Anmerkungen zur Netzwerksicherheit

Im März 2024 stieß ich auf einen scheinbar harmlosen Diskussionsbeitrag zu einem Artikel „IPv6 statt IPv4: Nicht wollen oder nicht können?“ (1), in dem mögliche Gründe dafür angerissen wurden, warum IPv6 immer noch ein Schattendasein führt, obwohl aufgrund der IPv4-Verknappung inzwischen immer mehr Provider nur noch CGNAT-Anschlüsse mit DS-Lite anbieten.

Die Frage in dem Beitrag lautete: „*Wie verhindere ich bei wechselnden IP-Adressen und ausgewürfelten MAC-Adressen im Heimnetz das Nach-Hause-Telefonieren eines beliebigen Gerätes (Drucker, Fernseher etc.) über IPv6?*“

Überraschend waren die Antworten, weil die meisten offenbar die Dimension der Frage gar nicht verstanden hatten, beispielsweise, weil sie Adressvergabe via SLAAC vs. DHCP diskutierten, dabei aber „IPv6 Privacy Extensions“ ausklammerten. Das war der Anstoß für dieses Dokument, dass sich etwas grundlegender mit den Fragen beschäftigen wird, ehe wir zu einer Antwort kommen.

Wieso diese Frage?

In der „alten“ (IPv4) Welt konnte man Firewall-Regeln für IPv4 Adressen erstellen, die für ein bestimmtes Gerät den Internet-Zugang verhindert.

Im Fall von IPv6 erfolgt die Adressvergabe meist via SLAAC (d. h. abgeleitet aus der EUI-64, also indirekt aus der MAC des Geräts) oder per DHCPv6, meist abgeleitet aus einem global routebaren IPv6-Präfix, der häufig dynamisch durch den ISP bereitgestellt wird. Eine statische Zuweisung ist möglich, muss aber den aktuellen IPv6-Präfix berücksichtigen, damit sie funktioniert.

Aufgrund des Umstands, dass ein Gerät unter IPv6 immer mehrere Adressen haben darf, kann man neben der ursprünglichen IPv6 mittels „IPv6 Privacy Extensions“ zusätzlich zufällig erzeugte Adressen für ausgehende Verbindungen verwenden, um ein Tracking zu verhindern oder zu erschweren.

Da diese IPv6-Adressen nicht vorab bekannt sind, kann man sie auch nicht per Firewall-Regel blockieren – zumindest nicht anhand der Quell-IP.

Blockieren möchte man einzelne Geräte z. B., damit es ihnen nicht möglich ist, „nach Hause zu telefonieren“, d. h. Verbindung zu ihrem Hersteller aufzunehmen und dabei ungewollt private Daten auszuplaudern oder mittels „Firewall-Piercing“ Sicherheitslücken aufzumachen, falls der Hersteller einer Attacke ausgesetzt wurde oder selbst Böses im Schilde führt.

Solche „Reverse Tunnel“ unterminieren den Zweck einer Firewall komplett, weil sie, obwohl eine Verbindungsaufnahme von außen eigentlich immer geblockt wird, in der Gegenrichtung genau die Schlupflöcher schaffen, die man nicht gern haben möchte.

Eine kleine Geschichte illustriert das: Ein Bekannter hatte zwei Webcams eines deutschen Herstellers gekauft. Diese stellten – ganz auf die Kundenwünsche abgestellt – den Zugriff auf das Bildmaterial per Cloudlösung zur Verfügung. Auf die potentielle Gefahr hingewiesen, erwiderte er, dass es ja ein deutsches Produkt sei, dem man ohne Weiteres vertrauen könne.

Etwas später nahm er anstelle einer Fritzbox eine OpnSense in Betrieb. Dort fiel auf, dass es eine dauernde Verbindung von seinen Webcams ins Internet gab. Eine Abfrage der Ziel-IPs ergab, dass

diese in der Alibaba Cloud lagen. Der Chinesische Alibaba-Konzern betreibt ähnlich wie Amazon, Google und andere Cloudservices und unterliegt der Einsichtnahme durch chinesische Behörden.

Was sich hier zeigte, war, dass das „deutsche Produkt“ nichts anderes als eine rebrandete Lösung eines chinesischen Herstellers war, die der deutsche „Hersteller“ nur unter eigenem Namen vertreibt – inklusive der integrierten, gebrandeten Cloud-Lösung. Willkommen in der wirklichen Welt!

Warum sind einige implizite Annahmen falsch?

