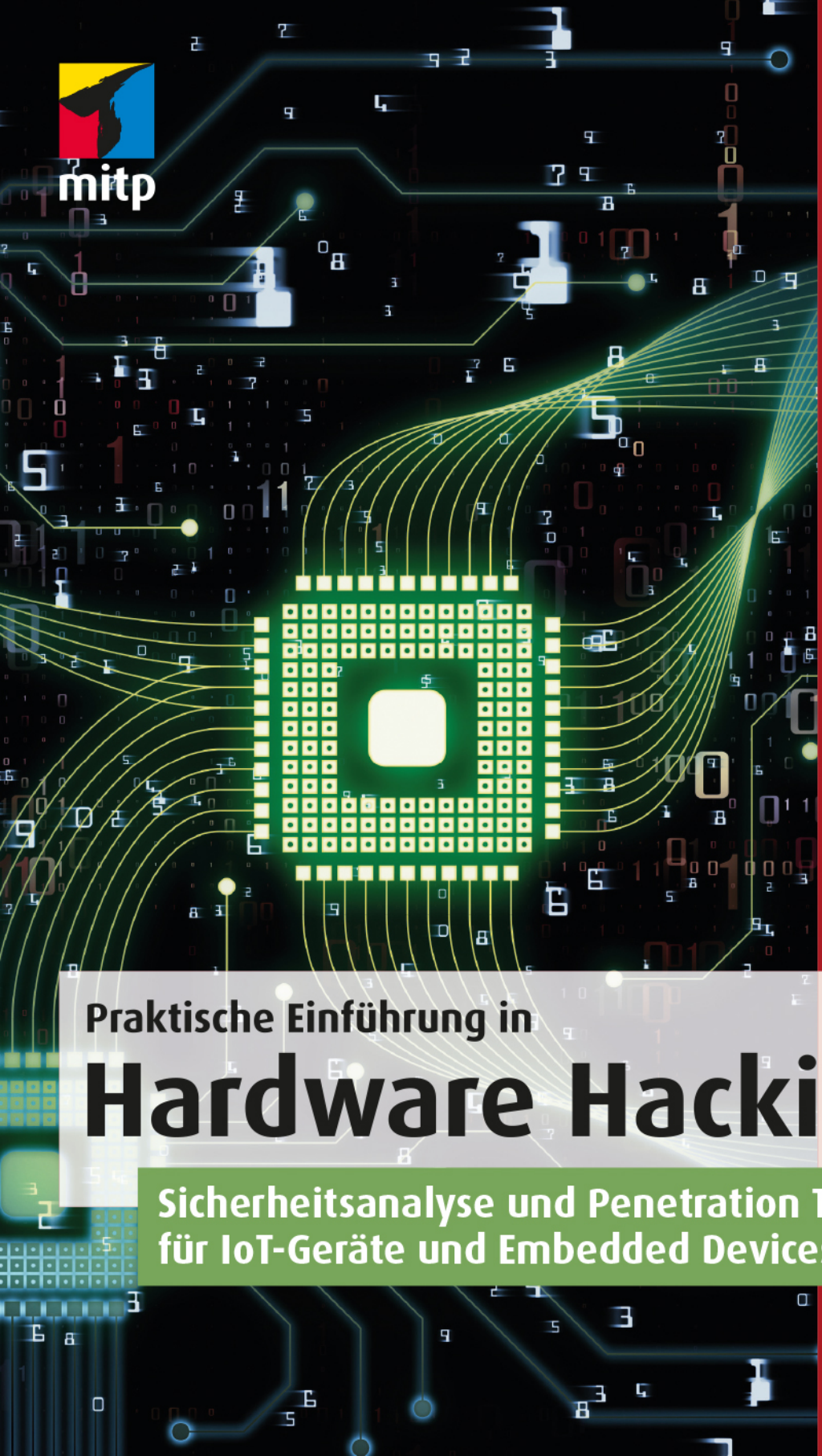

mitp

Marcel Mangel

Sebastian Bicchi

Praktische Einführung in

Hardware Hacking

Sicherheitsanalyse und Penetration Testing für IoT-Geräte und Embedded Devices

Inhaltsverzeichnis

Vorwort 9

Einleitung 11

Über die Autoren 15

Danksagungen 17

1 Vorbereitung 19
1.1 Organisatorische Vorbereitungen 19
1.2 Grundlegender Ablauf des Penetration Tests 20
1.3 Das Labor 24
1.4 Werkzeuge 26
1.4.1 Labornetzgerät 26
1.4.2 Multimeter 29
1.4.3 Computer 32
1.4.4 Oszilloskop 32
1.4.5 Logic Analyzer 32
1.4.6 Raspberry Pi als Universaltool 36
1.4.7 Abgreifklemmen und Adapter 41

2 OSINT 45
2.1 OSINT-Quellen 46
2.2 OSINT-Ziele im Produktumfeld 46
2.3 OSINT-Analyse 47
2.3.1 Hardware 48
2.3.2 Apps 51
2.3.3 Operations 55
2.3.4 Community 59

3 Hardware 61
3.1 Elektronikgrundlagen 61
3.1.1 Stromstärke, Spannung und Widerstand 62
3.2 Kleine Bausteinkunde 62
3.2.1 Diskrete Bauelemente 62
3.2.2 Integrierte Bauelemente (Integrated Circuit – IC) 67

3.2.3 Bussysteme und Schnittstellen 74
3.3 Schaltpläne und Layouts 78
3.4 Datenblätter 79

4 Physische Sicherheit 83
4.1 Gehäuse 84
4.1.1 Spezialschrauben 86
4.1.2 Verklebungen 86
4.1.3 Verschweißung (Kunststoff) 87
4.1.4 Siegel und Plomben 87
4.1.5 Elektronik und Elektromechanik 88
4.1.6 Bausteinverblendung 89
4.2 Designgrundlagen 95
4.2.1 8-Bit-Controller mit HL-Kommunikationsbaustein 96
4.2.2 32-Bit-Controller 98
4.2.3 Android Embedded Device 99
4.2.4 All-in-One-SoC 100
4.2.5 Kombinationen 100
4.3 Praktische Analyse 100
4.3.1 Visuelles Hilfsdokument 100
4.3.2 Entfernen des Schutzlacks 104
4.4 Firmware-Extraktion am Gerät 105
4.4.1 JTAG 106
4.4.2 SWD 107
4.4.3 Serielle Konsole (UART) 114
4.4.4 USB 122
4.4.5 SPI 123
4.4.6 Bus Sniffing & Injection 126

5 Firmware 131
5.1 Dateisysteme 133
5.2 Quellen von Firmware-Images 135
5.2.1 Download des Firmware-Images aus dem Internet 135
5.3 Firmware-Image entpacken 138
5.3.1 Entpacken vorbereiten 138
5.3.2 Manuelles Entpacken 139
5.3.3 Toolgestütztes Entpacken 142
5.3.4 Besondere Firmware-Images 143

5.4 Statische Firmware-Analyse 145
5.4.1 Manuelle Analyse 147
5.4.2 Toolgestützte Analyse 149
5.5 Firmware-Emulation 150
5.6 Dynamische Firmware-Analyse 154
5.7 Firmware-Manipulation 157

6 IoT-Referenzarchitekturen und Netzwerkprotokolle 163
6.1 Einführung in Protokolle 163
6.2 IoT-Referenzarchitekturen 164
6.3 Bluetooth Low Energy 167
6.3.1 Protokoll-Stack 168
6.3.2 Kommunikation zwischen Geräten 169
6.3.3 Bluetooth LE – Sicherheit 178
6.3.4 Klassische Bluetooth-Angriffe 179
6.3.5 Praktische Bluetooth-LE-Angriffe 180
6.4 Zigbee 195
6.4.1 Protokoll-Stack 195
6.4.2 Netzwerk 197
6.4.3 Zigbee-Schwachstellen 200

7 MQTT 205
7.1 Funktionsweise 206
7.2 Quality of Service (QoS) 209
7.3 Retained Messages 209
7.4 Last Will and Testament 210
7.5 Pakettypen 210

8 Apps 217
8.1 OWASP MASVS 218
8.2 Die App herunterladen 219
8.2.1 iOS 220
8.2.2 Android 221
8.3 Statische Analyse 222
8.4 Dynamische Analyse 224
8.4.1 BlackBox 225

9 Backend, Web und Cloud 233
9.1 Vorbereitung 233
9.1.1 Microsoft Azure 235

9.1.2 Amazon Web Service (AWS) 236
9.1.3 Google Cloud Service 237
9.2 Testen von Webapplikationen 237
9.2.1 OWASP Top 10 2017 238
9.2.2 OWASP Testing Guide 242

Stichwortverzeichnis 249

Vorwort

Eine bei Minusgraden gehackte smarte Heizung führte in Finnland zu eingefrorenen Rohrleitungen in einer Wohnanlage mit einigen Hundert Einheiten. US-Behörden riefen Herzschrittmacher zurück, weil Hacker sie angreifen konnten. Zwei Forscher schafften es, einen Jeep Grand Cherokee über das Internet vollständig fernzusteuern, von Aircondition über Fahrtrichtung, Geschwindigkeit und Sitzverstellung bis hin zur Zentralverriegelung.

Längst betrifft IT-Sicherheit nicht mehr nur den klischeehaften Hacker in seinem mit Pizzakartons zugemüllten dunklen Kabuff, sondern jeden von uns. Mit Cyber-Physical-Systemen haben Programmierfehler und Sicherheitslücken Auswirkungen auf das tägliche Leben.

Marcel Mangel und Sebastian Bicchi sind professionelle Penetration Tester, sie demonstrieren regelmäßig Sicherheitslücken. Auf Basis ihres Wissens und ihrer praktischen Erfahrungen ist das folgende Buch entstanden, in dem sie darlegen, wie Angreifer arbeiten – am Beispiel smarter Geräte.

Sie zeigen auf, wie oft simple Fehler zu gefährlichen Angriffen führen und wie die Angriffe funktionieren. Damit bieten sie Sicherheitstestern eine hervorragende Anleitung dazu, wie sie zur Qualitätssicherung vor der Markteinführung ihre Produkte testen sollten.

Gleichzeitig lernen die Entwickler, wie sie die Angriffsflächen in der Zukunft verringern und so die Sicherheit deutlich erhöhen.

Ich bin ein Verfechter von Qualitätssicherung in der Softwareentwicklung. Zur Qualitätssicherung gehört es, sowohl die Fehlermöglichkeiten zu kennen, um sie zu vermeiden, als auch durch gründliches Testen Fehler zu identifizieren. Zu beidem leistet dieses Buch einen wertvollen Beitrag.

Von Praktikern für Praktiker anhand praktischer Beispiele mit qualifiziertem theoretischem Hintergrund – eine lohnende Lektüre, für die ich den beiden Autoren sehr dankbar bin.

Prof. Dr. Tobias Eggendorfer
Lehrstuhl für IT-Sicherheit
Hochschule Ravensburg Weingarten

Einleitung

Die Vernetzung der Welt schreitet immer weiter voran. Das betrifft nicht nur traditionelle IT-Systeme, sondern mehr und mehr alle möglichen »smarten« Geräte. Diese können alle unter dem Stichwort **I**nternet **o**f **T**hings (IoT) subsummiert werden. Für die Hersteller technischer Geräte ist es heutzutage quasi ein Muss, diese in irgendeiner Weise »smart« zu machen. Ein Gerät wird in der Regel dadurch »smart«, dass es mit anderen Geräten oder Systemen vernetzt ist und Informationen austauschen kann.

