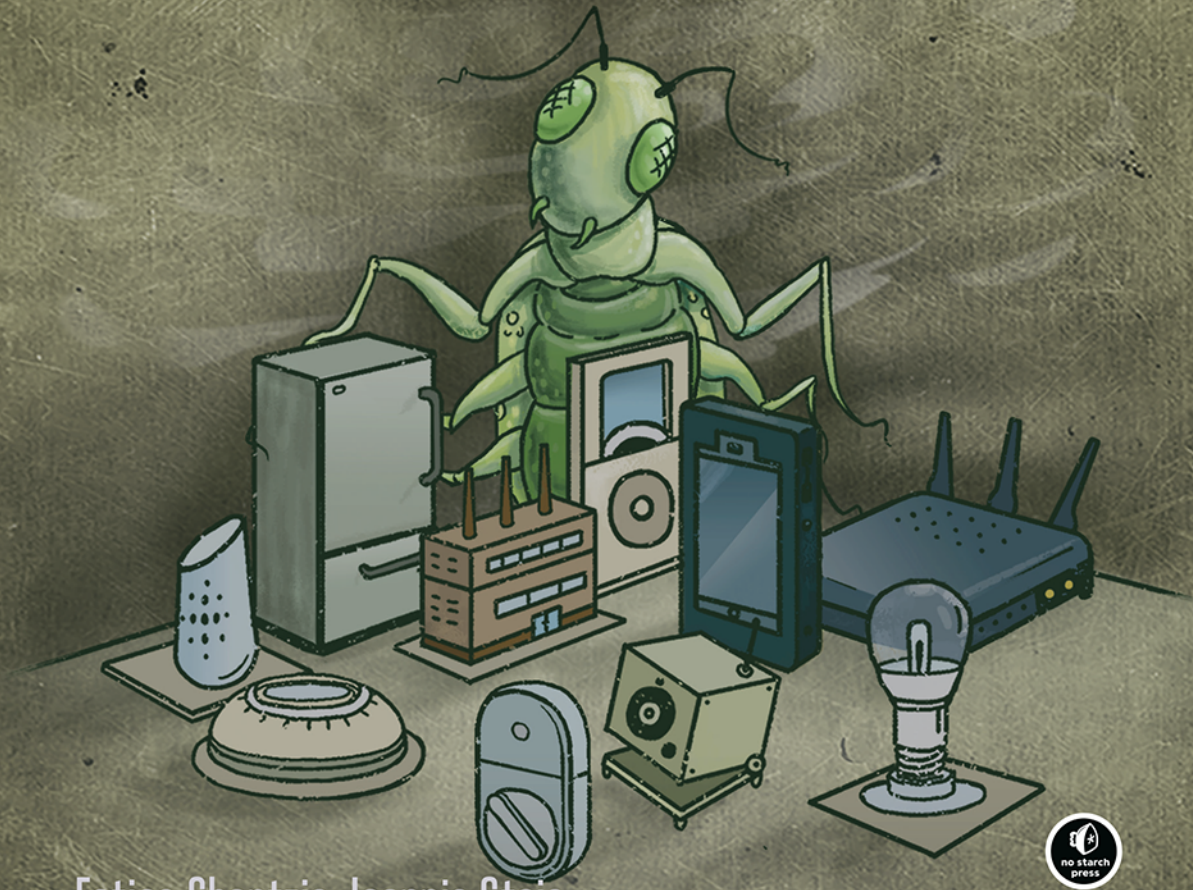
Hakowanie internetu rzeczy w praktyce

Przewodnik po skutecznych metodach atakowania IoT



Fotios Chantzis, Ioannis Stais
Paulino Calderon, Evangelos Deirmentzoglou, Beau Woods

no starch press
Helion

Tytuł oryginału: Practical IoT Hacking: The Definitive Guide to Attacking the Internet of Things

Tłumaczenie: Andrzej Watrak

ISBN: 978-83-283-8339-5

Helion S.A.
ul. Kościuszki 1c, 44-100 Gliwice
tel. 32 231 22 19, 32 230 98 63
e-mail: *helion@helion.pl*
WWW: *http://helion.pl* (księgarnia internetowa, katalog książek)

Drogi Czytelniku!
Jeżeli chcesz ocenić tę książkę, zajrzyj pod adres
http://helion.pl/user/opinie/hainrz
Możesz tam wpisać swoje uwagi, spostrzeżenia, recenzję.

Printed in Poland.

