4 | 19 Sonderdruck für DeviceTRUST

IT Administrator
Das Magazin für professionelle System- und Netzwerkadministration

deviceTRUST 19.1

www.it-administrator.de

deviceTRUST 19.1

Schutzschild

von Dr. Christian Knermann

Die Software deviceTRUST möchte Zugriffe reglementieren. Sie arbeitet hierzu als Erweiterung für virtuelle Desktops und Terminalserver, basierend auf dem Client-Sicherheitskontext. Durch Steuerung des Zugangs zum Desktop und zu einzelnen Anwendungen sorgt sie, wie unser Test zeigt, für mehr Sicherheit und ein besseres Benutzererlebnis. Und dies nicht nur bei der Anmeldung, sondern auch zur Echtzeit in laufenden Sitzungen.

Anbieter aus dem Bereich des End-User-Computing (EUC), wie etwa Citrix, Microsoft und VMware, propagieren seit vielen Jahren die Möglichkeiten mobilen Arbeitens. Mit Hilfe von Desktopvirtualisierung und Terminalservern können Anwender zu jeder Zeit von jedem Ort und mit beliebigen Endgeräten ihrer Arbeit nachgehen. Was die Anwender freut, stellt die Admins vor große Herausforderungen, müssen sie sich doch um die Einhaltung vieler Sicherheits-, Compliance- und regulatorischer Vorgaben kümmern. Und dies muss auch gelingen, wenn die Clients weltweit verstreut in teils unsicheren Netzen unterwegs sind und sogar beliebige, nicht vom Unternehmen verwaltete Endgeräte im Einsatz sind.

Hier kommt deviceTRUST des gleichnamigen deutschen Softwareherstellers ins Spiel, das den Kontext eines Clients detailliert analysiert und die entsprechenden Informationen in die Remote-Sitzungen übermittelt. Die Software erfasst über 200 Hardware-, Software-, Netzwerk-, Sicherheits-, Performance- und Standorteigenschaften. Das passiert bei der Anmeldung und auch kontinuierlich zur Laufzeit einer Sitzung, sodass die Gegenstelle auf dem virtuellen Desktop oder Terminalserver bei Änderungen des Sicherheitsstatus mit vom Admin definierten Aktionen reagieren und Zugriffe erlauben oder auch verbieten kann. deviceTRUST wird pro namentlich benannten Nutzer (Named User) lizenziert. Ein Benutzer darf mit beliebig vielen Endgeräten auf beliebig viele entfernte Desktops und Server zugreifen.

Für Windows, macOS, iOS und IGEL

Die Software besteht aus zwei Kernkomponenten, dem deviceTRUST-Client (DTC) und dem deviceTRUST-Host (DTH). Beide sind schnell installiert und lassen sich auch in Systeme zur automatischen Softwareverteilung und zum Provisionieren von Desktops aus einem Golden Image integrieren. Nach der initialen Installation kann deviceTRUST sich mittels einer eingebauten Update Engine selbsttätig darum kümmern, die Clients aktuell zu halten.

Der DTC ist für Microsoft Windows und auch macOS verfügbar. Zudem stellt der Bremer Anbieter IGEL Technology deviceTRUST ab Werk in seinen Linux-Thin-Clients bereit. Um die Funktionalität zu aktivieren, muss lediglich ein Haken im lokalen Setup des Thin Clients oder zentral im Konfigurationsprofil der "IGEL Universal Management Suite" gesetzt werden. Eine App für Apple iOS ist ebenfalls verfügbar, Unterstützung für

Android soll mit einer zukünftigen Version folgen.

Der DTH residiert auf den Systemen, die Remote-Sitzungen anbieten. Dort darf auch der Client zusätzlich installiert werden. Das ist nötig in sogenannten "Multihop"- oder "Doublehop"-Szenarien, wenn ein Benutzer sich etwa mit einem virtuellen Desktop verbindet und von dort eine zweite Sitzung auf einem nachgelagerten Terminalserver öffnet. Der DTC auf dem virtuellen Desktop kann dann den Kontext des Endgeräts auch auf den Terminalserver durchreichen.