Zum einen ist es eben nicht so, dass eine freie Adresswahl nicht auch unter IPv4 möglich wäre: Auch dort kann man mittels statischer Vergabe an DHCP vorbei andere IPs nutzen (auch zusätzlich), zudem ließe ich mit einer gefälschten MAC ein anderes Gerät emulieren, dass eine andere IP per DHCP erhielte. Bei modernen iOS-Versionen und Android, wird die MAC zur Vermeidung der Identifikation eines Geräts schon standardmäßig „gefälscht“.

Noch bevor ein Gerät überhaupt ins Netzwerk darf, müsste man es authentifizieren – wobei Mechanismen wie IEEE 802.1x (falls sie überhaupt im Heimnetz zur Verfügung stehen) gerade bei IoT-Geräten die ja das größte Problem darstellen, mangels Konfigurationsmöglichkeit nicht mit Zertifikaten umsetzbar sind, womit wir wieder bei MAC-basiertem Zugang wären, der leicht gefälscht werden kann. Für WLANs gilt das ebenso, denn damit das Gerät überhaupt im Netzwerk arbeiten kann, muss der Zugang per WLAN-Passwort ja erlaubt werden.

Außerdem: Wenn ein Gerät erst einmal im (V)LAN zugelassen ist, kann es danach oft immer noch statisch konfiguriert werden, wenn nicht andere Maßnahmen wie DHCP-Guarding greift und der Port auf die anfangs vorgelegte MAC beschränkt wird. Solche Möglichkeiten bietet nur Switches der Enterprise-Klasse.

Die „Zufälligkeit“ der verwendeten Adressen endet für praktische Zwecke aber bei IPv4 genau wie bei IPv6 ohnehin nicht bei der verwendeten Absender-IP, sondern bei der MAC. Deswegen ist es erstens hilfreich, wenn die Firewall MAC-basierte Regeln unterstützt (OpnSense z. B. beherrscht das) und zweitens, wenn man nicht versucht, bestimmte Geräte zu blockieren, sondern eher, bestimmten Geräten den Zugriff zu gewähren, aber generell für solche IoT-(V)LANs den Internet-Zugriff zu verbieten. Damit müsste ein IoT-Gerät, um sich Zugriff zu verschaffen, eine der zugelassenen MAC-Adressen kennen und diese fälschen, was nebenbei zu Adresskonflikten führt.

Netzsegmentierung und -zugangskontrolle

Voraussetzung für eine vernünftige Kategorisierung und Berechtigung von Geräten ist eine sinnvolle Segmentierung des Netzwerks in verschiedene Bereiche (VLANs / WLANs) und Regelung des Verkehrs zwischen diesen Bereichen.

Im Fall WLAN kann man auch im Heimnetzumfeld verschiedene Bereiche schaffen, wobei man durch Bekanntgabe der Credentials reglementieren kann, welches Gerät was darf – wichtig ist allerdings Disziplin bei der Weitergabe dieser Credentials (z. B. nicht Gästen „aus Versehen“ – oder Faulheit das WLAN-Passwort für das eigene LAN zu geben).

Bei Kabel-(V)LANs hilft entweder nur eine Technologie wie IEEE 802.1x, bei der Geräte identifiziert und dynamisch auf ihr entsprechendes VLAN aufgeschaltet werden oder eine physische Zugriffskontrolle – d. h. Verhinderung, dass eine Gerät an einen privilegierten Port angeschlossen wird, um höhere Rechte zu bekommen.

Eine sinnvolle Aufteilung könnte z. B. so aussehen:

Bezeichnung	**VLAN**	**WLAN-Name**	**Zweck / Rechte**
LAN	1	Family	Haupt-Netzwerk für vertrauenswürdige Geräte, d. h. PCs, Server, NAS usw. - Voller Internet-Zugriff - Zugriff untereinander - Zugriff auf IoT und DMZ
IoT	2	IoT	Netzwerk für Geräte, die „nach Hause telefonieren könnten“ und denen man nicht voll vertrauen kann, z. B. Webcams, Drucker, Handys, Smart-Home-Devices, Streaming-Boxen, Haushaltsgeräte usw. - Internet-Zugriff - Kein Zugriff auf andere VLANs - Im Idealfall „Netzwek Isolation“, d. h. keine Sichtbarkeit untereinander (falls verfügbar und nicht im Widerspruch zur angestrebten Nutzung)
Gast	3	Gast	Netzwerk als reines Zugangs-WLAN für Gäste, um diesen Internet-Zugang zu gewähren. - Internet-Zugriff - Zugriff maximal auf IoT, eventuell spezifisch, z. B. nur für Drucker - Netzwerk Isolation, d. h. keine Sichtbarkeit untereinander Eventuell ohne WLAN-Passwort, d. h. offenes WLAN, allerdings: „Störerhaftung“ und eventuell Nutzung durch jedermann (falls möglich: Bandbreitenbeschränkung)
DMZ	4	-	Betrieb von Diensten, die man per Internet erreichbar machen will, z. B. E-Mail-, Backup- oder Fileserver - Internet-Zugriff - Eingehender Zugriff (selektiv) - Kein Zugriff auf andere VLANs

