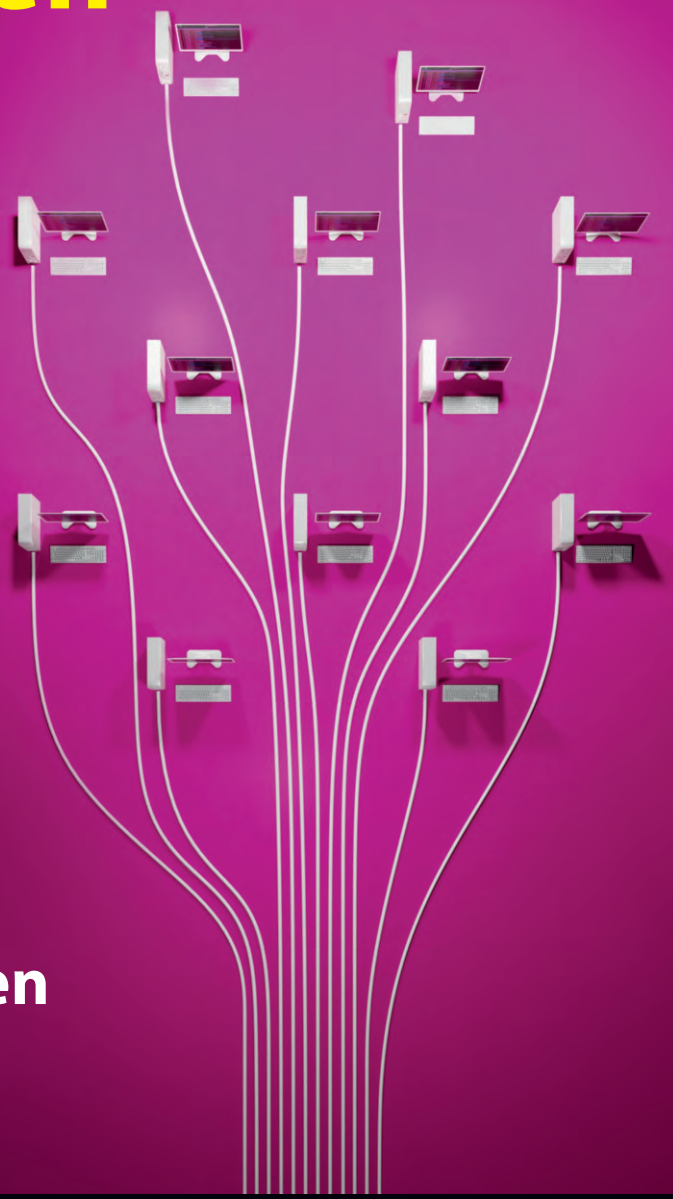
IT Administrator
Das Magazin für professionelle System- und Netzwerkadministration

Problemlose Automatisierung

Logging in Python-Skripten

Fehler schnell aufspüren



E-Mail aufgebohrt

Spike 3.70 im Test

IoT aufbauen

Azure Sphere einrichten

Freies Monitoring

IT mit Graphite überwachen

Lebendiges Lernen

Mit der neuen WAC-Serie stellt Samsung Electronics neue interaktive Displays für die Fortbildung vor. Die Geräte nutzen Android OS, um eine intuitive Bedienung mit anpassbarem Startbildschirm und mehreren Bildschirmen zu ermöglichen. Die WAC-Serie ist in 65, 75 und 86 Zoll erhältlich und verfügt über einen 3-in-1-USB-C-Anschluss. Darüber hinaus erlaubt der HDMI-Ausgang, bei Bedarf das Bild des WAC-Displays auf einen anderen Bildschirm zu übertragen. Der Infrarot-Touch will ein natürliches Schreiberlebnis bieten und 20-Punkt Multi-Touch und die doppelseitigen Stifte, bei denen jede Seite das Schreiben einer anderen Farbe ermöglicht, sorgen für zusätzlichen Komfort. Die Stereolautsprecher an der Vorderseite, ein Stifthalter und die Haltegriffe am Display runden die Features dieser neuen interaktiven Tafeln ab. *(jp)*

Samsung: www.samsung.com/de

Schnell gespeichert

KIOXIA Europe erweitert seine PCIe-4.0-SSD-Laufwerke um die BG6-Serie. Dabei kommt der neue BiCS FLASHTM-3D-Flash-Speicher der 6. Generation des Unternehmens zum Einsatz. Er soll im Vergleich zum Vorgängermodell fast die 1,7-fache Leistung erzielen. So sollen Nutzer mit den Client-SSDs der BG6-Serie die höheren Geschwindigkeiten von PCIe 4.0 erschließen können. Die SSDs sind dabei im Formfaktor M.2 2230 mit höheren Kapazitäten und verbesserter Energieeffizienz verfügbar. Versionen mit einseitigem Formfaktor M.2 2280 sind ebenfalls erhältlich.

Die verfügbaren Speicherkapazitäten reichen von 256 GByte über 512 und 1024 bis hin zu 2048 GByte. An Datentransfers schaffen die SSDs bis zu 6000 MByte/s sequenzielles Lesen und 5300 MByte/s sequenzielles Schreiben bei bis zu 850.000 IOPS (Lesen) und 900.000 IOPS (Schreiben). Die BG6-Serie von KIOXIA wird in der zweiten Jahreshälfte 2023 zur Bewertung für OEM-Kunden als Muster bereitgestellt. *(dr)*

KIOXIA Europe: www.kioxia.com

Kern-Switches

Mit den neuen 7520- und 7720-Universal-Switches weitet Hersteller Extreme Networks seine Universal-Switching-Platform auf den Netzwerk-Core aus. Die neuen Core- und Aggregation-Switches unterstützen auch Fabric Connect und automatisiertes Zero-Touch-Onboarding sowie Auto-Sensing für Geräte von Extreme und Drittanbietern. Daneben sollen sie eine verbesserte Netzwerksicherheit durch Hyper-Segmentierung erlauben. Der 7520-Switch wurde für 1/10/25-GBit-Server und Top-of-Rack-Anwendungen in RZ und Verteilerschränken entwickelt. Die beiden Modelle stehen für 1/10-GBit-Kupfer und 1/10/25-GBit-Glasfaser zur Verfügung. *(jp)*

Extreme Networks:
https://de.extremenetworks.com

Passwort wechsle dich

Keeper Security erweitert sein Privileged Access Management KeeperPAM um die Funktion "Password Rotation". Diese soll es Unternehmen ermöglichen, Dienstkonten und andere privilegierte Anmeldeinformationen automatisch bei Bedarf oder nach einem Zeitplan zu ändern. Die Passwortrotation erlaubt dabei, das Ändern und Zurücksetzen von Systemanmeldeinformationen zu automatisieren – einschließlich Active-Directory-Servicekonten, Azure-AD-Konten, AWS-IAM-Konten, SSH-Schlüssel, Datenbankpasswörter, lokale Windows-, Linux- oder auch Mac-Benutzer. Möglich sind dabei auch Aktionen nach der Rotation, wie etwa der Neustart von Diensten oder das Ausführen bestimmter Anwendungen. Sämtliche Aktionen werden dabei im Advanced Reporting and Alerts Module (ARAM) von Keeper und gegebenenfalls bei SIEM-Drittanbietern protokolliert. Die Passwortrotation von KeeperPAM ist über den Webtresor, die Desktop-App und die Verwaltungskonsole verfügbar. *(dr)*

Keeper Security: www.keepersecurity.com

Spike 3.70

E-Mail in Höchstform

von Thomas Bär

Quelle: eastmanphoto – 123RF

Trotz aller modernen Kommunikationswege hat sich die E-Mail als Lastesel beim Informationsaustausch immer wieder als standhaft erwiesen. Dabei wünschen sich viele Anwender eine bessere Vernetzung in Richtung Echtzeit oder Gruppenfunktionen, ohne jedoch zu weit vom E-Mail-Postfach abwandern zu müssen. Wie das geht, zeigt Spike eindrucksvoll in unserem Test.

Spike ist ein Clouddienst, der sich um eine bestehende E-Mail-Konfiguration legt und diese um eine Vielzahl von Kollaborationsfunktionen erweitert. Im Vergleich zu den vielen anderen Produkten, die wir uns im IT-Administrator angeschaut haben, ist Spike eine Besonderheit. Für Administratoren oder Systemverantwortliche gibt es recht wenig zu tun, sollte es in Ihrem Umfeld zum Einsatz kommen, denn Spike ist viel mehr eine Applikation als ein justier- und steuerbarer Service.

Der augenscheinlichste Unterschied, wenn ein Anwender mithilfe der Spike-Oberfläche auf sein Postfach zugreift, ist die gänzlich andere Darstellung und Organisation der bisherigen Kommunikation. Der Dienst macht aus den E-Mails faktisch einen Chat-Dialogverlauf, ohne dass es sich hierbei um ein neues Mailkonto handelt. Wer das erste Mal mit Spike auf sein Postfach zugreift, sieht folgerichtig die bisherigen Nachrichtenverläufe in einer Chatansicht. Spike leistet jedoch mehr als eine alternative E-Mail-Darstellung. Letztendlich geht es darum, die Zusammenarbeit zwischen Teammitgliedern, Kunden oder anderen Personen zu verbessern – und dies in einer möglichst selbsterklärenden und selbstverständlichen Art und Weise.

Bevor jedoch die dialogbasierte E-Mail, das intelligente Postfach, Videokonferenzen, Aufgabenplanung und der verbesserte Workflow – allesamt Schlagworte des Herstellers – dem Anwender zuteilwerden, gilt es, sich bei Spike anzumelden. Der Anbieter hat drei verschiedene Varianten im Portfolio, die sich in ihren Basisdiensten nur marginal voneinander unterscheiden.

Die kostenfreie Free-Edition, auf die wir hier ein Auge geworfen haben, eignet sich für Benutzer, die nur ein Postfach mit Spike nutzen möchten. Weitere Einschränkungen sind die Größenlimitierung bei zu teilenden Dokumenten und die Anzahl gleichzeitiger Personen in einem Videoanruf. Free-User können nur 1:1-Videositzungen machen, Nutzer der Pro-Edition sind maximal zu fünft und die Business-Variante erlaubt Videobesprechungen mit insgesamt zehn Personen. Die Suchfunktion von Spike in der kostenfreien Version durchsucht nur die Elemente der letzten 60 Tage, eine Einschränkung, die in den kostenpflichtigen Versionen nicht existiert.

Einfache Verknüpfung von Mail und Spike

Sofern keine kostenpflichtige Anbindung gewählt wurde, besucht der künftige Spike-Nutzer mit dem Browser die Webseite des Herstellers und klickt auf "Leg los". Anschließend die E-Mail-Adresse und das Passwort eingeben, und schon ist die kostenfreie Edition startklar.

Sofern bei der Benutzeranmeldung alles direkt glattgeht, muss sich der Anwender überhaupt nicht um irgendwelche Einstellungen kümmern. In unserem Test verwendeten wir ein bei 1&1/IONOS gehostetes Mailkonto. Allein durch die Eingabe der Mailadresse erkannte Spike, dass es sich um ein Konto bei ebendiesem Anbieter handelt. Einen Schritt weiter gilt es lediglich noch, das dem Benutzer bekannte Passwort einzugeben, das auch beim regulären Zugriff auf den Webclient des jeweiligen Anbieters zu nutzen ist. In einem weiteren Test verwendeten wir ein Mailkonto, das auf einem lokal betriebenen Exchange-Server ohne extern erreichbare Web-App arbeitet. In diesem Fall schlug Spike vor, den Zugriff per IMAP durchzuführen. Ein kleines Fragezeichen mit einem umrandeten Kreis um das Wort IMAP bringt den Anwender auf eine Liste der verschiedenen Anbieter, die mit Spike zusammenarbeiten.

Neben Google, Yahoo!, iCloud, Office 365, Outlook, Microsoft, Exchange, Mail.ru und AOL findet sich so auch der

im Vergleich eher generische Eintrag IMAP. Öffnet der Anwender die Auswahl unter "Erweiterte Einstellungen", kommen die klassischen Detailinformationen wie IMAP-Server, Port, Username, SMTP-Server, SMTP-Port und SMTP-Benutzername zum Vorschein. Erwartungsgemäß ließ sich Spike nicht mit einem lokal arbeitenden Exchange-Server nutzen, da dieser nicht entsprechend konfiguriert war. Auch bei der Auswahl "Exchange" scheiterte die Anbindung mit "Auto-discover failure". Selbiges geschah in unserem Test mit einem bei 1&1/IONOS gehosteten Exchange-Postfach. Dies ließ sich jedoch innerhalb einer Minute korrigieren, indem wir die spezifischen Verbindungsdaten, die nicht für Auto-Discover gepflegt waren, manuell eingaben.

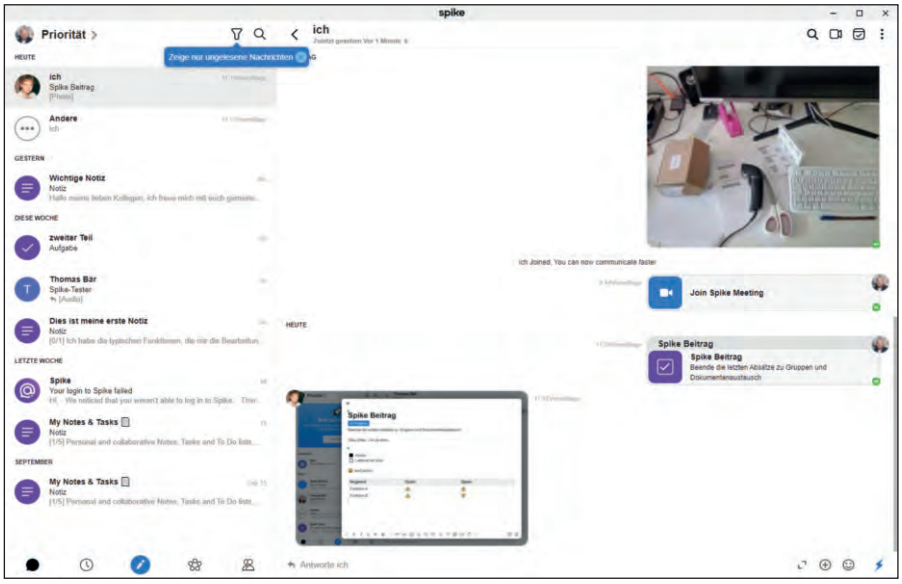


Bild 1: Mit Spike wird aus einem regulären E-Mail-Konto ein Chat-Message-Stream, in dem sich viele Aufgaben leichter und übersichtlicher erledigen lassen.

Zusammenfassend darf der Prozess der Anbindung von Spike an den jeweiligen Mailservice als weitgehend einfach bezeichnet werden. Nutzen Anwender die üblichen E-Mail-Anbieter, beschränkt sich der Vorgang auf die Eingabe von E-Mail-Adresse und Passwort. In den anderen Fällen benötigen die Mitarbeiter gegebenenfalls Unterstützung bestehend aus den entsprechenden Konfigurationsdetails. Anschließend führt Spike die Benutzer in der Webseite weiter und verschont diese mit weiteren Konfigurationsdetails. Lediglich eine Passwortänderung des Benutzers in der grundlegenden Mailkonfiguration könnte sich auf Spike auswirken.

Spike 3.70

Produkt
Onlinedienst für eine integrierte Kommunikation via E-Mail, Chat und Videoanrufe auf Basis des bestehenden E-Mail-Verkehrs.

Hersteller
Chatflow Ltd.
www.spikenow.com

Preise
Spike wird in drei Varianten angeboten. In der kostenfreien Free-Edition sind alle Funktionen nutzbar, beschränkt auf ein Postfach für den Benutzer, eine auf 60 Tage limitierte Suchfunktion und 1:1-Videoanrufe. Die Pro-Edition erlaubt einen Datei-Upload bis zu 100 MByte, 5 GByte Speicherplatz pro Benutzer, Fünfer-Videokonferenzen und bis zu drei E-Mail-Adressen pro Benutzer für monatlich rund 6 Euro. Die Business-Edition bietet eine unlimitierte Anzahl von E-Mail-Adressen je Benutzer, 20 GByte Speicher, 1 GByte-Uploads und Zehner-Videocalls für rund 12 Euro pro Monat.

Systemvoraussetzungen
Spike ist ein cloudbasierter Dienst und benötigt lediglich einen Browserzugriff. Alternativ gibt es eine gesonderte Desktop-App für Windows, macOS und eine Mobile-App für Android und iOS.

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

Die Postbox einmal ganz anders

Direkt nach der Anmeldung gibt Spike den neuen Blick auf das Postfach frei. Die Optik ist zeitgemäß, modern und frisch mit klarer Struktur. Am oberen linken Bildrand thront ein Hinweis mit einer stilisierten Rakete und dem Hinweis "Meet your new Inbox". Klickt der Anwender auf den verbundenen Link, bekommt er eine auf Englisch verfasste Beschreibung mit einem kleinen Videoclip angezeigt. In dieser Info wird erklärt, dass die Organisation der E-Mails durch Spike nicht nach Empfangszeitpunkt oder Betreffzeile stattfindet, sondern ganz in Chat-Client-Manier nach dem Absender.

Gesprächsverläufe sind so, zumindest nach unserer Einschätzung, deutlich angenehmer zu lesen. Wem eine solche Ansicht doch nicht zusagt, kann jederzeit in den Einstellungen wieder eine andere Organisationsform wählen. Während die regulären Dialoge oder auch die Einstellungen auf Deutsch verfasst sind, kommen Erklärungstexte oder die Datenschutzvereinbarungen ausschließlich in englischer Sprache daher. Sicherlich dürfte die Mehrzahl der Administratoren den kleinen Wechsel kaum bemerken, sofern sie Englisch verstehen. Sollten jedoch Anwender über keine oder nur rudimentäre Englischkenntnisse verfügen, könnte dies zu einer Herausforderung werden. Etwas ärgerlich ist in diesem Zusammenhang, dass die deutschsprachige Wissensdatenbank, die sich als Option nach einem Klick auf das Fragezeichensymbol anbietet, auf eine Fehlerwebseite führt – diese ist zwar mit einer entsetzt schauenden Katze hübsch dargestellt, aber dann doch keine Hilfe.