Dieser Informationsaustausch geschieht dabei in aller Regel über standardisierte Protokolle und Infrastrukturen wie z. B. das Internet. Für potenzielle Angreifer eröffnet diese Vernetzung eine ganze Reihe neuer Angriffsmöglichkeiten. Anstatt besonders gehärtete klassische IT-Systeme anzugreifen, ist es für Hacker deutlich interessanter geworden, ihren Fokus auf IoT-Geräte zu verlegen. Darunter finden sich eine Menge Geräte, die praktisch überhaupt nicht abgesichert sind. Da kann es, wie das Mirai-Botnetz[1] eindrucksvoll unter Beweis gestellt hat, schon ausreichen, die Geräte einfach mit Standardpasswörtern zu übernehmen und für weitere Angriffe zu missbrauchen. Es wurden insgesamt 64 verschiedene Standardpasswörter getestet, um Zugriff auf die entsprechenden Geräte zu erlangen. Eine genauere Analyse des Sourcecodes sei dem Leser an Herz gelegt.

Die Situation verschärft sich jedoch dramatisch, wenn es sich bei den angegriffenen Geräten nicht mehr um Router oder Kameras handelt, sondern um Geräte aus der Medizintechnik. Auch in dieser Branche werden die Geräte immer »smarter« und bieten damit immer größere Angriffsflächen. Viele Hersteller sind sich aktuell gar nicht richtig im Klaren über die Gefahren, die von der zunehmenden Vernetzung ausgehen. Nicht nur die Medizintechnik ist ein Beispiel für die wachsende Verzahnung von Security und Safety. Weitere Bereiche, für die das gilt, sind die sogenannten »kritischen Infrastrukturen«. Darunter fallen z. B. Energieversorger, der öffentliche Personenverkehr oder auch Krankenhäuser. Werden solche Systeme kompromittiert, kann das fatale Folgen haben. Ein sehr bekanntes Beispiel ist der Stromausfall in Brasilien. Nachdem Angreifer Industrial-Control-Systeme unter ihre Kontrolle gebracht hatten, kam es zu flächendeckenden Stromausfällen, von denen Millionen Menschen in Brasilien über mehrere Stunden betroffen waren.[2]

1 https://github.com/jgamblin/Mirai-Source-Code
2 https://www.wired.com/2009/11/brazil/

Auch hier waren nicht abgesicherte Systeme, die über das Internet erreichbar waren Ausgangspunkt des Angriffs.

Die Hersteller werden jedoch immer mehr zur Verantwortung gezogen. Während man das Thema Cyber Security vor einigen Jahren noch sehr stiefmütterlich behandeln konnte, wird es in den kommenden Jahren mehr und mehr an Wichtigkeit gewinnen. Zum Teil wird dies auch schon durch die Einhaltung von Compliance-Richtlinien wie der Datenschutzgrundverordnung erzwungen.

Ziel des Buchs

Als wir anfingen, uns mit dem Thema Penetration Testing für Internet-of-Things-Geräte zu beschäftigen, gestaltete es sich als überaus schwierig, Bücher zu diesem Thema zu finden. Insbesondere auf dem deutschsprachigen Markt war zu diesem Zeitpunkt kein geeignetes Buch verfügbar. Auch im englischsprachigen Raum gab es zwar schon einige Bücher, doch diese wurden alle nicht dem gerecht, wonach wir suchten. Im Internet auf diversen Blogs fanden wir einige sehr interessante und hilfreiche Informationen, jedoch keine, die das Thema umfassend abdeckten. Mit diesem Buch versuchen wir, genau diese Lücke zu schließen. Das Buch soll zum einen eine umfassende Informationsquelle zum Thema sein und zum anderen als eine Art Referenzwerk zum praktischen Testen von Geräten dienen.

Die OWASP IoT Top 10 stellen ein erwähnenswertes Projekt dar, in dem man sich ebenfalls mit der Sicherheit von IoT-Geräten beschäftigt. OWASP steht für **O**pen **W**eb **A**pplication **S**ecurity **P**roject, und es handelt sich dabei um eine Non-Profit-Organisation, die insbesondere durch die »OWASP Top 10«, eine Liste mit den am häufigsten vorkommenden Schwachstellen in Webanwendungen, bekannt geworden ist. Neben dieser Liste für Webanwendungen gibt es die »OWASP Mobile Top 10« sowie seit 2014 auch die »OWASP IoT Top 10«, in der die zehn am häufigsten vorkommenden Schwachstellen für IoT-Geräte aufgelistet sind. Nach der letztmaligen Aktualisierung im Jahr 2018 sind dies:

1. Schwache, erratbare oder hartcodierte Passwörter
2. Unsichere Netzwerkdienste
3. Unsichere Schnittstellen innerhalb des IoT-Ökosystems
4. Unsichere Update-Mechanismen
5. Verwendung unsicherer oder veralteter Komponenten
6. Nicht ausreichender Schutz von Benutzerdaten
7. Unsichere Transfers und unsichere Speicherung von Daten
8. Unzureichendes Management der Geräte
9. Unsichere Standardeinstellungen
10. Unzureichender Schutz gegen physische Angriffe

Aufbau des Buchs

Das Buch behandelt in acht Kapiteln die einzelnen Aspekte, die beim Testen eines IoT-Geräts vonnöten sind.

Kapitel 1 – Vorbereitung

In diesem Kapitel werden organisatorische sowie technische Projektvorbereitungen dargestellt, die für ein erfolgreiches Security-Testing-Projekt unabdingbar sind. Neben dem Thema Scoping wird auch intensiv auf den Aufbau eines entsprechenden Testing-Labors eingegangen. Abschnitt 1.3 »Das Labor« zeigt den praktischen Aufbau eines Elektroniklabors, welche Geräte benötigt werden und wofür. Die Grundbegriffe der Elektronik werden erläutert sowie Arbeitsweisen im elektronischen Labor.

Kapitel 2 – OSINT

Zu Beginn dieses Kapitels wird der Begriff *OSINT* (**O**pen **S**ource **Int**elligence) im Kontext der Sicherheit vernetzter Geräte erklärt. Weiterhin wird beschrieben, wie die systematische Sammlung und Analyse offen zugänglicher Informationen durchzuführen ist und wie diese zur Sicherheitsanalyse verwendet werden können.

Kapitel 3 – Hardware

Das Kapitel »Hardware« erklärt zunächst die notwendigen Elektronikgrundlagen und die verschiedenen Bauelemente in einer kleinen Baustein-Lehre. Zusätzlich werden Bussysteme und Schnittstellen theoretisch beleuchtet und integrierte Bausteine sowie verschiedene Aspekte der Herstellung, wie Layout und Schemata einer Platine, beschrieben.

Kapitel 4 – Physische Sicherheit

Das Kapitel »Physische Sicherheit« beschäftigt sich mit der Sicherheit des Gehäuses und Tamper-Protection-Maßnahmen. Außerdem werden Hardwaredesign-Grundlagen erklärt (8-/32-Bit-Controller) und verschiedene Angriffspunkte erläutert. Zu guter Letzt wird gezeigt, wie bei einem physischen Zugriff auf das Gerät die Firmware extrahiert werden oder ein Zugriff auf andere Daten in Bausteinen erfolgen kann (JTAG/SWD/UART/SPI).

Kapitel 5 – Firmware

Die Firmware ist quasi das Herzstück eines jedes IoT-Geräts und besitzt damit einen besonderen Stellenwert bei der Sicherheitsanalyse eines solchen. In diesem Kapitel werden zunächst unterschiedliche Möglichkeiten dargestellt, an die Firmware eines Geräts zu gelangen. Im Anschluss beschreibt das Kapitel ausführlich das Entpacken, die Analyse und die Emulation von Firmware-Images.

Kapitel 6 – IoT-Referenzarchitekturen und Netzwerkprotokolle

Das Kapitel beginnt mit einer Einführung in das Thema Protokolle und stellt zwei verschiedene in der Praxis relevante IoT-Referenzarchitekturen vor, bevor auf zwei konkrete Netzwerkprotokolle (Bluetooth Low Energy und Zigbee) im Detail eingegangen wird. Diese beiden Protokolle werden von zahlreichen IoT-Geräten verwendet und stellen damit ein oftmals zentrales Thema bei vielen Security Assessments dar. Neben den Grundlagen der beiden Protokolle werden praktische Angriffe und Tools vorgestellt.

Kapitel 7 – MQTT

Das achte Kapitel widmet sich dem wohl wichtigsten Protokoll auf Anwendungsebene im IoT-Umfeld: MQTT. Hier steht insbesondere eine ausführliche Darstellung der Funktionsweise sowie der wesentlichen Pakettypen im Vordergrund.

Kapitel 8 – Apps

Das App-Kapitel umfasst einen kurzen Einstieg in die OWASP-App *Security* und erklärt, welche Sicherheitsanforderungen an Apps allgemein gestellt werden.

Vertiefend wird auf die Spezifika vernetzter Geräte und Apps eingegangen, insbesondere auf sämtliche Verbindungen (direkt und indirekt) zum vernetzten Gerät.

Kapitel 9 – Backend, Web und Cloud

In diesem Kapitel werden zum einen die Rahmenbedingungen erläutert, die beim Testen von Backend-Systemen zu beachten sind, und zum anderen wird die am weitesten verbreitete Methodik zum Testen von Applikationen nach OWASP erläutert.

Zielgruppe

Das Buch richtet sich in erster Linie an Personen, die Penetration Tests von Internet-of-Things-Geräten durchführen möchten.

Neben den eigentlichen Testern kann das Buch durchaus auch für Projektmanager oder Security-Verantwortliche aufseiten der Hersteller interessant sein.

Was Sie benötigen

Als Leser dieses Buchs sollten Sie bereits über grundlegende Kenntnisse in IT-Sicherheit, insbesondere in den Bereichen Netzwerk- und Applikationssicherheit, verfügen. Zudem wird ein routinierter Umgang mit Linux vorausgesetzt.

Trotzdem legen wir Wert darauf, dass auch interessierte Einsteiger den Inhalten des Buchs gut folgen können.

Über die Autoren

Marcel Mangel verfügt über mehr als eine Dekade praktischer Erfahrung in IT-Sicherheit. Sowohl im defensiven als auch im offensiven Bereich war er mehrere Jahre für renommierte Unternehmen tätig und hat neben einem Master in Informatik noch über eine ganze Reihe von anerkannten Zertifikaten. Nebenbei arbeitet Marcel Mangel bereits seit mehreren Jahren im Rahmen von Lehraufträgen und Gastvorlesungen an diversen Universitäten und Fachhochschulen als Dozent. Zurzeit hält er die Master-Vorlesung »Vertiefung der IT-Sicherheit« an der Fachhochschule Rosenheim.

Sebastian Bicchi begann bereits in seiner Jugend mit dem Aufspüren von Sicherheitslücken in Webseiten und Anwendungen. Nach seinem IT-Security-Studium gründete er in Wien gemeinsam mit Studienkollegen das Unternehmen »Security Research« (sec-research.com) und beschäftigte sich insbesondere mit den technischen Aspekten der IT-Security. Die Synergien seiner Ausbildungen in verschiedenen Bereichen (Elektrotechnik/industrielle Informationstechnik, Informationstechnologien und Telekommunikation, IT-Security) schufen die Basis für sein fundiertes Wissen über IoT- und Hardware-Hacking. Sebastian Bicchi betätigt sich darüber hinaus freiwillig in nicht kommerziellen Organisationen für Informationssicherheit, zum Beispiel als Co-Chapter-Lead bei OWASP für das Chapter Vienna.

Danksagungen

Marcel Mangel

Mein besonderer Dank gilt meiner Lebensgefährtin Sarah sowie meiner gesamten Familie, die mich bei der Erstellung des Buchs außerordentlich unterstützt haben.

