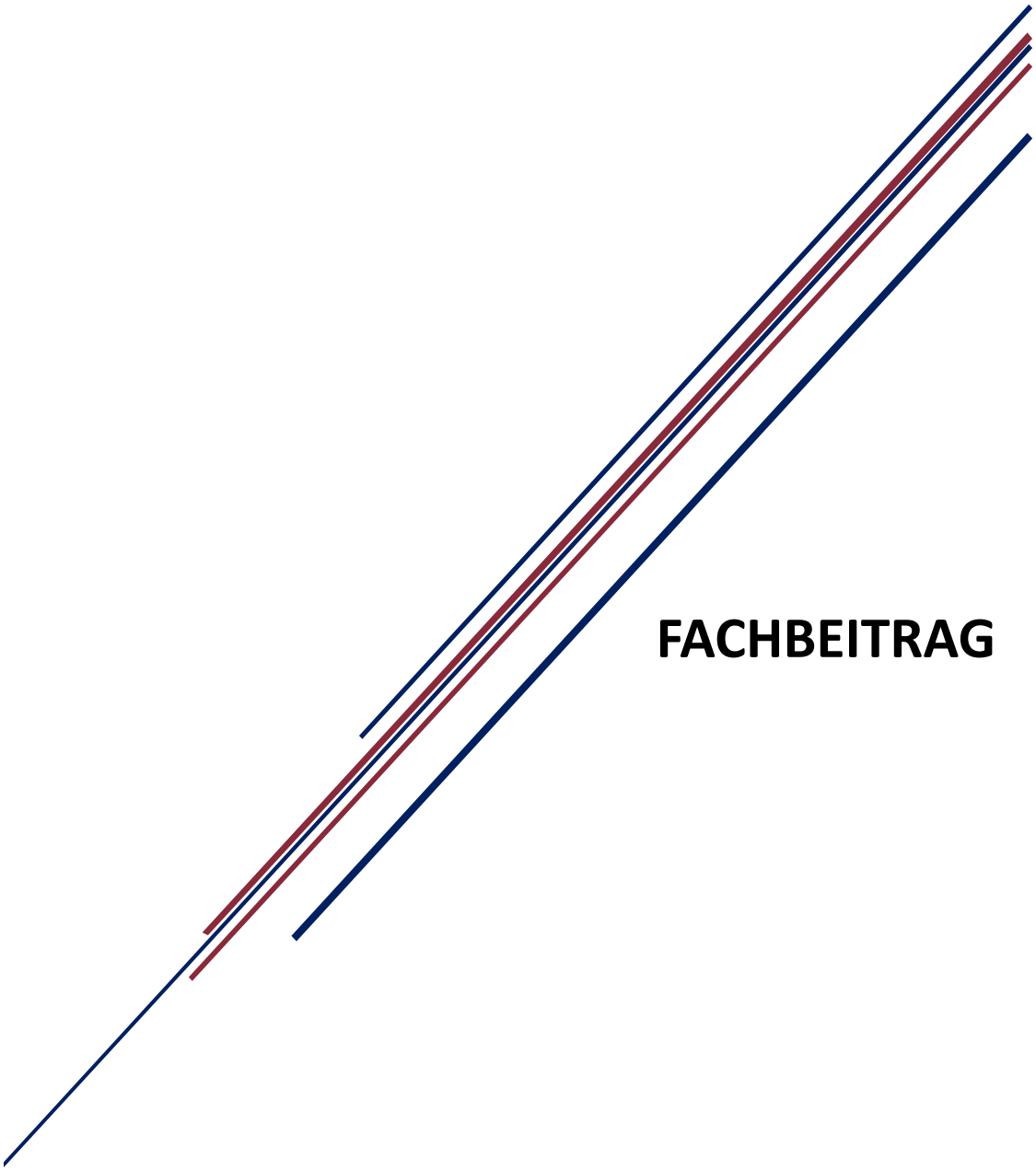
IT-SECURITY IN OT-UMGEBUNGEN

Vulnerabilität und Auftretenshäufigkeit ungeschützter Modbus/TCP-Systeme im öffentlichen IP-Adressbereich