Noch einmal zurück zu den Einstellungen, die sich auf den ersten Blick gar nicht so leicht finden lassen: Anstelle des üblichen "Hamburger"-Menüs mit den drei oder vier Streifen sind viele Befehle bei Spike über einen Rechtsklick beziehungsweise Langklick auf das persönliche Profilbild zugänglich. Die meisten der Einstellungen erklären sich von selbst und bestehen zumeist nur aus einem An- beziehungsweise Ausschalter. Beispielsweise die Einstellung, dass Spike E-Mails von unbekannten Personen in einen separaten Feed auslagert, um die Priorität auf die wichtigen und vom

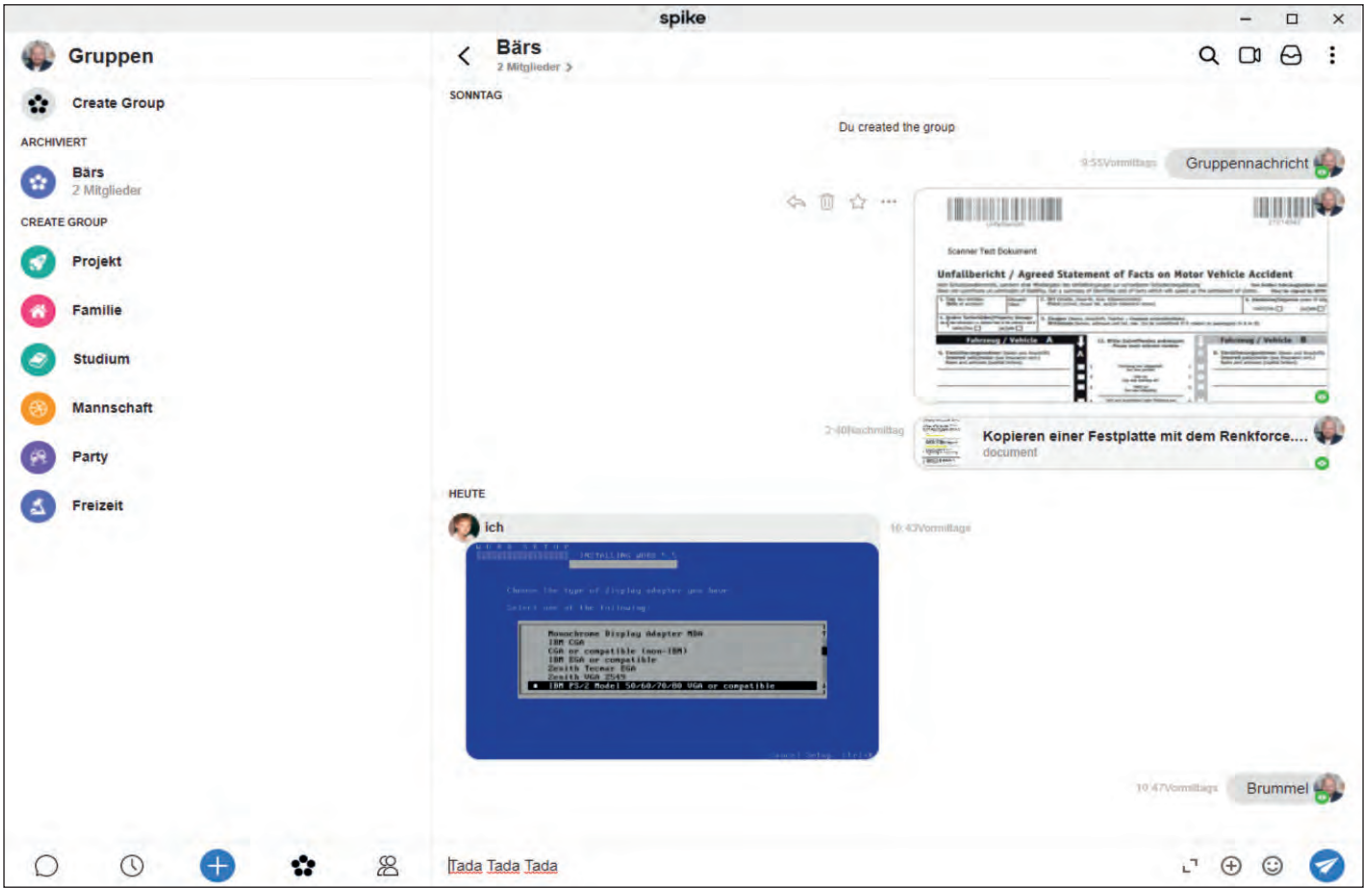


Bild 2: Mithilfe der Gruppenfunktion bietet Spike die einfache Verteilung von Nachrichten, Dateien und Aufgaben.

Anwender initiierten oder bereits bearbeiteten Dialoge zu lenken. Die Anpassbarkeit für die Arbeitsweise umfasst auch die Einstellungen zum "Snoozing" oder für das zeitgesteuerte Versenden von Nachrichten. Damit können Benutzer das Abarbeiten von Dateien, Aufgaben, Erinnerungen und Nachrichten auf einen späteren Zeitpunkt verschieben. In der Grundeinstellung gibt es "Morgen" und "Nächste Woche", aber auch, wie bereits geschrieben, die Möglichkeit, dies anzupassen.

Sofern Spike vom Benutzer im Browser geöffnet wurde und die entsprechende Zustimmung noch nicht vorliegt, erscheint am oberen Fensterrand ein blauer Balken mit dem Hinweis, dass Spike noch die Erlaubnis benötigt, um Desktop-Benachrichtigungen auszugeben. Der Klick auf das Kommando empfiehlt sich, da so Informationen wie eingehende Videoanrufe, das Eintreffen von Nachrichten oder Aufgabenhinweise die Aufmerksamkeit des Benutzers auf Spike lenken.

Es kommt bei der Verwendung von Spike als alternativen Mailclient wahrlich ein Chatgefühl auf. Beginnt der Gesprächspartner mit dem Verfassen einer Nachricht in Spike, wird dies durch "hüpfende drei Punkte" beim Empfänger symbolisiert. Der Empfänger weiß somit, dass alsbald eine Nachricht eintreffen wird. Wie die meisten modernen Mailprogramme zeigt auch Spike den Anhang gleich als Vorschau an. Diese Vorschau, gepaart mit der schnellen Übersicht als Chat-Stream, macht die Suche nach Anhängen einfacher. Leider hängen manche Firmen ihr Logo als PNG-Datei in den Footer jeder E-Mail. Diese Grafik prangt dann unter jedem Chatnachrichten-Element – auch unter einer Antwort, wenn diese den ursprünglichen Footer enthält.

Nur um es noch einmal hervorzuheben: Wenn jemand als Anwender Spike nutzt und ein anderer Kommunikationspartner oder ein Gruppenmitglied nur über einen herkömmlichen Mailclient verfügt, erhält dieser dennoch alle Informationen als gewöhnliche E-Mail, angereichert mit einigen Hinweisen auf das Programm Spike. Der E-Mail-Absender lautet auf den Namen, die E-Mail-Adresse stets auf "spike.group" mit einer kryptischen, führenden Buchstaben- und Zahlenfolge.

Gruppen, Videoanrufe und Dokumentenaustausch

Glücklicherweise beschränkt sich Spike nicht allein auf E-Mails. Das Gruppenfeature erlaubt, beliebige Empfänger als eine Gruppe zu deklarieren, um so E-Mails oder Dateien mit einer größeren Empfängergruppe zu teilen. Aufgabeneinträge und deren Bearbeitung in Spike gehen deutlich über die Funktionalität einer E-Mail hinaus, auch wenn diese Aktivitäten für einen Empfänger ohne Spike-Oberfläche wieder als E-Mail einsehbar sind. Eine Notiz kann verschiedene Elemente wie beispielsweise Checkboxen, Absätze, Texte oder Tabellen enthalten und lässt sich per Mausklick mit anderen Anwendern teilen. Nutzen diese ebenfalls Spike, sehen sie über farblich stilisierte Schreibmarken die Eingaben und Aktivitäten der anderen User. Ohne es jetzt übertreiben zu wollen, ergibt sich mithilfe der To-do-Listen und der Echtzeitbearbeitung ein ganz einfaches Werkzeug für das gemeinsame Projetmanagement.

Die Anlehnung an die Chatwelt geht noch weiter. Wie von Diensten wie WhatsApp bekannt, sind Benutzer in der Lage, Sprachnachrichten aufzunehmen und diese zu versenden oder auch Videoanrufe durchzuführen und dies je nach Edition auch in einer Gruppe. Auf der Roadmap von Spike steht die Möglichkeit, in Zukunft auch Videos aufzeichnen zu können.

Bild 3: Spike erlaubt auch die gleichzeitige Bearbeitung von Notizdokumenten – der kleine umgedrehte grüne Tropfen ist die Schreibmarke eines Mitwirkenden.

Markierungen helfen bei der Suche

Was uns im Test besonders gut gefiel und wir darum hervorheben möchten, ist das Tagging. Was selbst die ältesten Apple-Computer bereits in den 1990er Jahren unterstützten, fand erst langsam Einzug in andere Programme. Alle Elemente in Spike, ob es nun eine E-Mail ist oder eine Notiz, kann der Benutzer mit einem beliebigen "Tag" versehen – ähnlich der Kategorien in Outlook. In der Suchfunktion kann sich der Benutzer relativ rasch einen Überblick über die vorhandenen Elemente gemäß der Markierung mit dem "Tag" machen. Eine echte Bereicherung für den hart gesottenen Information-Worker, der nicht unbedingt mit den typischen Stichworten arbeiten kann. Gute Tag-Bezeichnungen wären Kunden-, Vorhaben- oder Projektkürzel.

Da die Suchfunktion vom 60-Tage-Limit der Free-Edition betroffen ist, zeigt sich zumindest die kostenfreie Version diesbezüglich eingeschränkt. Leider synchronisiert Spike zudem die vergebenen Tags nicht an andere Teilnehmer, auch nicht innerhalb einer Gruppe. Eine solche übergreifende Markierung – also eine Art Gruppen-Tagging von Inhalten – wäre noch das i-Tüpfelchen bei der Zusammenarbeit.

Keine direkte Konkurrenz

Die Überarbeitung der Arbeitsweisen "E-Mail / Chat / Kollaboration" ist keine Idee, die nur die beiden israelischen Spike-Gründer Dvir Ben-Aroya und Erez Pilosof im Jahr 2014 allein hatten. Hinter dem Dienst verbirgt sich die israelische Firma Chatflow. Auch andere Firmen haben sich dem Thema verschrieben. Der bekannteste Konkurrent dürfte Slack sein, der jedoch an sich auf E-Mail, integrierte Notizen und eine traditionelle Inbox mit "Priorität" verzichtet. Gmail von Google bietet ebenfalls eine gewisse Ähnlichkeit, führt aber E-Mail und Chat nicht in dieser Form zusammen. Spark und Outlook bieten ebenfalls nicht die genaue Mischung, die Spike für sich in Anspruch nimmt.

Auch wenn es nicht mehr unüblich ist, seine Zugangsdaten an zig Stellen im Internet einzugeben, bleibt doch auch gern ein mulmiges Gefühl: Wer hat Zugang zu meinen Daten? Spike speichert den eigenen Angaben nach alles mit einer AES-Verschlüsselung mit 256 Bit in Form eines privaten Schlüssels pro Nachricht. Die Sicherheit und Privatsphäre der Daten ist im Fall von Spike der grundlegende Unternehmenswert und daher sei den Mitarbeitern der Zugriff auf Benutzerdaten untersagt. Laut den Informationen auf der Webseite nutzt Spike technische Kontrollmechanismen und Auditrichtlinien, um sicherzustellen, dass alle Zugriffe protokolliert sind. Bei Nichtnutzung löscht Spike die Daten nach 60 Tagen und über eine E-Mail an die Adresse "chat@spikenow.com" könne dieser Vorgang manuell initiiert werden. Wir wollten Spike noch ein wenig länger im Einsatz behalten und nutzten die Möglichkeit nicht.

Fazit

Für uns war die Betrachtung von Spike insgesamt eine freudige Angelegenheit. Es war spannend zu sehen, wie die Basisfunktion – das eigene Postfach – sich in wenigen Minuten zu einem deutlich moderneren Hybridsystem von E-Mail, Videotelefonie und Gruppenchat zu verwandeln wusste. Die Funktionen sind allesamt bekannt, es gibt nichts wirklich Neuartiges, aber es ist mit einer großen Portion an Pfiffigkeit angereichert worden, sodass alle Arbeitsschritte flüssiger von der Hand gehen. Dies erweist sich als äußerst praktisch, wenn Nutzer in ansonsten getrennten Kommunikationssystemen zusammenarbeiten. *(dr)*

So urteilt IT-Administrator

Einbindung des Mailaccounts	8
Organisation der E-Mails	8
Videotelefonie	6
Gruppenfunktionen	6
Flexibilität der Nutzung	9

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

Dieses Produkt eignet sich

optimal für kleine und große Teams, die in verschiedenen Kommunikationssystemen organisiert zusammenarbeiten, ohne ihre Systeme anpassen zu wollen.

bedingt für Einzelpersonen, die bereits einen modernen Webclient für den Zugriff nutzen.

nicht für Firmen, die ihre Arbeitsgruppen kontrolliert in eigenen Umgebungen organisieren wollen oder müssen.

Azure Sphere für IoT-Anwendungen

Alles vernetzen

von Dr. Christian Knermann

Mit Azure Sphere hat Microsoft eine Plattform geschaffen, um Mikrocontroller ins Internet of Things zu integrieren. Dabei liegt der Fokus auf der Sicherheit und umfasst nicht nur die Referenzarchitektur für die Mikrocontroller selbst, sondern auch deren Betriebssysteme sowie einen Cloudservice, der sich um Updates kümmert. IT-Administrator hilft bei ersten Gehversuchen mit Azure Sphere.

Der große Trend der Digitalisierung bestimmt zunehmend das Geschäftsleben. Oftmals hängt der wirtschaftliche Erfolg an den Fähigkeiten eines Unternehmens, Prozesse in Produktion und Logistik digital abzubilden und zu optimieren. In diesen Kontext passen das Internet of Things (IoT) und dessen professioneller Ableger, das Industrial IoT (IIoT), als Untermengen des Obergriffs Digitalisierung. Das IoT stellt die Verbindung zwischen Informationstechnik und realer Welt her, schafft also die Grundlage dafür, dass Software und künstliche Intelligenz (KI) zur Anwendung kommen können.

Damit eine KI die physische Welt erfassen, messen, daraus Schlüsse ziehen und auf die reale Ebene zurückwirken kann, braucht es in Software realisierte digitale Zwillinge von physischen Anlagen. Und umgekehrt benötigen die Maschinen in der echten Welt Sensoren, die Daten erheben, sowie Aktoren, mittels derer Software ins Geschehen eingreifen kann.

Elemente des IoT

Dies stellt IT-Administratoren vor neue Herausforderungen, denn es drängen Geräte ins Netzwerk, die nach anderen Spielregeln als das klassische Client-Server-Betriebsmodell verlangen. Sahen sich Admins bislang schon mit einer Vielzahl an Clients und Servern konfrontiert, kommen bei einer vollständigen Digitalisierung von Maschinen und Anlagen schnell sehr viele neue Endpunkte hinzu.

Typischerweise handelt es sich dabei um Mikrocontroller (Micro Control Unit, MCU). Die kleinsten Vertreter dieser Gattung sind etwa die Geräte der Produktfamilien espressif ESP8266 sowie ESP32, die wir im Rahmen eines früheren Workshops für erste Gehversuche im IoT nutzten [1]. Ebenfalls beliebt sind die Systeme der Arduino-Familie. Es handelt sich dabei um eine quelloffene MCU-Plattform, die sich mittels einer Entwicklungsumgebung (der ebenfalls quelloffenen Arduino-IDE) in C++ programmieren lässt. Sowohl Hard- als auch Software stehen unter einer Open-Source-Lizenz.

Dabei ist der eigentliche Controller meist auf einem Entwicklungsboard verbaut, das direkt per USB-Anschluss an ein System mit installierter IDE findet. Ein solches Board vereint Prozessor, Speicher- und Timer-Bausteine, Digital-Analog-Wandler sowie Konnektivität per WLAN oder Bluetooth. Zum weiteren Ausbau der Funktionalität kommen als Hat oder Shield bezeichnete Erweiterungsplatinen zum Einsatz, die auf das Board aufgesteckt etwa Sensoren, Aktoren oder kleine Displays ergänzen. Alternativ sind sogenannte Grove Shields erhältlich. Ein solches Shield führt die Anschlüsse des MCU auf einfache Steckplätze heraus, sodass sich alle weiteren Komponenten ganz ohne Löten mittels standardisierter Steckkabel verbinden lassen.

MCU suchen örtlich begrenzt per WLAN oder auf größere Distanzen per LoRaWAN Anschluss ans Netz. Auf der Ebene der logischen Datenübertragung sei der Klassiker HTTP erwähnt, auf dessen Basis der Representational State Transfer (REST) mittels gängiger HTTP-Methoden wie GET und POST die Kommunikation von und mit IoT-Gerätschaften ermöglicht. Sehr verbreitet im IoT sind weiterhin Protokolle wie etwa das Advanced Message Queuing Protocol (AMQP) und Message Queuing Telemetry Transport (MQTT). Letzteres kommt mit minimalem Overhead aus und ist darauf optimiert, Geräten mit wenig Ressourcen eine zuverlässige Datenübertragung auch über eher unzuverlässige Netze zu erlauben.

MQTT folgt dem Publisher-Subscriber-Funktionsprinzip. Clients können dabei

Nachrichten mit einem bestimmten Betreff (Topic) versenden, während andere Clients diese abonnieren, so wie etwa von sozialen Netzwerken her bekannt. Im Zentrum der Kommunikation steht ein MQTT-Broker, der die Daten entgegennimmt und sich um das Verteilen von Nachrichten im Push-Verfahren kümmert.

Azure Sphere für IoT im Unternehmen

Aufgrund des einfachen Einstiegs sind Arduinos und vergleichbare MCU auch im Consumer-Bereich beliebt, doch bringt das Entwickeln auf dieser Plattform Herausforderungen mit sich, sobald ein Projekt auf Produktionsmaßstäbe skalieren soll. Was ist zu beachten, wenn nicht mehr wenige Prototypen, sondern hunderte oder gar tausende an Endpunkten Ihr Netzwerk bevölkern und geschäftskritische Prozesse nicht nur messen, sondern auch beeinflussen können?

In diesem Fall ist die flächendeckende Verteilung von Firmware- und Anwendungsupdates an die Endpunkte ebenso sicherzustellen wie eine Absicherung der Kommunikation sowie Authentifizierung und Autorisierung. Diese zeitraubende Tätigkeit bringt in der Regel die Beschäftigung mit Sicherheitskonzepten wie zertifikatsbasierter Verschlüsselung mit sich und wird auf dem schnellen Weg vom Entwicklungsstadium zum produktiven Einsatz leider nur zu oft nicht oder nur unzureichend berücksichtigt. An dieser Stelle setzt Microsoft mit Azure Sphere an [2].

Mit Azure Sphere hat Microsoft eine eigene Plattform für das IoT geschaffen, die weit mehr als die Hardware der MCU umfasst. Azure Sphere verknüpft die drei Elemente Hardware, Software und Clouddienst. Dabei ist die Sicherheit keine optionale Ergänzung, die später mühsam nachgerüstet werden muss, sondern von Anfang an integraler Bestandteil der Plattform.