Daneben möchte ich mich ganz herzlich bei meinem Koautor und Freund Sebastian Bicchi bedanken, ohne den dieses Buch nicht entstanden wäre.

Außerdem bedanke ich mich sehr herzlich beim mitp-Verlag für die Möglichkeit der Veröffentlichung sowie insbesondere bei unserer Lektorin Frau Janina Bahlmann für die tolle Unterstützung und Zusammenarbeit.

Und zu guter Letzt bedanke ich mich ebenfalls sehr herzlich bei Christian Salzmann, der die Abschnitte über Bluetooth und Zigbee beigesteuert hat, und bei Tobias Eggendorfer für den stetigen Austausch und das Vorwort.

Sebastian Bicchi

Mein besonderer Dank gilt meiner Freundin Sandra, die mich in allem, was ich mache, unterstützt und mir, während ich dieses Buch geschrieben habe, alle nur erdenklichen Lasten abgenommen hat.

Weiterhin möchte ich mich bei Marcel Mangel bedanken – für die Möglichkeit, dieses Buch zu schreiben, und für die Zusammenarbeit bei der Erstellung des Buchs und allen weiteren spannenden Projekten, die wir zusammen bearbeitet haben.

Ein großes Dankeschön möchte ich an dieser Stelle auch dem Verlag und Frau Bahlmann für ihre Unterstützung während des gesamten Erstellungsprozesses des Buchs und für das außerordentlich gute Feedback ausrichten.

Mein weiterer Dank gilt den Personen, die meine Ausbildung zu einem besonderen Weg gemacht und mir letztendlich ermöglicht haben, dieses Buch zu schreiben: Manuel Koschuch (FH Campus Wien) und Bernd Stanzl (HTL Mödling). Beide haben mich über alle Maßen unterstützt, sämtliche Fragen, auch über den Lehrplan hinaus, immer mit genügend Zeit und Geduld beantwortet und mich inspiriert, Wissen weiterzugeben.

Ich möchte mich ebenfalls bei Zlatko Sehanovic und Herbert Dowalil für das Vorab-Review und das inhaltliche Feedback bedanken.

Kapitel 1

Vorbereitung

1.1 Organisatorische Vorbereitungen

Bei der Durchführung eines Penetration Tests auf einem IoT-Gerät ist ein gut strukturierter Projektablauf maßgeblich für den Erfolg.

Ein wesentlicher Aspekt, dem ausreichend Aufmerksamkeit geschenkt werden sollte, ist das Thema »Scoping«, also die Festlegung der thematischen Abdeckung bzw. des Geltungsbereichs des Tests. Das Scoping muss zwingend in enger Abstimmung mit dem Auftraggeber erfolgen. Nach unserer Erfahrung liegt hier eine häufige Ursache in der Unzufriedenheit vieler Auftraggeber am Ende eines Projekts. Wichtig ist es, sicherzustellen, dass der Auftraggeber und der Tester die »gleiche Sprache sprechen«. Häufig sind die Tester so tief technisch verwurzelt, dass es bei Projektbesprechungen zu Missverständnissen kommt und insbesondere Fachbegriffe unterschiedlich ausgelegt werden. Deshalb ist es zu Beginn eines Projekts sinnvoll, einige wesentliche Begriffe kurz zu definieren und für ein gemeinsames Verständnis zu sorgen.

Darüber hinaus sollten folgende Punkte abgestimmt werden:

- Testmethodik (BlackBox, GreyBox oder WhiteBox)
- Testtiefe
- Testziele

Bei einem *BlackBox*-Test verfügt der Tester über keinerlei nicht öffentliche Informationen über das zu testende Gerät bzw. System. Der Tester schlüpft also in die Rolle eines externen Angreifers, der sich alle notwendigen Informationen selbstständig besorgen muss. Im Gegensatz dazu stehen dem Tester bei einem *WhiteBox*-Test viele nützliche nicht öffentliche Dokumente wie z. B. Netzwerkdiagramme, Funktionsdiagramme, interne technische Dokumentationen, Datenflussdiagramme oder auch der Quellcode der Applikationen zur Verfügung. Der *GreyBox*-Ansatz stellt den Mittelweg zwischen BlackBox und WhiteBox dar. Hier verfügt der Tester über einige nicht öffentliche Informationen. Im Regelfall stellt der Auftraggeber zumindest Netzwerkdiagramme und interne technische Dokumentationen zur Verfügung, während der Quellcode von Applikationen nicht herausgegeben wird.

Bei der Abstimmung der Testtiefe geht es in erster Linie darum, festzulegen, inwieweit identifizierte Schwachstellen ausgenutzt werden sollen. In der Regel wer-

den dazu sogenannte Proof-of-Concept-Exploits erstellt, die belegen, dass die entsprechende Schwachstelle ausnutzbar ist.

Es ist sinnvoll, für jedes Projekt Ziele zu definieren, die beim Testen verfolgt werden sollten. Diese können auch in Form von Fragen formuliert werden.

Einige sinnvolle Fragen, deren Beantwortung nach Abschluss des Tests möglich sein sollte, sind:

- Ist es möglich, unautorisierten Zugriff auf das Gerät zu bekommen?
- Ist es möglich, an sensitive Daten zu gelangen?
- Ist die Angriffsfläche des Produkts minimal gehalten, um potenzielle Angriffe zu erschweren?

Das Motiv vieler Auftraggeber ist zunehmend das Thema »Compliance«, z. B. im Rahmen einer PCI-Zertifizierung oder der Datenschutzgrundverordnung. Aus unserer Sicht sollte die Einhaltung der Compliance zwar mit berücksichtigt werden, jedoch niemals das einzige Motiv für eine Überprüfung sein. Die Sicherheit des IoT-Geräts bzw. dessen Ökosystem sollte immer das zentrale Motiv eines solchen Tests darstellen.

1.2 Grundlegender Ablauf des Penetration Tests

Obwohl ein Penetration Test dynamisch abläuft und eine Aufgabe in die andere greift, gibt es ein grundsätzliches Vorgehensmuster:

1. **Auftragsvergabe, Vorbereitung, Klärung des Scopes:** Bevor Sie mit dem Test beginnen, müssen sämtliche formellen Punkte geklärt werden. Dazu zählt der Testmodus, der Scope (Testziel), die Angriffsfreigabe, und natürlich müssen Sie die Testobjekte (IoT-Geräte) erhalten.
2. **Informationsbeschaffung:** Wie funktioniert das Gerät? Welche öffentlichen Informationen gibt es? Dieser Teil wird in Kapitel 2 behandelt.
3. **Inbetriebnahme nach Vorgabe:** Nehmen Sie das Gerät wie durch den Hersteller vorgesehen in Betrieb. Zeichnen Sie dabei jeglichen Netzwerkverkehr auf – die erste Inbetriebnahme beinhaltet sehr oft sicherheitskritische Abläufe wie Updates, Kopplung oder Registrierung. Die erste Inbetriebnahme bzw. Aufzeichnung wird im nächsten Abschnitt erläutert.
4. **Firmware-Analyse:** Dieser Schritt hängt davon ab, ob Sie die Firmware im Rahmen der OSINT-Analyse oder anderweitig bereits erhalten konnten. Ist dies der Fall, kann die Firmware-Analyse durchgeführt werden – siehe hierzu Kapitel 5. Haben Sie die Firmware noch nicht, ziehen Sie die Hardware- oder App-Analyse vor.
5. **Hardwareanalyse und physische Sicherheit:** Im Rahmen der Hardwareanalyse sind Sie unter Umständen in der Lage, die Firmware zu extrahieren. Führen Sie diese Analyse durch und verbinden Sie den Vorgang mit der Analyse der physischen Sicherheit (Kapitel 3 und Kapitel 4).

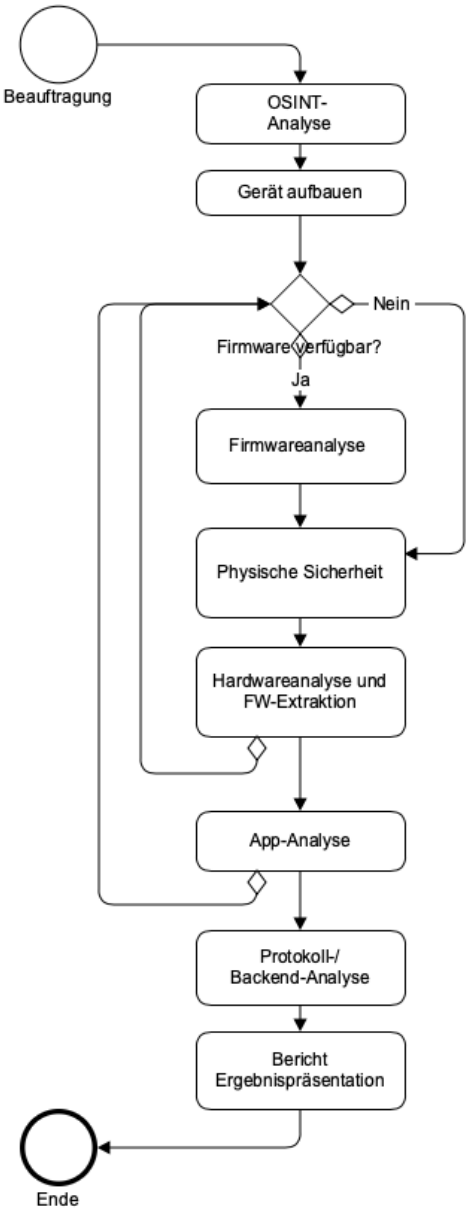

Abb. 1.1: Ablaufdiagramm eines IoT-Penetration Tests

Wichtig

Wenn Sie nur ein Testobjekt zur Verfügung haben, verschieben Sie die Hardwareanalyse ans Ende des Tests, da das Testobjekt beschädigt werden könnte und Sie danach nicht mehr in der Lage sein könnten, die anderen Tests durchzuführen!

6. **App-Analyse:** Führen Sie die App-Analyse wie in Kapitel 8 beschrieben durch.
7. **Protokollanalyse:** Im Rahmen der vorangegangenen Tests sollten nun einige Protokollangriffspunkte sichtbar geworden sein. Führen Sie jetzt den Penetration Test auf die IoT-Protokolle und Backends durch, wie in den Kapitel 6, Kapitel 7, und Kapitel 9 beschrieben.
8. **Schwachstellen zusammenfügen:** Das holistische Bild ist sehr wichtig, da reale Angriffe sehr oft eine Kombination unterschiedlicher Schwachstellen ausnutzen. Verbinden Sie alle gefundenen Schwachstellen und beschreiben Sie eventuelle Zusammenhänge.
9. **Reporting:** Erstellen Sie einen Bericht und führen Sie die Nachbereitung mit dem Auftraggeber durch. Das Reporting besteht normalerweise aus einer Präsentation der Schwachstellen und Empfehlungen dazu, wie diese behoben werden können. Auch wird oft ein Nachtest besprochen, um die Behebung der Schwachstellen zu prüfen.

Dieser gesamte Testablauf ist grafisch in Abbildung 1.1 dargestellt.

Aufzeichnung der ersten Inbetriebnahme

Bevor der eigentliche Test durchgeführt wird, sollten Sie das Gerät zunächst wie durch den Hersteller vorgesehen in Betrieb nehmen. So erfahren Sie, wie das Gerät grundsätzlich funktioniert bzw. wie die vom Hersteller gedachte Arbeitsweise ist.