FACHBEITRAG

Nils Kruchem
Bücker IT-Security GmbH

Einleitung

In zahlreichen Produktionsumgebungen, jedoch auch in der Gebäudeautomation und in Energieanlagen sowie weiteren Anlagen der Versorgungstechnik werden großflächig industrielle Datenübertragungsprotokolle eingesetzt, welche in der herkömmlichen IT-Infrastruktur weitgehend unbekannt sind. Derartige Protokolle werden zumeist von Feldbus-Systemen genutzt und dienen ganz überwiegend der Abwicklung steuerungs- und regelungstechnischer Operationen. Dabei zeichnet sich schon seit vielen Jahren die Abkehr von seriellen Übertragungsstandards in sich geschlossener Systeme hin zu ip-basierten Feldbussen ab. Darüber hinaus werden seriell kommunizierende Anlagenabschnitte mit Serial-to-Ethernet-Convertern an die ip-basierte Welt angebunden. Bessere Vernetzbarkeit, einfachere Konfiguration und Multimasterfähigkeit sind nur einige Vorteile, die aus diesen Entwicklungen resultieren. Da die genutzten Protokolle leistungsfähig, universell, einfach zu handhaben und bei technischen Spezialisten etabliert sind, hat sich an ihnen oft schon seit Jahrzehnten nur wenig geändert. Vor allem die oft fehlende oder nur schwache Verschlüsselung und das mangelnde Bewusstsein über die Erreichbarkeit im öffentlichen IP-Adressbereich bei vernetzten Anlagen gehen massiv zulasten der informationstechnischen Sicherheit. Aufgrund der enormen Nachholbedarfe in diesem Bereich soll anhand des weltweit stark verbreiteten Feldbusses Modbus/TCP exemplarisch gezeigt werden, wie angreifbar derartige Systeme sind und wie häufig man sie komplett ungeschützt auffindet. Modbus wurde bereits Ende der 1970er Jahre entwickelt und kennt mit dem rein analogen Modbus/ASCII, dem digitalen Modbus/RTU (Remote Terminal Unit) und dem Modbus/TCP drei Grundausprägungen. In diesem Beitrag wird es aufgrund der weltweit stark ausgeprägten Nutzung ip-basierter Systeme nur um den Modbus/TCP gehen, welcher stets gemeint ist, wenn von Modbus die Rede ist.

1. Kurze Beschreibung des Modbus-Protokolls

Da sich dieser Beitrag nicht nur an Fachexperten für OT-Security richtet, sondern auch an informationstechnische Mitarbeiter:innen aus der klassischen IT-Infrastruktur, die möglicherweise eine OT-Umgebung mit zu betreuen haben und nicht über entsprechende Protokollkenntnisse verfügen, werden an dieser Stelle die für die weiteren Ausführungen relevanten Teile des Modbus-Protokolls kurz erklärt. Bis auf die im Jahr 2018 veröffentlichte Variante des Modbus/TCP mit TLS-Verschlüsselung, welche bislang nicht flächendeckend eingesetzt wird, ist das Protokoll unverschlüsselt und besitzt keine native Authentifizierung. Das Protokoll kennt vier Datentypen im 1-Bit- und 16-Bit-Bereich, von denen zwei nur gelesen und zwei sowohl gelesen und auch überschrieben werden können. Relevant ist hier der Datentyp Holding Register, welcher 16 Bit hat und somit einen zweistelligen Analogwert abbilden kann. Holding Register lassen sich sowohl auslesen als auch