Dazu nutzt deviceTRUST die virtuellen Kanäle der Remote-Protokolle Citrix ICA/HDX, Microsoft RDP oder PC-over-IP (PCoIP). Letzteres kommt bei den Amazon AWS WorkSpaces sowie VMware Horizon zum Einsatz. Für Remote-Clients unter Apple iOS, wo die virtuellen Kanäle nicht nutzbar sind, betreibt der Hersteller einen Dienst in der Microsoft-Azure-Cloud. Wer iOS-Geräte in Verbindung mit deviceTRUST nutzen möchte, muss die entsprechende App installieren und diese online im Kundenportal registrieren.

Ansonsten benötigt deviceTRUST weder einen zusätzlichen Server noch eine Datenbank, sondern setzt auf vorhandene Infrastruktur auf. Der Host erhält seine Einstellungen über Gruppenrichtlinien, wofür die Software eigene ADMX-Vorlagen für die Richtlinien und eine grafische Konsole mitbringt, die den Gruppenrichtlinienverwaltungs-Editor erweitert.

deviceTRUST 19.1

Produkt
Kontextabhängige Clientsicherheit für virtuelle Desktops und Terminalserver.

Hersteller
deviceTRUST GmbH
https://devicetrust.de

Preis
Mietlizenz auf Subscription-Basis ab 23,76 Euro pro Named User für 12 Monate, Mengenstaffeln sowie Rabatte bei längeren Laufzeiten (24 oder 36 Monate) auf Anfrage.

Systemvoraussetzungen
Unterstützte Betriebssysteme (32-/64-bit): Microsoft Windows 7/8.x/10, Microsoft Windows Server 2008 R2/2012/2012 R2/2016/2019, Apple macOS 10.12 oder höher

Unterstützte mobile Betriebssysteme: Apple iOS 8.x oder höher

Unterstützte IGEL-Thin-Client-Betriebssysteme:
IGEL OS 10.3.500 oder höher

Unterstützte Remoting-Technologien: Microsoft Remote Desktop Protocol (RDP), Citrix Independent Computing Architecture (ICA), Amazon WorkSpaces, PC-over-IP und VMware Horizon (PCoIP)

Technische Daten:
www.it-administrator.de/downloads/datenblaetter

Installation mit minimalem Aufwand

Wie genau deviceTRUST funktioniert, erklärt sich am besten im praktischen Einsatz und so haben wir die Software in unserer Testumgebung mit einem Domänencontroller sowie zwei Remotedesktop-Sitzungshosts unter Windows Server 2016 installiert. Unsere Clientcomputer sollten sich zum ersten Terminalserver verbinden und von dort eine weitere Sitzung auf dem zweiten Terminalserver als "Multihop" eröffnen.

Zunächst installierten wir den Host in der 64-Bit-Variante auf beiden Terminalservern. Nach dem Abnicken der Lizenzbestimmungen meldete die Installationsroutine, dass sie weder in der lokalen noch in einer domänenweiten Richtlinie eine Lizenz finden konnte. Darum wollten wir uns später kümmern und so konnten wir die Meldung getrost ignorieren, die Installation abschließen und die Server neu starten. Das ist nötig, da sich die Software in das Benachrichtigungssystem von Windows einklinkt. Auf unseren Clients unter Windows 10 und dem ersten Terminalserver installierten wir dann den DTC. Hier ist kein Neustart erforderlich, es reicht, den Benutzer einmal ab- und wieder anzumelden.

Welche Variante des Clients die richtige Wahl ist, hängt nicht nur vom Clientbetriebssystem, sondern auch vom verwendeten Remote-Protokoll ab. Microsofts RDP-Clients unterstützen durchgängig 64-Bit. Für Citrix ICA/HDX oder VMware PCoIP ist hingegen die 32-Bit-Variante erforderlich, da diese beiden Hersteller ihre Clients noch nicht auf 64-Bit portiert haben. Im Zweifelsfall lassen sich auch beide Clients parallel installieren, um für alle Anwendungsfälle gerüstet zu sein.