Die erste Inbetriebnahme ist von besonderer Bedeutung: Oft werden hier einmalige Vorgänge gestartet, die sich später nicht einfach reproduzieren lassen, zum Beispiel ein Firmware-Update oder die Registrierung des Geräts mit dem Server. Daher ist es notwendig, die Inbetriebnahme von Beginn an ausreichend zu dokumentieren. Dazu zählt auch insbesondere der Netzwerkverkehr des Geräts.

Je nachdem, wie das Gerät mit dem Netzwerk zu verbinden ist, stehen verschiedene Aufzeichnungsmöglichkeiten zur Auswahl. Verbindet sich das Gerät mittels WLAN mit dem Netzwerk, kann die Konfiguration in Abbildung 1.2 verwendet werden.

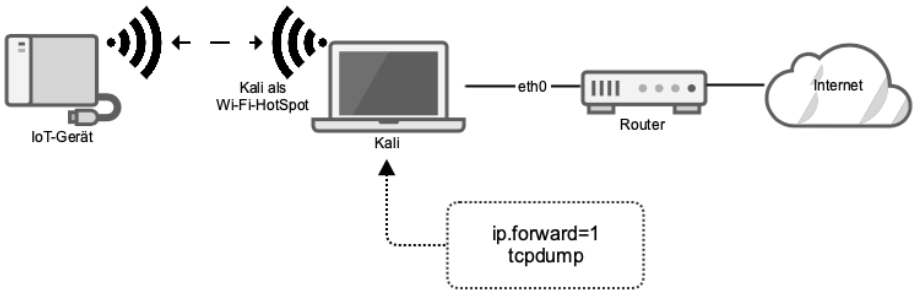

Abb. 1.2: Capturen der Wi-Fi-Verbindung

In dieser Konfiguration können Sie folgenden Befehl verwenden, um die Aufzeichnung durchzuführen:

```
tcpdump -i wlan0 -s 65535 -w ./capture-01.pcap
```

Besitzt das Gerät hingegen einen Ethernet-Anschluss bzw. muss die Ersteinrichtung über Ethernet durchgeführt werden, können Sie einerseits auf Hardware wie das Hak5 PacketSquirrel[1] zurückgreifen, das in Abbildung 1.3 dargestellt ist.

Abb. 1.3: PacketSquirrel

Mit einer entsprechenden Konfiguration (entnehmen Sie diese bitte der Anleitung bzw. dem Hak5-Forum) und einem USB-Stick zeichnet das PacketSquirrel den Verkehr zwischen dem Gerät und dem Netzwerk zuverlässig auf. Diese Methode ist komfortabel, da kein Computer benötigt wird, jedoch muss das PacketSquirrel hierzu gekauft werden.

Sie können die Aufzeichnung jedoch auch mit einem Computer oder dem Raspberry Pi durchführen. Hierzu benötigen Sie eine zweite Ethernet-Karte – hat Ihr Computer diese nicht eingebaut, können Sie auch eine externe USB-Ethernet-Karte verwenden, wie in Abbildung 1.4 gezeigt.

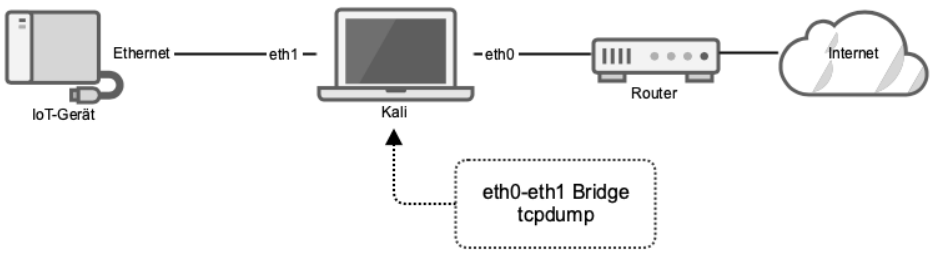

Abb. 1.4: Paket Capture Ethernet

Die beiden Karten müssen nun zu einer Netzwerkbrücke konfiguriert werden, sodass Pakete einfach zwischen den Ports durchgereicht werden. Installieren Sie

1 https://docs.hak5.org/hc/en-us/categories/360000982574-Packet-Squirrel

hierzu die bridge-utils – unter Debian oder Kali mit dem Kommando `aptitude install bridge-utils`.

Um nun eine Netzwerkbrücke zwischen den beiden Karten – zum Beispiel eth0 und eth1 – zu erstellen, verwenden Sie folgende Kommandos:

```
brctl addbr br0
brctl addif br0 eth0 eth1
```

Dadurch erhalten Sie ein neues virtuelles Netzwerkgerät namens br0, das die beiden realen Ethernet-Adapter eth0 und eth1 miteinander verbindet. Mittels `tcpdump` können Sie nun den Verkehr aufzeichnen.

```
tcpdump -i br0 -s 65535 -w ./capture-01.pcap
```

Wenn sich das IoT-Testgerät über Bluetooth mit einer App verbindet, sollten Sie ein erweitertes Setup verwenden. Aktivieren Sie hierzu bei Android zunächst die Funktion *Bluetooth HCI Snooping* in den Entwickleroptionen. Dadurch wird der Bluetooth-Verkehr aufgezeichnet (Details hierzu finden Sie im Abschnitt 6.3.5).

Verwenden Sie dann das Setup, das Abbildung 1.5 darstellt.

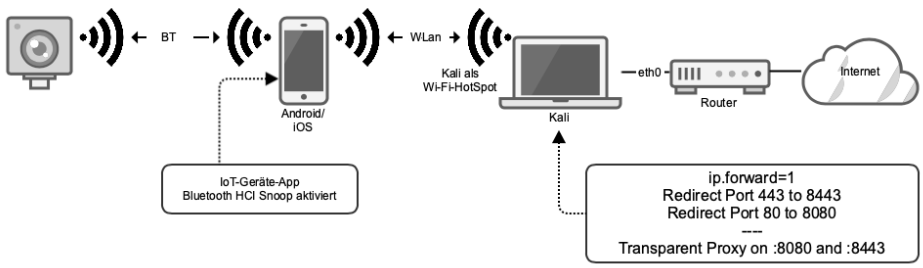

Abb. 1.5: Paket Capture Bluetooth

Zusätzlich sollten Sie zuvor noch die Einrichtungsschritte aus dem Abschnitt 8.4.1 »Netzwerkverkehr« durchführen und gleichzeitig den Netzwerkverkehr der App mitschneiden.

1.3 Das Labor

Das Labor ist der Arbeitsbereich für die Tests und sollte daher entsprechend ausgestattet sein. Im Labor werden Sie die Geräte zerlegen, die Hardware analysieren sowie Bluetooth-, WLAN- und LAN-basierte Angriffe durchführen.

Hinweis

Sollte sich das Labor am Arbeitsplatz befinden, sind spezielle Regelungen hinsichtlich Belastung der Mitarbeiter mit Schadstoffen zu beachten. Insbesondere Schwermetalle (z. B. bleihaltiges Lötzinn), Lötdämpfe und Lösungsmittel unterliegen in der Arbeitswelt strengen Regulierungen. Der Arbeitgeber hat an dieser Stelle für die Sicherheit der Mitarbeiter zu sorgen.

Eine ausreichende Arbeitsfläche, mindestens 2,00 × 1,20 Meter, sollte vorhanden sein. Diese Fläche sollte mit einer ESD(**E**lectro**s**tatic-**D**ischarge)-sicheren Matte oder einer ESD-Schutzeinrichtung ausgestattet sein. Die Arbeitsfläche muss beständig sein gegen Hitze (Löten, Heißluft) und Chemikalien.

Zu den unbedingt benötigten Geräten gehören:

- Feinmechanikwerkzeug (zum Beispiel von iFixIt)
- Labornetzgerät
- Lötkolben und Lötzubehör (Lötzinn, Entlötlitze, Flussmittel)
- Klemmen, »Fliegenfüße«, SMD-Abgreifzangen, Stift- und Sockelleisten, Jumper-Kabel und weitere Adapter/Kabel
- Multimeter
- Lupe oder Mikroskop
- Computer mit mindestens zwei Netzwerkkarten (1× WLAN + 1× Ethernet), besser jedoch vier (2 + 2)
- Internetanschluss mit möglichst wenig Zwischenkomponenten, am besten direkt, nach Möglichkeit ohne (Mehrfach-)NAT bzw. mit öffentlicher IP-Adresse[2]
- Raspberry Pi-Einplatinencomputer

Die Werkzeuge werden im nächsten Abschnitt genauer erklärt.

Folgende Geräte sind empfehlenswert bzw. werden für bestimmte Analysen benötigt:

- Oszilloskop
- Logic Analyzer
- RS232-USB-Wandler
- Shikra-USART/UART-USB-Wandler
- Adafruit Buspirate

2 Einige Internetanbieter, insbesondere von mobilen Verbindungen wie 3G/LTE, vergeben keine Public-IP, sondern eine interne private IP, die zusätzlich von transparenten Proxy-Servern gefiltert wird. Dies kann sich nachteilig auf die Tests auswirken und zu nicht reproduzierbaren Ergebnisse führen.

- Facedancer (USB-Analyse)
- Hack RF One
- UberTooth
- Micro:bit-Einplatinencomputer
- Android-Smartphone, wenn möglich gerootet
- iOS-Smartphone, nach Möglichkeit mit JailBreak

Diese Auflistung ist nicht vollständig, und im Laufe Ihrer weiteren IoT-Hacking-Erfahrung wird sich Ihr Werkzeugkasten stets erweitern. Nicht alle Geräte werden immer benötigt – so können einige dieser Hardwaretools zum Beispiel durch einen Raspberry Pi ersetzt werden, wie später in diesem Buch beschrieben wird.

Vorsicht!

Schließen Sie die zu testenden Geräte niemals ohne galvanische Trennung (Labornetzgerät, Batterie oder Trenntrafo) an das Stromnetz an, wenn Sie an diesen geöffnet arbeiten. Interne Netzgeräte sind manchmal nicht galvanisch getrennt. Dies kann zu fehlerhaften Messungen, Spannungsverzug oder gefährlichen Spannungsspitzen führen. Dadurch können Sie sich ernsthaft verletzen oder Geräte zerstören.

1.4 Werkzeuge

(Hardware-)Werkzeuge sind für das Testen von vernetzten Geräten unerlässlich, da Angriffe auf die Hardware durchgeführt werden müssen und das Verständnis der Hardware essenziell für eine ausführliche Analyse ist. Sie sollten eine gewisse Fertigkeit mit den Werkzeugen entwickeln, bevor Sie sie im echten Einsatz verwenden. Nachfolgend werden die wichtigsten Hardwarewerkzeuge erklärt sowie Grundeinrichtung und Verwendung eines Standardlabors.

1.4.1 Labornetzgerät

Das Labornetzgerät dient als Stromquelle für sämtliche zu analysierenden Geräte. Im Idealfall können Sie auf dem Gerät Spannung und Strombegrenzung exakt einstellen. So bietet Ihnen das Gerät zusätzlichen Schutz gegen einen Kurzschluss im Verlauf der Arbeiten.

Das Labornetzgerät ist insbesondere sehr nützlich, wenn Sie an offenen Geräten arbeiten. Zwar können viele Geräte auch ohne entsprechendes Netzgerät versorgt werden, jedoch ist es vor allem im Automotive-Bereich üblich, dass die Geräte durch das Bordnetz versorgt werden. Daher muss in diesem Fall eine externe regu-

lierbare Quelle zur Verfügung stehen, wenn das Gerät außerhalb des Vehikels getestet werden soll.

Auch bei batteriebetriebenen Geräten kann die Arbeit am geöffneten Gerät äußerst mühsam werden, da die Kontakte oft durch das Gehäuse gehalten werden und ohne Gehäuse keine richtige Stromversorgung sichergestellt werden kann.