Zertifizierte Hardware erforderlich

Der erste wichtige Baustein ist ein für Azure Sphere zertifizierter MCU. Microsoft betätigt sich allerdings nicht selbst als Hardwarehersteller, sondern arbeitet mit Drittanbietern zusammen, die Geräte nach Spezifikation fertigen. So finden sich am Markt bereits einige kompatible Geräte verschiedener Hersteller [3]. Herzstück aller Geräte bildet der Prozessor MT3620 des Herstellers MediaTek mit der Prozessorarchitektur ARM32 und insgesamt fünf Kernen (Bild 1).

Einer dieser Kerne ist Microsofts Sicherheitschip Pluton, der grundsätzlich vergleichbar ist mit einem Trusted Platform Module (TPM) und doch deutlich über dessen Funktionen hinausgeht. Pluton fungiert als Hardware-Vertrauensanker (Root-of-Trust) sowie als Zufallszahlengenerator, sichert den MCU mittels Secure-Boot ab und implementiert Verschlüsselungsfunktionen. Der Chip sorgt dafür, dass ein Gerät keinen nicht signierten Code ausführt und dass mittels Remote-Attestation in Zusammenarbeit mit dem Azure-Sphere-Cloud-Service die Integrität eines jeden Geräts gewährleistet bleibt. Jeder MCU ist dazu weltweit eindeutig identifizierbar.

Der MCU verfügt weiterhin über einen ARM-Cortex-A-Kern, der auf besonders niedrigen Stromverbrauch ausgelegt ist, sowie zwei ARM-Cortex-M Kerne. Letztere sind für Funktionen der Echtzeitsteuerung optimiert. Der MCU kann sämtliche Anschlüsse für Peripherie (siehe Kasten "Schnittstellen") jeweils exklusiv einem dieser Kerne zuordnen und sicherstellen, dass von diesem Kern ausgeführter Code nicht auf die anderen Kerne zugreifen kann. Zu guter Letzt implementiert ein separater Kern das WLAN-Subsystem. Der MCU funkt darüber im Dual-Band nach den Standards 802.11a/b/g/n. Der MT3620 isoliert alle Sicherheitsfunktionen und das WLAN von jeglichem Code der Endanwender.

Linux-Betriebssystem und lokale Apps

Zertifizierte MCU bringen ab Werk das Azure Sphere OS mit. Es handelt sich dabei um ein von Microsoft angepasstes und quelloffenes Linux-System mit einem gehärteten Kernel, speziell optimiert für IoT-Anwendungen. Das Betriebssystem ist entsprechend auf die notwendigsten Funktionen reduziert und verfügt weder über eine Shell noch über einen Paketmanager.

Anwendungscode in Form eines kompilierten Images befördern Sie entweder lokal per USB auf ein Gerät oder liefern Ihre Images über den Azure Sphere Security Service in der Cloud aus (Firmware-Over-the-Air, FOTA). Letzteres ist bei einer größeren Anzahl an Geräten der Weg der Wahl, zumal die Endpunkte im produktiven Einsatz oftmals geografisch verteilt, in Anlagen fest verbaut und physisch gar nicht mehr zugänglich sind. Die direkte

Schnittstellen

Mikrocontroller verfügen über eine Vielzahl an Schnittstellen und Bausteinen, um mit Sensoren, Aktoren und anderer Peripherie zu interagieren:

- GPIO (General Purpose Input / Output): Nicht zweckgebundene Schnittstelle zur bidirektionalen Übertragung digitaler Signale, Funktion vollständig in Software definierbar. Jeweils ein Pin bildet dabei die Werte 0 und 1 ab – im einfachsten Fall etwa, um eine LED aufleuchten zu lassen (Output) oder die Stellung eines Schalters abzufragen (Input).
- PWM (Pulse Width Modulation): Generiert variable analoge Signale, geeignet etwa zur Steuerung von Motoren oder der Helligkeit von Leuchtdioden.
- TDM (Time-Division Multiplexing): Digitale Schnittstelle zur Übertragung multipler Datenströme in einem einzelnen Signal.
- I2S (Inter-integrated Sound): Digitaler serieller Bus zur Übertragung von Audiosignalen.
- UART (Universal Asynchronous Receiver/Transmitter): Bidirektionale serielle Schnittstelle zur Kommunikation mit angeschlossenen Geräten, etwa zur klassischen Terminalverbindung mit einem per USB angeschlossenen PC zu Debugging-Zwecken. Zudem gibt es auch Sensoren, die per UART kommunizieren.
- I2C (Inter-integrated Circuit): Bidirektionale serielle Kommunikation ähnlich UART, typischerweise genutzt für Module und Sensoren.
- SPI (Serial Peripheral Interface): Ebenfalls genutzt zur bidirektionalen seriellen Kommunikation, schneller als UART und I2C.
- ADC (Analog-to-Digital Converter): Dieser Baustein wandelt analoge Signale in digitale um, kommt etwa für Sensoren zur Messung von Temperatur, Luftfeuchte oder elektrischer Spannung zum Einsatz.

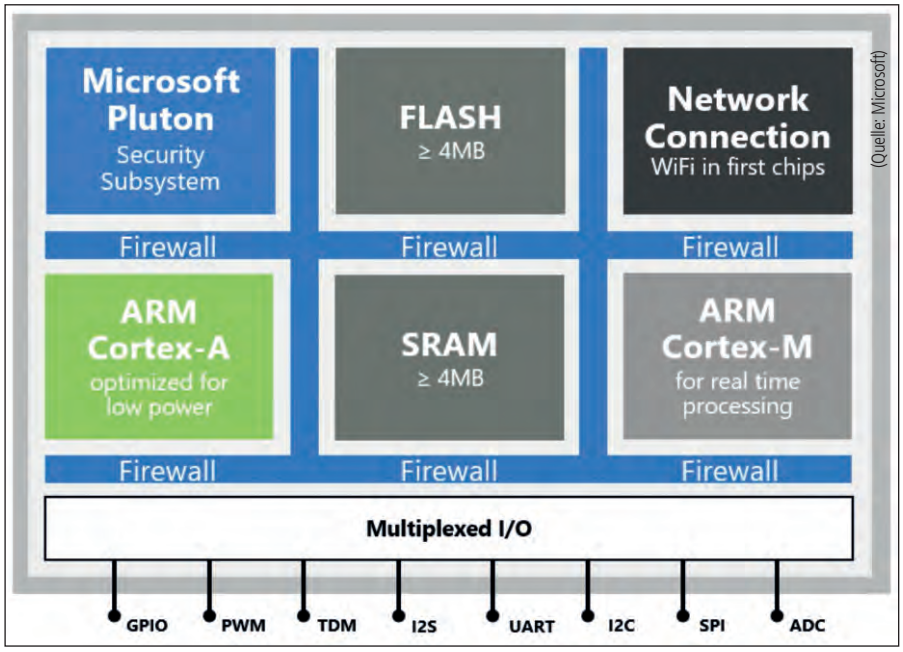


Bild 1: Der MediaTek-MT3620-Mikrocontroller verfügt über fünf Prozessorkerne nebst SRAM.

Übertragung von Images per USB wird auch als Side-Loading bezeichnet, doch auch dies funktioniert nur, wenn das jeweilige Gerät beim Cloudservice registriert ist.

Microsoft unterscheidet zwischen High-Level-Apps, die auf dem A-Kern der MCU laufen, und Low-Level-Apps, die die M-Kerne nutzen [5]. Eine High-Level-App läuft im Benutzerkontext und darf nur definierte Bibliotheken und API-Funktionen verwenden. Sie vermittelt zertifikatsbasierte authentifizierte Verbindungen zwischen Geräten und der Cloud, interagiert mit den Schnittstellen, wie GPIO oder UART, und kommuniziert mit Low-Level-Apps. Letztere greifen direkt oder über ein zusätzliches Echtzeit-Betriebssystem (Real-Time Operating System, RTOS) auf die Hardware zu. Jede Low-Level-App läuft vollständig isoliert und kann nicht direkt, sondern nur über eine High-Level-App mit der Außenwelt interagieren.

Updates via Clouddienst

Der Azure Sphere Security Service kümmert sich um die Remote-Attestation, stellt also sicher, dass es sich um einen echten und unverfälschten MCU mit integrem und aktuellem Betriebssystem handelt. Der Cloudservice sichert mittels Authentifizierung auf Basis von Clientzertifikaten die Kommunikation von Gerät zu Gerät sowie vom Gerät zur Cloud. Den Kontakt zum Clouddienst benötigen Geräte nur für Updates von Betriebssystem und Anwendungsabbildern. Abseits davon funktionieren die Geräte auch im Offlinebetrieb.

Betriebssystem und Cloudservice sind im Kaufpreis des MCU enthalten. Mit jedem Controller erhalten Sie zehn Jahre lang Unterstützung und Updates für das Betriebssystem und ebenso lang das Recht zur Nutzung des Clouddiensts. Darüber hinaus entstehen Ihnen keine weiteren Kosten – sofern Sie in Verbindung mit Azure-Sphere-Geräten nicht Microsofts hauseigene Azure-IoT-Dienste nutzen, für die Microsoft separate Gebühren berechnet. Doch Azure IoT Central und der Azure IoT Hub sind optional. Der Azure Sphere Security Service läuft zwar innerhalb der Azure Cloud, Sie können jedoch Ihre Azure-Sphere-Geräte zur Steuerung und weiteren Verarbeitung von Daten mit jedem beliebigen Dienst in einer öffentlichen oder privaten Cloud integrieren und etwa Daten per MQTT an einen Broker in Ihrem lokalen Netz schicken.

Loslegen mit einem Entwicklungsboard

Die meisten Code-Beispiele für Einsteiger beziehen sich auf das Azure Sphere MT 3620 Development Kit des Herstellers Seeed Technology [6]. Dieses Entwicklungsboard bringt zwei integrierte WLAN-Antennen sowie zwei Anschlüsse für externe Antennen mit. Es verfügt außerdem über zwei Funktionstasten, eine Reset-Taste, mehrere Status-LED und einen Micro-USB-Anschluss, der die Stromversorgung und eine Schnittstelle für Programmierung und Debugging bereitstellt (Bild 2).

Die Anschlüsse für Peripheriegeräte sind über zwei Reihen doppelter Pins, Header genannt, zugänglich. Zusammen mit dem Grove Starter Kit desselben Herstellers bildet das Entwicklungsboard eine solide Basis für erste Schritte [7]. Allerdings ist es aufgrund seines Layouts leider nicht kompatibel zu existierenden Shields für Arduino-Systeme.

Microsoft führt mit einfach nachvollziehbaren Anleitungen [8] durch die Inbetriebnahme. Dabei haben Sie die Wahl, ob Sie die Images für das Gerät mithilfe des ausgewachsenen IDE Visual Studio oder des vielseitigen Editors Visual Studio Code entwickeln möchten. Im Falle von Visual Studio eignet sich auch die Community-Edition, die Microsoft allerdings nur für Schüler und Studenten, reine Open-Source-Projekte sowie einzelne Entwickler kostenfrei anbietet. Der Editor Visual Studio Code, den wir auch im Rahmen unserer Tests eingesetzt haben, ist demgegenüber auch für kommerzielle Projekte und Unternehmen jeder Größenordnung kostenlos verfügbar.

Unabhängig davon, welche Entwicklungsumgebung Sie bevorzugen, installieren Sie im ersten Schritt das Azure Sphere Software Development Kit (SDK), das den Befehl "azsphere" mitbringt, der sowohl auf der klassischen Kommandozeile als auch in PowerShell-Sitzungen funktioniert [9]. Installieren Sie außerdem entsprechend Microsofts Dokumentation Visual Studio Code, gefolgt von den zusätzlichen Paketen CMake und Ninja. Anschließend starten Sie Visual Studio Code und binden die Erweiterungen für Azure Sphere ein.

Verbinden Sie Ihr Entwicklungsboard per USB mit dem Computer. Sie können daraufhin per azsphere-Kommando mit dem Gerät kommunizieren, das Sie mittels der integrierten Lizenz berechtigt, ein neues

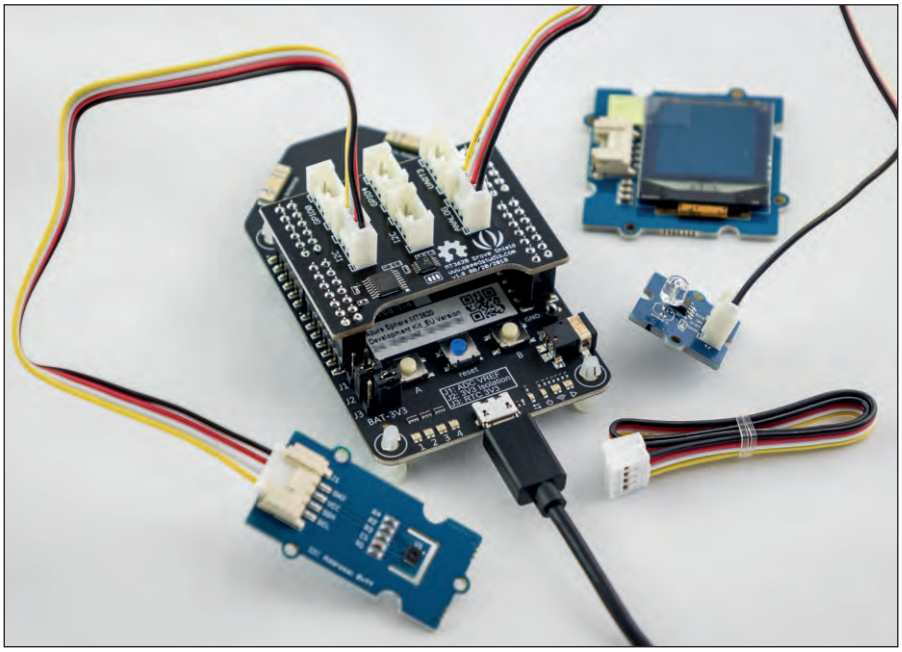


Bild 2: Das Grove Starter Kit erweitert das Azure Sphere MT3620 Development Kit ganz ohne Löten.

Kundenkonto, den sogenannten Tenant, im Azure Sphere Security Service anzulegen und das Gerät zu registrieren. Im Falle unseres Entwicklungsboards mussten wir allerdings zunächst das Betriebssystem neu installieren ("recover"), da die mitgelieferte Version schlicht zu alt war:

```
azsphere register-user --new-user
  <Ihre Mail-Adresse>

azsphere device recover

azsphere tenant create --name
  <Ihr Name>

azsphere tenant select --tenant
  <uid>

azsphere device claim
```

Den Namen Ihres Tenants dürfen Sie frei wählen, doch Obacht: Das folgende Anfordern des Geräts (claim) verbindet das Gerät irreversibel mit Ihrem Tenant. Hier hat die Sicherheit oberste Priorität. Der MCU nimmt bei einer Cloudbereitstellung keine Images außerhalb Ihres Tenants an. Eine feindliche Übernahme durch einen anderen Tenant ist somit ausgeschlossen.

Erste Signale aus der Cloud

Zunächst aktivieren Sie das Gerät aber für die lokale Ausführung von Code. Microsofts Lernpfad führt Sie daraufhin zum denkbar einfachsten Beispiel. Während angehende Programmierer in nahezu jeder Sprache zuerst lernen, den String "Hello World" auszugeben, gilt es im Bereich der Mikrocontroller, eine LED des Geräts blinken zu lassen. So stellt auch Microsoft den passenden Beispiel-Code hierzu bereit, den Sie mittels Visual Studio Code kompilieren, per USB auf den MCU übertragen und ausführen [10].

Deutlich spannender ist jedoch eine Cloudbereitstellung, denn in diesem Fall kann der Azure-Sphere-Clouddienst seine Stärken ausspielen. So erstellen Sie in der Cloud ein "Produkt" als Verwaltungseinheit für eine bestimmte Anwendung [11]. Diesem Produkt ordnen Sie dann ein oder mehrere MCU zu, die, eine korrekte WLAN-Konfiguration vorausgesetzt, fortan Images Ihrer Anwendungen als FOTA aus der Cloud beziehen. Der Azure-Sphere-Service legt dazu für jedes Produkt automatisch Gerätegruppen, darunter Entwicklung, Feldtest und Produktion, an und unterstützt so optimal den Entwicklungsprozess.

Fazit

Während bei herkömmlichen Ansätzen der Entwicklung für das IoT die Sicherheit leider oft auf der Strecke bleibt oder aber erst mühsam nachträglich ihren Weg ins fertige Produkt findet, hat Microsoft mit Azure Sphere den Schutz der gesamten Plattform von Anfang an mitgedacht. Microsofts Ansatz, die Security fix in der Hardware der MCUs zu verankern und mit einem Cloudservice zu verknüpfen, adressiert eher professionelle Anwendungsfälle und sorgt dafür, dass Sie sich ganz auf Ihre Anwendungslogik konzentrieren können. Um die Rahmenbedingungen kümmert sich Azure Sphere und stellt auch gleich die nötige Infrastruktur für einen abgestuften Prozess von der Entwicklung über Tests bis zum produktiven Einsatz bereit.

Sie finden bereits zu Azure Sphere kompatible Hardware am Markt und auch zahlreiche Codebeispiele auf GitHub. Der einzige Wermutstropfen ist die fehlende Kompatibilität zum Arduino-Ökosystem. Diese Plattform bietet ein größeres Angebot an Hardware und Anwendungsbeispielen, macht den Einstieg entsprechend einfach, ist aber abseits der Maker-Szene im professionellen Umfeld weniger verbreitet. *(jp)*

Link-Codes

[1] **Workshop zum Aufbau eines IoT in IT-Administrator August 2019**
n2p31

[2] **Azure Sphere**
n2p32

[3] **Für Azure Sphere zertifizierte Hardware**
n2p33

[4] **Azure IoT Hub**
n2p34

[5] **Applikationen in Azure Sphere**
n2p35

[6] **Azure Sphere MT3620 Development Kit**
n2p36

[7] **Grove Starter Kit für Azure Sphere MT3620 Development Kit**
n2p37

[8] **Erste Schritte mit Azure Sphere**
n2p38

[9] **Schnellstarts zum Einrichten Ihres Azure-Sphere-Geräts**
n2p39

[10] **Erstellen einer allgemeinen Anwendung**
n2p30

[11] **Erstellen einer Cloudbereitstellung**
n2p3a

Phishing-Tests mit Gophish

Im eigenen Teich

von Dr. Holger Reibold

Schutzmaßnahmen für die IT-Infrastruktur vernachlässigen oft das Einfallstor E-Mail. Zwar bringen die meisten Unternehmen einen Spam-Filter an den Start, doch dem Phishing schenken sie viel zu wenig Aufmerksamkeit. Das Phishing-Framework Gophish erlaubt mit eigenen Phishing-Kampagnen, Sicherheitslecks zu finden und auch die Anwender für diese Gefahren zu sensibilisieren.