Spis treści

O autorach 13
O współautorach 13
O korektorze merytorycznym 14

PRZEDMOWA 15

PODZIĘKOWANIA 17

WPROWADZENIE 19
Koncepcja książki 19
Dla kogo jest ta książka? 20
Kali Linux 21
Struktura książki 21

Część I. Krajobraz zagrożeń IoT 25

1
BEZPIECZEŃSTWO W ŚWIECIE IOT 27
Dlaczego bezpieczeństwo IoT jest ważne? 28
Czym różni się bezpieczeństwo IoT od tradycyjnego bezpieczeństwa IT? 30
Co jest specjalnego w hakowaniu IoT? 31
Normy, regulacje i wytyczne 32
Studium przypadku: identyfikowanie, zgłaszanie i ujawnianie problemów z bezpieczeństwem IoT 36
Zdaniem eksperta: poruszanie się po świecie IoT 37
Regulacje dotyczące hakowania IoT 37
Rola rządu w bezpieczeństwie IoT 39
Bezpieczeństwo urządzeń medycznych z perspektywy pacjentów 40
Podsumowanie 42

2
MODELOWANIE ZAGROŻEŃ 43
Modelowanie zagrożeń IoT 43
Regulacje dotyczące modelowania zagrożeń 44
Identyfikacja architektury urządzenia 45
Podział architektury na komponenty 46

Określenie zagrożeń 47
Wykrywanie zagrożeń za pomocą drzewa ataku 54
Ocena zagrożenia przy użyciu klasyfikacji DREAD 55
Inne modele zagrożeń, podejścia i narzędzia 56
Typowe zagrożenia IoT 57
Zakłócanie sygnału 58
Odtwarzanie danych 58
Zniekształcanie ustawień 58
Naruszenie integralności sprzętu 58
Klonowanie węzłów 58
Naruszenie bezpieczeństwa i prywatności danych 59
Niska świadomość zagrożeń 59
Podsumowanie 59

3

METODYKA TESTÓW BEZPIECZEŃSTWA 60

Pasywny rekonesans 62
Warstwa fizyczna lub sprzętowa 65
Interfejsy peryferyjne 65
Środowisko rozruchowe 66
Blokady 66
Zabezpieczenia przed modyfikacjami i wykrywanie modyfikacji 66
Oprogramowanie układowe 67
Interfejsy diagnostyczne 67
Fizyczna odporność 67
Warstwa sieciowa 68
Rekonesans 68
Ataki na protokoły i usługi sieciowe 71
Testy protokołów bezprzewodowych 72
Testy aplikacji WWW 73
Tworzenie mapy aplikacji 73
Kontrolki klienckie 74
Uwierzytelnianie użytkowników 74
Zarządzanie sesjami 75
Kontrola dostępu i autoryzacja 75
Weryfikacja danych wejściowych 75
Błędy w algorytmie 76
Serwer aplikacyjny 76
Przegląd konfiguracji hosta 76
Konta użytkowników 76
Siła haseł 77
Uprawnienia kont 77
Poziom poprawek 78
Zdalne utrzymanie 78
Kontrola dostępu do systemu plików 79
Szyfrowanie danych 79
Błędy w konfiguracji serwera 79
Testy aplikacji przenośnych i chmurowych 80
Podsumowanie 81

Część II. Hakowanie sieci 83

4

OCENIANIE SIECI 85

Skok w sieć IoT 85
- Sieci VLAN i przełączniki sieciowe 86
- Imitowanie przełącznika 87
- Podwójne tagowanie 90
- Imitowanie urządzeń VoIP 91

Identyfikowanie urządzeń IoT w sieci 93
- Uzyskiwanie haseł poprzez badanie odpowiedzi usług 94
- Tworzenie własnych sygnatur usług 98

Hakowanie protokołu MQTT 100
- Przygotowanie środowiska testowego 102
- Tworzenie modułu dla programu Ncrack do hakowania poświadczeń w protokole MQTT 104
- Test modułu MQTT 114

Podsumowanie 114

5

ANALIZA PROTOKOŁÓW SIECIOWYCH 115

Badanie protokołów sieciowych 116
- Gromadzenie informacji 116
- Analiza 118
- Prototypowanie i tworzenie narzędzi 119
- Ocena bezpieczeństwa protokołu 119

Tworzenie dekodera protokołu DICOM w języku Lua dla programu Wireshark 121
- Język Lua 121
- Protokół DICOM 121
- Generowanie ruchu DICOM 123
- Włączenie języka Lua w programie Wireshark 123
- Zdefiniowanie dekodera 125
- Zdefiniowanie głównej funkcji dekodera 125
- Skompletowanie dekodera 126

Tworzenie dekodera żądań C-ECHO 127
- Wyodrębnienie ciągów znaków z tytułu jednostki aplikacji 128
- Uzupełnienie funkcji dekodującej 128
- Analiza pól o zmiennych długościach 129
- Test dekodera 130

Tworzenie skanera usługi DICOM dla silnika skryptowego Nmap 131
- Utworzenie biblioteki Nmap dla protokołu DICOM 131
- Identyfikatory DICOM i stałe wartości 132
- Zdefiniowanie funkcji tworzącej i usuwającej gniazdo sieciowe 133
- Zdefiniowanie funkcji wysyłającej i odbierającej pakiet DICOM 134
- Utworzenie nagłówka pakietu DICOM 135
- Utworzenie funkcji wysyłającej żądanie A-ASSOCIATE 136
- Odczytanie parametrów skryptu w silniku Nmap 138
- Zdefiniowanie struktury żądania A-ASSOCIATE 138
- Analiza odpowiedzi A-ASSOCIATE 140
- Utworzenie finalnego skryptu 140