Nils Kruchem

T +49 5703 / 930 360
nils.kruchem@buecker-edv.de

Bücker IT-Security GmbH
Ein Unternehmen der Bechtle-Gruppe

Nordhemmer Straße 97,
32479 Hille

T +49 5703 / 930 0

info@buecker-edv.de
www.buecker-it-security.de

überschreiben. Zur Abbildung längerer Analogwerte können mehrere Holding Register miteinander verbunden werden. Damit Holding Register angesprochen werden können, hat jedes Register eine spezifische Registeradresse. Registeradressen können die Werte 0 und alle natürlichen Zahlenwerte bis 65335 annehmen. Folglich sind bis zu 65336 verschiedene Registeradressen theoretisch möglich. Gemeinhin werden in etwa die Registeradressen 0 bis 120 am häufigsten genutzt. Ein möglicher Anwendungsfall für Holding Register wäre zum Beispiel das Auslesen der Ist-Temperatur und das Festlegen der Soll-Temperatur von industrieller Thermoprozesstechnik. Die Datentypen werden über unterschiedliche Funktionscodes angesprochen. Mit dem Funktionscode 3 (0x03) lassen sich ein oder mehrere Holding Register auslesen. Ein weiterer Funktionscode, der für die weiteren Ausführungen relevant ist, ist der Funktionscode 16 (0x10) mit welchem sich mehrere Holding Register überschreiben lassen. Das gesamte Protokoll ist nicht gerätebezogen und bietet eine hohe Universalität. Hersteller können den Datentypen, Funktionen und Registeradressen individuell Bedeutungen zuweisen, welche den Benutzerhandbüchern der Geräte im Regelfall zu entnehmen sind. Der Umstand, dass viele Benutzerhandbücher im Internet zur freien Verfügung stehen, erleichtert Angreifern die Manipulation der Systeme. Der Standardport für Modbus/TCP ist der Port 502. Die Abbildungen 1 und 2 zeigen, dass sich Modbus-Datenverkehre aufgrund der fehlenden Verschlüsselung in Klartext mitlesen und aufgrund der oft fehlenden Authentifizierung zudem leicht manipulieren lassen. Im vorliegenden Beispiel wurde die Software zur Analyse von Datenpaketen in Netzwerkverkehren Wireshark eingesetzt.[1]

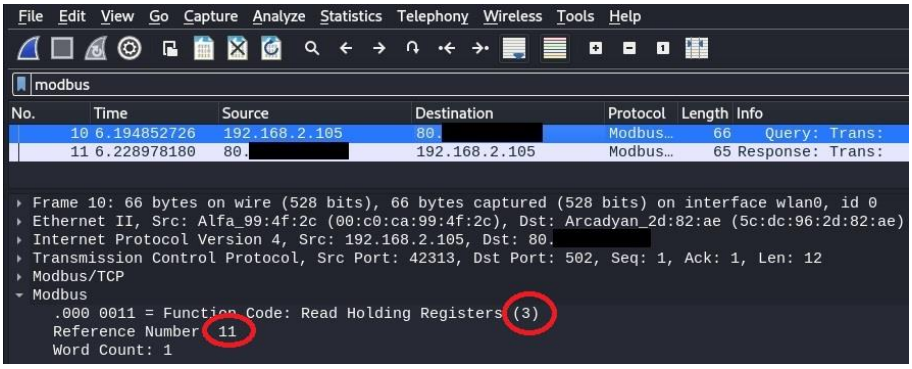


Abb. 1: Modbus Request, mit dem Funktionscode 3 (Read Holding Registers) wird der Wert des Holding Registers mt der Registeradresse 11 abgefragt. Die relevanten Klartextdaten sind rot markiert. Quelle: Wireshark

[1] Wireshark, Software zur Analyse von Datenpaketen in Netzwerkverkehren, Version 4.0.10 für Linux