Konfiguration und granulare Steuerung mittels Gruppenrichtlinien

Die zur Konfiguration nötigen Gruppenrichtlinien wollten wir direkt auf unserem Domänencontroller pflegen und installierten dort entsprechend die deviceTRUST-Konsole ebenfalls in der 64-Bit-Variante. Der DTH erhält seine Konfiguration als Group Policy Object (GPO) auf Maschinenebene. Somit platzierten wir unsere Terminalserver in einer separaten OU im Active Directory und erzeugten dann in der GPMC ein neues GPO, das wir mit dieser OU verknüpften. Nun konnten wir das GPO editieren und uns der grundlegenden Konfiguration widmen. Die entsprechenden Einstellungen finden sich unter "Computer Configuration / Policies / Administrative Templates... / deviceTRUST". Dort aktivierten wir die Option "Enable deviceTRUST" und hinterlegten den Schlüssel unserer Eval-Lizenz, die wir vom Hersteller erhalten hatten.

Weiterhin schalteten wir die Funktion "Blacklist of users not controlled by deviceTRUST" für die Gruppe der Domänen-Admins ein, um uns bei den folgenden Tests auf den Terminalservern nicht versehentlich selbst auszusperren. Im Bereich der "Properties" ist zudem die Einstellung "Define host properties to include in MULTIHOP properties" nützlich. Wir aktivierten diese und trugen den Wert "HOST_NAME" ein, sodass auf nachgelagerten Servern in einem Multihop-Szenario sichtbar ist, von welchem System aus eine Sitzung aufgebaut wurde.

Sobald wir auf den Terminalservern die Gruppenrichtlinien aktualisiert hatten, konnten wir uns bereits davon überzeugen, dass deviceTRUST betriebsbereit ist. Wir öffneten dazu von einem unserer deviceTRUST-Clients eine RDP-Sitzung auf dem ersten Terminalserver. Innerhalb der Sitzung fanden wir die von deviceTRUST

bereitgestellten Informationen über die Sitzung an drei Stellen wieder. Zum einen schreibt die Software ihre Informationen in Umgebungsvariablen. Der set-Befehl auf der Kommandozeile förderte eine Vielzahl neuer Variablen zu Tage. Alle Variablen, die mit "DEVICE_" beginnen, liefern Details zu Eigenschaften des Clients, alle mit dem Präfix "HOST_" Informationen zum Server selbst. So konnten wir serverseitig etwa auslesen, wie der Client heißt ("DEVICE_NAME") und ob er Mitglied einer Domäne ist ("DEVICE_DOMAIN_JOINED"). Ist Letzteres der Fall, ermittelt deviceTRUST nicht nur den Namen der Domäne, sondern auch ihre SID, sodass zweifelsfrei feststellbar ist, ob es sich um einen vertrauenswürdigen Client handelt.

Mit den zahlreichen weiteren Variablen geht deviceTRUST ans Eingemachte und erfasst auch, welcher Virenscanner auf dem Client installiert ist, ob dieser aktiv und aktuell ist und ob die Firewall eingeschaltet ist. Weitere Informationen schließen Hardware- und Netzwerkeigenschaften, Zertifikate oder auch die auf dem Client vorhandenen Drucker mit ein. Auch zum Host liefern die Variablen eine geradezu erschlagende Fülle an Eigenschaften. Da wir nicht alle davon wirklich benötigten, konnten wir über die Knoten "Device Filter" und "Host Filter" im GPO, unterteilt in Rubriken wie Hardware, Netzwerk, Drucker und Zertifikate, granular steuern, welche Eigenschaften deviceTRUST erfassen soll. So konnten wir die Liste der Variablen ausdünnen und übersichtlicher gestalten.