Podsumowanie 142

6

EKSPLORACJA SIECI SAMOKONFIGURACYJNYCH 143

Eksploracja protokołu UPnP 144
- Stos protokołu UPnP 145
- Typowe wady protokołu UPnP 146
- Otwieranie przejść w zaporze sieciowej 147
- Atakowanie protokołu UPnP poprzez interfejs WAN 153
- Inne ataki na protokół UPnP 158

Eksploracja protokołów mDNS i DNS-SD 158
- Jak działa protokół mDNS? 159
- Jak działa protokół DNS-SD? 159
- Rekonesans przy użyciu protokołów mDNS i DNS-SD 160
- Atak na operację sondowania w protokole mDNS 162
- Ataki typu „człowiek pośrodku" na protokoły mDNS i DNS-SD 163

Eksploracja protokołu WS-Discovery 172
- Jak działa protokół WS-Discovery? 172
- Imitowanie kamery 173
- Ataki na protokół WS-Discovery 180

Podsumowanie 181

Część III. Hakowanie sprzętu 183

7

EKSPLORACJA UART, JTAG I SWD 185

Interfejs UART 186
- Narzędzia sprzętowe wykorzystujące interfejs UART 187
- Identyfikacja pinów interfejsu UART 187
- Określenie prędkości transmisji interfejsu UART 191

Interfejsy JTAG i SWD 192
- Interfejs JTAG 192
- Jak działa interfejs SWD? 193
- Narzędzia sprzętowe wykorzystujące interfejsy JTAG i SWD 193
- Identyfikacja pinów interfejsu JTAG 194

Hakowanie urządzenia za pomocą interfejsów UART i SWD 196
- Hakowanie mikrokontrolera STM32F103C8T6 (black pill) 197
- Przygotowanie środowiska diagnostycznego 198
- Utworzenie programu w środowisku Arduino IDE 200
- Załadowanie i uruchomienie programu 203
- Diagnozowanie urządzenia 209

Podsumowanie 217

8

INTERFEJSY SPI I I²C 218

Narzędzia do komunikacji z interfejsami SPI i I²C 219

Interfejs SPI 220
- Jak działa interfejs SPI? 220
- Odczyt zawartości pamięci EEPRM/flash za pomocą interfejsu SPI 221

Interfejs I²C 226
- Jak działa interfejs I²C? 226
- Utworzenie szyny I²C typu kontroler – urządzenie peryferyjne 227
- Hakowanie interfejsu I²C za pomocą urządzenia Bus Pirate 232

Podsumowanie 236

9
HAKOWANIE OPROGRAMOWANIA UKŁADOWEGO 237
Oprogramowanie układowe i system operacyjny 237
Uzyskanie oprogramowania układowego 238
Hakowanie routera Wi-Fi 241
Wyodrębnienie systemu plików 242
Statyczna analiza zawartości systemu plików 243
Emulacja oprogramowania układowego 246
Analiza dynamiczna 252
Otwieranie ukrytych wejść do oprogramowania układowego 254
Hakowanie mechanizmu aktualizacji oprogramowania układowego 259
Kompilacja i konfiguracja 260
Kod klienta 261
Uruchomienie usługi aktualizacji 264
Luki w bezpieczeństwie usługi aktualizacji oprogramowania 265
Podsumowanie 267

Część IV. Hakowanie radia 269

10
RADIO KRÓTKIEGO ZASIĘGU: NADUŻYWANIE RFID 271
Jak działa RFID? 271
Zakresy częstotliwości radiowych 272
Pasywne i aktywne technologie RFID 273
Architektura tagu RFID 275
Tagi RFID niskiej częstotliwości 276
Tagi RFID wysokiej częstotliwości 277
Atakowanie systemów RFID za pomocą urządzenia Proxmark3 277
Przygotowanie narzędzia Proxmark3 278
Aktualizacja urządzenia Proxmark3 278
Identyfikacja tagów niskiej i wysokiej częstotliwości 280
Klonowanie tagu niskiej częstotliwości 281
Klonowanie tagu wysokiej częstotliwości 282
Symulowanie tagu RFID 287
Modyfikacja tagu RFID 287
Atakowanie karty MIFARE za pomocą aplikacji Android 288
Ogólne polecenia dla nieoznaczonych i niekomercyjnych tagów RFID 289
Podsłuchiwanie komunikacji między tagiem a czytnikiem 293
Wyodrębnianie klucza sektora z zarejestrowanych danych 294
Atakowanie czytnika RFID 295
Automatyzacja ataków przy użyciu skryptów Proxmark3 296
Zakłócanie czytnika RFID za pomocą własnego skryptu 297
Podsumowanie 301

11
TECHNOLOGIA BLE 302
Jak działa technologia BLE? 303
Profile GAP i GATT 304

Korzystanie z technologii BLE 305
Urządzenia BLE 305
BlueZ 306
Konfiguracja interfejsów BLE 307
Wykrywanie urządzeń i wyświetlanie charakterystyk 308
Narzędzie GATTTool 308
Narzędzie Bettercap 309
Uzyskiwanie listy charakterystyk, usług i deskryptorów 310
Odczytywanie i zapisywanie charakterystyk 311
Hakowanie technologii BLE 312
Przygotowanie projektu BLE CTF Infinity 312
Pierwsze kroki 313
Flaga 1 — zbadanie charakterystyk i deskryptorów 315
Flaga 2 — uwierzytelnienie 316
Flaga 3 — podszycie się pod adres MAC 317
Podsumowanie 319