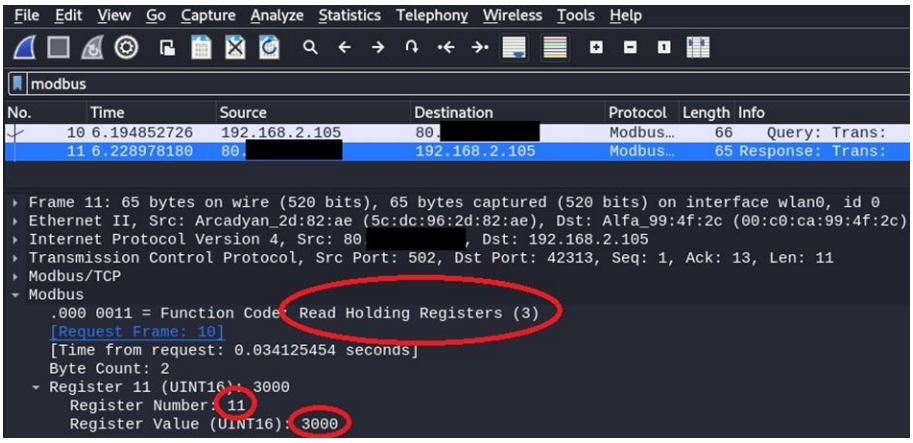


Abb. 2: Modbus Response, der Wert des Holding Registers mit der Adresse 11 beträgt 3000, Quelle: Wireshark

2. Deskriptiv-statistische Betrachtungen mithilfe der Spezialsuchmaschine Shodan

Mit einer ersten Abfrage bei Shodan ausschließlich nach dem offenen Port 502 werden weltweit 452.117 Treffer (Stand 29.10.2023) angezeigt.[2] Nach Analyse einiger Stichproben ist davon auszugehen, dass ein Großteil der Treffer Honeypots sind, also unechte Ziele, welche gemeinhin überwiegend zur Erhebung von Daten zu aktuellen Angriffswellen oder aber zur Ablenkung von einem echten Ziel ins Netz gestellt worden sind. Aufgrund ihrer spezifischen Beschaffenheit und Verhaltensweise werden Honeypots über Shodan leicht erkannt und mit einem Tag versehen, um sie gezielt finden oder bei Suchabfragen ausschließen zu können. Da sich bei Modbus-Systemen per Shodan oder mit speziell dafür geeigneten Netzwerkscannern die Device Identification über den Port 502 in Klartext auslesen lässt und Systeme, bei denen weder Lese- noch Schreibzugriffe gestattet sind, einen Fehlercode zurückgeben, lässt sich der Search Query nun unproblematisch so anpassen, dass man in seiner Ergebnisgesamtheit einen sehr hohen Anteil an Modbus-Systemen erhält, bei denen sich die Datentypen unproblematisch auslesen und überschreiben lassen. Da es selten ist, dass sich Werte in Modbus-Systemen mit dem zuvor beschriebenen Merkmal der in Klartext lesbaren Device Identification nicht überschreiben lassen, bekommt man mit dem angepassten Search Query folglich eine Vielzahl von Informationen über vulnerable Systeme präsentiert. Konkret bietet sich hier der Search Query *„port:502 "device identification" -Illegal -Error“* an. Dieser liefert immer noch 6.715 Ergebnisse (Stand 29.10.2023) weltweit. Davon fallen 182 Ergebnisse auf Deutschland. Da über Shodan nicht alles gefunden werden kann, zumal es vor kurzer Zeit aufgrund einer Systemmigration neu aufgesetzt werden musste, dürfte die Anzahl tatsächlich im Netz erreichbarer Modbus-Systeme noch wesentlich höher sein.

[2] Shodan auf www.shodan.io, Zugriff: 29.10.2023, 15:47 Uhr MEZ

Abb. 3: Auslesen der Device Identification mit Shodan. Im vorliegenden Beispiel handelt es sich um eine SPS des Herstellers Schneider Electric vom Typ TM251MESE, der bei Shodan übliche MMH3-Hash zur Auffindbarkeit wurde unkenntlich gemacht, Quelle: Shodan.io