Indem all diese Informationen über die Umgebungsvariablen serverseitig verfügbar sind, lassen sie sich nicht nur in Verbindung mit dem DTH, sondern auch in eigenen Skripten und anderweitigen Programmen verwenden. Über die Umgebungsvariablen hinaus bringt deviceTRUST zudem ein eigenes Werkzeug für die Kommandozeile mit: *dtcmd get* liefert ebenfalls die Informationen zur Umgebung zurück.

Weiterhin finden sich diese Informationen im Registrierungsschlüssel "HKCU \ Software \ deviceTRUST \ Properties" wieder. deviceTRUST verknüpft die Properties mittels im GPO definierter Logik zu Kontexten, die mit dem Präfix "CONTEXT_" ebenfalls bei den Systemvariablen sowie in der Registrierung unter "HKCU \ Software \ deviceTRUST \ Contexts" auf dem Host verfügbar sind.

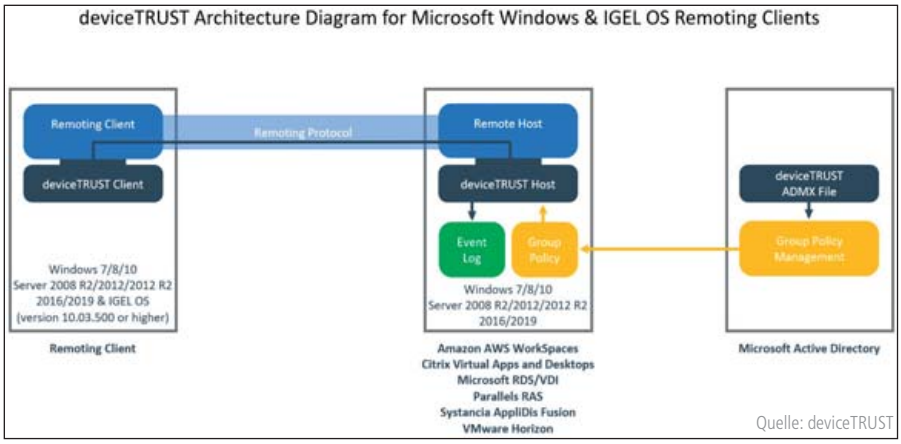

Bild 1: Die Architektur von deviceTRUST: Client und Host kommunizieren über virtuelle Kanäle des Remote-Protokolls.

Ermittlung des Kontexts durch logische Abfragen

Wie genau das Ganze funktioniert, zeigt am besten das lebende Objekt und so begaben wir uns im Gruppenrichtlinien-Editor zum Knoten "Computer Configuration / Policies / deviceTRUST Console". Hier klinkt sich der Editor für deviceTRUST-Regeln ein. Die Startseite begrüßte uns mit den Optionen, wahlweise über das "Sharing"-Symbol oben rechts im Bild einzusteigen oder aber über die prominent mittig auf der Seite platzierten Schaltflächen "Context" und "Actions" eine manuelle Konfiguration zu beginnen.

Das Sharing-Symbol führt zur Aktion "Import from Template" und hilft bei einem schnellen Start. Unterteilt in vier Kategorien für Hardware, Sicherheit, Bildschirmschoner und Leistung bringt es vom Hersteller vorkonfigurierte Kontexte und Aktionen mit. Die lassen sich einfach an eigene Bedürfnisse anpassen. Anschließend exportiert die Sharing-Option diese im JSON-Format in die Zwischenablage und importiert sie von dort auch wieder.