12

RADIO ŚREDNIEGO ZASIĘGU: HAKOWANIE WI-FI 320

Jak działa Wi-Fi? 320
Sprzęt do oceniania bezpieczeństwa Wi-Fi 321
Ataki na klientów sieci Wi-Fi 321
Ataki dysocjacyjne i blokujące usługę 322
Ataki asocjacyjne 324
Wi-Fi Direct 329
Ataki na punkty dostępu 332
Łamanie szyfrowania WPA/WPA2 333
Łamanie szyfrowania WPA/WPA2 Enterprise i przechwytywanie poświadczeń 338
Metodyka testów bezpieczeństwa 339
Podsumowanie 340

13

RADIO DALEKIEGO ZASIĘGU: LPWAN 341

LPWAN, LoRa i LoRaWAN 342
Przechwytywanie danych w sieci LoRaWAN 343
Przygotowanie płytki Heltec LoRa 32 343
Przygotowanie klucza LoStik 348
Klucz USB CatWAN jako rejestrator pakietów 352
Dekodowanie protokołu LoRaWAN 357
Format pakietu LoRaWAN 357
Dołączanie do sieci LoRaWAN 359
Hakowanie sieci LoRaWAN 361
Atak bit-flipping 362
Generowanie kluczy i zarządzanie nimi 365
Ataki odtworzeniowe 365
Podsłuchiwanie komunikacji 366
Fałszowanie potwierdzeń 366
Ataki aplikacyjne 366
Podsumowanie 367

Część V. Celowanie w ekosystem IoT 369

14

ATAKI NA APLIKACJE MOBILNE 371

Zagrożenia aplikacji mobilnych IoT 372
- Komponenty środowiska aplikacji mobilnych 372
- Identyfikacja zagrożeń 372

Zabezpieczenia w systemach Android i iOS 374
- Ochrona danych i szyfrowany system plików 374
- Odizolowane otoczenie aplikacji, bezpieczna komunikacja międzyprocesowa, usługi 375
- Podpisy aplikacji 376
- Uwierzytelnienie użytkownika 376
- Odizolowane komponenty sprzętowe i zarządzanie kluczami 377
- Zweryfikowany i bezpieczny rozruch 377

Analiza aplikacji dla systemu iOS 377
- Przygotowanie środowiska testowego 378
- Wyodrębnienie i ponowne podpisanie pakietu IPA 379
- Analiza statyczna 380
- Analiza dynamiczna 383
- Wstrzykiwanie danych 390
- Magazyn łańcucha kluczy 391
- Dekompilacja pliku binarnego 391
- Przechwytywanie i badanie danych sieciowych 393
- Omijanie wykrywania włamań poprzez wprowadzanie dynamicznych zmian w kodzie 394
- Omijanie wykrywania włamań poprzez wprowadzanie statycznych zmian w kodzie 395

Analiza aplikacji dla systemu Android 397
- Przygotowanie środowiska testowego 397
- Wyodrębnienie pliku APK 398
- Analiza statyczna 399
- Dekompilacja pliku binarnego 400
- Analiza dynamiczna 400
- Przechwytywanie i badanie danych sieciowych 405
- Boczne kanały wycieku danych 405

Omijanie wykrywania włamań poprzez wprowadzanie statycznych zmian w kodzie 406
- Omijanie wykrywania włamań poprzez wprowadzanie dynamicznych zmian w kodzie 408

Podsumowanie 408

15

HAKOWANIE INTELIGENTNEGO DOMU 409

Uzyskanie fizycznego dostępu do budynku 410
- Sklonowanie karty RFID do inteligentnego zamka 410
- Zakłócanie bezprzewodowego systemu alarmowego 413

Odtwarzanie strumienia wideo z kamery IP 417
- Protokoły strumieniowe 418
- Analiza danych przesyłanych przez kamerę IP 418
- Wyodrębnienie strumienia wideo 420

Hakowanie inteligentnej bieżni treningowej 423
- Inteligentna bieżnia i system Android 424
- Przejęcie kontroli nad inteligentną bieżnią 425