Neun von zehn Unternehmen sind in Deutschland bereits Opfer von Datendiebstahl, Spionage und Sabotage geworden. Der Brancheverband Bitkom kalkuliert die jährliche Schadenssumme durch diese Angriffe auf über 200 Milliarden Euro. Dabei handelt es sich längst nicht mehr nur um schlecht organisierte Attacken, sondern die Art und Weise der Angriffe wird immer professioneller.

Während die meisten Unternehmen ihre sicherheitsspezifischen Maßnahmen auf die Absicherung der eigenen Infrastrukturen konzentrieren, übersehen sie, dass die eigentlichen Gefahren woanders lauern: 85 Prozent der Verletzungen der Cybersicherheit sind auf menschliches Versagen zurückzuführen und 94 Prozent aller Malware findet per E-Mail ihren Empfänger. Über 80 Prozent der sicherheitsrelevanten Ereignisse sind Phishing-Attacken. Dabei konzentrieren sich die Angreifer längst nicht mehr nur auf die scheinbar attraktiven Konzerne und Großunternehmen, sondern zunehmend auf kleine und mittlere Unternehmen (KMU). Diese KMU rücken gerade auch deshalb verstärkt in das Visier der Angreifer, weil sie deutlich weniger in ihre Sicherheitsarchitektur investieren (können).

Die Konsequenzen aus diesen Erkenntnissen: Unternehmen müssen verstärkt in ihre E-Mail-Sicherheit investieren; insbesondere den Schutz gegen Phishing-Attacken gilt es, signifikant zu verbessern. An diesem Punkt setzt Phishing Penetration Testing an, mit dessen Hilfe sich Firmen hinsichtlich der Anfälligkeit gegenüber Phishing-Attacken prüfen lassen. Mit Gophish [1] steht Ihnen ein Open-Source-Framework für exakt diese Aufgabe zur Verfügung.

Gophish im Überblick

Angesichts der hohen Relevanz der Phishing-Problematik und der damit verbundenen Bedrohungslage erstaunt es, dass sich die meisten Unternehmen auf Erweiterungen von etablierten Filterprogrammen wie SpamAssassin verlassen, die meist erst über ein Plug-in den effizienten Kampf gegen Phishing-Mails ermöglichen. Doch dabei genügt es nicht, kritische Nachrichten herauszufiltern; vielmehr muss der IT-Verantwortliche die eigene Infrastruktur auf etwaige Schwachstellen hin überprüfen. Dabei stellen dynamische Umgebungen und temporär eingeloggte Clients wie die Notebooks von Außendienstmitarbeitern, Tablets und Smartphones eine besondere Herausforderung dar.

An diesem Punkt setzt Gophish an. Es handelt sich um ein Framework, das die Simulation von Phishing-Angriffen erlaubt und so Phishing-Training für jeden Unternehmenstypus ermöglicht. Gophish ist in der Programmiersprache Go geschrieben. Der zentrale Gewinn dabei: Die kompilierten Binärdateien besitzen keinerlei Abhängigkeiten.

Inbetriebnahme

Sie können die Software einfach herunterladen und ausführen – eine Installation entfällt. Bei der Quellcode-basierten Installation müssen Sie das Zusammenspiel mit einem MySQL-Server konfigurieren, außerdem benötigen Sie SSL-Zertifikate und private Schlüssel. Schließlich müssen Sie über die "config.json"-Datei, die sich im Root-Verzeichnis der Gophish-Installation befindet, verschiedene Anpassungen wie die IP-Adressen- und Port-Konfiguration vornehmen. Über [2] stehen die kompilierten Pakete für Linux, macOS und Windows zum Download bereit. Zur Inbetriebnahme genügt das Entpacken und Starten des Gophish-Servers.

Penetration-Tester greifen bei ihrer Arbeit meist zu Kali Linux. Gophish ist zwar nicht in Kali vorinstalliert, doch das lässt sich mit wenigen Handgriffen ändern. Nach dem Download entpacken Sie Gophish in ein Verzeichnis Ihrer Wahl. Dann weisen Sie die notwendigen Berechtigungen zu:

```
chmod +x gophish
```

Anschließend passen Sie die Konfiguration über das config.json-File an. Neben der IP-

Adresse geben Sie insbesondere die Pfade zu den SSL-Schlüsseln und -Zertifikaten an. Zum Start verwenden Sie *./gophish*.

Die Durchführung von Phishing-Kampagnen ist durch drei Schritte gekennzeichnet. Im ersten generieren Sie die Templates und bestimmen die Ziele. Dann bringen Sie die Phishing-Mails auf den Weg – bei Bedarf auch zeitgesteuert. Der letzte Schritt dient dem Tracking der Ergebnisse. Diese visualisiert Gophish in seinem Dashboard in Echtzeit.

Eine Kampagne vorbereiten

Unter Windows starten Sie die Umgebung durch die Ausführung von "gophish.exe", unter macOS und Linux entsprechend über die jeweiligen Binaries. Der Zugriff auf das Webinterface erfolgt standardmäßig über die URL "https://127.0.0.1:3333/". Der Benutzername lautet "admin", das Passwort wird auf der Konsolenebene ausgegeben. Bevor Sie auf die Administrationszentrale zugreifen können, müssen Sie ein neues Passwort anlegen. Anschließend steht Ihnen die Umgebung zur Verfügung. Bei der Erstinstallation informiert Sie Gophish darüber, dass noch keine Kampagne existiert.

Im ersten Schritt legen Sie nun ein sogenanntes Sendeprofil an. Dazu wechseln Sie zum Menü "Sending Profiles" und erstellen mit "New Profile" eine erste Konfiguration. Unsere nachfolgende Beschreibung nutzt eine VM mit der IP-Adresse "192.168.178.100" und dient als Sender der Phishing-Mails. Diesem Sender weisen Sie die typischen Daten für den E-Mail-Versand zu und anschließend wartet er unter der angegebenen E-Mail-Adresse auf Nachrichten. Wichtig ist, dass Sie einen gültigen Sender-Port verwenden. Außerdem können Sie einen benutzerdefinierten Header verwenden.

Bei ersten Gehversuchen bietet es sich an, eine Testmail zu versenden, um die Funktionalität zu prüfen. Dazu klicken Sie auf "Send Test Email". Mit einem weiteren Klick auf "Save Profile" sichern Sie die erste Profilkonfiguration.

Vor dem Start einer Phishing-Kampagne müssen Sie die Ziele bestimmen, die Sie ins Visier nehmen möchten. Dabei können Sie sich verschiedener Instrumente bedienen. Wenn Sie E-Mail-Adressen aus öffentlichen Informationen sammeln wollen, um ein möglichst realistisches Szenario zu simulieren, können Sie beispielsweise zu OSINT greifen. Um die lokale Infrastruktur zu testen, benötigen Sie die lokalen E-Mail-Adressen. Unabhängig von der Datenquelle müssen Sie im Menü "Users & Groups" eine erste Gruppe anlegen. Das geschieht mit einem Klick auf "New Group". Hier vergeben Sie einen Namen und hinterlegen die Adressen Ihrer Zielgruppe. Am einfachsten geht das mit der Bulk-Importfunktion einer CSV-Datei. Damit der Import funktioniert, muss diese die Header-Werte Vorname, Nachname, E-Mail-Adresse und Position besitzen. Für erste Gehversuche können Sie einige Testempfänger auch manuell anlegen. Mit einem Klick auf "Save changes" speichern Sie die Gruppe.

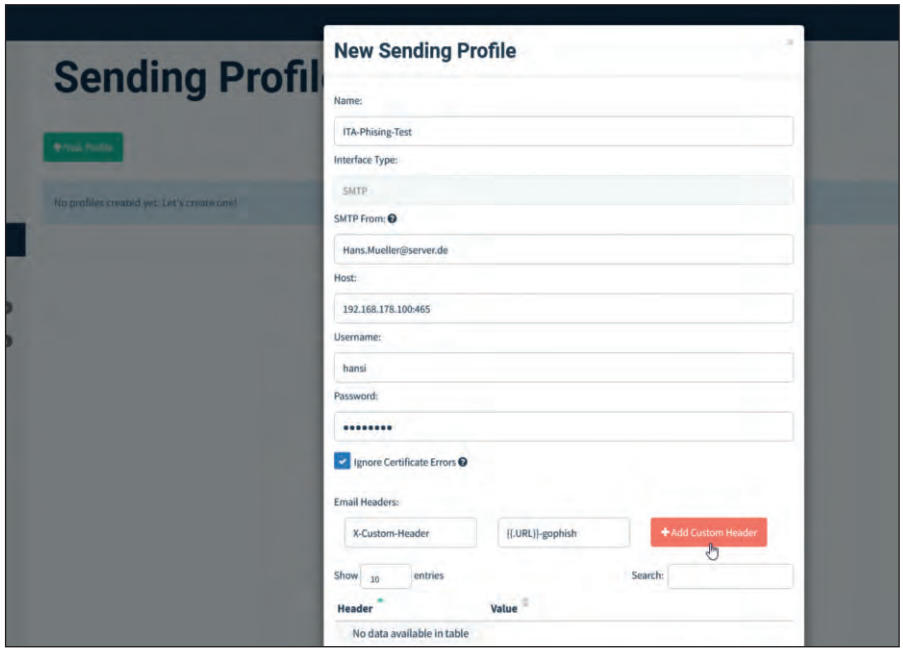


Bild 1: Den Ausgangspunkt des Phishing-Tests bildet das Anlegen eines Senderprofils.

Template für Phishing-Mail erstellen

Sind die Empfänger Ihrer ersten Phishing-Kampagne angelegt, ist des Erstellen eines Templates dran. Dahinter verbirgt sich die eigentliche Phishing-Mail, genauer gesagt deren Inhalt. Dazu wechseln Sie zum Menü "Email Template". Mithilfe der Importfunktion können Sie bestehenden E-Mail-Content übernehmen. Ein typischer Angriffsvektor bei Phishing-Mails ist es, den Empfänger der Nachricht zu einem Reset des Passworts zu bewegen. Dazu stellen Ihnen die Vorlagen verschiedene Funktionen zur Verfügung und Sie können mit Variablen arbeiten. In der Betreffzeile verwenden Sie folgende Konfiguration, um alle Empfänger aus einem E-Mail-Adressenpool zu kontaktieren: "Passwort zurücksetzen für {{.Email}}".

Im Textfeld beginnen Sie mit der Eingabe des Nachrichteninhalts, den Sie über die Registerkarte "HTML" gestalten. Gophish verfügt auch über einen einfachen visuellen Editor; diesen öffnen Sie mit einem Klick auf die Schaltfläche "Source". Um die Funktionsweise des Templates zu verdeutlichen, geben Sie folgenden Text in der HTML-Ansicht ein:

```
Hallo {{.FirstName}},
Ihr Passwort für {{.Email}} ist
  abgelaufen. Bitte beantragen Sie
  hier ein neues Passwort.
Viele Grüße,
Ihr Support-Team.
```

Nun müssen Sie dem Nachrichtenempfänger noch den "Hier"-Link anbieten. Dazu markieren Sie das Wort "hier", klicken auf das Ketten-Symbol und weisen dem Verweis auf der Registerkarte "Link Info" den Link-Typ "URL", das Protokoll "http://" und die Ziel-URL zu. Anstelle einer fixen URL verwenden Sie wieder eine Variable, dieses Mal "{{.URL}}". Diese

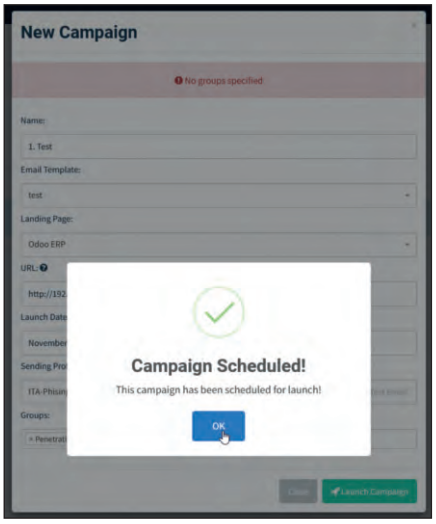


Bild 2: Die Kopie der Phishing-Landing-Page sieht der Standard-Login-Seite sehr ähnlich.

Konfiguration sorgt dafür, dass Sie verschiedenen Kampagnen individuelle URLs zuweisen können. Stellen Sie sicher, dass das Hinzufügen von Tracking-Images aktiviert ist, um die Nachverfolgung der User sicherzustellen.

Auf der Landing Page Passwörter abgreifen

Das Ziel von Phishing-Mails ist es, die personenbezogenen Informationen zu gewinnen. Dabei geht es vorrangig darum, Zugangsdaten abzugreifen und diese für weitere Betrügereien einzusetzen. Dazu locken die Angreifer ihre Opfer auf Websites, die oftmals perfekt nachgebildet sind. Daher erfordert auch Ihr Phishing-Test ein entsprechendes Portal. Dies erledigen Sie mit den Funktionen des Menüs "Landing Pages".

Dieser Schritt gestaltet sich recht einfach. Mit einem Klick auf "New Page" generieren Sie schnell eine Kopie der Seite, deren URL Sie über "Import Site" hinterlegen. In unserem Beispiel simulieren wir den Zugriff auf den Admin-Bereich einer webbasierten ERP-Installation. Nach dem Import zeigt sich der HTML-Code der Seite und ein Klick auf die Schaltfläche "Source" liefert eine Vorschau.

Damit die Formulareingaben der Opfer aufgezeichnet werden, aktivieren Sie die Schaltfläche "Capture Submitted Data". Anschließend findet sich eine zweite Option namens "Capture Passwords". Diese Eingaben sind nicht verschlüsselt und wandern im Klartext in die Gophish-Datenbank. Hat das Opfer diese Daten hinterlassen, leiten Sie es zu einer weiteren Seite weiter, die beispielsweise die Änderung der Passwortaktualisierung bestätigt, damit sich das Opfer in Sicherheit wiegt. Die Weiterleitungs-URL geben Sie im Eingabefeld "Redirect to:" an. Mit einem abschließenden Klick auf "Save Page" speichern Sie die Zielseite.

Unterstützte Variablen

Innerhalb der E-Mail-Templates und der Landing Page können Sie verschiedene Variablen verwenden. Beachten Sie, dass die Software bei Vorlagen zwischen Groß- und Kleinschreibung unterscheidet. Folgende Variablen stehen Ihnen zur Verfügung:

- {{.RId}}: Eindeutige ID der Zielperson.
- {{.FirstName}}: Vorname der Zielperson.
- {{.LastName}}: Nachname des Empfängers.
- {{.Position}}: Berufliche Position.
- {{.Email}}: E-Mail-Adresse.
- {{.From}}: Gefälschte E-Mail-Adresse des Absenders.
- {{.TrackingURL}}: URL zum Tracking-Handler.
- {{.Tracker}}: Alias für .
- {{.URL}}: Phishing-URL.
- {{.BaseURL}}: Basis-URL ohne Pfad und rid-Parameter. Sinnvoll zum Erstellen von Links zu statischen Dateien.

Eine Kampagne durchführen

Damit sind die Vorbereitungen abgeschlossen und Sie können eine erste Phishing-Kampagne auf den Weg bringen. Dazu wechseln Sie zum Menü "Campaigns" und legen mit "New Campaign" eine erste ebensolche an. Die meisten Einstellungen sind selbsterklärend: Sie weisen der Kampagne eine Bezeichnung zu, wählen das E-Mail-Template aus und bestimmen die Landing-Page. Einzig die Konfiguration des Eingabefelds "URL" bedarf der Erklärung. Hier geben Sie die IP-Adresse des Gophish-Servers an. Wichtig ist, dass dieser während der Kampagne erreichbar ist und so die Clientaktionen verfolgen und aufzeichnen kann.

Die Kampagnenkonfiguration erlaubt über das Eingabefeld "Launch Date" die zeitliche Steuerung. Geben Sie weiter das Senderprofile und die Zielgruppe an. Mit einem Klick auf "Lauch Campaign" bringen Sie ihren ersten Test auf den Weg. Nach dem Start der Kampagne werden Sie automatisch auf die Ergebnisseite weitergeleitet. Dort können Sie in Echtzeit den E-Mail-Versand und geöffnete Nachrichten verfolgen. Die Visualisierung zeigt die Anzahl der verschickten, der geöffneten sowie die Zahl der E-Mails, bei denen die Zielpersonen dem Link gefolgt sind und sogar Formulareingaben übermittelt haben.

Im Abschnitt "Details" listet das Dashboard Detailinformationen wie den Namen der Zielpersonen und den jeweiligen Status. Der Statusspalte können Sie entnehmen, welche Mitarbeiter auf die Phishing-Mail hereingefallen sind. Möglicherweise lassen sich Muster bei erfolgreichen Attacken erkennen und daraus Konsequenzen für die Anpassung der Infrastruktur ableiten. Oft hilft es auch, die Mitarbeiter noch einmal für die Problematik zu sensibilisieren.

Hinter der Reportfunktion verbirgt sich das Modul "Goreport" [3] samt Exportmöglichkeiten, die eine Weiterverarbeitung erlauben. Dazu folgen Sie im Dashboard dem Link "View Results" am Ende der jeweiligen Testkonfiguration. In der Ergebnisansicht listet Goreport die Details einer Kampagne. Aus der Detailansicht heraus ist ein Export der Ergebnisse beziehungsweise der Rohdaten in eine CSV-Datei möglich.

Für benutzerdefinierte Berichte, um beispielsweise die Ergebnisse von mehreren Kampagnen zu bündeln, steht die Gophish-API zur Verfügung. Die Entwickler stellen einen Python-API-Client zur Implementierung entsprechender Funktionen bereit. Um eine Kampagne zu beenden, führen Sie in der Ergebnisübersicht den Befehl "Complete" aus. Das automatische Beenden von Kampagnen sieht die aktuelle Gophish-Version nicht vor.

Berichte per E-Mail erhalten

Wenn Sie aufmerksam die Konsolenausgabe während des Startvorgangs von Gophish verfolgt haben, ist Ihnen vermutlich nicht entgangen, dass sich hier auch das Starten des IMAP-Managers vorfindet. Da Gophish bislang noch über keine ei-

gene Funktion für den Versand von Report-E-Mails verfügt, haben sich die Entwickler dazu entschieden, einen Meldemechanismus zu implementieren. Der Hintergrund: Im Idealfall ist nur ein geringer Teil der Benutzer auf die gefälschten E-Mails hereingefallen, doch der Administrator kann nur dann aktiv werden, wenn er über entsprechende Vorfälle informiert ist. Daher sieht Gophish einen Meldemechanismus vor. Dazu müssen Sie eine E-Mail-Adresse einrichten, die die entsprechenden Hinweise entgegennimmt.