Wir wählten zunächst den Weg der manuellen Konfiguration über die Schaltflächen "Context / Create new Context". So gelangten wir zum grafischen Editor, der beim Bau von Entscheidungsbäumen hilft. Diese verknüpfen logische Operationen zu Pfaden, die anhand einzelner Prüfungen einen Kontext mit einem Wert belegen. Zuoberst auf der Seite gaben wir unserem ersten Kontext den Namen "Sicherheitsstatus" und eine optionale Beschreibung. In der Zeile darunter konnten wir daraufhin erkennen, dass sich der Wert unseres Kontexts später auf dem Host in der Variablen "CONTEXT_SICHERHEITSSTATUS" wiederfinden würde. Der Editor startete mit zwei blau gefärbten Zuständen und einer logischen Operation dazwischen, die wir nun nach unseren Wünschen ausbauen konnten.

Dem Ausgangspunkt gaben wir den Namen "Unbekannt". Die erste Prüfung mit der Regel "Connected Equals True" prüft, ob der DTH erfolgreich eine Verbindung zum Client etablieren konnte. Wir klickten auf das "+"-Zeichen darunter, um eine weitere Prüfung zu ergänzen. Der folgende Dialog erschließt nach Kategorien sortiert sämtliche Informationen, die deviceTRUST vom Client abfragen kann. Wir wählten den Punkt "OS", den wir im nächsten Dialogschritt noch detaillieren konnten. So wollten wir die Eigenschaft "Name" des Clientbetriebssystems abfragen und konfigurierten entsprechend, dass es sich um eine Eigenschaft des Clients ("Device") handelt, die in unserem Fall dem Wert "Windows" entsprechen sollte.

Analog dazu ergänzten wir weitere Abfragen aus der Kategorie "Security Product", mit denen wir erheben wollten, ob auf dem Client Antispyware- und Anti-

virus-Tools sowie die Firewall aktiv sind. Nur wenn diese Abfragen zutreffen, führt der Pfad zum blau hinterlegten Zustand, dem wir den Wert "Sicher" gaben. Oberhalb der Abfrage des Betriebssystems fügten wir dann mit dem Pfeil nach rechts einen weiteren Zustand ein, dem wir den Wert "Unsicher" gaben.

Der standardmäßig eingestellte "Evaluation Mode" mit der Bezeichnung "Left-Most Path" besagt, dass deviceTRUST sich beim Durchlaufen des Entscheidungsbaums nach links orientiert und entlang der zutreffenden Abfragen seinen Weg sucht. Sobald eine Abfrage aber nicht zutrifft, kehrt deviceTRUST zur letzten Abzweigung zurück und biegt nach rechts ab. Ein ähnlicher Kontext, erweitert um Abfragen für verwaltete IGEL-Thin-Clients und iOS-Geräte, ist auch bei den Vorlagen enthalten und leicht an den eigenen Bedarf anzupassen.

Was das nun in der Praxis bedeutet, sahen wir in unseren Benutzersitzungen, sobald wir die Richtlinie gespeichert und auf den Terminalservern aktualisiert hatten. Dort fanden wir in der Registrierung und bei den Umgebungsvariablen den neuen Eintrag "CONTEXT_SICHERHEITSSTATUS" mit dem Wert "Sicher". Sobald wir aber auf unserem Windows-Client den Antivirus-Client oder die Firewall deaktivierten, wechselte der Wert der Variablen in der entfernten Sitzung umgehend zu "Unsicher".

Auch im Ereignisprotokoll des Terminalservers fanden wir detaillierte Informationen zur Benutzersitzung und deren Statusänderungen. deviceTRUST bringt dazu ein eigenes Applikations-spezifisches Ereignisprotokoll mit den beiden Unterordnern "Usage" und "Admin" mit. Ersterer enthält Informationen zur Lizenznutzung. Für den Betrieb interessanter ist der Admin-Bereich mit technischen Informationen zur Funktionsweise der Software.

Ist der Client nicht sicher, wird er ausgesperrt

Dank deviceTRUST konnten wir also bereits Kontextänderungen des Clients auf dem Host erkennen. Nun blieb nur noch, angemessen darauf zu reagieren. Dazu wählten wir in der deviceTRUST-Konsole die Option "Actions / Create new Action" und gaben der neuen Aktion den Namen "Sicherheitsstatus prüfen". Auch die Aktionen konnten wir wiederum einfach grafisch konfigurieren. Sie reagieren auf ausgewählte Kontexte oder auf Trigger. Dabei kann es sich um An- und Abmelden, Trennen oder Wiederverbinden von Sitzungen sowie Kontextänderungen handeln.