3. Mögliche Angriffsszenarien und ihre Folgen

Ein Angreifer kann versuchen, ein Modbus-System durch wiederholte Anfragen zu überlasten, um den normalen Betrieb zu stören. Dies kann durch Fluten des Netzwerks mit Modbus-Anfragen oder durch Senden ungültiger Anfragen erreicht werden, die vom System verarbeitet werden müssen. Diese DoS-Angriffe (Denial of Service) können dazu führen, dass das Modbus-System verzögert reagiert oder es gar zu einem Systemausfall kommt.[3] Dies kann die Prozesse in Industrieumgebungen stören und Qualitätsprobleme sowie erhebliche finanzielle Verluste verursachen. Befindet sich der Angreifer schon im Netzwerk, sind Man-in-the-Middle-Angriffe möglich. Ein Angreifer kann beispielsweise versuchen, den Datenverkehr zwischen einer Steuerung und einem Aktor zu manipulieren. Dies ermöglicht es dem Angreifer, Daten zu lesen, zu ändern oder zu injizieren, ohne dass die beteiligten Parteien davon Kenntnis nehmen. Beispielsweise wäre es denkbar, ein Abschaltsignal an einen Elektromotor automatisiert per Skript in ein Anschaltsignal umzuwandeln, um den Versuch der Abschaltung des Motors wirkungslos werden zu lassen.[4] Auch ist es bei ungeschützten Modbus-Systemen möglich, von außen sämtliche Datenwerte zu überwachen, um sensible Informationen wie Prozessdaten und Konfigurationsdetails zu sammeln. Diese Informationen könnten für zukünftige Angriffe oder Spionagezwecke genutzt werden. Ebenfalls sehr gefährlich sind hochaktive Angriffe, bei welchen von außen versucht wird, Daten über das Modbus-System in großem Stil zu überschreiben. Dies kann zu schweren Prozessstörungen bis hin zu Produktionsausfällen führen. Ebenso ist die Wahrscheinlichkeit im Falle eines derartigen Angriffs sehr hoch, dass die grundsätzliche Integrität der Systemdaten gefährdet wird, was zu fehlerhaften Entscheidungen und Sicherheitsproblemen und im schlimmsten Fall auch zu Verletzten und Toten führen kann. Ein Angriff aus letzterer Kategorie wird im nächsten Abschnitt dargestellt und sein Gelingen mit eigenentwickelter Software untermauert.

[3] Vgl.: Tongxin et. al.: Research on DoS Attack Detection Method of Modbus TCP in OpenPLC, in: Journal of Computer and Communications 9, S. 73-90, Wuhan (China): Scientific Research Publishing, 30.07.2021.
[4] Vgl.: Gebeshuber, Klaus: Modbus – Angriffe im lokalen Netzwerk, Informatik aktuell, Frankfurt: Alkmene Verlag, 06.07.2021, auf: https://www.informatik-aktuell.de/betrieb/sicherheit/modbus-angriffe-im-lokalen-netzwerk.html, Zugriff: 29.10.2023, 21:26 Uhr MEZ