Gophish bietet die Zugriffsmöglichkeit auf ein IMAP-Postfach. Sobald sie eine Kampagnen-E-Mail identifiziert, meldet die Software dieses Ergebnis. Dazu müssen Sie zuvor für jeden Gophish-Benutzer die IMAP-Einstellungen über das Menü "Account Settings / Reporting Settings" konfigurieren. Mit den erweiterten Einstellungen unter "Advanced Setting" bestimmen Sie die Ordner und die Polling-Frequenz. Auch hier besteht die Möglichkeit, die Konfiguration mit einem Klick auf "Test Settings" zu prüfen.

E-Mail-Anlagen einsetzen

Ein echtes Highlight von Gophish ist die Funktion "Attachment Tracking". Sie können den Gophish-Templates Anhänge mit den Dateitypen DOCX, DOCM, PPTX, XLSX, XLSM, TXT, HTML und ICS mitgeben. Beim Start einer Kampagne werden die in diesen Dokumenten platzierten Variablen durch die entsprechenden Werte ersetzet. Der Gewinn ist offensichtlich, denn so sind Sie beispielsweise bei Office-Attachments in der Lage festzustellen, ob die Opfer diese öffnen. Dies gelingt, da beim Öffnen eines präparierten Dokuments die Office-Applikation versucht, das Bild nachzuladen. Diesen Zugriff registriert dann der Gophish-Server.

Dazu erzeugen Sie beispielsweise ein Word-Dokument und fügen über das Menü "Einfügen" im Abschnitt "Text" einen Schnellbaustein ein. Hier wählen Sie "Feld", geben in dessen Eigenschaften "{{.TrackingURL}}" ein und in den Feldoptionen aktivieren Sie die Option "Daten nicht in Dokument gespeichert". Um auch die Variable für den Vor- und Nachnamen im Word-Dokument zu verwenden, müssen Sie die Grammatik- und Rechtschreibprüfung deaktivieren, da Word ansonsten einen Fehler registriert. Gophish kann nach diesem Muster auch die Ausführung von Makros registrieren. In den Template-Einstellungen hängen Sie das modifizierte Dokument an.

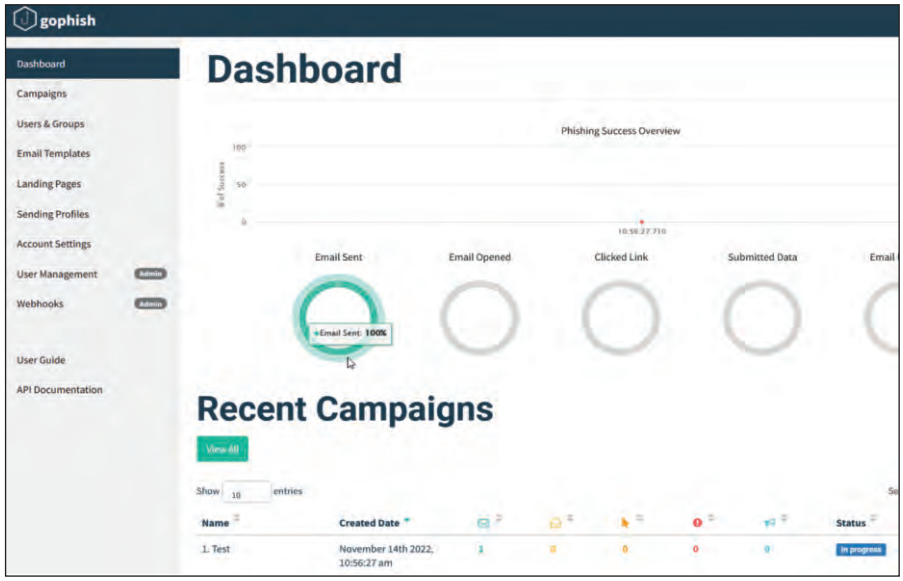


Bild 3: Das Gophish-Dashboard liefert die Ergebnisse der Phishing-Kampagne in Echtzeit.

Gophish verwalten

Die internen administrativen Funktionen von Gophish sind überschaubar und beschränken sich auf das Benutzermanagement, die Webhook-Konfiguration und das Logging. Neben dem Admin-User, den Sie bei der Inbetriebnahme von Gophish angelegt haben, können Sie im Menü "User Management" mit "New User" weitere Benutzer anlegen. Im zugehörigen Dialog weisen Sie neben dem Benutzernamen und dem Passwort eine Rolle zu. Sie haben die Wahl zwischen der Admin- und Standardbenutzerrolle. Das Hinzufügen weiterer Rollen sieht die aktuelle Version nicht vor.

Grundsätzlich bietet Gophish die Möglichkeit, die Ergebnisse über die API abzurufen. Doch in der Praxis ist es oftmals wünschenswert, dass Updates unmittelbar nach der Registrierung eines Ereignisses gemeldet werden. Dieses Problem löst Gophish durch die Webhook-Unterstützung. Bei einer Webhook-Konfiguration sendet Gophish eine HTTP-Anforderung an einen spezifischen Endpunkt – bei Bedarf auch signiert. Diese Anfrage enthält den JSON-Text des gerade registrierten Ereignisses. Das Ereignis lässt sich dann in einer Drittanwendung weiterverarbeiten. Die Webhook-Konfiguration erfolgt im gleichnamigen Menü.

Die Logging-Funktionen fallen ebenfalls rudimentär aus. Standardmäßig werden die Protokolle an der Standardfehlerausgabe ("stderr") ausgegeben. Sollen die Protokolle in eine Datei geschrieben werden, verwenden Sie den Befehl:

```
gophish.log 2>&1
```

Auf diesem Weg ist auch der Einsatz eines externen SIEM-Systems möglich.

Fazit

Gophish eröffnet eine neue Dimension bei der Bekämpfung von Phishing-Mails. Die Phishing-Penetration-Testing-Umgebung erlaubt es IT-Verantwortlichen, dieses Problem aus der Perspektive der Angreifer anzugehen und aus den gewonnenen Erkenntnissen Rückschlüsse bezüglich der Optimierung der Sicherheitsstrukturen zu ziehen. Trotz gewisser Einschränkungen bietet Gophish einen bedeutenden Mehrwert. *(jp)* IT

Link-Codes

[1] Gophish
n2z81

[2] Gophish auf GitHub
n2z82

[3] Goreprt
n2z83

Praxis-Know-how für Admins: Das IT-Administrator Sonderheft I/2023

Erfahren Sie auf 180 Seiten alles rund um das Thema:

Cloud Security

Workloads und Daten schützen

Bestellen Sie jetzt zum Abonnenten-Vorzugspreis* von nur 24,90 €!

Heinemann Verlag
Im Dialog mit Spezialisten.

Abo- und Leserservice IT-Administrator

vertriebsunion meynen
Herr Stephan Orgel
D-65341 Eltville
Tel: 06123/9238-251
Fax: 06123/9238-252
leserservice@it-administrator.de

it-a.eu/sh0123

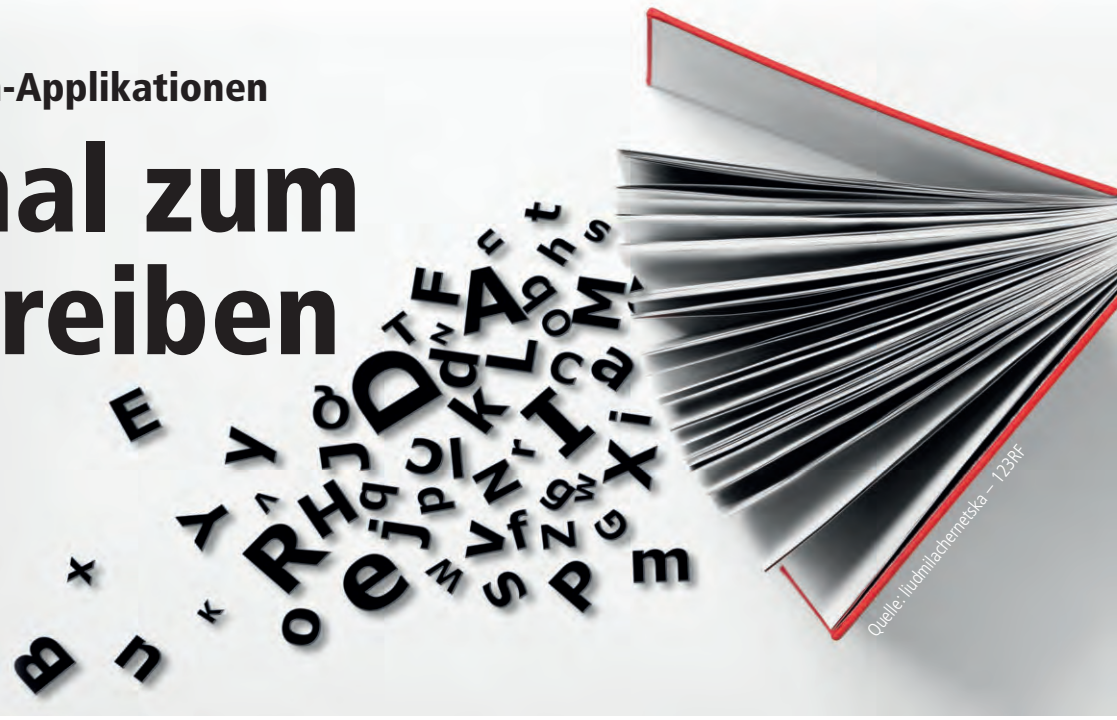
shop.heinemann-verlag.de

*Als Abonnent erhalten Sie Ihr Sonderheft zum Vorzugspreis von 24,90 € gegenüber dem Listenpreis von 29,90 €. Alle Preise verstehen sich inklusive Versandkosten und Mehrwertsteuer.

Logging von Python-Applikationen

Nochmal zum Mitschreiben

von Tam Hanna



Wie jeder andere Code können auch Skripte zur Automatisierung Fehler enthalten oder nicht genau das erledigen, was der Verfasser damit im Sinn hatte. Um solchen Problemen auf die Spur zu kommen, bietet es sich an, das Skript seine eigenen Aktionen mitschreiben zu lassen. Wie sich unter Python detaillierte Protokolldaten erzeugen lassen, zeigt dieser Workshop.

Für die Integration von Logging in Programme ist es ein guter Start, wenn die eingesetzte Sprache ein Interface für die Verarbeitung von Logdaten vorsieht. Der Nutzer muss sich dann lediglich auf das Aufrufen der vorgesehenen Funktionen beschränken, während Drittanbietermodule eine API zur Verarbeitung vorfinden. Im Fall von Python gibt es die unter [1] im Detail beschriebene API.

Einstieg in das Python-Logging

Für ein erstes Beispiel im Rahmen unseres Workshops bietet es sich an, die Ausgabe von drei Statusmeldungen in Python zu befehlen. Interessant ist, dass das Logging-Objekt eine Gruppe von Methoden darstellt – jede von ihnen übernimmt eine Nachricht, die Wichtigkeit erschließt sich aus der benutzten Methode:

```
import logging
logging.error('Dies ist ein Fehler')
logging.warning('Dies ist
  eine Warnung')
logging.info('Dies ist
  eine Information')
```

Das kleine Programm erzeugt ein Ergebnis, in dem jedoch eine der von uns angelegten Ausgaben fehlt. Die Menge der im Prompt angezeigten Informationen lässt sich durch Umgebungsvariablen steuern. Die reduzierte Darstellung in den im Artikel verwendeten Screenshots erfolgte durch die Eingabe von *export PS1='\u@\h: '*.

Das Logging-Framework in Python bietet die Möglichkeit, Logging-Meldungen Wichtigkeiten zuzuweisen. Von Haus aus gibt es die unten dargestellten Optionen, bestehend aus Log-Level und zugehörigem Ausgabewert. Angemerkt sei, dass Log-Levels in Form von Konstanten vorliegen – es ist Ihnen erlaubt, eigene Werte zwischen den vorgegebenen Leveln zu schreiben:

```
- logging.NOTSET: 0
- logging.DEBUG: 10
- logging.INFO: 20
- logging.WARNING: 30
- logging.ERROR: 40
- logging.CRITICAL: 50
```

Zur Anzeige der Meldung ist eine Anpassung der Filtereinstellungen erforderlich. Dies erledigen Sie durch die basicConfig-Methode:

```
import logging
logging.basicConfig(level=
  logging.INFO)

logging.error('Dies ist ein Fehler')
logging.warning('Dies ist
  eine Warnung')
logging.info('Dies ist
  eine Information')
```

Das basicConfig-Kommando lädt Standardeinstellungen in das Logging-Objekt, das als Basis des Aufrufs dient. Durch Übergeben des level-Parameters legt der Code fest, welche Mindestwichtigkeit notwendig ist. Lohn der Ausführung ist das Aufscheinen der Meldung "INFO: root:Dies ist eine Information" in der Kommandozeile.

Verwalten von Logging-Konfigurationen

Logging lohnt sich insbesondere in komplexen Applikationen. Das Nutzen eines statischen Objekts in mehreren Modulen wirkt sich negativ auf die Wartbarkeit des Codes aus. Im Fall des Logging-Systems von Python gilt, dass die Runtime eine Liste aller existierenden Logging-Objekte verwaltet. Dabei hat es sich als Best Prac-

tice etabliert, "__name__" zur Ermittlung eines "Lokalkontexts" heranzuziehen:

```
import logging
tamslogger = logging.
  getLogger(__name__)
tamslogger.setLevel(logging.INFO)
tamslogger.error('Dies ist
  ein Fehler')
tamslogger.warning('Dies ist
  eine Warnung')
tamslogger.info('Dies ist
  eine Information')
```

Die Konstante "__name__" erlaubt die Verortung von Code per Reflektion. Reflektion bedeutet in der Programmierung, dass ein Programm seine eigene Struktur kennt beziehungsweise diese modifizieren kann. Zum Überprüfen bietet es sich an, den Wert via *print(__name__)* auszugeben. Lohn der Mühen ist das Erscheinen des folgenden Strings:

```
tamhan@tamhan-gf65: python3
  worker.py
__main__
```

Bringen wir unser Programm nun abermals zur Ausführung, zeigt sich ein Ergebnis wie in Bild 1, wobei jedoch der setLevel-Befehl nicht funktioniert. Damit dieser korrekt arbeitet, ist folgende Anpassung erforderlich:

```
tamslogger = logging.
  getLogger(__name__)
tamslogger.setLevel(logging.INFO)
console = logging.StreamHandler()
tamslogger.addHandler(console)
```

Im Hintergrund jedes Logging-Objekts arbeiten Handler. Das sind Hilfsobjekte [2], die eingehende Logging-Daten in Richtung einer "Datensenke" weiterreichen. Unser neu erzeugtes Logging-Objekt hat keinen Handler, weshalb der unter [3] beschriebene und nicht zur Gewichtung von Logs befähigte "Handler of Last Resort" zum Einsatz kommt. Sodann erzeugen Sie durch die Eingabe von *code modula.py* ein neues Modul mit folgendem Code:

```
import logging
def modulaHallo():
  print(__name__)
```

Im Hauptprogramm folgt eine Aktivierung der Testfunktion, dabei informiert Sie "__name__" in der Rückmeldung über den Ausführungskontext:

```
import logging
import modula
modula.modulaHallo()
```

Im nächsten Schritt rufen wir die Modul-Methode im Hauptteil des Programms wie folgt auf:

```
tamslogger = logging.
  getLogger(__name__)
tamslogger.setLevel(logging.INFO)
console = logging.StreamHandler()
tamslogger.addHandler(console)
modula.modulaHallo()
```

Nun erhält "modulaHallo()" zusätzliche Befehle. Amtshandlung eins ist das abermalige Aufrufen von "logging.getLogger (__name__)", danach erfolgt die Ausgabe von Logging-Nachrichten:

```
import logging
def modulaHallo():
  tamslogger =
  logging.getLogger(__name__)
  tamslogger.error('Dies ist
  ein Fehler')
  tamslogger.warning('Dies ist
  eine Warnung')
  tamslogger.info('Dies ist
  eine Information')
  print ("- - - - - - -")
```

Bei der Ausführung zeigt sich, dass im Modul die per "info()" ausgegebene Nachricht nicht auf dem Bildschirm erscheint.

```
tamhan@tamhan-gf65: python3 worker.py
Dies ist ein Fehler
Dies ist eine Warnung
tamhan@tamhan-gf65:
```

Bild 1: Der Aufruf von "setLevel" bleibt zunächst wirkungslos.

Zur Behebung des Problems versorgen wir "modulaHallo" mit einem Parameter, der "getLogger" entsprechend beeinflusst:

```
import logging
def modulaHallo(whichName):
tamslogger =
  logging.getLogger(whichName)
```

Ein neuer Aufruf erlaubt das Übergeben von "Name":

```
modula.modulaHallo(__name__)
```

Da der an "getLogger" übergebene String in beiden Fällen konstant ist, liefert der Logging-Manager dasselbe Logging-Objekt zurück.

Zusätzliche Informationen anzeigen

Ein Vergleich der beiden Ausgaben zeigt, dass der "allgemeine" Logger mehr Informationen über den Systemzustand liefert. Sie finden unter [4] eine Liste mit rund zwei Dutzend Strings, die Informationen über den Systemzustand bereitstellen. Um diese zu nutzen, ist es erforderlich, dass Sie dem Logging-Prozess einen "Formatter" verpassen:

```
console = logging.StreamHandler()
formatter = logging.Formatter
  ('%(asctime)s:%(levelname)s:
```

```
tamhan@tamhan-gf65: python3 worker.py
2023-03-18 00:11:16,887:ERROR:Dies ist ein Fehler
2023-03-18 00:11:16,887:WARNING:Dies ist eine Warnung
2023-03-18 00:11:16,887:INFO:Dies ist eine Information
- - - - - - -
2023-03-18 00:11:16,887:ERROR:Dies ist ein Fehler
2023-03-18 00:11:16,887:WARNING:Dies ist eine Warnung
2023-03-18 00:11:16,887:INFO:Dies ist eine Information
tamhan@tamhan-gf65: python3 worker.py
2023-03-18 00:12:17,018:ERROR       :Dies ist ein Fehler
2023-03-18 00:12:17,018:WARNING     :Dies ist eine Warnung
2023-03-18 00:12:17,018:INFO        :Dies ist eine Information
- - - - - - -
2023-03-18 00:12:17,018:ERROR       :Dies ist ein Fehler
2023-03-18 00:12:17,018:WARNING     :Dies ist eine Warnung
2023-03-18 00:12:17,018:INFO        :Dies ist eine Information
tamhan@tamhan-gf65:
```

Bild 2: Die Definition der Länge erhöht die Übersichtlichkeit der Ausgabe des Loggings.

```
  %(message)s')
console.setFormatter(formatter)
tamslogger.addHandler(console)
```

Neben der Auswahl der auszugebenden Informationen ist der Befehl zum Anpassen der Anzeige zu mehr in der Lage: Durch Übergeben des Strings "-12" lässt sich eine Mindestlänge zuweisen, um die Ausgabe übersichtlicher zu gestalten (das Ergebnis sehen Sie in Bild 2):

```
formatter =
  logging.Formatter('%(asctime)s:%
  (levelname)-12s:%(message)s')
```

Handler und Dateiinformationen

Die Handler-Infrastruktur steigert die Flexibilität bei der Verarbeitung der Logdaten. Die durch die Meldemethoden angelieferten Informationen lassen sich mit Bordmitteln in rotierende Logdateien oder E-Mails verpacken. Das manuelle Implementieren des Interfaces erlaubt die Realisierung exotischerer Vorgehensweisen wie etwa der Umwandlung in einen per RS232 übertragbaren Datenstrom.