Podsumowanie 438

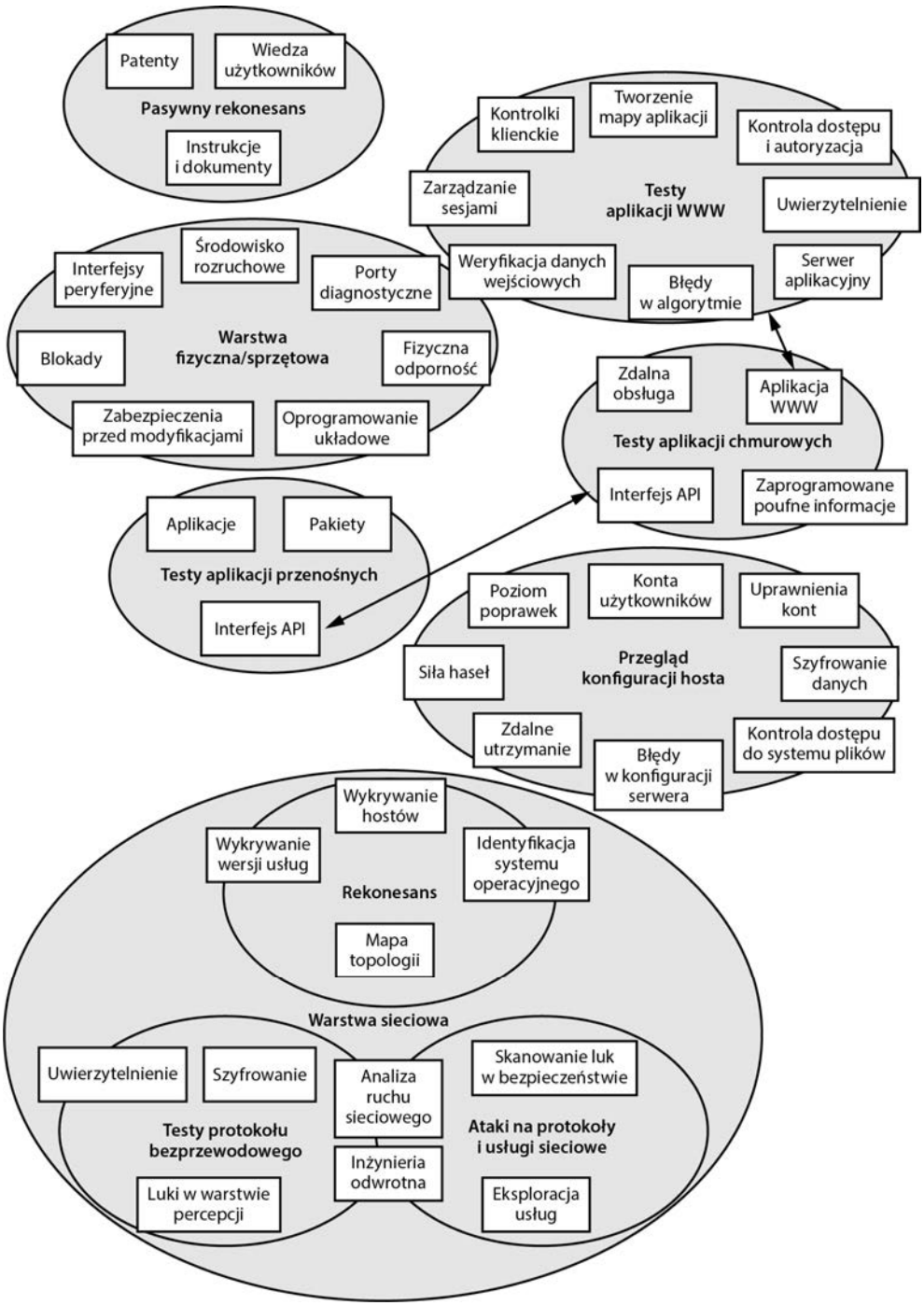
3

Metodyka testów bezpieczeństwa

Od czego zacząć, przystępując do testów podatności systemu IoT na ataki? Jeżeli obszar ataku jest niewielki, na przykład jest to strona WWW do sterowania kamerą monitorującą, zaplanowanie testu jest proste. Jednak nawet w takim przypadku, jeśli nie przyjmie się ustalonej metodyki, można przeoczyć krytyczne cechy aplikacji.

Ten rozdział zawiera listę kroków, które bezwzględnie należy wykonać podczas testów penetracyjnych. Obszar ataku IoT podzieliliśmy na warstwy koncepcyjne pokazane na rysunku 3.1.

Do testowania systemów IoT będziesz potrzebować solidnej metodyki oceny, takiej jak niżej opisana, ponieważ często systemy te składają się z wielu współpracujących ze sobą komponentów. Przeanalizujmy przykład rozrusznika serca podłączonego do domowego urządzenia monitorującego. Urządzenie za pośrednictwem sieci 4G wysyła dane pacjenta do portalu w chmurze, aby lekarz mógł kontrolować anomalie tętna. Lekarz może również konfigurować rozrusznik za pomocą programatora wykorzystującego technologię NFC (ang. *Near Field Communication*, komunikacja na krótkim zasięgu) i zastrzeżony protokół komunikacyjny. Cały system składa się z wielu części, z których każda stanowi szerokie pole do ataku. Stosując nieuporządkowaną metodykę oceny bezpieczeństwa, prawdopodobnie nie udałoby się tych zagrożeń rozpoznać. Aby ocena była rzetelna, przeprowadzimy pasywny rekonesans, a następnie opiszemy metody testowania warstw fizycznej i sieciowej, aplikacji internetowej i przenośnej, hosta, aplikacji oraz chmury.