Vorsicht!

Arbeiten Sie niemals an geöffneten mit Strom versorgten Geräten, wenn diese mit Netzspannung oder Spannungen über 24 V versorgt werden, es sei denn, Sie besitzen entsprechende Kenntnisse bzw. eine Elektrotechnikausbildung. Geräte mit integrierten Netzteilen sollten Sie niemals geöffnet an das Stromnetz anschließen, da hier an manchen Stellen lebensgefährliche Netzspannung anliegen kann. Geräte mit einer Versorgungsspannung unter 24 V können im Betrieb und geöffnet analysiert werden. Hier besteht unter Spannung trotzdem weiterhin Stromschlag- und Kurzschlussgefahr (jedoch sind keine gesundheitlichen Auswirkungen für gesunde Personen zu erwarten).

Abb. 1.6: Labornetzgerät der Firma VOLTCRAFT (Eigenmarke der Elektronikkette Conrad Electronic)

Bei integrierten Netzgeräten ist es Best Practice, das integrierte Netzgerät abzuklemmen und durch ein Labornetzgerät zu ersetzen. Dazu muss bekannt sein, welche Ausgangsspannung durch das interne Netzgerät erzeugt wird. Dies kann beispielsweise durch Identifikation der Spannungsregler eruiert werden.

Geräte wie das in Abbildung 1.7 sollten Sie niemals geöffnet an Netzspannung betreiben, wenn Sie keine ausreichende Qualifikation im Bereich Elektrotechnik besitzen.

Tipp

Wenn Sie noch kein Labornetzgerät besitzen, empfiehlt sich bei der Anschaffung ein Mehrkanalgerät, also eines mit zwei oder mehr Ausgängen. Leistung ist im IoT-Bereich meistens nebensächlich. Ein Spannungsbereich von 0 bis 30 V und Ausgangsstrom von bis zu 2,5 A sollten für die meisten Aufgaben ausreichend sein. Genaue Einstellungsmöglichkeiten (digital) sind von Vorteil.

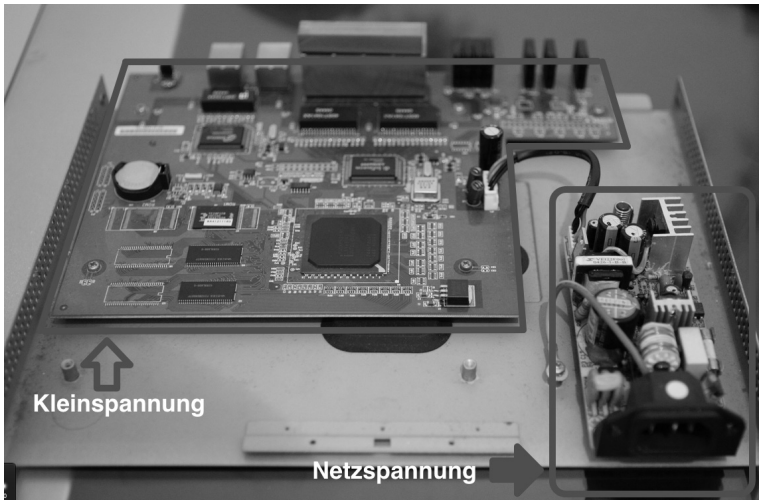

Abb. 1.7: Integriertes Netzgerät – hier besteht Lebensgefahr, wenn das Gerät geöffnet an das Netz angeschlossen wird!

Die richtige Spannung

Die Ausgangsspannung muss an das zu testende Gerät entsprechend angepasst werden. Um diese Spannung zu eruieren, gibt es drei Methoden:

- **Die Spannung ablesen:** Oftmals ist die Eingangsspannung am Typenschild vermerkt. Das ist insbesondere bei Geräten, die ein externes Netzgerät besitzen, der Fall – manchmal ist die zulässige Betriebsspannung jedoch auch bei batteriebetriebenen Geräten im Handbuch vermerkt.
- **Die Spannung berechnen:** Bei batteriebetriebenen Geräten können Sie die Spannung einfach berechnen. Bei in Serie geschalteten Batterien addiert sich die Spannung, bei parallel geschalteten Batterien bleibt die Spannung gleich, aber die Kapazität verdoppelt sich. In der Praxis kommt nahezu ausschließlich die Serienschaltung vor, da eine Parallelschaltung unerwünschte Nebeneffekte hat (z. B. Ausgleichsströme zwischen den Zellen). Ausnahmen bilden Lithium-Akkumulatoren mit Ausgleichsbrücken, hier ist die Parallelschaltung ebenfalls gängig. Auch Kombinationen sind möglich.

- **Die Spannung messen:** Mit dem Multimeter einfach den Spannungseingang abgreifen.

Beispiel

Nachfolgend als Beispiel das Danalock. Es handelt sich um ein batteriebetriebenes Gerät mit vier in Serie geschalteten 3-V-Lithium-Batterien CR123A. Durch die Serienschaltung ergibt sich: 4 * 3 V = 12 V Eingangsspannung.

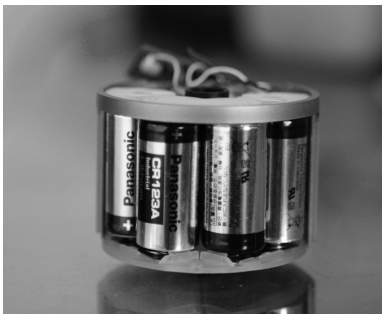

Abb. 1.8: Danalock mit Batterien

Abbildung 1.9 zeigt die Verbindungen für die Serienschaltung der Batterien.

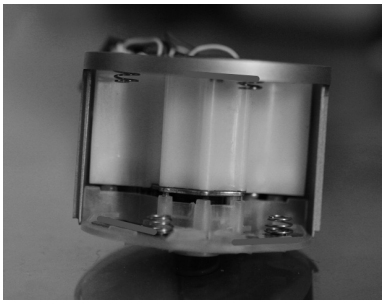

Abb. 1.9: Danalock ohne Batterien – Serienschaltung sichtbar

1.4.2 Multimeter

Ein Multimeter kann mehrere Messgrößen erfassen und ist ein unverzichtbares Universalwerkzeug für Elektronikarbeiten. Im Rahmen der Analyse sind besonders die Widerstandsmessung und die Spannungsmessung von Interesse, da mit diesen Belegung und Funktion der Bausteine und -gruppen grob analysiert werden können.

So können mit dem Multimeter die Verbindungen der Bausteine zur Versorgung und zu anderen Bauteilen untersucht werden. Grundsätzlich gilt: Widerstands- und Durchgangsmessungen werden niemals im Betrieb des zu prüfenden Geräts durchgeführt.

Spannungsmessungen müssen natürlich im Betrieb durchgeführt werden – dies jedoch nur, wenn die Versorgungsspannung keine Gefahr darstellt (siehe Kasten in Abschnitt 1.4.1).

Abb. 1.10: Multimeter der Firma FLUKE

Die Investition in ein gutes Multimeter zahlt sich zu Beginn einer IoT-Reverse-Engineering-Karriere durchaus aus und bewahrt Sie vor Frust.

Spannung messen

Elektrische Spannung ist ein Potenzial zwischen zwei Leitern. Daher wird sie immer zwischen zwei Bezugspunkten gemessen. In den meisten Fällen in diesem Buch ist der Bezugspunkt Masse (Ground, GND) mit dem Potenzial null, da die Messungen auf Frequenzsignale oder differenzielle Signale mit Logic Analyzern oder mittels Oszilloskop erfolgen.

Im IoT-Kontext sind fast ausschließlich Gleichspannungsmessungen von Interesse. Am Multimeter sind diese mit »VDC« oder »V=« bezeichnet (wobei bei dem =-Symbol die untere Linie gestrichelt ist).

Dies ist aber nicht immer der Fall. So sind die Potenziale bei differenziellen Bussystemen nicht gegen GND, sondern jeweils zueinander zu verstehen. Grundsätz-

lich können True-RMS-Messgeräte Wechselspannungen zwar exakt erfassen, für Bussignale sind sie aber nicht geeignet. Hier ist ein Logic Analyzer vonnöten.

Beispiel

Besitzt ein Gerät mehrere Spannungsschienen, zum Beispiel 1,8 V, 3,3 V und 5 V, sind diese gegen den gemeinsamen Bezugspunkt Masse angegeben. Gegeneinander besitzen diese Spannungsschienen ebenfalls ein Potenzial. So besitzt 5 V gegenüber der 3,3-V-Schiene ein Potenzial (eine Spannung) von 1,7 V. Würde man die 1,8-V-Schiene gegen die 5-V-Schiene messen, erhielte man 3,2 V.

Widerstand messen

Die Widerstandsmessung erfolgt immer im ausgeschalteten Zustand, also ohne Betriebsspannung. Bei der Widerstandsmessung verhält sich das Messgerät wie eine *Konstantstromquelle*. Das bedeutet, das Messgerät versucht, einen (sehr geringen) Strom zwischen den Messpunkten zu erhalten. Gemäß dem ohmschen Gesetz wird hierzu die Spannung der Quelle verändert, bis diese den gewünschten Stromfluss erzielen kann.

Dies ist in den allermeisten Fällen kein Problem, jedoch kann es bei sehr empfindlichen ICs oder Bauteilen zu Beschädigungen kommen. Ein Beispiel sind CMOS-Bausteine, die durch ihren hohen Innen- oder Eingangswiderstand die Stromquelle dazu veranlassen, eine hohe Spannung zu liefern.

Im eingebauten Zustand (also wenn das Bauteil in einer Platine eingelötet ist) kann es außerdem dazu kommen, dass nicht das gewünschte Bauteil gemessen wird, sondern die Schaltung um dieses Bauteil herum (siehe Abbildung 1.11).

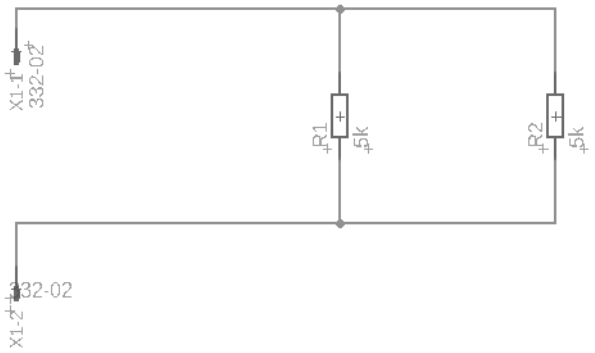

Abb. 1.11: Würde hier an R1 gemessen werden, erhielte man den Parallelwiderstand 2,5 kΩ von R1 und R2 und nicht den Wert 5 kΩ von R1.

1.4.3 Computer

Als Herzstück des Labors ist der Rechner unabdinglich. An dieser Stelle empfehlen wir für den Einstieg die Linux-Distribution Kali. Diese kann unter `https://www.kali.org/downloads/` heruntergeladen werden. Kali hat bereits sehr viele Penetration-Testing-Tools an Bord, und es existiert eine ausgeprägte Community, die bei Fragen sehr hilfreich ist.

Zwar lässt sich Kali auch in einer virtuellen Maschine installieren, für das Testen von IoT-Geräten ist diese Installationsart aber eher ungeeignet. Eine physische Maschine mit Kali erspart Stress und Konfigurationseskapaden.

1.4.4 Oszilloskop

Ein Oszilloskop nimmt Signale auf und zeichnet sie als Spannungs-Zeit-Diagramm. Dies ist bei analogen oder unbekannten Signalen sehr hilfreich, um sie zu identifizieren. Für digitale Signale ist es zwar geeignet, aber nicht unbedingt notwendig.