Alternative Zero Trust

Eine andere Variante wäre, das Heimnetzwerk grundsätzlich mittels „Zero Trust“ zu verwenden, d. h. alle Geräte stets so zu behandeln, als befänden sie sich in einem offenen, potentiell feindlichen Umfeld. Das setzt voraus, dass sämtliche ggf. auf den angeschlossenen Geräten verfügbaren Dienste mittels Passworten oder Zertifikaten geschützt und idealerweise auch verschlüsselt sind.

Dazu kann man LetsEncrypt-Zertifiate oder eine eigene CA verwenden.

Randbemerkungen zu Port-Scans o. ä.

Ein verbreiteter Irrtum zu IPv6 ist, dass damit anders als bei IPv4 keine Port-Scans mehr durchgeführt werden können, weil der Adressbereich so groß ist, dass man ihn nicht mehr sinnvoll scannen kann.

Das ist nämlich auch gar nicht mehr notwendig:

Um Geräte aus dem Internet verfügbar zu machen, muss man entweder einen Reverse-Proxy (Caddy, HAproxy, Apache oder NGinx) verwenden, der den Traffic an der Firewall terminiert und dann z. B. bei HTTP(S) namensbasiert an das Ziel-Gerät weiterleitet oder mindestens einen Port öffnen. Um die Firewall oder das Endgerät zu adressieren, wird man typischerweise einen dynamischen DNS-Eintrag nutzen.

Die Reverse-Proxy-Lösung bietet scheinbar den Vorteil, dass man somit hinter dem *einen* offenen Port 443 (für HTTPS) *mehrere* Namen für interne Geräte bzw. Services „verstecken“ kann – wer den Namen nicht kennt, kann den Dienst bzw. das Gerät nicht erreichen und somit auch nicht auf Schwachstellen testen.

Einerseits muss es dann allerdings so sein, dass die DNS-Namen nicht per Zonentransfer abfragbar sein dürfen oder dass man mit einem DNS-Wildcard arbeitet, der keine Rückschlüsse auf die verwendeten Namen zulässt.

Das allein reicht aber nicht aus – nutzt man z. B. LetsEncrypt, kann man mittels der Authentifizierungsmethode http-01 nur konkrete Namen in ein Zertifikat aufnehmen. Damit hier DNS-Wildcards möglich sind, muss man zwingend dns-01 als Authentifizierungsmethode verwenden.

Warum das eine Rolle spielt?

Einerseits müsste man im Reverse Proxy dafür sorgen, dass das oder die Zertifikate nicht auf erste Anforderung ohne konkreten Zugriff auf einen speziellen Hostnamen vorgelegt wird. Sonst bekäme ein Angreifer beim zufälligen Finden eines offenen Ports ja die möglichen Hostnamen bereits präsentiert – eventuell sogar in einem Multi-Domain-Zertifikat.

Andererseits gibt es heutzutage „Certificate Trust“: Seit einigen Jahren veröffentlichen alle nennenswerten Zertifizierungsstellen sämtliche ausgestellten Zertifikate. Wenn man z. B. ein Zertifikat für abc.xyz.de ausstellen lassen hat, ist das im Internet publik.

Ein potentieller Angreifer hat damit die Möglichkeit, Scans nicht einfach zufällig per IP, sondern gezielt mittels des Namens, der im Zertifikat genannt wurde, durchzuführen. Damit bietet sich die Möglichkeit, Scans auch für IPv6-Services sehr gezielt und effizient zu machen.

Unsere Beobachtungen zeigen, dass diese Möglichkeit tatsächlich genutzt wird, so erfolgten Zugriffe auf Systeme per IPv6 hinter Carrier-Grade-NAT-Anschlüssen der Deutschen Glasfaser, die konkret auf die EUI-64-IPv6 der Firewall erfolgten – es ist extrem unwahrscheinlich, dass diese zufällig entdeckt wurden. Auslöser war eine IPv6, die mittel ipinfo.io einem Betreiber „internet-measurement.com“ zugeordnet werden konnten. Greift man auf dessen Webseite zu, findet man den Dienst Driftnet.IO, der auf seiner Webseite konkret die Recherche per Domain-Namen anbietet, bei der man u. a. über den „Certificate Trust“ verfügbare Informationen auslesen kann.