Rysunek 3.1. Warstwy koncepcyjne testowane podczas oceny bezpieczeństwa

Pasywny rekonesans

Pasywny rekonesans, powszechnie nazywany *białym wywiadem* (ang. *open source intelligence*, OSINT), to proces gromadzenia informacji o celu ataku bez bezpośredniego komunikowania się z nim. To jeden z pierwszych kroków w każdej metodyce oceny i należy go zawsze wykonywać, aby uzyskać ogólny obraz systemu. Obejmuje on m.in. pobieranie i przeglądanie instrukcji obsługi, danych katalogowych chipsetów, przeglądanie forów internetowych i mediów społecznościowych, przeprowadzanie wywiadów z użytkownikami i personelem technicznym. Można także zbierać wewnętrzne nazwy hostów zapisane w certyfikatach TLS udostępnianych w ramach projektu Certificate Transparency, w którym urzędy podają do publicznej wiadomości wydawane certyfikaty.

Instrukcje i dokumenty

Instrukcje systemowe są skarbnicą wiedzy o wewnętrznych szczegółach funkcjonowania urządzeń. Zazwyczaj są one dostępne na oficjalnych stronach internetowych producentów. Gdybyś nie mógł ich tam znaleźć, zastosuj zaawansowane wyszukiwanie dokumentów PDF w serwisie Google, na przykład po nazwie urządzenia wpisz frazę `inurl:pdf`.

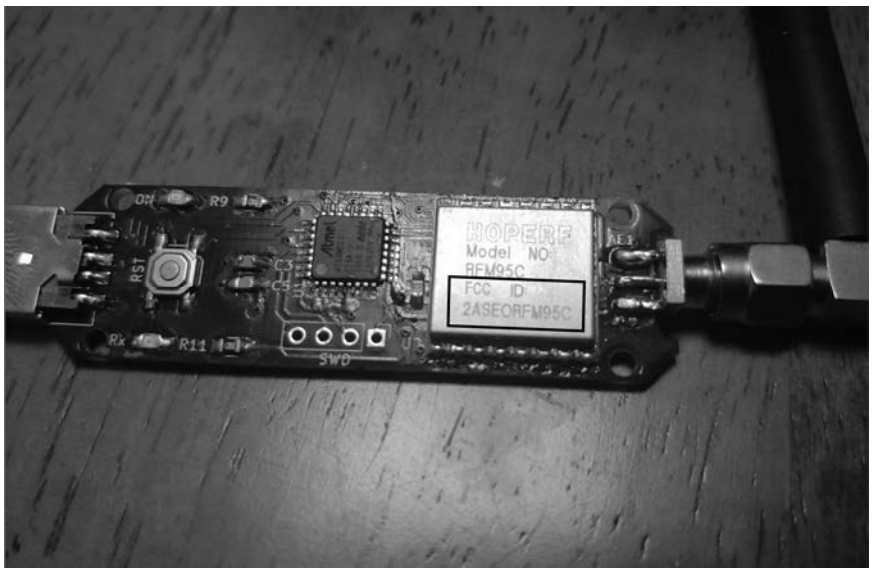
Zaskakujące jest, jak wiele ważnych informacji można znaleźć w instrukcjach obsługi. Z naszego doświadczenia wynika, że zawierają one domyślne nazwy użytkowników i hasła, które często są stosowane niezmienione w środowiskach produkcyjnych. Ponadto w instrukcji można znaleźć szczegółową specyfikację systemu i jego komponentów, diagramy sieci i architektury oraz wskazówki diagnostyczne umożliwiające identyfikację słabych punktów.

Jeśli uda Ci się zidentyfikować wykorzystane chipsety, poszukaj ich kart katalogowych (instrukcji obsługi komponentów elektronicznych), ponieważ mogą one zawierać opisy pinów diagnostycznych (na przykład interfejsy JTAG opisane w rozdziale 7.).

Innym przydatnym źródłem informacji o urządzeniach wykorzystujących komunikację radiową jest internetowa baza identyfikatorów FCC, dostępna pod adresem *https://fccid.io*. Jest to wykaz unikatowych identyfikatorów urządzeń zarejestrowanych przez amerykańską Federalną Komisję Łączności. Każde urządzenie emitujące sygnał radiowy musi posiadać taki identyfikator. Na jego podstawie można uzyskać szczegółowe informacje o częstotliwości i sile sygnału radiowego, fotografie wnętrza urządzenia, instrukcje obsługi i inne materiały. Identyfikator FCC jest zazwyczaj wygrawerowany na obudowie elementu elektronicznego lub urządzenia (patrz rysunek 3.2).

Patenty

Patenty zawierają informacje o wewnętrznych szczegółach funkcjonowania niektórych urządzeń. Poszukaj na stronie *https://patents.google.com* nazwy wybranego dostawcy i sprawdź, jakie informacje o nim są dostępne. Na przykład po wpisaniu frazy `medtronic bluetooth` wyszukasz patent z 2004 r. na protokół komunikacyjny wykorzystywany w implantowanych urządzeniach medycznych.