Abb. 1.12: Oszilloskop von ROHDE & SCHWARZ, 4 Kanäle, Mixed-Signal

Ein Oszilloskop ist eine größere Investition und am Beginn der IoT-Testing-Karriere oft noch nicht zwingend notwendig. Interessant sind sogenannte Mixed-Signal-Oszilloskope, die klassisches Oszilloskop und Logic Analyzer in sich vereinen.

Wichtige Kenngrößen für Oszilloskope sind die Abtastrate in Samples pro Sekunde, die Genauigkeit, die Speichertiefe sowie Kanäle und Ausstattung. Mindestens zwei Kanäle und eine Abtastrate von 2 GS/s sollte ein Oszilloskop für den Einstieg besitzen.

1.4.5 Logic Analyzer

Ein Logic Analyzer ist ähnlich wie ein Oszilloskop ein Signal-Interpreter. Der Unterschied besteht in der Ausführung: Während Oszilloskope zwischen einem

und vier Kanälen besitzen, beginnen Logic Analyzer erst bei vier Kanälen – meistens jedoch sind acht Kanäle und mehr Standard. Außerdem ist der Logic Analyzer auf das Abtasten und Interpretieren digitaler Signale ausgerichtet, mit denen in diesem Buch bzw. während der IoT-Analyse hauptsächlich gearbeitet wird. So können UART- und I2C/SPI-Signale erkannt und interpretiert werden. Während manche Logic Analyzer auch in der Lage sind, analoge Signale abzutasten und anzuzeigen, besteht der große Vorteil der Analyzer darin, dass digitale Signale entsprechend umgesetzt bzw. interpretiert werden können – insbesondere bei höheren Geschwindigkeiten.

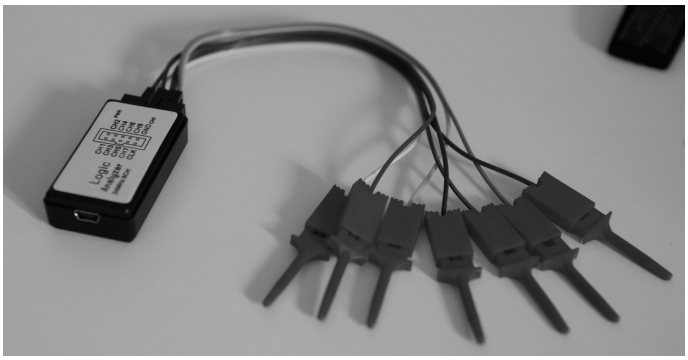

Abb. 1.13: Ein günstiger Logic Analyzer

Wichtige Kenngrößen für Logic Analyzer sind die Abtastrate (Sampling Rate), die Speichertiefe und die Kanalanzahl. Für das Hardware-Testing empfiehlt sich zu Beginn ein Logic Analyzer mit mindestens acht Kanälen und einer Sampling Rate mit ca. 10 MS/s bis 50 MS/s.

Je nach Gerät gibt es eine eigene Software. Sie können jedoch auch das Open-Source-Projekt »sigrok« und die dazugehörige GUI »pulseview« verwenden.

Abb. 1.14: Hochwertiger Logic Analyzer der Firma Saleae

sigrok besitzt ein breites Repertoire an Signaldecodern und lässt diese auch aufeinander »stacken«. Das erspart oft mühsames Decodieren von Befehlen und Daten. Außerdem lässt sich einfach ein neuer Decoder entwickeln, wenn noch kein passender vorhanden ist.

Sie können auf Ihrem Kali-Rechner sigrok und pulseview via `apt` installieren:

```
apt install pulseview sigrok sigrok-*
```

Oft ist jedoch die durch Kali-Paketquellen verfügbare Version deutlich älter als die aktuelle Entwicklerversion. Diese können Sie direkt von der Entwicklerseite kompilieren und installieren: `https://sigrok.org/wiki/Building`.

Arbeiten mit dem Logic Analyzer

Um mit dem Logic Analyzer zu arbeiten, empfiehlt es sich, ihn zunächst »kennenzulernen«, sprich einfache Versuchsaufbauten durchzuführen, deren Ergebnis oder Inhalt bekannt ist, bevor man sich an die Analyse von richtigen Geräten wagt.

Sie können einen einfachen Versuchsaufbau durchführen, indem Sie einen Raspberry Pi an einen I2C-Baustein anschließen.

In Abbildung 1.15 ist der Analog-Digital-Konverter ADS1115 von Adafruit über I2C an einen Raspberry Pi angeschlossen. Die einzelnen Pins sind so einfach abnehmbar.

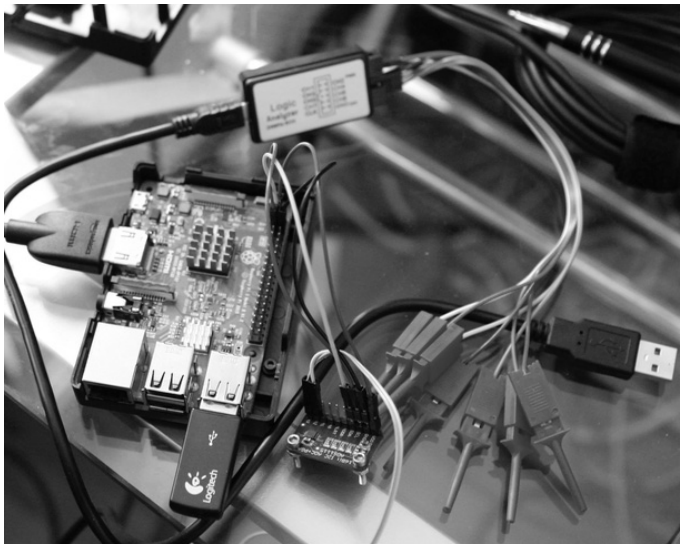

Abb. 1.15: Beispielaufbau, um den Logic Analyzer zu testen – LDA1115 mit Raspberry Pi und einfachem Logic Analyzer

Sie können für den Aufbau auch ein Breadboard verwenden – dieses ermöglicht ein einfacheres Zusammenstecken der benötigten Kontakte. Ein Tutorial und ein Beispielprogramm für den ADS1115 finden Sie auf `adafruit.com`[3].

Sobald der Basisaufbau steht, verbinden Sie GND des Logic Analyzer mit dem GND des Raspberry Pi, um eine gemeinsame Referenzmasse zu schaffen. Verbinden Sie die Kanäle 1 und 2 mit SCL und SDA.

Vorsicht!

Nehmen Sie Verbindungen mit dem Analyzer und Verdrahtungen immer im ausgeschalteten Zustand vor. Die Klemmen können leicht Kurzschlüsse auslösen. Dies kann zur Beschädigung der Komponenten führen.

Starten Sie nun pulseview auf Ihrem Computer und richten Sie die Kanäle korrekt ein. Wählen Sie dazu Ihren Logic Analyzer und verwenden Sie folgende Einstellungen:

- Abtastrate: 5 MSa/s oder mehr
- Sampletiefe 50 MSamples (10 Sekunden)

Starten Sie nun das Beispielprogramm auf dem Raspberry Pi – es zeigt die Eingangswerte des Analog-Digital-Konverters (ADC) an.

Starten Sie möglichst davor oder zeitgleich den Aufzeichnungsvorgang in pulseview. Nun sehen Sie in pulseview die Daten, die gelesen werden.

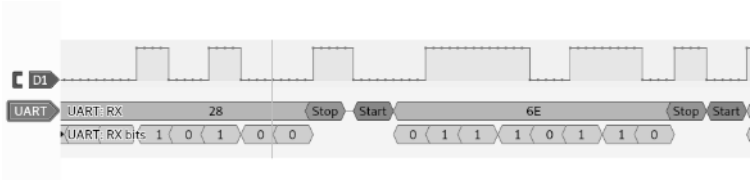

Abb. 1.16: UART in pulseview

Nyquist-Shannon-Abtasttheorem

Gemäß dem Abtasttheorem von Nyquist-Shannon[4] gehen Informationen verloren, wenn die Abtastrate kleiner als das Doppelte der höchsten vorkommenden zu erfassenden Frequenz ist.

Dies bedeutet in der Praxis, dass ein Bussignal mit 10 MHz mit mindestens 20 MHz (entspricht 20 MS/s) abgetastet werden muss, um alle Informationen

3 `https://learn.adafruit.com/adafruit-4-channel-adc-breakouts/`
4 `https://de.wikipedia.org/wiki/Nyquist-Shannon-Abtasttheorem`

vollständig zu erfassen. Je nach Bustyp und Informationsfluss kann das bereits herausfordernd werden. Während UART-Signale typischerweise bei 115.000 Baud liegen (Baud ist im Gegensatz zur SPI-Taktfrequenz in MHz die Datenübertragungsrate, da es bei asynchronen Schnittstellen kein getaktetes Clock-Signal gibt – bei 115.200 Baud ist die höchste vorkommende Frequenz 115.200 Hz, also 115,2 kHz) und damit bei einer Abtastfrequenz von 500 kHz bereits »oversampled« sind, können SPI-Signale durchaus 20 MHz bis 80 MHz erreichen.

Manche Bausteine erlauben es allerdings, den Datendurchsatz zu erhöhen, indem aus einem Input-Pin ein Output-Pin wird.

1.4.6 Raspberry Pi als Universaltool

Der kostengünstige Raspberry Pi-Einplatinencomputer ist durchaus leistungsfähig und kann das Labor um einige Funktionen erweitern. So können Sie die Anschaffung von zusätzlichen Tools umgehen und gleichzeitig die elektronischen Kenntnisse testen bzw. erweitern.

Der Raspberry kann eingesetzt werden als:

- **SPI-Programmierer:** Statt eines Shikra- oder eines ähnlichen Programmierers kann der Raspberry durch seine Ausstattung SPI-Chips lesen und schreiben.
- **JTAG-Debugger:** Mit etwas Installationsaufwand kann der Raspberry mit OpenOCD als JTAG-Debugger verwendet werden.
- **SWD-Debugger:** Ebenso kann der Raspberry als SWD-Debugger verwendet werden.
- **I2C-Tester/-Injecter:** Der Raspberry besitzt einen zugänglichen I2C-Port und kann so Speicher und Sensoren ansprechen. Er kann jedoch nur als I2C-Master agieren, nicht als Slave.
- **UART-/serielle Konsole:** Durch die zugänglichen UART-Pins kann der Raspberry auch als serielles Terminal verwendet werden.
- **Bluetooth-/LE-Sniffer und Interface:** Manche Versionen des Raspberry besitzen bereits einen Bluetooth-Chip onboard und können so für manche Bluetooth-Angriffe verwendet werden.

Der Raspberry ist also – korrekt konfiguriert – ein durchaus leistungsfähiger Helfer im Hardware-/IoT-Bereich.

Wir empfehlen die Durchführung dieser Vorbereitungsarbeiten, um bereits ein gewisses Gefühl für Hardware Testing und den Raspberry Pi zu bekommen.

SPI-basierte Bausteine – flashrom

Um SPI-Flash Bausteine auszulesen, müssen Sie auf dem Raspberry Pi zunächst SPI aktivieren. Verwenden Sie das Betriebssystem Raspbian, können Sie dies mittels der Einstellungs-GUI durchführen.

Wählen Sie hierzu den Menüeintrag EINSTELLUNGEN | RASPBERRY-PI-KONFIGURATION aus.

Wechseln Sie auf die Registerkarte SCHNITTSTELLEN und aktivieren Sie SPI. Sie können auch zusätzlich I2C und SERIAL PORT anschalten. Dies ist für spätere Beispiele nützlich.