Als Demonstration sollen Informationen in eine lokale Logdatei wandern. Beim Einsatz des basicConfig-Befehls reicht das Übergeben von "filename" aus:

```
logging.basicConfig(filename=
  'example.log', encoding='utf-8',
  level=logging.DEBUG)
```

Verwenden Sie zur Reduktion der Koppelung alleinstehende Logging-Objekte, müssen Sie ein FileHandler-Objekt ergänzen:

```
tamslogger.addHandler(console)

filehdlr = logging.FileHandler
  ('filehandler.txt')
filehdlr.setFormatter(formatter)
tamslogger.addHandler(filehdlr)
```

Logging-Objekte sind zur gleichzeitigen Verarbeitung mehrerer Handler befähigt – platzieren Sie den zweiten Aufruf von "addHandler" unter den ersten, kommen beide beim Eingehen einer Message zum Zug. Die Logdatei wächst dann stark an.

Durch Übergeben des Modus-Parameters "w" (für "write") lässt sich der FileHandler anweisen, bei jedem Durchlauf des Programms eine neue Logdatei anzulegen:

```
filehdlr = logging.FileHandler('fi-
  lehandler.txt', mode='w')
filehdlr.setFormatter(formatter)
```

Nur kurze Zeit laufende Programme sind hierbei unproblematisch, weil sie in der Praxis bei Unix-Systemen nur selten vorkommen. Werkzeuge wie newsyslog oder logrotate betreffen derartige Beispiele nicht – so die Utilities im produktiven Einsatz eine geöffnete Logdatei löschen, kommt es beim normalen FileHandler zu undefiniertem Verhalten, wenn dieser versucht, den nicht mehr gültigen Datei-Deskriptor zum Schreiben zu verwenden. Der "WatchedFileHandler" [5] schafft Abhilfe.

Zu beachten ist jedoch, dass Windows keine Dateisystem-Überwachungs-API bietet, sodass der WatchedFileHandler nur unter unixoiden Systemen nutzbar ist. Dessen Nutzung setzt ein zusätzliches Paket voraus:

```
import logging.handlers
```

Zur Inbetriebnahme des WatchedFileHandler kommt derselbe Konstruktor zum Einsatz. Das Einschreiben des Formatters und die Zuweisung des Handlers in den Logger sind jedoch auch hier erforderlich:

```
filehdlr = logging.handlers.Watched-
  FileHandler('filehandler.txt',
  mode='w')
filehdlr.setFormatter(formatter)
tamslogger.addHandler(filehdlr)
```

Um das korrekte Verhalten zu überprüfen, ist ein lang anhaltender Logging-Prozess notwendig. Dieser lässt sich für einen Test durch eine Endlosschleife abbilden, die alle drei Sekunden neue Logdaten erzeugt:

```
while 1==1:
    tamslogger.error('Dies ist
        ein Fehler')
    time.sleep(3)
```

Der Test erfolgt in zwei Terminalfenstern. In Fenster eins läuft der Python-Code, während die Logdatei in Fenster zwei per *rm* gelöscht wird. Diese Manipulation erfolgt ohne Beeinflussung des Programmverhaltens, womit wir die gewünschte Funktionalität verifiziert haben.

FileHandler und seine Verwandten löschen Logdateien auf Wunsch bei jedem Durchlauf des Programms. Dies beschränkt den verbrauchten Speicherplatz, es besteht allerdings das Risiko, keine Daten zu einem kritischen Vorfall mehr vorzufinden. In der Praxis ist es besser, Logdateien zu rotieren. Darunter ist der zeitgesteuerte Austausch von Daten unter Vorhalten einer Reserve mit aktuellen Informationen zu verstehen. Alle für dieses Verhalten vorgesehenen Handler sind Varianten des "BaseRotatingHandler", der folgende Basisparameter erfodert:

```
class logging.handlers.BaseRotating-
  Handler(filename, mode, encoding=
  None, delay=False, errors=None)¶
```

In der Praxis lässt sich die abstrakte Klasse jedoch nicht einsetzen – die häufigste verwendete Implementierung ist der RotatingFileHandler:

```
rh = logging.handlers.RotatingFile-
  Handler('rotated.log', maxBy-
  tes=128, backupCount=5)
```

Mit "maxBytes" legen Sie die maximale Größe einer Logdatei fest, während Sie mit "backupCount" die Anzahl der vorzuhaltenden Backups bestimmen. Zu

Listing 1: Anpassungen am Echoserver

```
import socket

HOST = '127.0.0.1'
PORT = 5000

with socket.socket(socket.AF_INET,
   socket.SOCK_STREAM) as s:
    s.bind((HOST, PORT))
    s.listen()
    conn, addr = s.accept()
    with conn:
        print ("Neuer Client")
        while True:
            data = conn.recv(1024)
            print (data)
```

dessen Überprüfung bietet sich folgender Code an:

```
rh = logging.handlers.
  RotatingFileHandler
  ('rotated.log', maxBytes=128,
   backupCount=5)
rh.setFormatter(formatter)
tamslogger.addHandler(rh)
while 1==1:
    tamslogger.error
       ('Dies ist ein Fehler')
```

Am daraufhin erzeugten Ergebnis ist erkennbar, dass RotatingFileHandler die per backupCount und maxBytes festgelegten Constraints einhält.

Das in der Klasse angelegte Verhalten lässt sich durch Hilfsmethoden anpassen:

```
def namer(name):
  return name + ".gz"

def rotator(source, dest):
  with open(source, 'rb') as f_in:
    with gzip.open(dest, 'wb') as
       f_out:
          shutil.copyfileobj
             (f_in, f_out)
  os.remove(source)

rh.rotator = rotator
rh.namer = namer
```

Hierbei hat "Namer" die Aufgabe, in das Erzeugen der Dateinamen einzugreifen. Die Methode bekommt den von der im FileHandler enthaltenen Logik generierten Namen und kann diesen – hier durch das Anfügen einer Dateiendung – adaptieren. In "Rotator" findet sich Logik für die Dateiumbenennung und Dateiverschiebung.

Sofern Logdateien nach einer gewissen Zeit verfallen, bietet sich der "TimedRotatingFileHandler" an. Er nutzt die unter [6] gezeigten und an CRON erinnernden Strings zum Festlegen des Vorhaltezeitraums:

```
class logging.handlers.
  TimedRotatingFileHandler
  (filename, when='h',
  interval=1, backupCount=0,
  encoding=None,
  delay=False, utc=False,
  atTime=None, errors=None)
```

Handler liefern Nachrichten ins SysLog

Die Überwachung der SysLog-Inhalte ist häufig bereits implementiert. Das Einrichten eines unabhängigen zweiten Logging-Prozesses lässt sich vermeiden, wenn Python seine Daten in das SysLog schreibt. Dies ist Aufgabe von "SysLogHandler". Wir illustrieren seinen Einsatz unter Ubuntu 22.04LTS – da verschiedene Betriebssysteme unterschiedliche Logging-Infrastrukturen mitbringen, ist eventuell eine Anpassung erforderlich.

Bevor wir damit starten noch der Hinweis, dass SysLogHandler nur unter Unix-Betriebssystemen zur Verfügung steht. Windows-Plattformen verwenden die Klasse "NTEventLogHandler" – sie verhält sich analog, loggt aber in das Windows-SysLog.

Von Haus aus verbindet sich der SysLogHandler unter Nutzung des UDP-Protokolls mit der Endstelle "localhost:514". Ubuntu weist an dieser Stelle keinen Logging-Dienst auf, die korrekte Parametrierung sieht daher folgendermaßen aus:

```
loghdler = logging.handlers.SysLog-
  Handler(address = '/dev/log')
loghdler.setFormatter(formatter)
tamslogger.addHandler(loghdler)
```

Der Befehl

```
cat /var/log/syslog | tail -f
```

liefert Ihnen nun die vom Python-Programm angelieferten Informationen auf den Bildschirm.

Handler exportieren Logdaten ins Netz

Das lokale Vorhalten von Logdateien ist in kritischen Umgebungen nicht empfehlenswert, denn Angreifer finden in den Logging-Speichern angegriffener Systeme wertvolle Informationen. Python begegnet diesem Problem durch Handler, die Logdaten ins Netzwerk übertragen. So kümmern sich "SocketHandler" und "DatagramHandler" um das Absenden von Nachrichten über TCP beziehungsweise UDP. Als Gegenstelle erwarten die Klassen eine beliebige Netzwerkapplikation, die per Berkeley-Socket-API realisierbar ist. Dies ist eine Netzwerk-API, die auf so gut wie allen Betriebssystemen gleichermaßen umgesetzt ist [7].

Dieser Artikel soll nicht in einem Tutorial der Netzwerkprogrammierung ausarten, daher greifen wir auf ein fertiges Beispielprogramm [8] zurück. Dieses liegt für so gut wie alle Socket-Implementierungen als sogenannter Echoserver vor. Sie nehmen die eingehenden Informationen entgegen und schicken diese an den Client zurück. Für die folgenden Schritte adaptieren wir das Programm wie in Listing 1, sodass es die angelieferten Informationen nicht mehr in den Socket zurückschreibt, sondern sie in Richtung der Kommandozeile ausgibt.

Relevant ist die Nutzung des Ports 5000, denn Verbindungen zu Ports kleiner als 1024 setzen auf unixoiden Betriebssystemen das Vorhandensein von Root-Rechten voraus, was wir vermeiden wollen. Da Server und das Logging-Programm auf dem gleichen System arbeiten, können Sie außerdem den String "localhost" anstatt einer IP-Adresse verwenden.

Dank der Modularität des Logging-Systems sind nur minimale Eingriffe erforderlich. Das soeben verwendete Programm zur Ausgabe in den Systemlog lässt sich durch die Klasse "logging.handlers.SocketHandler" in einen Socket-Logger umwandeln:

```
loghdler = logging.handlers.
  SocketHandler('localhost', 5000)
loghdler.setFormatter(formatter)
tamslogger.addHandler(loghdler)
```

Listing 2: SMTP-Handler nutzen

```
import queue
from logging.handlers import QueueHandler,
   QueueListener
log_queue = queue.Queue(-1)
queue_handler = QueueHandler(log_queue)
remote_handler = RemoteLogHandler()
remote_listener = QueueListener(log_queue,
   remote_handler)
logging.getLogger().addHandler(queue_handler)
remote_listener.start()
```

Laufen Logger und Server parallel, präsentieren sich die Nachrichten seltsam formatiert. Ursache ist eine als "Pickler" bezeichnete Komponente. Er wandelt eingehende Nachrichten in ein über das Netzwerk übertragbares Binärformat um.

Da wir in unserem Beispiel Zeichenketten verarbeiten wollen, müssen wir den Pickler adaptieren. Hierbei bieten sich die objektorientierten Funktionen von Python an. Durch Vererbung lässt sich eine vorhandene Klasse als Blaupause für eine neue heranziehen, deren Eigenschaften veränderbar werden. Für "PlainTcpHandler" ist ein neuer Pickler erforderlich:

```
class PlainTcpHandler(logging.
  handlers.SocketHandler):
    def makePickle(self, record):
        message = self.formatter.
          format(record) + "\r\n"
        return message.encode()
```

Der mit "class" beginnende Programmteil folgt den Import-Deklarationen. Danach ist noch eine Anpassung notwendig:

```
loghdler = PlainTcpHandler('local-
  host', 5000)
```

Die Ausführung dieses Programms bringt die Nachrichten in die Kommandozeile. Angemerkt sei, dass das Python-Loggingsystem auch E-Mails senden kann. Hierzu ist ein SMTP-Server erforderlich, der wie folgt Teil des Konstruktors der SMTPHandler-Klasse wird:

```
class logging.handlers.SMTP-
  Handler(mailhost, fromaddr,
  toaddrs, subject,
  credentials=None,
  secure=None, timeout=1.0)¶
```

Allerdings ist SMTPHandler kaum mit kommerziellen Mailservern kompatibel, denn die Klasse bietet keine Unterstützung für TLS-Übertragungen, was die Verwendung von Systemen wie Gmail und Co. erschwert.

Wichtig ist außerdem, dass der Handler normalerweise pro Message eine E-Mail sendet. Dies führt in der Praxis zum Überlaufen der Mailbox und Logging-Nachrichten werden nach kurzer Zeit ungelesen gelöscht. Zur Lösung bietet sich die MemoryHandler-Klasse [9] an. Dabei handelt es sich um einen Handler, der eingehende Informationen im ersten Schritt in einem Puffer sammelt und die Logdaten in einem zweiten Schritt blockweise schreibt.

Logging in durchsatzkritischen Szenarien

Die in Python implementierte Logging-Infrastruktur ist prinzipiell Thread-sicher ausgeführt. Das bedeutet, dass das gleichzeitige Aufrufen von Methoden in einem Logger aus verschiedenen Threads keine Probleme verursacht. Kritisch ist lediglich die Nutzung einer Logdatei aus mehreren Prozessen.

Ein Grund für die hohe Thread-Sicherheit der Logger ist, dass Handler per se blockierend angelegt sind. Nutzt Code beispielsweise einen SMTP-Handler, läuft das Programm erst nach dem erfolgreichen Versand der E-Mail weiter (Stichwort Netzwerklatenz). Zur Lösung des Problems hat sich in der Python-Community der in Listing 2 gezeigte Code etabliert. Die Klasse "QueueListener" lebt hier in einem eigenen Thread, der sich um die Abarbeitung der RemoteLogHandler-Instanzen beziehungsweise der bei ihnen eingehenden Informationen kümmert.

Fazit

Die Anreicherung von Python-Code mit Logfunktionen sorgt dafür, dass der Administrator im Ernstfall auf umfangreiche Informationen zu aufgetretenen Fehlern zurückgreifen kann. Bei geschickter Kombination der verschiedenen Handler ist zum Einrichten eines flexiblen und redundanten Loggings in Python-Applikationen nur wenig Handarbeit erforderlich. *(jp)*

Link-Codes

[1] **Logging-Modul für Python**
n6z21

[2] **Logging-Handler**
n6z22

[3] **Last-Resort-Handler**
n6z23

[4] **Attribute von LogRecord**
n6z24

[5] **WatchedFileHandler**
n6z25

[6] **TimedRotatingFileHandler**
n6z26

[7] **Socket-Programmierung**
n6z27

[8] **Beispielprogramm zu Echoserver**
n6z28

[9] **Überlaufen der Logging-Nachrichten verhindern**
n6z29

Quelle: bozenafulawka – 123RF

Echtzeit-Monitoring mit Graphite

Alles auf einen Haufen

von Dr. Holger Reibold

Die Open-Source-Software Graphite verspricht ein Echtzeitmonitoring für IT-Umgebungen. Dabei zeichnet es sich als gewissenhafter und schneller Datensammler aus, der jedoch bei Visualisierung und Alarmierung auf Drittanwendungen angewiesen ist. Ist diese Hürde jedoch überwunden, profitieren Admins von einem Monitoring, dass Daten aus nahezu allen Systemen sammeln und verarbeiten kann.

Graphite [1], das im Jahr 2006 als Nebenprojekt eines Überwachungswerkzeugs eines Flug- und Reiseportals startete, hat sich bis heute zu einem mächtigen System entwickelt, das laut Angaben des Entwicklers Chris Davis die Unternehmen Etsy, Booking.com, GitHub, Salesforce, Reddit und vielen anderen zur Überwachung ihrer Geschäftsprozesse dient. Seit 2008 steht die Software unter der Open-Source-Lizenz Apache 2.0.

Im Kern leistet Graphite zwei Aufgaben: Es speichert numerische Zeitreihendaten und rendert die Daten in Form von Diagrammen. Dabei gestaltet sich die Übertragung von Rohdaten in das System besonders einfach. IT-Verantwortliche profitieren dabei von einem inzwischen stark angewachsenen Ökosystem, das für alle gängigen Anwendungsszenarien die notwendigen Collection-Agenten und Sprachanbindungen bietet.

Das Innenleben von Graphite

Um möglichst unterschiedliche Rohdaten bündeln zu können, setzt Graphite auf das Zusammenspiel mit drei Softwarekomponenten. Dabei fragt Carbon die verschiedenen Zeitreihendaten ab, Whisper übernimmt als Datenbankbibliothek die Speicherung dieser Daten und Graphite-web stellt das User-Interface und die API für die Darstellung von Diagrammen und Dashboards bereit. Der Carbon-Dienst ist für die Einspeisung der verschiedenen Rohdaten in den Stack zuständig, der seinerseits der langfristigen Speicherung der Daten in der Whisper-Datenbanken dient. Administratoren interagieren mit der Graphite-Web-UI oder der API, die wiederum Carbon und Whisper nach den Daten abfragt.

Der zentrale Gewinn des Tools besteht darin, dass es die verschiedensten Ausgangsdaten bündelt, konsolidiert und für Drittanwendungen verfügbar macht. Graphite unterstützt diverse Ausgabestile und -formate, beispielsweise CSV, XML und JSON. Damit ist die Voraussetzung für das Einbetten von benutzerdefinierten Diagrammen in externe Websites oder Dashboards gegeben.

Hinsichtlich der Übermittlung der Rohdaten an Graphite sind Sie flexibel. Die Monitoringumgebung unterstützt dafür drei Hauptmethoden: Klartext, das Python-spezifische Datenformat Pickle und AMQP (Advanced Message Queuing Protocol). Die an Graphite gesendeten Informationen werden von Carbon und Carbon-Relay verwaltet, während das Graphite-Webinterface diese Daten entweder aus dem Cache oder direkt von einem Speichermedium liest.

Welche Übertragungsmethode im Einzelfall zum Einsatz kommen sollte, ist dabei primär von den auslesenden Applikationen oder den verwendeten Skripten abhängig. Manche davon verfügen über spezielle Tools oder APIs, die Ihnen helfen können, Daten nach Carbon zu übertragen. Um die Übertragungsspezifika kennenzulernen, verwenden Sie am einfachsten das Klartextprotokoll.

Hinter Carbon verbergen sich verschiedene Daemons, die das Storage-Backend bilden. Eine einfache Graphite-Installation verwendet üblicherweise lediglich einen Daemon ("carbon-cache.py"). Bei größeren Umgebungen kommen in der Regel "carbon-relay.py" und "carbon-aggregator.py" hinzu, um die Last der Metrikenverarbeitung zu verteilen beziehungsweise benutzerdefinierte Aggregationen durchzuführen. Grundsätzlich erwarten alle Carbon-Daemons Zeitreihendaten von Drittquellen und können diese über gängige Übertragungsprotokolle empfangen. Allerdings unterscheiden sich die verschiedenen Daemons in der Verarbeitungsform der entgegengenommenen Daten. Die Kenntnisse dieser unterschiedlichen Ver-

arbeitungsmöglichkeiten ist bei der Implementierung von anspruchsvollen Speicher-Backends notwendig.