Wir empfehlen, dies einmal mit den eigenen (DynDNS-)Domains auszuprobieren: https://driftnet.io

Ein weiterer beobachteter Scan scheitete daran, dass wir per Geolocation bestimmte Länder komplett blockieren. Der versuchte Zugriff konnte der Universität von Shenzen zugeordnet werden – ein Schuft, der Böses dabei denkt.

Übrigens: eine eigene CA kann man natürlich nutzen – diese wird auch nichts per „Certificate Trust“ verraten. Nur: Wildcard-Zertifikate funktionieren in modernen Browsern nur noch für Subdomains, nicht mehr für Top-Level-Domains. Wenn man also z. B. im internen Netzwerk Namen wie abc.local verwendet möchte und dafür *.local als Zertifikat nutzen will, um sich Arbeit für jedes neue Gerät zu sparen, funktioniert das nicht. Ein Zertifikat für *.xyz.local würde dagegen akzeptiert.

Also - **wenn** Dienste angeboten werden:

1. Patchen, Patchen, Patchen!
2. Idealerweise hinter einem Proxy mit Wildcard-Zertifikaten
3. DNS-Einträge ohne Möglichkeit für Zonentransfers und / oder DNS-Wildcards – *keine* Multi-Domain-Zertifikate
4. Folglich im Fall von LetsEncrypt: Verifikation per DNS-01
5. Betrieb ausschließlich in einer DMZ, damit bei etwaigen Einbrüchen kein „lateral movement“ in andere Teile des Netzwerks erfolgen kann

Fazit

Erinnern Sie sich noch an die anfangs gestellte Frage? Die vollständige Antwort darauf lautet:

1. Das zugrundeliegende Problem ist, dass es Geräte im Heimnetz geben könnte, die unter Nutzung nahezu beliebiger Adressen ein „Verbot“, sich mit dem Internet zu verbinden, umgehen könnten. Es wird konkret vermutet oder indirekt postuliert, dass dies zu verhindern, mit IPv6 schwieriger (oder unmöglich) sei, während es mit IPv4 leicht zu realisieren ist. Die Annahme ist, dass man mittels IPv6 Privacy Extensions Absende-IPs verwenden kann, die durch eine fiktive Verbots-Regel umgangen würden.
2. Zunächst muss man sich klarmachen, dass bereits mit IPv4 die Nutzung nahezu beliebiger Adressen möglich ist. Dies gilt sogar, wenn man den Netzzugang per 802.1x reglementiert,

wenn parallel kein IP Guarding betrieben wird – was meist außerhalb der Möglichkeiten eines normalen Heimnetzbetreibers liegt.

3. Man sieht sehr schnell, dass es weniger um das „Verbieten“ bestimmter Aktivitäten für einzelne Geräte, als das „Erlauben“ solche Aktivitäten für andere geht. Eine Netzsegmentierung in verschiedene VLANs und Klassifizierung von Gerätetypen bietet einen ersten Ansatz. Sofern man nicht die Möglichkeit zur Kontrolle des Netzzugangs per 802.1x hat, ist ein physischer Zugangsschutz für LAN-Geräte und eine Sicherung der WLANs hoffentlich gegeben. Damit lassen sich VLANs schaffen, in denen Geräte, die eben gar keinen Internet-Zugang haben sollen, gebündelt werden.

4. Die scheinbar „bequeme“ Möglichkeit, ein bestimmtes Gerät zu sperren, stößt ohnehin an ihre Grenzen. Um zu verhindern, dass einzelne Geräte Sicherheitslücken aufreißen, kommt man eigentlich um das Gegenteil, nämlich die Auflistung derjenigen Geräte, denen man unterstellt, dass sie das nicht tun und die deshalb Internet-Zugang erhalten, also eine Whitelist.

5. Bei IPv6 kann man hoffentlich deren EUI-64-Adresse nutzen, bei IPv4 hilft DHCP Guarding, bei dynamischen IPv6 Präfixen hilft nur eine Firewall, die den Präfix in Regeln abstrahiert oder Einsatz von ULAs und NAT. in jedem Fall hilft eine Firewall, bei der man Regeln auf Basis von MAC-Adressen definieren kann. Man beachte: Ein Angreifer, der die MAC-Adressen fälschen kann, kann diese Schutzarten immer noch umgehen, indem er sein Gerät für ein anderes ausgibt.

(1) https://www.heise.de/forum/heise-online/Kommentare/IPv6-statt-IPv4-Nicht-wollen-oder-nicht-koennen/forum-536762/comment/

(2) https://www.heise.de/forum/heise-online/Kommentare/IPv6-statt-IPv4-Nicht-wollen-oder-nicht-koennen/Heimnetz-DEVICE-sperren-bei-IPv6/posting-43780117/show/