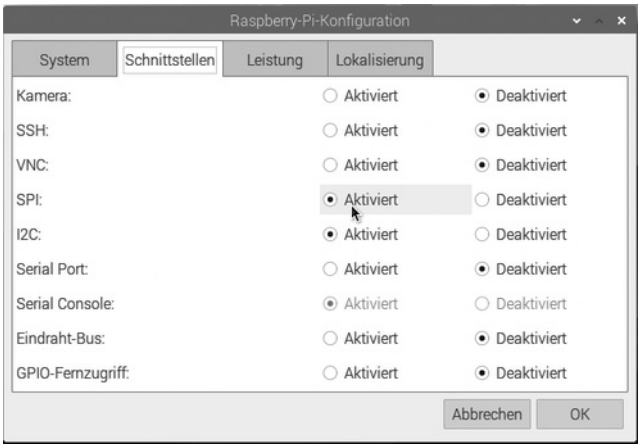

Abb. 1.17: Aktivieren Sie die entsprechenden Bausteine: I2C, SPI und eventuell den seriellen Port.

Mit dem Kommando

```
sudo apt install flashrom
```

können Sie nun das Programm »flashrom« installieren. Dieses Programm wird benötigt, um Speicherbausteine (ROM, Flash etc.) auszulesen oder zu schreiben.

Verdrahtung

Um mit dem Raspberry nun einen Flash-Speicher (unabhängig davon, mit welcher Schnittstelle dieser ausgestattet ist – SPI, I2C etc.) zu lesen oder zu schreiben, muss der Raspberry mit dem Speicher verbunden und der Speicher muss natürlich mit Spannung versorgt werden.

Vorsicht!

Sämtliche GPIO-Pins am Raspberry Pi sind mit 3,3 V zu beschalten! Benötigen Sie andere Spannungslevel, müssen Sie einen Level-Konverter verwenden!

Mit dem Tool »pinout« können Sie die aktuelle Pin-Konfiguration des Raspberry abfragen. Dabei werden jedoch SPI-Pins als GPIO-Pins angezeigt.

```
pi@raspberrypi:~ $ umount /dev/sda2
pi@raspberrypi:~ $ pinout
,--------------------------------.
|  oooooooooooooooooo J8     +====|
|  1ooooooooooooooooooo      | USB|
|                            +====|
|       Pi Model 3B  V1.2        |
|      +----+                +====|
|  |D|  |SoC |                | USB|
|  |S|  |    |                +====|
|  |I|  +----+                   |
|                     |C|    +======|
|                     |S|    |   Net|
|  pwr        |HDMI|  |I||A|  +======|
`-| |--------|    |----|V|-------'

Revision           : a52082
SoC                : BCM2837
RAM                : 1024Mb
Storage            : MicroSD
USB ports          : 4 (excluding power)
Ethernet ports     : 1
Wi-fi              : True
Bluetooth          : True
Camera ports (CSI) : 1
Display ports (DSI): 1

J8:
   3V3  (1) (2)  5V
 GPIO2  (3) (4)  5V
 GPIO3  (5) (6)  GND
 GPIO4  (7) (8)  GPIO14
   GND  (9) (10) GPIO15
GPIO17 (11) (12) GPIO18
GPIO27 (13) (14) GND
GPIO22 (15) (16) GPIO23
   3V3 (17) (18) GPIO24
GPIO10 (19) (20) GND
 GPIO9 (21) (22) GPIO25
GPIO11 (23) (24) GPIO8
   GND (25) (26) GPIO7
 GPIO0 (27) (28) GPIO1
 GPIO5 (29) (30) GND
 GPIO6 (31) (32) GPIO12
GPIO13 (33) (34) GND
GPIO19 (35) (36) GPIO16
GPIO26 (37) (38) GPIO20
   GND (39) (40) GPIO21

For further information, please refer to https://pinout.xyz/
pi@raspberrypi:~ $
```

Abb. 1.18: Ausgabe von pinout auf einem Raspberry Pi 3B

In folgender Tabelle sind die SPI-Pins bzw. die für flashrom benötigten Pins am Raspberry Pi aufgelistet:

RPI-Pin	Bezeichnung	Beschreibung/Verbindung
17	+3.3V VCC	Versorgungsspannung 3,3 V. Wenn benötigt, an **VCC** des Flash-Speichers.
19	MOSI	**M**aster **O**ut, **S**lave **I**nput: Daten vom Master (Raspberry) an den Slave (Flash-Chip). Wird an den **DI**- oder **SI**-Pin des Speichers angeschlossen.
21	MISO	**M**aster **I**n, **S**lave **O**ut: Daten von Slave an den Master. Wird an den **DO**- oder **SO**-Pin des Speichers angeschlossen.
23	SCLK	Clock: Takt, wird an **CLK** oder **SCLK** des Speichers angeschlossen.
24	SPI0 CE0	**C**hip **E**nable/Chip Select: Dieser Pin wird bei SPI verwendet, um den angesprochenen Teilnehmer zu adressieren. Wird an **CS** des Speichers angeschlossen.
25	GND	Ground: Muss mit dem **GND** des Speichers verbunden werden, um ein gemeinsames Massepotenzial zu schaffen.

Im Gegensatz zu den JTAG-/SWD-Pins sind diese Pins *nicht* veränderbar.

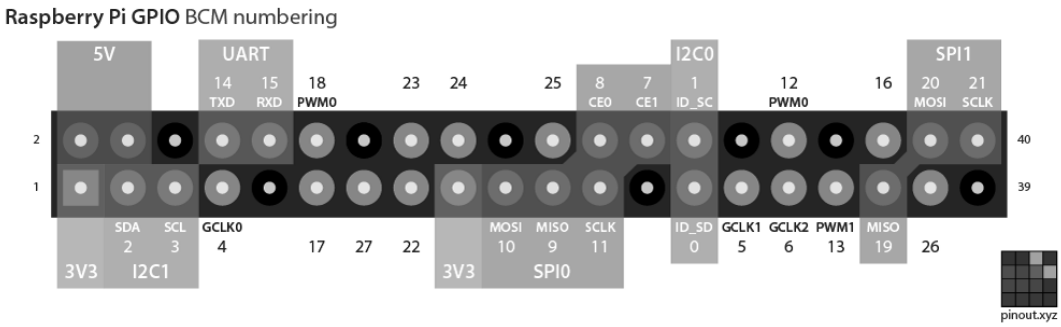

Abb. 1.19: Raspberry Pi-Pin-out (Quelle: `https://github.com/gadgetoid/Pinout.xyz`)

Zur Verkabelung empfehlen wir SMD-Klemmen, wenn keine Stift- oder Sockelleisten mit den benötigten Pins verfügbar sind, SOIC-Klemmen bei Chips oder eine direkte Verlötung von Testkabeln an Löt-/Prüfpunkten der Platine.

Ein Beispiel hierzu finden Sie in Abschnitt 4.4.5.

Raspberry als SWD-/JTAG-Adapter

Um den Raspberry als JTAG- oder SWD-Adapter nutzen zu können, müssen Sie zunächst die Software OpenOCD installieren. Dies kann aus den Paketquellen erfolgen, besser ist es jedoch, die aktuellste Version selbst zu kompilieren. Dazu müssen Sie zunächst die Build-Essentials installieren und dann den OpenOCD-Quellcode vom entsprechenden Git-Repository laden und kompilieren.

OpenOCD via `apt` installieren:

```
sudo apt install openocd
```

OpenOCD direkt aus dem Quellcode erstellen:

```
sudo apt-get update
sudo apt-get install git autoconf libtool make pkg-config libusb-1.0-0
libusb-1.0-0-dev
git clone http://openocd.zylin.com/openocd
cd openocd-code
./bootstrap
./configure --enable-sysfsgpio --enable-bcm2835gpio
make
sudo make install
```

Dieser Vorgang benötigt ca. 20 Minuten auf einem Raspberry Pi 3, danach ist OpenOCD in der aktuellen Version kompiliert und installiert.

OpenOCD arbeitet hauptsächlich mit Konfigurationsdateien, im Normalfall sollte eine Konfigurationsdatei für den Adapter (in diesem Fall den Raspberry) und eine für das Target angegeben werden (IC, zu dem die Verbindung aufgebaut werden soll).

Die Konfigurationsdateien für die Adapter finden Sie unter:

```
/usr/local/share/openocd/scripts/interface/
```

Für den Raspberry wird die Konfigurationsdatei `raspberrypi2-native.cfg` verwendet (diese funktioniert auch für das Modell 3).

Die Konfigurationsdateien für die Targets finden Sie unter:

```
/usr/local/share/openocd/scripts/target/
```

Ist das gewünschte Ziel nicht enthalten, lohnt es sich oft, öffentliche Git-Repositories zu durchsuchen.

Konfiguration anpassen

Die Standardkonfiguration für den Raspberry verwendet Ports, die auch gleichzeitig anderweitig genutzt werden könnten. Daher sollten Sie die Konfiguration ändern.

Vorsicht!

Die Nummer der GPIO-Pins entspricht nicht der physikalischen Pin-Nummer (siehe hierzu Abbildung 1.19).

Die wichtigen Zeilen in der Konfiguration sind die Zeilen 3 und 7 im folgenden Listing:

```
# Each of the JTAG lines need a gpio number set: tck tms tdi tdo
# Header pin numbers: 23 22 19 21
bcm2835gpio_jtag_nums 11 25 10 9
# Each of the SWD lines need a gpio number set: swclk swdio
# Header pin numbers: 23 22
bcm2835gpio_swd_nums 11 25
```

Ändern Sie diese wie folgt (legen Sie dazu eine Kopie in einem lokalen Verzeichnis an):

```
# #old# bcm2835gpio_jtag_nums 11 25 10 9
bcm2835gpio_jtag_nums 4 22 23 24
[…]
# #old# bcm2835gpio_swd_nums 11 25
bcm2835gpio_swd_nums 23 24
```

Somit bleiben die Pins für SPI und UART frei. Außerdem sind so die Pins für JTAG und SWD »geshared«, man benötigt also insgesamt weniger Pins und kann mit einem Kabel für SWD und JTAG arbeiten.

1.4.7 Abgreifklemmen und Adapter

Abgreifklemmen gibt es in verschiedenen Ausführungen zu verschiedenen Zwecken. So gibt es die Elektrotechnik-Abgreifklemmen mit 4-mm-Bananenstecker-anschluss. Diese sind geeignet, um ein Gerät mit Strom zu versorgen, solange es nicht mehr als 2A bezieht.

Außerdem gibt es SMD-Klemmen in verschiedenen Ausführungen, die dazu geeignet sind, SMD-Bausteine abzugreifen. Zur Versorgung eignen sie sich aufgrund der geringen Belastbarkeit nicht, es sei denn, es handelt sich um ein Gerät mit äußerst niedrigem Energiebedarf.

Abb. 1.20: Verschiedene Messklemmen

Bei SOIC-/SMD-Bausteinen kann dieses Abgreifen bereits zu einer sehr anstrengenden Feinarbeit ausufern, wie Abbildung 1.21 zeigt. Hier muss besonders darauf geachtet werden, dass keinerlei Kurzschluss entsteht. Außerdem sind diese Verbindungen meistens nicht sehr stabil und führen oft aus Platzgründen zu wackeligen Verbindungen.