Wir definierten zwei einfache Aktionen, die auf alle Trigger ansprechen und unseren Kontext "Sicherheitsstatus" abfragen. Ist dieser unbekannt oder unsicher, soll die Aufgabe "Shell Access / Deny Access" folgen. Hierbei konnten wir eine individuelle Nachricht eingeben, die dem Benutzer angezeigt wird, und eine Zeitspanne festlegen, nach der seine Sitzung automatisch getrennt oder abgemeldet wird. Unsere zweite Aktion bewirkte das Gegenteil. Ist der Sicherheitsstatus sicher, so erlaubt deviceTRUST den Zugriff auf die Sitzung.

Nachdem wir auch dies gespeichert und die Terminalserver aktualisiert hatten, konnten wir uns erneut von den Auswirkungen im Betrieb überzeugen. Mit deaktivierter Firewall oder Virenabwehr verweigerte der Terminalserver die Anmeldung und auch für laufende Sitzungen entfaltete deviceTRUST die gewünschte Wirkung. Sobald wir mit einer angemeldeten Sitzung auf dem Client den Virenwächter abgeschaltet hatten, sperrte der Terminalserver umgehend die Sitzung und präsentierte uns den Hinweis, dass der Client nicht mehr sicher ist. Erst als wir die Sicherheit wiederhergestellt hatten, konnten wir auf dem Server weiterarbeiten.

Die Abfragen konnten wir nun nach Belieben verfeinern und uns dabei aus dem umfangreichen Angebot an Informationen bedienen, die deviceTRUST über Client und Host ermittelt. So prüft die Software etwa auch, ob der Client über Zertifikate mit bestimmten Eigenschaften verfügt, ob es sich anhand von BIOS Seriennummer oder Domänen-Mitgliedschaft um ein dem Unternehmen bekanntes Gerät handelt oder ob das WLAN, in dem sich der Client befindet, ausreichend sicher verschlüsselt ist.

Sicherer Bildschirmschoner

Die möglichen Aktionen leisten aber noch deutlich mehr, als nur pauschal den Zugriff auf eine Sitzung zu erlauben oder zu verbieten. So konnten wir auch innerhalb der Sitzung Änderungen von Zuständen abhängig vom Client auslösen, was wir am Beispiel eines sicheren Bildschirmschoners erfolgreich getestet haben. Dabei hatten wir nicht nur die Sicherheit, sondern auch Komfort für den Endanwender im Sinn. Entsprechend wollten wir die Idee umsetzen, dass innerhalb einer Remote-Sitzung ein Bildschirmschoner mit Passwortschutz nur