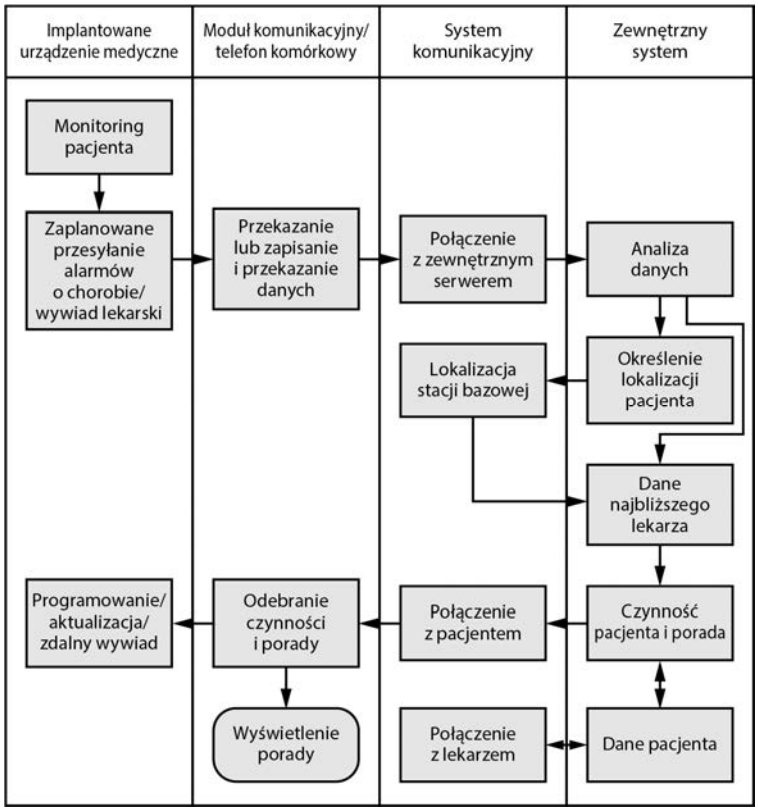


Rysunek 3.2. Identyfikator FCC układu RFM95C zastosowanego w urządzeniu USB CatWAN, które wykorzystamy w rozdziale 13. do hakowania sieci LoRa

Patenty prawie zawsze zawierają schematy blokowe, na podstawie których można ocenić kanał komunikacyjny między danym urządzeniem a innymi systemami. Rysunek 3.3 przedstawia prosty diagram przepływu informacji w systemie zawierającym implantowane urządzenie medyczne. Potwierdza on podatność urządzenia na ataki. Zwróć uwagę na strzałki wchodzące i wychodzące z kolumny *Implantowane urządzenie medyczne*. Moduł *Odebranie czynności i porady* zewnętrznego systemu może inicjować połączenia z urządzeniem. Podążając za ciągiem strzałek, można się przekonać, że taką czynnością może być zmiana ustawień urządzenia, która może zaszkodzić pacjentowi. Zatem zewnętrzny system stwarza ryzyko włamania do urządzenia za pośrednictwem niezabezpieczonej aplikacji przenośnej lub rzeczywistego systemu zdalnego (zwykle zaimplementowanego w chmurze).

Wiedza użytkowników

Niewiarygodne, ile informacji można znaleźć w mediach społecznościowych, na forach internetowych i czatach. Źródłem wiedzy mogą by nawet opinie użytkowników publikowane w serwisach Amazon i eBay. Poszukaj narzekań klientów na niektóre funkcje urządzenia. Błędne działanie może czasami sygnalizować poważną lukę w bezpieczeństwie. Załóżmy, że jakiś użytkownik skarży się, że w pewnych warunkach urządzenie ulega awarii. Warto zbadać ten trop, ponieważ może to doprowadzić do błędu w kodzie lub luki w zabezpieczeniu pamięci ujawniającej się po wprowadzeniu określonych danych wejściowych. Ponadto wielu użytkowników publikuje szczegółowe opisy, specyfikacje i zdjęcia rozmontowanych urządzeń.



Rysunek 3.3. Schemat blokowy w patencie firmy Medtronic pokazujący, że urządzenie może się komunikować z zewnętrznym systemem w obu kierunkach za pośrednictwem telefonu komórkowego. Oznacza to dużą podatność na atak

Oprócz tego warto przeglądać profile i wpisy w serwisach LinkedIn i Twitter. Inżynierowie i specjaliści IT pracujący dla producenta określonego systemu IoT mogą ujawniać cenne ciekawostki techniczne. Jeżeli na przykład ktoś pisze, że ma szeroką wiedzę o architekturze pewnego procesora, jest bardzo prawdopodobne, że wiele urządzeń danego producenta wykorzystuje ten właśnie procesor. Jeśli inny pracownik narzeka na konkretną platformę (lub ją chwali, choć zdarza się to rzadziej), z dużym prawdopodobieństwem oznacza to, że firma używa jej do tworzenia oprogramowania.

Ogólnie rzecz biorąc, każda branża IoT ma swoich ekspertów, z którymi można się konsultować w celu uzyskania przydatnych informacji. Na przykład podczas oceniania bezpieczeństwa elektrowni rozmowa z operatorami lub technikami o ich pracy może dostarczyć przydatnych informacji o potencjalnych możliwościach przeprowadzenia ataku. W świecie medycznym administratorami i głównymi operatorami systemów IoT są zazwyczaj pielęgniarki, które posiadają głęboką wiedzę na temat tajników urządzeń. Dlatego, jeśli to możliwe, należy się z nimi skonsultować.