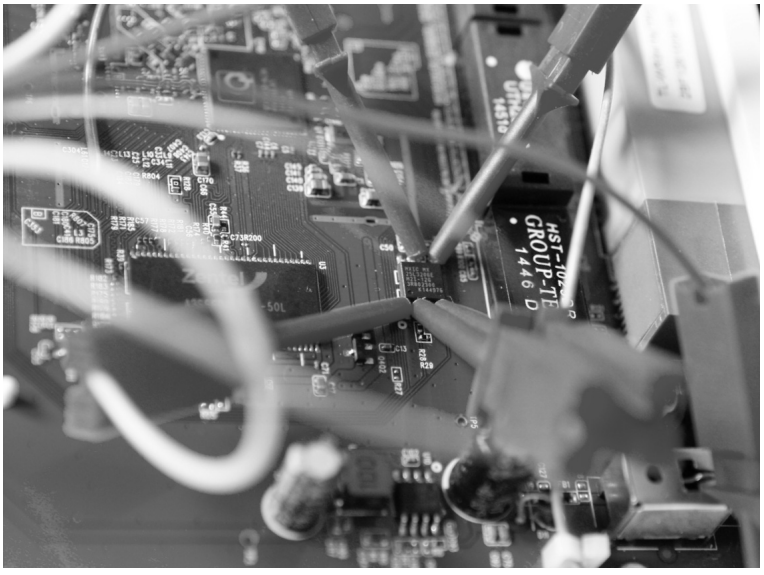

Abb. 1.21: Abgreifen mit SMD-Klemmen – nicht immer einfach

Bei achtpoligen ICs ist dieses Vorgehen gerade noch durchführbar, darüber sollten Sie auf SOIC-Adapter ausweichen. Diese Klemmen greifen den gesamten IC-Baustein *In-Circuit* ab (also aufgelötet, während er sich in dem Schaltkreis befindet) und bieten Anschlüsse für den Logic Analyzer oder den SPI-Adapter.

Der SOIC-Adapter wird auf den Chip geklemmt – je nach Ausführung ist er mit oder ohne Verkabelung zu haben. Die verlötete Verkabelung bietet einen besseren Kontakt, sofern sie korrekt durchgeführt wurde.

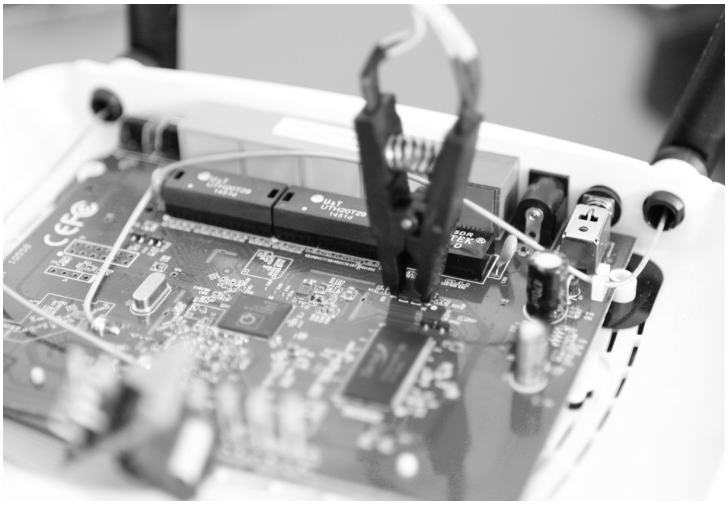

Abb. 1.22: Besser: mit dem SOIC-Adapter den Chip direkt abgreifen

Auf dem Markt erhältlich sind Sets für verschiedenste Größen und Ausführungen. Die SOIC-Ausführung ist durchaus gängig – in der Regel sind achtpolige Adapter am häufigsten zu verwenden. Sie müssen sich zu Beginn also kein ganzes Set anschaffen, wie in Abbildung 1.23 gezeigt.

Wichtig!

Die SOIC-Adapter funktionieren nur, wenn der IC komplett eingelötet ist. Wird eine Verbindung unterbrochen, wie im Beispiel in Abschnitt 4.4.5 »SPI«, lässt sich der Adapter oft nur schwer anbringen. In diesem Fall sind SMD-Klemmen besser geeignet.

Vorsicht!

Verbinden Sie niemals VS(S) oder VD(D) (positive Spannungsversorgung), wenn das Gerät, auf dem sich der IC befindet, unter Spannung ist, mit den Versorgungspins des Raspberry oder des Adapters!

Abb. 1.23: SOIC-Klemmenset – für Ambitionierte

Stichwortverzeichnis

0-Ω-Widerstand 62
32-Bit-Controller 98
8-Bit-Mikrocontroller 96

A
Abgreifklemmen 41
Abtasttheorem von Nyquist-Shannon 35
Advertisement Channel 170
Aktuator 71
All-in-One-SoC 70, 100
Amazon Web Service (AWS) 236
Android Embedded Devices 99
APK 51
Application Framework 196
Application Support Sub-Layer 196

B
Backdoor 157, 159
Baud 36
Baudrate 116
Bausteinverblendung 89
BGA 68
Bindshell 157, 159
Binvis 139
Binwalk 139
BlackBox-Analyse 46, 126
    8-Bit-Controller 98
Bluebugging 180
Bluejacking 180
Bluesnarfing 179
Bluesniffing 180
Bluetooth Low Energy 167
    Fuzzing 193
    Pairing 171
    Protokoll-Stack 168
    Sicherheit 178
Bluetooth-Sniffer 36
Bluetooth-Stack *siehe* BT-Stack
Broker 205
BtleJuice 190
BT-Stack 96
Bussystem 74
busybox 153

C
Canary 46
CE *siehe* Chip Enable
Chip Enable 74
Chip Select 74
Compliance 20
Compressed ROM File System 134
CONNACK 211, 214
CONNECT 211, 212
Controller 67
Controller mit Funkbaustein 97
Coordinator 197
CPIO 143
cpio 141
Crackle 192
CramFS 134
CS *siehe* Chip Select

D
Datenblatt 79
Diode 66
DIP/DIL 68
DISCONNECT 211
Diskrete Bauelemente 62
DSGVO 46
Dynamische App-Analyse 224

E
Elektrische Spannung 30
Entropie
    Binärdatei 138
    Checks 138
Epoxyd-Harz 91

F
FCC-ID 48
firmwalker 149
Firmware 131, 158
    dynamische Analyse 154
    Emulation 150
    Extraktion 105
    Image entpacken 138
    Manipulation 157

manuelle Analyse 147
statische Firmware-Analyse 145
firmware-analysis-toolkit 151
firmware-mod-kit 160
Fixed Header 211
Flash 127
flashrom 36, 123
benötigte Pins 38
Flash-Speicher extrahieren 109

G
Galvanische Trennung 26
GATT 176
Gattacker 185
Gehäuse
Tamper Protection 84
Generic Access Profile 168
Generic Attribute Profile 169
Glasfaserradierstift 104
GND 30
Google Cloud Service 237
Gradle 52
Ground 30
GSM 127
GTKTerm 118

H
Hardware, Design 95

I
I2C 50, 75
I2C-Tester 36
IC 67
IC, Typen 68
iMazing 220
In-Circuit 43
Induktivität 65
Infrastructure as a Service (IaaS) 234
Integrierter Schaltkreis *siehe* IC
Internal Photos 49
IoT-Referenzarchitekturen 164

J
jadx 51
JFFS 133
JFFS2 143
JFFS/JFFS2 133
J-Link 109
Joint Test Action Group *siehe* JTAG
Journaling Flash File System 133
JTAG 106
JTAG-Adapter 39
JTAG-Debugger 36
JTagulator 106

K
Killerbee 201
Kondensator 63

L
Labornetzgerät 26
Layout 79
LE-Sniffer 36
Level-Konverter 72
LGA 68
Link Key 199
Linux-Distribution Kali 32
LogFS 135
Logic Analyzer 32
Logik-Gatter 66
Lötstopplack 104
LSB first 74
LTE 127
LZMA 141

M
M2M 205
Masse 30
Mass-Erase 105
MASVS 218
Memory Technology Device *siehe* MTD
Metallblenden 90
Microsoft Azure 235
Mikrocontroller 69
MISO 74
MOSI 74
MQTT 205
Last Will and Testament 210
Quality of Service 209
Retained Messages 209
MQTT Control Packet Typ 211
MSB first 74
MTD 120
Multimeter 29

N
Network Key 199
Netzspannung 27
NTFS 134

O
Obfuscation 51
OE *siehe* Output Enabled
Open Source Intelligence *siehe* OSINT

OpenOCD 39, 111
Operations-Security 55
OSINT 45
  Findings dokumentieren 50
Oszilloskop 32
OTA-Update *siehe* Over-the-Air-Update
Output Enabled 73
Over-the-Air-Update 107
OWASP IoT Top 10 12

P

PCB 67
PFS 134
PGA 68
Philips Hue 201
PINGREQ 211
PINGRESP 211
pinout 37
Platform as a Service (PaaS) 234
PLIST 229
Proof-of-Concept-Exploits 20
Protokolle 163
PUBACK 211
PUBCOMP 211
PUBLISH 211, 215
Publisher 205
Publish-Subscribe-Architektur 205
PUBREC 211
PUBREL 211
Pull-down-Widerstand 92
Pull-up-Widerstand 92
pulseview 33

R

Raspberry Pi-Einplatinencomputer 36
Read Protection 97
Read-out-Protection 105
Replay-Angriffe 200
Reporting 22
Responsible Disclosure 58
RFS 135
romFS 134
Router 197
RX 76

S

Saleae Logic Analyzer 116
Schaltplan 78
Schnittstelle 74
Schutzlack 104
Schutzsiegel 87
SCL 76
SCLK 74
Scope 20
Scoping 19
SDA 76
Security Manager 169
Segger J-Link 109
Sensor 71
Serial Peripherial Interface *siehe* SPI
Serial Wire Debug *siehe* SWD
Serielle Konsole 36, 114
Serielle Schnittstelle 76
Shikra 106
Shodan 55
sigrok 33
Slave Select 74
SmartDevices 99
SMD-Bausteine 42
SMD-Package 68
SoC 67, 69
Software as a Service (Saas) 234
SOIC-Bausteine 42
Spannung
  ablesen 28
  berechnen 28
  messen 29
Spannungswandler 70
Speicher 71
SPI 74, 123
SPI-Programmierer 36
Spule 65
sqlite 231
SquashFS 133, 143
SS *siehe* Slave Select
Staging-Eintrag 53
Statische App-Analyse 222
strings 146
SUBACK 211
SUBSCRIBE 211, 216
Subscriber 205
Supply-Chain-Angriff 87
SWD 50, 107
SWD-Adapter 39
SWDCLK 107
SWD-Debugger 36
SWDIO 107
SWDO 107
Symmetric-Key Key Establishment Protocol 200

T

Tamper Detection 83, 88
Tamper Evidence 83
Tamper Prevention 83

Tamper Protection 50, 83
Gehäuse 84
Tamper Response 83
TAP 106
Test Access Port *siehe* TAP
Testmethodik
BlackBox 19, 46
GreyBox 19
WhiteBox 19
Testtiefe 19
Testziele 20
TExFAT 135
TFAT 135
TFS4 135
Threat-Modell 84
Topics 206
Transistor 66
Transparente Netzwerk-Bridge 226
TrueFFS 135
TTL-Level 73
TX 76

U

UART 50, 76, 114
UART-Konsole 36
Ubertooth One 181
UbiFS 134
UICR 114
Universal Asynchronous Receiver Transmitter *siehe* UART
UNSUBACK 211
UNSUBSCRIBE 211
USB 50, 122
User Information Configuration Register *siehe* UICR

V

Variable Header 212
VFat 134

W

White-Lable-Produkt 47
Widerstand 62
Widerstandsmessung 31

Y

YAFFS 134
Yet Another Flash File System 134

Z

Zigbee 195
Protokoll-Stack 195
Schwachstellen 200