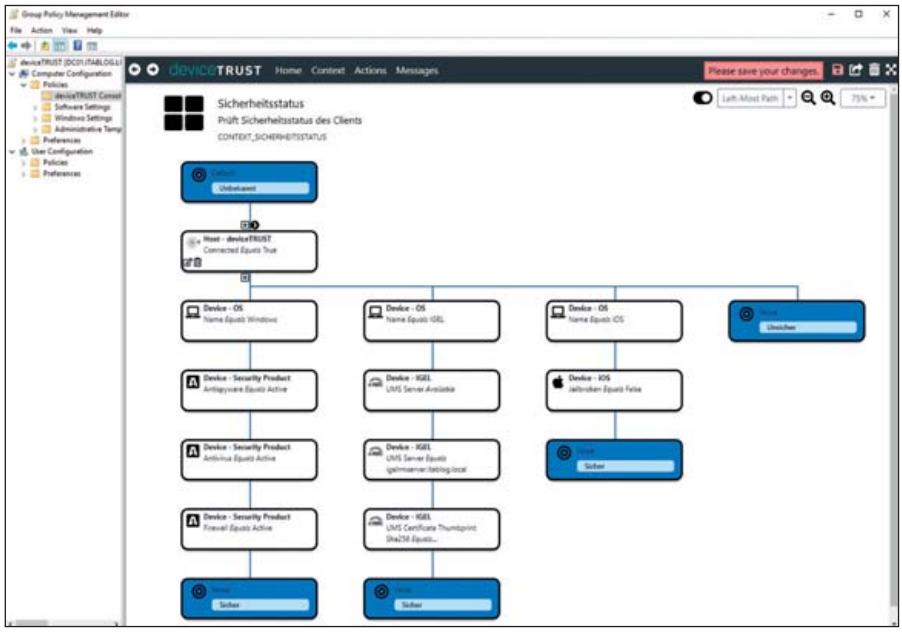

Bild 2: Die deviceTRUST-Konsole erweitert den Gruppenrichtlinienverwaltungs-Editor.

dann notwendig ist, wenn der Client nicht bereits über einen solchen verfügt.

Dazu konfigurierten wir zunächst den neuen Kontext "CONTEXT_SECURE_SCREENSAVER" ausgehend von der Annahme, dass dessen Wert zunächst "False" ist. Im sehr einfachen Entscheidungsbaum kombinierten wir mehrere Abfragen aus der Kategorie "Screen Saver", um festzustellen, ob ein Bildschirmschoner aktiv ist, ob dieser unter "c:\windows\system32" liegt, über einen Passwortschutz verfügt und nach mindestens 900 Sekunden aktiv wird. Nur wenn alle Bedingungen erfüllt sind, sollte der Kontext den Wert "True" bekommen.

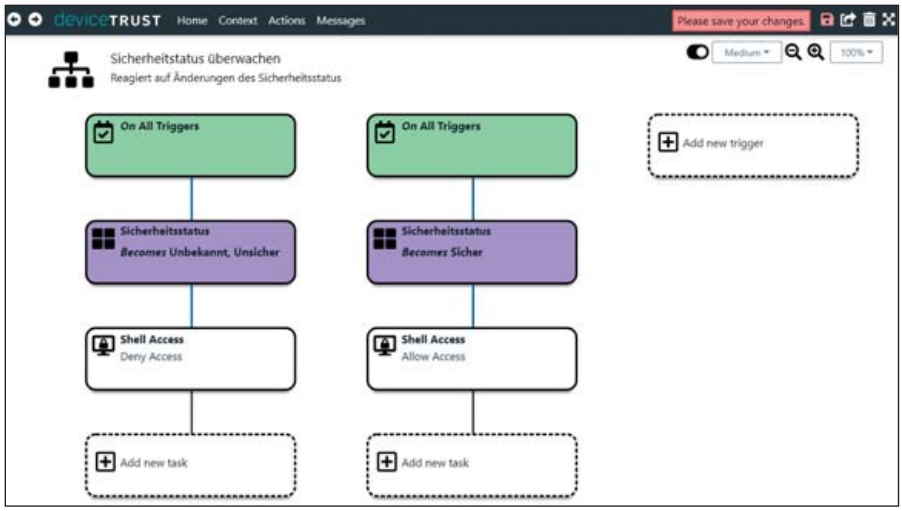

Bild 3: deviceTRUST-Aktionen reagieren dynamisch auf Kontextänderungen und erlauben oder sperren so den Zugriff.