4. Beschreibung eines großflächigen Angriffs am Beispiel eines selbst entwickelten Skriptes

Um zu zeigen, wie einfach sich großflächige Angriffe auf Modbus-Systeme realisieren lassen, habe ich ein Skript geschrieben, welches dazu in der Lage ist, mit geringem Aufwand möglichst großen Schaden anzurichten. Das Skript dient ausschließlich der Warnung vor solchen Angriffen und erbringt lediglich den Beweis, dass diese funktionieren. Der ethische Gedanke hinter dem Skript ist die Aufklärung über diesen immer noch weit verbreiteten Angriffsvektor. Das Skript wurde nur in einer entschärften Variante innerhalb der eigenen Testumgebung verwendet, welche jedoch in ihrer Grundfunktionalität nicht vom ursprünglichen Code abweicht. Das ursprüngliche Skript sucht nach sämtlichen, angreifbaren Modbus-Systemen innerhalb einer Internet-TLD. Diese kann per Eingabe in der Konsole festgelegt werden. Das Skript ist in der Programmiersprache Python der Version 3 geschrieben und nutzt unter anderem die Bibliotheken Pymodbus, um eine grundsätzliche Modbus-Protokollunterstützung zu erreichen und um Modbus-Verbindungen aufbauen zu können sowie Shodan für Python, um IP-Listen mit vulnerablen Zielen aus der zuvor definierten Internet-TLD erstellen zu können. Der Code ist so angelegt, dass vollautomatisiert unter Zuhilfenahme der Shodan-Bibliothek eine Liste mit den gefundenen IP-Adressen erstellt und anschließend die weiteren Codezeilen kompiliert werden. Das Kernstück des Skriptes ist eine sich bis zum Abbruch durch den Benutzer wiederholende Endlos-Schleife, in welcher für jede gefundene IP-Adresse Modbus-Aktionen durchgeführt werden und nach Abarbeiten der IP-Liste mit dieser wieder von vorne begonnen wird. Da sich bei einigen Systemen Registerdaten sehr schnell ändern, greift das Skript ein Ziel mit jedem Durchlauf an, bis es abgebrochen wird. Im Falle wiederholter Angriffe auf ein Ziel ist davon auszugehen, dass die Störung des Systems schwerwiegender ausfällt, da Daten frequentierter verändert werden. Innerhalb der Endlos-Schleife wird für jede IP-Adresse eine Verbindung zum Modbus-Gerät hergestellt. Dann werden die Holding Register der Registeradressen 0 bis 90 ausgelesen und mit Nullen überschrieben, insofern sie nicht bereits den Wert 0 haben. Die Werte der Register werden zur Kontrolle vor und nach dem Überschreiben angezeigt. Fehlercodes werden angezeigt, um erfolgreiche und fehlgeschlagene Angriffe sichtbar zu machen. Die oben erwähnte Testversion des Skriptes wurde dahingehend entschärft, dass sich damit ausschließlich die routbaren IP-Adressen der eigenen Testumgebung angreifen lassen. Alles Weitere einschließlich der Textausgaben in der Konsole ist identisch geblieben. Ein Ansatz zur Weiterentwicklung des Skriptes wäre, die Möglichkeit der Abarbeitung paralleler Threads zu integrieren, um noch schneller die Holdingregister überschreiben zu können. Dies könnte möglicherweise über die Python-Module Threading oder Multiprocessing mit der entsprechenden Hardware- und Netzwerkbeschaffenheit abbildbar sein.

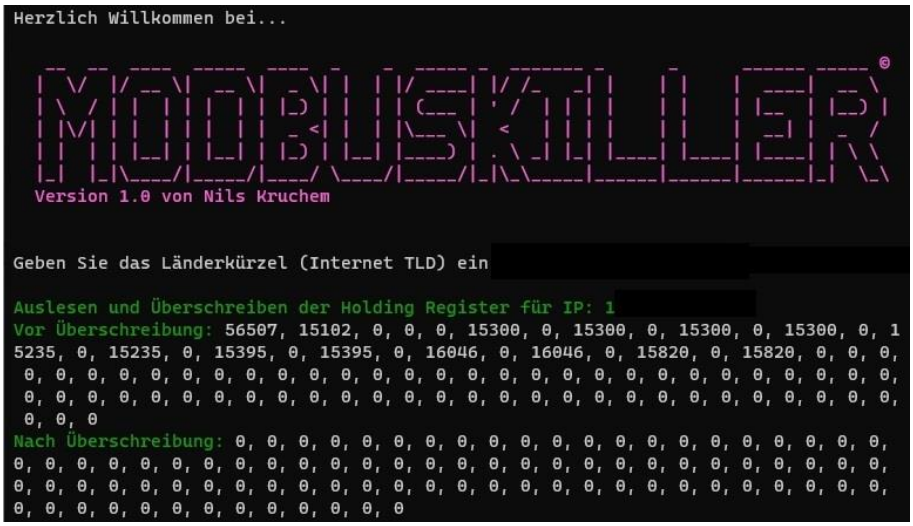
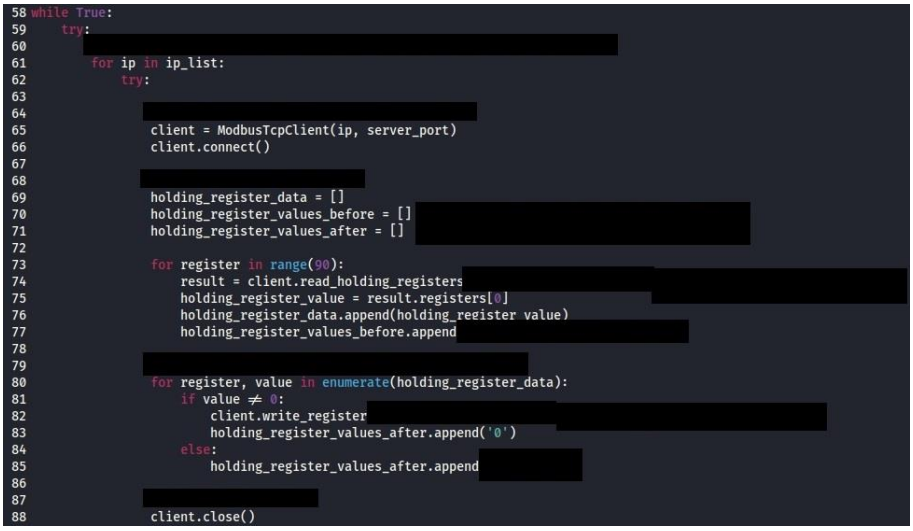