Installation

Graphite ist eine komplexe Linux-basierte Umgebung, die primär in Python programmiert wurde. Für das Rendern der Graphiken verwendet die Umgebung die Cairo-Grafikbibliothek. Daraus ergeben sich verschiedene Abhängigkeiten, wie sie typischen Serverinstallationen üblicherweise nicht bereitstellen. Bei der quellenbasierten Installation steht Ihnen das Skript "check-dependencies.py" zur Prüfung zur Verfügung. Am einfachsten gestaltet sich die Installation von Graphite in Docker – so ist auch eine Evaluierung auf einem Windows-System möglich. In diesem Fall führen Sie folgende Befehle aus:

Bild 1: Das Graphite-Dashboard liefert wie hier zu erkennen lediglich ein einfaches Werkzeug für die Visualisierung metrischer Daten – besser klappt es mit Grafana.

```
docker run -d \
 --name graphite \
 --restart=always \
 -p 80:80 \
 -p 2003-2004:2003-2004 \
 -p 2023-2024:2023-2024 \
 -p 8125:8125/udp \
 -p 8126:8126 \
 graphiteapp/graphite-statsd
```

Bei einer Standardinstallation müssen Sie neben Python ab Version 2.7 folgenden Voraussetzungen schaffen:

- cairocffi
- Django 1.11.19 oder höher
- django-tagging 0.4.6
- pytz
- scandir
- fontconfig
- WSGI und ein Webserver (bei Apache ist außerdem das Modul "mod_wsgi" erforderlich)

Außerdem benötigen Sie die Graphite-Webapp, Carbon und die Whisper-Datenbankbibliothek, die Teil des Graphite-Projekts ist. Abhängig von optional genutzten Funktionen sind weitere Python-Module wie "python-memcache", "python-lpad" und "python-rrdtool" erforderlich.

Dass die Installation nicht zwingend eine Herausforderung darstellen muss, zeigt die Verwendung von Synthesize [2], einem Installationsskript für Graphite und den zugehörigen Diensten auf Linux. Allerdings ist der Einsatz von Synthesize auf Ubuntu 18.04 LTS begrenzt. Jedoch existiert mit REsynthesize [3] ein Fork, der für CentOS 8.1 oder höher konzipiert ist.

Carbon-Modul konfigurieren

Das Carbon-Modul ist das Herzstück der Graphite-Umgebung. Seine Steuerung erfolgt über verschiedene Konfigurationsdateien, die im Verzeichnis "/opt/graphite/conf/" liegen. Da die Erstinstallation keine Konfigurationsdatei anlegt, müssen Sie dies händisch übernehmen. Dabei greifen Sie am einfachsten auf die verschiedenen Beispieldateien ("conf.example") zurück. Kopieren Sie diese in das Konfigurationsverzeichnis und entfernen Sie den Zusatz ".example". Graphite kennt folgende Konfigurationsdateien:

- "carbon.conf" ist die Hauptkonfigurationsdatei, mit der Sie die Einstellung für jeden Carbon-Daemon definieren.
- "storage-schemas.conf" bestimmt die Aufbewahrungsraten für die Speicherung von Metriken.
- "storage-aggregation.conf" legt fest, wie Daten geringerer Genauigkeit aggregiert werden.
- "relay-rules.conf": Mithilfe von Relay-Regeln werden Metriken an spezifische Backends übermittelt.
- "aggregation-rules.conf": Die sogenannten Aggregationsregeln bündeln verschiedene Metriken.
- "rewrite-rules.conf" erlaubt das Umbenennen von Metriken mithilfe regulärer Ausdrücke.

Bei einer Erstinstallation sind primär die Anpassungen der "carbon.conf" und der "speicher-schemas.conf" relevant. Die Hauptkonfigurationsdatei "carbon.conf" bündelt die Einstellungen der verschiedenen Daemons, wobei die Konfiguration in Abschnitte unterteilt ist: In "[cache]" steuern Sie "carbon-cache", mit "[relay] das Modul "carbon-relay" und entsprechend mit dem Abschnitt "[aggregator]" den Aggregator "carbon-aggregator". Die Entwickler empfehlen bei der Erstverwendung das Augenmerk auf den Abschnitt "[cache]" zu legen.

In der "storage-schemas.conf"-Datei weisen Sie den Mustern Metrikpfade zu und hinterlegen für die Komponente "whisper" Frequenz und Dauer der Datenspeicherung. Bei den Mustern handelt es sich um reguläre Ausdrücke, die sich in drei Zeilen definieren. In der ersten weisen Sie der Regel eine Bezeichnung zu. Es folgen der reguläre Ausdruck, den Sie mit "pattern=" angeben und die Retentionsrate ("retentions="). Ein einfaches Beispiel hierzu:

```
[garbage_collection]
pattern = garbageCollections$
retentions = 10s:14d
```

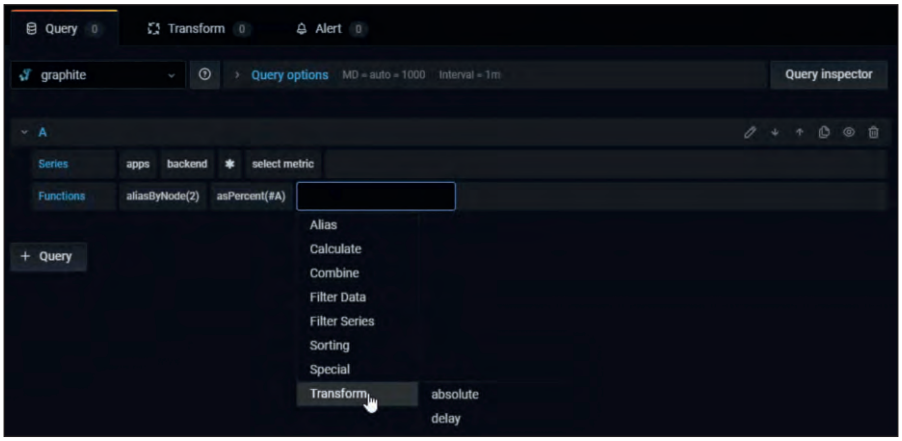


Bild 2: Grafanas Query-Editor ermöglich die die Integration von Graphite-Daten.

Im Beispiel dient die Bezeichnung "[garbage_collection]" primär der Dokumentation. Graphite protokolliert diesen und die entsprechenden Metriken in der Protokolldatei "creates.log". Dieses Muster kommt für alle Metriken zur Anwendung, die mit "garbageCollections" enden. Grundsätzlich gilt es zu beachten, dass Graphite die Python-Syntax für reguläre Ausdrücke verwendet. Die Retentions-Zeile besagt, dass jeder Datenpunkt zehn Sekunden repräsentiert und Daten der letzten 14 Tagen verfügbar sein sollen.

Neben den verschiedenen Konfigurationsdateien steuern Sie die Verwendung von Metriken mit White- und Black-Listen. Im Fall der Whitelist-Funktionalität akzeptiert Graphite nur die Metriken, die explizit in der Whitelist aufgeführt sind – Metriken der Blacklist werden entsprechend abgewiesen. Der Vorteil ist offensichtlich: Sie können explizit all jene Metriken herausfiltern, die nicht für Ihr Informationsbedürfnis relevant sind. Diese Filterfunktion aktivieren Sie in der "carbon.conf"-Datei mit dem "USE_WHITELIST"-Flag. Danach durchsucht Graphite das mit der Option "GRAPHITE_CONF_DIR" definierte Verzeichnis nach den beiden Black- und Whitelist-Konfigurationen ("whitelist.conf" und "blacklist.conf"). Haben Sie keine Whitelist-Konfiguration angelegt oder ist diese leer, lässt die Software alle Metriken durch.

Metriken ein- und auslesen

Die Herausforderung beim Einsatz von Graphite besteht im Einlesen der unterschiedlichen Daten, wobei sich die Monitoringumgebung jedoch sehr flexibel zeigt. Sie können sich der drei bereits erwähnten Methoden Klartext, Pickle und AMQP bedienen. Die eingelesenen Daten wandern in die beiden Module Carbon sowie Carbon-Cache und werden von diesen verwaltet. Welche Methode Sie für das Speisen von Graphite verwenden, ist von der jeweiligen Umgebung und den zu verarbeitenden Daten abhängig. Ihnen stehen hierfür verschiedene Tools und APIs zur Verfügung. Bei Testdaten verwenden Sie am einfachsten das Klartextprotokoll, bei großen Datenmengen empfiehlt sich Pickle. AMQP ist die geeignete Methode, wenn Carbon auf einen Message Bus hört.

Der einfachste Weg stellt die Verwendung des Klartextprotokolls dar. Dabei müssen die gesendeten Daten das Format "<Metrischer Pfad> <Metrischer Wert> <Metrischer Zeitstempel>"verwenden. Carbon kümmert sich anschließend um die Übersetzung dieser Textzeile in eine Metrik, die Webinterface und Whisper-Datenbank verstehen. Zu Testzwecken nutzen Sie unter Unix das Programm "netcat", um einen Socket zu generieren und Daten an Carbon zu übermitteln:

```
PORT=2003
SERVER=graphite.system
echo "local.random.diceroll 4 `date
  +%s`" | nc ${SERVER} ${PORT}
```

Im Grunde dient Graphite dazu, numerische Daten zu sammeln und diese an Carbon zur Auswertung zu übermitteln. Dabei besitzt jede Datenreihe einen eindeutigen Bezeichner, der auf der metrischen Bezeichnung und verschiedenen Tags basiert. Die Entwicklung eines Bezeichnungssystems ist essenziell. Der zweite Schritt dient der Konfiguration der Datenaufbewahrung, dabei verschiedene Fragen zu beantworten: Wie oft werden die Daten produziert? Welche Genauigkeit ist gewünscht? Welchen Zeitraum wollen Sie erfassen?

Anschließend erzeugen Sie ein Namensschema, in dem Sie die "/opt/graphite/conf/storage-schemas.conf"-Datei anpassen. Graphite verlangt das Nachrichtenformat, das aus dem Metrik-Namensraum, dem Wert, den Sie der Metrik zu diesem Zeitpunkt zuweisen wollen und dem Zeitstempel besteht. Hier ein einfaches Beispiel für die Verwendung dieses Formats:

```
echo "test.bash.stats 42 `date +%s`"
  | nc localhost 2003
```

Integration externer Software

Graphite ist für das Zusammenspiel mit rund einhundert verschiedenen Tools gerüstet. Die Entwickler stellen eine Übersicht der unterstützten Werkzeuge unter [4] zur Verfügung. Die Software spielt beispielsweise mit "collectd" zusammen, dem bekannten Daemon zum Sammeln von Systemdaten. Um collectd-Metriken an Carbon zu übermitteln, verwenden Sie das "write-graphite"-Plug-in von collectd.

Alternativ bezieht Graphite die Metriken aus den RRD-Dateien von collectd, indem Sie die RRD-Dateien zu "STORAGE_DIR/rrd" hinzufügen. In der Praxis können Sie beispielsweise die Datei "host.name/load/load.rrd" von collectd mit" rrd/collectd/host_name/load/load.rrd" verknüpfen, um den Graph "collectd.host_name.load.load.{short,mid,long}term" zu generieren.

Daten visualisieren

Für die Visualisierung der Daten ist das Graphite-Dashboard zuständig und erlaubt die Darstellung verschiedener Quellen. Der Zugriff auf das Dashboard erfolgt über die URL "http://mein.graphite.host/dashboard", alternativ über den sogenannten Composer. Leider sind die Vi-

sualisierungsmöglichkeiten von Graphite recht beschränkt. Sie erlauben aber beispielsweise, Funktionen auf die Daten anzuwenden oder die Graphen von verschiedenen Hosts zu bündeln.

Doch wie erwähnt erweisen sich die integrierten Visualisierungsfunktionen in der Praxis als sperrig. Daher greifen die meisten Administratoren auf benutzerfreundlichere Werkzeuge zurück, die aussagekräftige Auswertungen liefern. Großer Beliebtheit erfreut sich Grafana [5], gerade auch deshalb, weil es über ein natives Plug-in für die Integration von Graphite-Datenquellen verfügt. Mit Hilfe des "Grafana Query Editor" gestaltet sich die Integration einfach.

Alarmierung einrichten

Das Monitoring von relevanten Systemen und die Visualisierung der Daten ist das eine, doch Administratoren müssen primär auf kritische Ereignisse reagieren können. Hierfür kommen typische Alarmierungs- und Benachrichtigungsfunktionen zum Einsatz. Per Definition inkludiert ein Monitoringsystem auch die Ausgabe von Warnhinweisen bei der Erfüllung spezifischer Werte, doch Graphite verfügt nicht über entsprechende Möglichkeiten. Hierfür sind Sie auf Drittprodukte angewiesen.

Wiederum gestaltet sich die Alert-Konfiguration beim Zusammenspiel mit Grafana als besonders einfach. Diese Umgebung verfügt über den Alerting-Manager, mit dem Sie das gewünschte Regelwerk anlegen. Dabei profitieren Sie von der Möglichkeit, dass Sie ein- und mehrdimensionale Regeln generieren können. Der Alerting-Manager stellt den Query-Manager zur Verfügung, der die Integration von Graphite-Metriken mit wenigen Mausklicks erlaubt.

Alternativ bietet sich der Einsatz eines Werkzeugs wie "graphite-beacon" [6]. Dabei handelt es sich um eine einfache Alarmierungsanwendung für Graphite, die asynchron arbeitet und Benachrichtigungen auf Basis von Graphite-Metriken versendet. Der Vorteil ist, dass es außer dem Tornado-Paket keine Abhängigkeiten aufweist und einfach zu implementieren ist. Zur Installation mit Pip verwenden Sie folgenden Befehl:

```
pip install graphite-beacon
```

Alternativ klappt das auch mit "apt-get":

```
apt-get update

apt-get install graphite-beacon
```

Die Konfiguration findet in einer "config.json"-Datei statt, die sich im gleichen Verzeichnis wie "graphite-beacon" befindet. Darin geben Sie zunächst die URL des Graphite-Systems an und legen mithilfe regulärer Ausdrücke fest, wann eine Warnung generiert wird. Außerdem müssen Sie einen E-Mail-Handler definieren, der für den E-Mail-Versand der Hinweise sorgt. Neben Graphite-Alert können Sie mit diesem Instrument für webbasierte Umgebungen URL-Warnungen konfigurieren.

Fazit

Die Graphite-Entwickler ordnen ihre Anwednung gerne als Monitoringwerkzeug ein – wohl auch, um eine größere Zielgruppe anzusprechen. Doch ein zweiter Blick zeigt, dass es sich primär um einen Aggregator handelt, der die verschiedensten Metriken zusammenführt. Diese Aufgabe bewältigt Graphite par excellence – nicht zuletzt wegen des umfassenden Ökosystems, das um das Tool herum entstanden ist. Sein ganzes Potenzial entfaltet Graphite allerdings erst im Zusammenspiel mit Visualisierungsspezialisten. *(jp)*

Link-Codes

[1] Graphite
n5z81

[2] Synthesize
n5z82

[3] REsynthesize
n5z83

[4] Tools, die mit Graphite zusammenarbeiten
n5z84

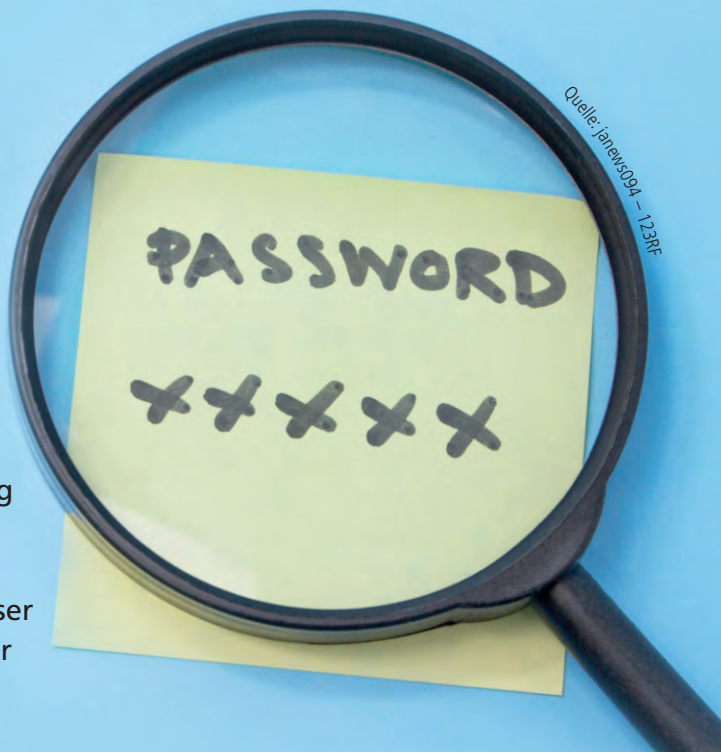
[5] Grafana
j7z91

[6] graphite-beacon
n5z85

Passwortlose Authentifizierung mit FIDO2

Ausgedient

von Thorsten Scherf



Immer mehr Unternehmen setzen in Ihrer IT-Umgebung auf Zero Trust, wozu eine ganze Reihe an Sicherheitsmaßnahmen gehört. Wir werfen im Open-Source-Tipp diesen Monat einen Blick auf den FIDO2-Standard. Dieser hilft beim Verzicht auf tendenziell unsichere Passwörter bei der Authentifizierung von Benutzern.

Die FIDO-Allianz (Fast IDentity Online) hat bereits in der Vergangenheit eine ganze Reihe an Standards veröffentlicht, die dabei helfen sollen, die Sicherheit bei der Authentifizierung von Benutzern zu verbessern. So haben wir bereits den damaligen Industriestandard U2F als Teil der FIDO1-Spezifikation vorgestellt. Dieser sieht vor, dass Benutzer mit geeigneten Tokens zusätzlich zum Passwort einen weiteren Faktor für eine erfolgreiche Authentifizierung benötigen.

FIDO2 wurde im März 2019 von der FIDO-Allianz in Zusammenarbeit mit dem World Wide Web Consortium (W3C) herausgegeben. Es hat allerdings einige Zeit gedauert, bis die Soft- und Hardwarehersteller diesen Standard unterstützt haben. Wie schon bei FIDO1 gehören diverse unterschiedliche Spezifikationen zu FIDO2.