Daraufhin richteten wir eine neue Aktion mit zwei Abläufen ein, um auf den Kontext zu reagieren. Beide Aktionen fragten im ersten Schritt ab, ob es sich bei der Sitzung um einen "Double-Hop" handelt. Nur wenn dies nicht zutrifft, sollte die Aktion weiterlaufen, da auf nachgelagerten Hosts natürlich kein Bildschirmschoner nötig ist, wenn dieser bereits in der ersten Sitzung aktiv ist. Beim Wert "False" sollte die Aktion innerhalb der Sitzung den Windows-eigenen Bildschirmschoner "scrnsave.scr" mit Passwortschutz aktivieren, beim Wert "True" dagegen den Bildschirmschoner wieder abschalten. Auch diese Aufgabe meisterte deviceTRUST mit Bravour. Wir konnten uns davon überzeugen, dass die Software den Bildschirmschoner nicht nur bei der Anmeldung, sondern auch in einer laufenden Sitzung dynamisch passend zum Zustand des Clients änderte. Auf dem zweiten Server, den wir als "Multihop" ansteuerten, passierte wie erwartet nichts.

Integration mit AppLocker sorgt für zuverlässigen Schutz

Zu guter Letzt widmeten wir uns der Integration von deviceTRUST mit Microsoft-AppLocker-Richtlinien. Diese mussten wir zunächst in der Gruppenrichtlinie grundsätzlich aktivieren und konnten anschließend basierend auf Zuständen unserer Kontexte gezielt einzelne Anwendungen sperren oder erlauben. So erzeugten wir einen neuen Kontext, um festzustellen, ob ein Client Mitglied unserer Domäne ist.

Ist dies nicht der Fall, sollte deviceTRUST eine AppLocker-Richtlinie erlassen, die den Zugriff auf eine geschäftskritische Anwendung – in unserem Fall Notepad++ – sperrt und bei Bedarf bereits laufende Prozesse dieser Anwendung terminiert. Für Clients, die der Domäne angehören, sollte diese Beschränkung wieder aufgehoben werden.

Auch in diesem Fall erfüllte deviceTRUST unsere Erwartungen. Wir konnten nachvollziehen, dass wir unsere wichtige Anwendung von fremden Clients aus nicht nutzen konnten. Unser Versuch, deviceTRUST und AppLocker auszutricksen, war erfolglos: So öffneten wir eine Sitzung von einem Domänenmitglied aus und starteten darin Notepad++. Dann trennten wir die Sitzung, um sie von einem fremden Client aus wieder zu verbinden. Doch deviceTRUST ließ sich davon nicht täuschen, erkannte den geänderten Kontext zuverlässig und konfigurierte AppLocker entsprechend. Nach einer Warnmeldung und dem von uns definierten Countdown beendete deviceTRUST dann auch die bereits laufende Anwendung.

Fazit

deviceTRUST setzt neue Maßstäbe bei der Absicherung von Remote-Sitzungen. Während herkömmliche Mechanismen, die auf die Analyse der Clientsicherheit abzielen, diese nur beim initialen Aufbau einer Sitzung prüfen, wirkt deviceTRUST kontinuierlich auch in laufenden Sitzungen und reagiert unmittelbar auf Änderungen. Die Konfiguration mittels Gruppenrichtlinien ist dank des grafischen Editors und der mitgelieferten Vorlagen schnell erledigt. Überzeugt hat uns dabei auch, dass deviceTRUST ohne zusätzliche Infrastruktur auskommt und sich in die marktgängigen Remote-Protokollen integriert. *(jp)*

So urteilt IT-Administrator

Sicherheit bei Kontextänderungen	10
Unterstützung Clientbetriebssysteme	7
Unterstützung Remoteprotokolle	8
Abfragbare Clienteigenschaften	8
Integration in Gruppenrichtlinien	6

Die Details unserer Testmethodik finden Sie unter www.it-administrator.de/testmethodik

Dieses Produkt eignet sich

optimal für Unternehmen mit hohen Ansprüchen an Sicherheit, die mobiles Arbeiten durch virtuelle Desktops und Terminalserver unterstützen.

bedingt für Unternehmen, die weder virtuelle Desktops noch Terminalserver einsetzen, aber den deviceTRUST-Host zur Verwaltung der Fat Clients nutzen möchten.

nicht für Unternehmen, die ihre Clients oder Remote-Sitzungen primär unter Linux betreiben.