Abb. 4: Zu sehen ist hier die Textausgabe einer entschärften Version des Skriptes, nachdem bei einem Modbus-System die Holding Register ausgelesen und überschrieben wurden. Die öffentlichen IP-Adressen, welche Teil der Testumgebung des Autors sind, wurden geschwärzt. Quelle: Eigeneentwickeltes Programm



```
while True:
    try:
        for ip in ip_list:
            try:

                client = ModbusTcpClient(ip, server_port)
                client.connect()

                holding_register_data = []
                holding_register_values_before = []
                holding_register_values_after = []

                for register in range(90):
                    result = client.read_holding_registers
                    holding_register_value = result.registers[0]
                    holding_register_data.append(holding_register value)
                    holding_register_values_before.append

                for register, value in enumerate(holding_register_data):
                    if value != 0:
                        client.write_register
                        holding_register_values_after.append('0')
                    else:
                        holding_register_values_after.append

                client.close()
```

Abb. 5: Zu sehen ist das Kernstück der Hauptschleife des Skriptes. Die Variablen wurden zuvor definiert. Um Nachahmer zu vermeiden, wird nicht das gesamte Skript veröffentlicht. Im vorliegenden Ausschnitt wurden zudem einige Teile unkenntlich gemacht. Quelle: Eigenentwickelter Programmcode

Das Skript ist in der Lage, innerhalb kurzer Zeit eine Vielzahl von Modbus-Systemen innerhalb einer Internet-TLD anzugreifen. Welche Anlagen und Geräte sich hinter den Modbus-Systemen befinden, könnte zwar in vielen Fällen durch weitere Recherchen leicht herausgefunden werden, ist für die Schadwirkung des Angriffs aufgrund dieser Vielzahl jedoch eher nachrangig. Trotz der Begrenzung empirischer Reliabilität und Validität dieser Methode ist es eine bemerkenswerte Randnotiz, dass die KI-Software ChatGPT[5] dem Skript das Potenzial attestiert, erhebliche Schäden anzurichten. Entsprechende Bildschirmausschnitte können vorgelegt werden. Für Leser mit guten Kenntnissen in Python, Shodan und dem Modbus-Protokoll dürfte die Realisierbarkeit und der Aufbau des Skriptes leicht nachvollziehbar sein.

[5] ChatGPT 3.5 auf https://chat.openai.com, Zugriff: 25.10.2023, 22:25 Uhr MEZ

5. Ergebnisse und Handlungsempfehlungen

Es konnte gezeigt werden, dass es weltweit tausende ungeschützte Modbus-Systeme gibt, welche mit einfachen Methoden angreifbar sind. Auch die Anzahl der Systeme mit IP-Adressen, die Deutschland zuzuordnen sind, ist mit 182 immer noch relativ üppig. Dabei ist Modbus leider nur ein Beispiel von vielen. Situationen aus meinem Berufsalltag haben gezeigt, dass häufig diejenigen, die sich mit industriellen Datenübertragungsprotokollen befassen auch diejenigen sind, die sich nur peripher oder gar nicht der informationstechnischen Sicherheit widmen, während diejenigen, deren Arbeitsalltag sich auf informationstechnische Sicherheit konzentriert, häufig nur geringe Kenntnisse in industriellen Datenübertragungsprotokollen sowie deren Angreifbarkeit und Schutz aufweisen. Beide der eben genannten Parteien müssten vermutlich mehr miteinander kommunizieren, um die hier gezeigten, fatalen Sicherheitslücken zu vermeiden.

