
                .\HardeningKitty.psm1
PS > Invoke-HardeningKitty ↵
          -EmojiSupport -Mode Audit ↵
                        -Log -Report
```

HardeningKitty überprüft diverse Konfigurationen, wobei es die einzelnen Resultate je nach Kritikalität farblich unterschiedlich codiert, etwa Gelb für „Severity: Medium“ oder Rot für „Severity: High“.

Weitere Testwerkzeuge, die nach Möglichkeiten zur lokalen Rechteerweiterung suchen, sind LinPEAS für Linux und WinPEAS für Windows (beide zu finden über ix.de/z7qk).

Fazit

Einige der in diesem Artikel angesprochenen Bereiche wie AD-Sicherheit und Netzwerkfreigaben in den Griff zu bekommen, ist ein mühsames und langfristiges Unterfangen.

Trotzdem kommt man nicht umhin, diese Themen anzugehen, denn das Härten der eigenen Umgebung ist essenziell, um sie resilienter gegen Angreifer sowohl von innen als auch von außen zu machen. Die vorgestellten Werkzeuge können dabei einen wichtigen Beitrag leisten, zunächst einen Überblick zu erlangen. Sie geben zudem Empfehlungen für die Priorisierung und oft auch für die Behebung der gefundenen Probleme.

Penetrationstests sind gut und unverzichtbar, jedoch betrachten sie häufig nur die Sicherheit einzelner Anwendungen. Die ganzheitliche Sicht ist nicht im Fokus. Da eine Umgebung jedoch immer nur so sicher sein kann wie ihr schwächstes Glied, ist eine ganzheitliche Betrachtung wichtig – und dazu gehören auch gerne vernachlässigte Bereiche wie Netzwerkfreigaben. (ur@ix.de)

Quellen

[1] Stephan Brandt; Sich selbst hacken: Scannen der eigenen Systeme; iX 8/2023, S. 40

[2] Frank Ully; Ins Netz gegangen; Angriffe auf das Azure Active Directory und auf Azure-Dienste; iX 4/2022, S. 50

[3] Frank Ully; Fette Beute; Passwörter und Hashes – wie Angreifer die Domäne kompromittieren; iX 11/2020, S. 94

[4] Frank Ully; Himmelsgeschenk; Active Directory: Komfortable IT-Schaltzentrale mit Schwachpunkten; iX 10/2020, S. 40

[5] Sandro Affentranger; Schwierige Wahl; Passwortsicherheit (nicht nur) im Active Directory; iX 1/2022, S. 116

[6] Frank Ully; Frisch geröstet; Roasting, Rechte, Richtlinien: Wie Angreifer sich im Active Directory Zugriff verschaffen; iX 12/2020, S. 92

[7] Die im Artikel erwähnten Werkzeuge und Artikel sind über ix.de/z7qk zu finden.

GEORG BUBE

ist Senior Penetration Tester bei der Oneconsult Deutschland AG in München. Im Laufe der Jahre hat er zahlreiche Penetrationstests in unterschiedlichsten Branchen durchgeführt.