Eine Kernkomponente stellt sicherlich die "Web Authentication"(WebAuthN) dar. Diese soll dafür sorgen, dass bei der Anmeldung von Benutzern an einem Webservice zukünftig keine Passwörter gefragt sind. Im Zuge der Zero-Trust-Strategie [1] möchten viele Unternehmen nämlich auf den Einsatz von Passwörtern so weit wie möglich verzichten. FIDO2 stellt somit einen wichtigen Baustein zur Umsetzung von Zero Trust dar.

Listing 1: LDAP-passkey-Schema

```
attributeTypes: ( 2.16.840.1.113730.3.8.24.27
   NAME 'passkey' DESC 'Passkey mapping'
   EQUALITY caseExactMatch SYNTAX
   1.3.6.1.4.1.1466.115.121.1.15 )
objectclasses: ( 2.16.840.1.113730.3.8.24.9
   NAME 'passkeyUser' DESC 'Passkey user'
   AUXILIARY MAY passkey)
```

Asymmetrische Kryptographie

Das Prinzip von WebAuthN basiert auf asymmetrischer Kryptographie, wobei ein öffentlicher und ein privater Schlüssel für einen Benutzer zum Einsatz kommen. Der öffentliche ist jeweils an die Webservices zu übertragen, die WebAuthN unterstützen und auf denen sich der Benutzer anmelden möchte. Der private Schlüssel verbleibt hingegen lokal. Optional können Sie den Private Key zusätzlich mit einer PIN schützen. Für die Schlüsselerzeugung kommt zumeist ein Hardwaretoken in Form eines USB-Devices zum Einsatz. Sehr beliebt und oft eingesetzt sind die Tokens der Forma YubiKey [2], wobei es auch eine Vielzahl an anderen Herstellern gibt, die Hardwaretoken für FIDO2 anbieten.

In unserem Open-Source-Tipp möchten wir jedoch einen anderen Einsatzzweck von FIDO2 aufzeigen. Es geht dabei darum, wie Sie sich mithilfe eines entsprechenden Tokens und eines zentral verwalteten Benutzeraccounts an Ihrem Linux-System anmelden können. Das Konto kann dabei entweder in einem klassischen LDAP-Server oder einem FreeIPA-System vorliegen. Letzteres bietet den Vorteil, dass Sie bei der Anmeldung auch direkt ein Kerberos-Ticket für diesen Benutzer ausgestellt bekommen. Hiermit können Sie dann ohne die Eingabe des Passworts auf weitere Services zugreifen, solange diese Kerberos als Authentifizierungsmethode unterstützen.

Fedora 39, das im Herbst 2023 erscheint, wird eine der ersten Linux-Distributionen sein, die FIDO2 und WebAuthN von Haus aus unterstützt. Hierfür findet ein zusätzliches Softwarepaket namens sssd-passkey seinen Weg in die Distribution.

Vorab testen

Wer die neue Software schon jetzt testen möchte, kann diese über ein Fedora-COPR-Repository [3] beziehen. Die Aktivierung des Repositories erfolgt über den Befehl

```
dnf copr enable
  ipedrosa/passkey-auth
```

Im Anschluss haben Sie unmittelbar Zugriff auf sämtliche Pakete aus diesem Software-Repository und können die aktuelle Version des System Security Services Daemons (sssd) installieren. Und natürlich

steht hierüber nun auch das neue Paket sssd-passkey zur Verfügung:

```
dnf install sssd sssd-passkey
```

Haben Sie ein FIDO2-kompatibles Token an Ihrem Rechner angeschlossen, erzeugen Sie mit dem folgenden Kommando einen öffentlichen und privaten Schlüssel für den gewünschten Benutzer. Die Domäne bezieht sich dabei auf den Bereich, in dem der Account im LDAP- oder FreeIPA-System verwaltet wird:

```
sssctl passkey-register --username=
  tscherf --domain=ipa.test
```

Das sssctl-Tool erzeugt nun einen Schlüsselbund und gibt als Ergebnis eine längere Zeichenkette auf dem Display aus. Diese können Sie dem jeweiligen Benutzerkonto hinzufügen. Abhängig davon, welches Backendsystem Sie verwenden, unterscheidet sich die Vorgehensweise hierfür. Stellen Sie zunächst aber auch sicher, dass die Konfigurationsdatei des SSSD-Services ("/etc/sssd/sssd.conf") über die passkey-Anweisung verfügt:

```
[pam]
pam_passkey_auth=true
```

Starten Sie im Anschluss den Dienst erneut, um die Konfigurationsanweisung zu aktivieren: *systemctl restart sssd*.

Benutzerkonto mit FIDO2-Schlüssel

Am einfachsten erfolgt die Konfiguration mit FreeIPA als zentrales Identity-Management-System. Für einen bereits vorhandenen Benutzer können Sie hier einfach auf das Kommando *ipa user-add-passkey* zurückgreifen, um den FIDO2-Schlüssel einem Benutzer-Account hinzuzufügen:

```
ipa user-add-passkey tscherf
  passkey:<Schlüssel>
```

Mittels *ipa user-show* überzeugen Sie sich dann davon, dass der Schlüssel nun auch Teil des Benutzer-Objektes im LDAP ist:

```
ipa user-show tscherf

User login: tscherf
First name: tscherf
Last name: doe
Home directory: /home/tscherf
Login shell: /bin/sh
Principal name: tscherf@IPA.TEST
Principal alias: tscherf@IPA.TEST
Email address: tscherf@ipa.test
UID: 805400005
GID: 805400005
Passkey mapping: passkey:<Schlüssel>
Account disabled: False
Password: False
Member of groups: ipausers
Kerberos keys available: False
```

Melden Sie sich nun am System an oder wechseln mittels *su* zu einem Konto, das über einen FIDO2-Schlüssel verfügt, fängt das Hardwaretoken an zu blinken. Nach einem Tastendruck auf dem Token startet der Authentifizierungsvorgang und Sie werden angemeldet, ohne ein Passwort eingeben zu müssen.

Listing 2: LDIF-Datei user.ldif

```
dn: uid=foobar,dc=example,dc=com
mail: foobar@example.com
uid: foobar
givenName: Foo
objectClass: top
objectClass: person
objectClass: organizationalPerson
objectClass: inetorgperson
objectClass: posixAccount
objectClass: inetuser
objectClass: passkeyUser
sn: Bar
cn: Foo Bar
uidNumber: 5000
gidNumber: 5000
homeDirectory: /home/foobar
loginShell: /bin/bash
gecos: foobar
passkey: passkey:<Schlüssel>
```

LDAP-Server im Backend

Setzen Sie im Backend einen regulären LDAP-Server ein, müssen Sie im ersten Schritt sicherstellen, dass dieser über die notwendige Objektklasse verfügt, damit Sie das passkey-Attribut im nächsten Schritt einem Benutzerobjekt zuweisen können.

Die Vorgehensweise unterscheidet sich je nach eingesetztem LDAP-Server. Für den 389 Directory Server [4] können Sie beispielsweise die Datei "/etc/dirsrv/slapd-backend/schema/60base.ldif" um die beiden Einträge aus Listing 1 erweitern und dann im nächsten Schritt mittels *dsconf* die Schemaänderungen aktivieren:

```
dsconf -D "cn=Directory Manager" -W
  backend schema reload
```

Grundsätzlich können Sie natürlich ebenso ldapmodify einsetzen, um Anpassungen am Schema vorzunehmen. Gleiches gilt für den Benutzeraccount, den Sie im LDAP-Server anlegen – ob Sie hierfür auf ein grafisches Frontend zurückgreifen oder auch wieder ldapmodify oder ldapadd einsetzen, spielt keine Rolle. Wichtig ist lediglich, dass das LDAP-Objekt über das Attribut "passkey" verfügt, indem Sie den öffentlichen Schlüssel des Benutzers speichern, den Sie zuvor mithilfe von *sssctl passkey-register* erzeugt haben. Listing 2 zeigt ein Besipiel für eine einfache LDIF-Datei, die Sie mittels ldapadd in den LDAP-Server laden können:

```
ldapadd -D "cn=Directory Manager"
  -W -H ldap://localhost -x -f
  /tmp/user.ldif
```

Hat alles geklappt, können Sie nun ebenfalls eine Anmeldung mit diesem Konto durchführen, ohne hierfür ein Passwort zu verwenden.

Fazit

Mittels FIDO2 und eines passenden Hardwaretokens können sich zentral verwaltete Benutzer ohne die Eingabe eines Passworts an einem System anmelden. Der vom Token erzeugte Schlüssel ist hierfür dem Benutzerobjekt im LDAP in einem zusätzlichen Attribut hinzuzufügen. FIDO2 ist somit ein hilfreicher Baustein bei der Umsetzung einer Zero-Trust-Strategie. *(dr)*

Link-Codes

[1] Zero-Trust-Security
n3pc3

[2] YubiKey-FIDO2-Token
n6pc2

[3] Fedora-COPR-Repositories
k2z61

[4] 389 Directory Server
n6pc5

Monitoringdaten mit Detectree visualisieren

Stets im Bilde

von Dr. Matthias Wübbeling

Die Visualisierung von Log- und Eventdaten unterstützt Securityanalysten beim Erkennen von Mustern und Zusammenhängen. Entsprechende Werkzeuge kommen in IT-Abteilungen präventiv und reaktiv zum Einsatz und helfen den Mitarbeitern, den Überblick bei der Beobachtung der Infrastruktur zu behalten. Der Security-Tipp in diesem Monat stellt Ihnen mit Detectree eine simple Möglichkeit zur grafischen Darstellung von Sachverhalten vor.

Detectree [1] ist eine Open-Source-Entwicklung des Unternehmens WithSecure (ehemals F-Secure) zur Angriffserkennung und -abwehr. Die Idee dahinter ist die Analyse von Log- und Eventdaten mit einem besonderen Fokus auf Zusammenhänge zwischen Aktivitäten und Infrastrukturelementen. Die Entwickler haben sich dabei bewusst für Baumstrukturen zur grafischen Darstellung entschieden. Damit sind, im Gegensatz zu einfachen Graphen, nicht nur grundsätzliche Beziehungen, sondern auch Hierarchien darstellbar. Detectree fokussiert dabei vor allem Prozessbäume, also die Abhängigkeiten beziehungsweise Hierarchien von Prozessen.

Installation über Repository

Um Detectree für Ihre Analysearbeit auszuprobieren, klonen Sie das Repository aus dem offiziellen Git [2]. Leider scheint das Repository von den Entwicklern nicht besonders gut gepflegt zu werden. Im Branch "dev" finden Sie etwas aktuellere Sourcen für die Installation. Detectree ist in TypeScript entwickelt, daher benötigen Sie eine Node.JS-Installation auf Ihrem Computer. Um die benötigten Abhängigkeiten zu installieren und Detectree im Development-Modus zu starten, rufen Sie innerhalb des detectree-Verzeichnisses die folgenden Kommandos auf:

```
npm install

npm run dev
```

Sie erhalten einige Debug-Ausgaben und Informationen des Svelte-Frameworks, das die Entwickler für die GUI-Berechnungen verwenden. Nach dem Start können Sie mit Ihrem Browser unter "https://localhost:3000" auf das Werkzeug zugreifen. Da noch kein Datenbackend konfiguriert ist, sehen Sie zunächst ein leeres Interface.

Detectree unterstützt in der vorliegenden Version ausschließlich Elasticsearch als Speicherort für die Logdaten. Mit etwas Erfahrung bei der Entwicklung mit TypeScript fügen Sie Adapter für weitere Backends hinzu. Sie können sich dafür an der Datei "elastic.ts" im Ordner "src/backend_adapters/" orientieren. Sie müssen am Ende nur die Funktion "query" implementieren, um die Daten aus der Datenbank zu erhalten. Beachten Sie dabei aber, dass Sie das später diskutierte Mapping der Felder innerhalb des Adapters erfolgt.

Leider ist die Fehlerausgabe von Detectree nicht immer hilfreich, wenn eine Fehlkonfiguration oder Inkompatibilität des Datenbank-Backends vorliegt. In unseren Tests haben unterschiedliche Versionen der eingesetzten Elasticsearch-Instanz und der von Node installierten Elasticsearch-Bibliothek verhindert, dass Daten aus der Datenbank erfolgreich abgefragt werden können. Achten Sie also darauf, dass Sie die Version der Bibliothek in der Datei "package.json" entsprechend Ihrer Elasticsearch-Version anpassen.

Für den Start ist es hilfreich, auf eine bereits existierende Elasticsearch-Instanz mit Logdaten zugreifen zu können, allerdings müssen Sie die Dokumente darin möglicherweise etwas anpassen. Um Detectree zu testen, bietet sich an, zunächst eine lokale Instanz aufzusetzen und einige vorhandene Daten in diese Instanz zu importieren. Für die Konfiguration der Datenbank nennen Sie die mitgelieferte "schema.yml.example" in "schema.yml"

um. Unter "backend" hinterlegen Sie die Verbindungsparameter zu Ihrer Datenbank sowie den verwendeten Index. Wenn Sie statt eines API-Keys für die Authentifikation lieber Benutzername und Passwort verwenden möchten, können Sie die Zeile mit dem "apiKey" kommentieren und jeweils eine Zeile für "username" und "password" hinzufügen.

Damit Detectree die Daten grafisch aufbereiten kann, müssen verschiedene Felder existieren. Zunächst passen Sie die in der Datei "schemal.yml" referenzierten Felder für die "primary ID" und das "timeField" an. Dabei handelt es sich um die ID des Dokuments sowie um den Zeitstempel. Diesen benötigt Detectree, um die Suchergebnisse zeitlich einzuschränken. Im Feld "source" können Sie eine Kategorie der Dokumente in der Datenbank berücksichtigen.

Mapping konfigurieren

Detectree benötigt weitere, als notwendig und optional deklarierte Felder. Das Bild zeigt eine entsprechende Übersicht. Da vermutlich nicht alle Felder mit der korrekten Bezeichnung in Ihrer Datenbank vorliegen, nutzt Detectree ein konfigurierbares Mapping, um die Informationen in Ihrer Datenbank korrekt zu interpretieren.

Die Konfiguration des Mappings ist die wichtigste Aufgabe bei der Einrichtung von Detectree. Dafür öffnen Sie wieder die Datei "schema.yml" und schauen sich die einzelnen Mappings an. Für die unterschiedlichen Felder können Sie nun die tatsächlichen Felder Ihrer Datenbank einsetzen, auch mit unterschiedlicher Tiefe der Felder innerhalb eines Dokuments. JSON-typisch trennen Sie hier die Ebenen einfach mit einem Punkt ab. Der Datenbank-Adapter kümmert sich anschließend bei der Abfrage der Daten um die entsprechende Zuordnung der Werte.

Es ist möglich, dass sich nicht alle notwendigen Felder in Ihrer Datenbank einsortieren lassen. Leider sind innerhalb des Mappings keine Default-Werte festlegbar. So haben Sie vielleicht keine Entsprechung für "Detection Name" in Ihren Daten und gelegentlich fehlt auch ein Äquivalent zu "Severity". Um Detectree trotzdem verwenden zu können, haben Sie nun zwei Möglichkeiten. Entweder Sie erweitern alle existierenden Dokumente mit eigenen Default-Werten und passen die Felder aus Ihrem Logging für zukünftige Einträge an oder Sie bearbeiten auch in diesem Fall den bereits erwähnten Datenbank-Adapter und setzen dort einfach vor der Rückgabe entsprechende Default-Werte.

Informationen sammeln

Wenn Sie in Detectree gezielt nach Einträgen suchen möchten, können Sie das Suchfeld auf der Webseite verwenden. Ist dieses in Ihrer Ansicht noch nicht geöffnet, klicken Sie einmal auf das Baumsysmbol in der linken oberen Ecke. Wählen Sie nun ein Zeitintervall aus, das Sie betrachten möchten. Dafür geben Sie das Start- und das Enddatum neben dem Suchfeld an. Bei der Suche verwenden Sie die klassische Syntax "Feld: Wert", wie Sie diese bereits von anderen Elasticsearch-Werkzeugen kennen. Um etwa nach allen Dokumenten zu suchen, die sich auf iexplore.exe beziehen, geben Sie "Process Name":"iexplore.exe" ein.

Field	Required
Endpoint ID	yes
Category	yes
Severity	yes
Detection Name	yes
Parent Process Name	yes
Parent Process ID	yes
Process Name	yes
Process ID	yes
User	no
Command Line	no
File Name	no
Registry Key	no
Network Address	no
Target Process Name	no
Target Process ID	no

Notwendige und optionale Felder für die Darstellung in Detectree.

Als Analyst haben Sie vor allem Interesse daran, viele Informationen übersichtlich aufbereitet zu bekommen. Die in der Oberfläche angezeigten Informationen sind aber doch erst einmal eher einfach gehalten – Detectree fokussiert ja auf Beziehungen und Hierarchien der Objekte. Wenn Sie nun aber mit der Maus über ein Objekt fahren, erhalten Sie in einem Tooltipp weitere Informationen zu diesem, vor allem auch die Werte der als optional gekennzeichneten Felder.

Detectree unterstützt unterschiedliche Typen, die Sie in der Toolleiste auf der linken Seite auswählen können. Hier bestimmen Sie, ob einzelne (nicht verbundene) Objekte, Dateien, Netzwerk- oder Registry-Objekte angezeigt werden sollen oder nicht. Je nachdem, wie viele Logdaten dargestellt werden, kann das vereinzelte Ausblenden bei der Orientierung helfen.

Fazit

Der grundlegende Ansatz, Zusammenhänge bei der Sicherheitsanalyse grafisch darzustellen, ist nicht neu. Die Vereinfachung der Analyse sowie die Darstellung von Zusammenhängen und Hierarchien in Form von Baumdiagrammen sind vorgebliches Ziel von Detectree, das Ihnen der Security-Tipp in diesem Monat vorgestellt hat. Dabei überzeugt die zunächst schlicht gehaltene Oberfläche des Tools durchaus. Weitere Informationen werden bei Bedarf über Popups eingeblendet.

Die Interaktionsmöglichkeiten, Exporte und Importe weiterer Informationen sowie die Verbindung mit anderen Analysetools benötigen für einen regelmäßigen Einsatz aber noch etwas mehr Aufmerksamkeit der Entwickler. Auch die Konfiguration der Mappings vor der ersten Verwendung ist etwas verwirrend ausgestaltet und benötigt ein weitergehendes Verständnis in der Funktionsweise des Programms. *(dr)* IT

Link-Codes

[1] WithSecure Detectree
n3pd1

[2] Repository auf GitHub
n3pd2

12 Monatsausgaben im Print- & E-Paper-Format

+ Zwei Sonderhefte im Print- & E-Paper-Format

+ Zugang zum Heftarchiv online sowie Jahres-CD

mit allen Monatsausgaben im PDF-Format

Das Abo All-Inclusive

shop.heinemann-verlag.de

Abo- und Leserservice IT-Administrator

vertriebsunion meynen
Herr Stephan Orgel
D-65341 Eltville
Tel: 06123/9238-251
Fax: 06123/9238-252
leserservice@it-administrator.de

IT Administrator