Grundsätzlich ist für einen externen Zugriff auf OT-Umgebungen gleichermaßen wie in der klassischen Büro-IT immer mindestens eine 256-Bit-Verschlüsselung (beispielsweise ein VPN mit IPSec) zu empfehlen, wenn dieser Zugriff denn dringend notwendig ist. Sichere Netzübergänge und Firewall-Cluster sind obligatorisch. Eine Authentifizierung sollte konsequent über MFA erfolgen. Umfangreiche IT-Security-Assessments samt Attack Surface Analysen mit der Frage danach, wie man von außen von einem Angreifer wahrgenommen wird, sind unabdingbar, um Schäden zu vermeiden. Insbesondere dann, wenn über einfach manipulierbare Datenübertragungsprotokolle kommuniziert werden muss, ist ein hoher Grad der Netzwerksegmentierung von Vorteil. Dringend anzuraten ist zudem die Etablierung von Softwareprodukten der Netzwerkzugangskontrolle (NAC) mit Security-Funktion, die mindestens eine Überwachung des ARP-Cache bietet, sowie von Lösungen der verhaltensbasierten Anomalie-Erkennung im Netzwerk (NDR) und Netzwerkinventur.[6] Diese Lösungen sollten sich zum Schutz vor Wechselwirkungen möglichst Passiv im OT-Netzwerk verhalten. Die Versendung von Datenpaketen zwecks Betriebs von Hard- und Software der informationstechnischen Sicherheit in OT-Netzwerken muss immer hinsichtlich all ihrer Konsequenzen genau abgewogen werden.

[6] Vgl.: Ennenbach, Martin et. al.: Leitfaden Cyber-Sicherheits-Check OT, Berlin: ISACA Germany Chapter e. V., S. 55 – 58, 65, September 2021

Quellen

ChatGPT auf https://chat.openai.com, Zugriff: 25.10.2023, 22:25 Uhr MEZ

Ennenbach, Martin et. al.: Leitfaden Cyber-Sicherheits-Check OT, Berlin: ISACA Germany Chapter e. V., September 2021

Gebeshuber, Klaus: Modbus – Angriffe im lokalen Netzwerk, Informatik aktuell, Frankfurt: Alkmene Verlag, 06.07.2021, auf: https://www.informatik-aktuell.de/betrieb/sicherheit/modbus-angriffe-im-lokalen-netzwerk.html, Zugriff: 29.10.2023, 21:26 Uhr MEZ

Shodan auf www.shodan.io, Zugriff: 29.10.2023, 15:47 Uhr MEZ

Skript „Modbuskiller 1.0“, Eigenentwicklung von Nils Kruchem, programmiert in Python 3

Tongxin et. al.: Research on DoS Attack Detection Method of Modbus TCP in OpenPLC, in: Journal of Computer and Communications 9, S. 73-90, Wuhan (China): Scientific Research Publishing, 30.07.2021

Wireshark, Software zur Analyse von Datenpaketen in Netzwerkverkehren, Version 4.0.10 für Linux

Abkürzungsverzeichnis

Abb. = Abbildung

ARP = Adress Resolution Protocol

DoS = Denial of Service

IP = Internet Protocol

IPsec = Internet Protocol Security

MFA = Multi Factor Authentication

NAC = Network Access Control

NDR = Network Detection and Response

OT = Operational Technology

TCP = Transmission Control Protocol

TLD = Top Level Domain

vgl. = vergleiche

VPN = Virtual Private Network