Warstwa fizyczna lub sprzętowa

Jednym z podstawowych celów ataków w urządzeniu IoT jest sprzęt. Jeśli haker zdobędzie komponenty sprzętowe systemu, prawie na pewno uda mu się uzyskać rozszerzone uprawnienia, ponieważ system niemal zawsze „ufa" każdemu użytkownikowi, który ma do niego fizyczny dostęp. Mówiąc wprost, jeśli haker ma fizyczny dostęp do Twoich systemów, możesz się poddać. Wyobraźmy sobie, że najbardziej zmotywowani cyberprzestępcy, na przykład finansowani przez obce rządy i dysponujący praktycznie nieograniczonym czasem i zasobami, mają do dyspozycji fizyczną kopię urządzenia. Na internetowych platformach handlu elektronicznego mogą nabyć nawet specjalistyczny sprzęt (na przykład duże ultrasonografy), beztrosko wyprzedawany przez firmy, a nawet pochodzący z kradzieży. Hakerzy nie potrzebują nawet konkretnej wersji urządzenia. Często luki w bezpieczeństwie dotyczą całej generacji systemów.

Ocena warstwy sprzętowej powinna obejmować testowanie interfejsów peryferyjnych, środowiska rozruchowego, fizycznych blokad, zabezpieczeń przed modyfikacjami, oprogramowania układowego, portów diagnostycznych i fizycznej odporności.

Interfejsy peryferyjne

Interfejsy peryferyjne to fizyczne porty komunikacyjne, służące do podłączania zewnętrznych urządzeń, na przykład klawiatury, twardych dysków czy kart sieciowych. Sprawdzaj, czy są aktywne porty USB lub gniazda kart PC i czy można za ich pomocą uruchomić system operacyjny. Wielokrotnie udawało nam się uzyskać administracyjny dostęp do różnych systemów x86 i omijać techniczne zabezpieczenia, uruchamiając własne systemy operacyjne, montując niezaszyfrowane systemy plików, wyodrębniając kody i hasła oraz instalując własne oprogramowanie. Można również wyjmować twarde dyski i odczytywać lub zapisywać na nich dane, nie mając dostępu do rozruchowych portów USB. Jednak ta technika jest mniej wygodna. Należy pamiętać, że podczas manipulowania przy sprzęcie w celu wyjęcia dysków można uszkodzić inne komponenty.

Porty USB mogą być celami ataków z jeszcze innego powodu. Niektóre urządzenia, szczególnie wykorzystujące system Windows 10, mogą pracować w *trybie kiosku* z ograniczonym interfejsem użytkownika. Rozważmy przykład bankomatu, z którego wypłacamy gotówkę. Mimo że wewnątrz może on wykorzystywać wbudowany system operacyjnym Windows XP, użytkownik ma do dyspozycji wyłącznie ograniczony interfejs graficzny, oferujący określony zestaw opcji. Wyobraźmy sobie, co można byłoby zrobić, gdyby udało się podłączyć klawiaturę USB do odsłoniętego portu. Za pomocą określonych kombinacji klawiszy, na przykład *Ctrl+Alt+Delete* lub klawisza *Windows*, można byłoby wyjść z trybu kiosku i uzyskać bezpośredni dostęp do całego systemu.

Środowisko rozruchowe

W urządzeniach opartych na procesorach x86 i x64 sprawdzaj, czy wykorzystywany w nich konwencjonalny system BIOS oraz program rozruchowy są chronione hasłami oraz jaka jest skonfigurowana kolejność nośników rozruchowych. Jeżeli w pierwszej kolejności jest wykorzystywany wymienny nośnik, haker będzie w stanie uruchomić własny system operacyjny bez modyfikowania ustawień BIOS. Sprawdzaj również, czy jest włączone i jaki ma priorytet środowisko PXE (ang. *Preboot Execution Environment*, przedrozruchowe środowisko wykonawcze) umożliwiające uruchomienie systemu uzyskanego za pomocą protokołów DHCP (ang. *Dynamic Host Configuration Protocol*, protokół dynamicznego konfigurowania hostów) i TFTP (ang. *Trivial File Transfer Protocol*, trywialny protokół przesyłania plików). Haker może je wykorzystać, uruchamiając w sieci nielegalny serwer rozruchowy. Jednak nawet wtedy, gdy sekwencja rozruchowa jest poprawnie skonfigurowana i wszystkie ustawienia są chronione hasłami, istnieje możliwość zresetowania systemu BIOS (na przykład poprzez usunięcie baterii) i przywrócenia domyślnych, niezabezpieczonych ustawień. Jeżeli wykorzystywany jest interfejs UEFI (ang. *Unified Extensible Firmware Interface*, ujednolicony, rozszerzalny interfejs oprogramowania układowego) i mechanizm bezpiecznego rozruchu (*Secure Boot*), kontroluj jego implementację. Mechanizm ten sprawdza, czy oprogramowanie rozruchowe nie zostało zmodyfikowane (na przykład przez wirusa). Weryfikuje również podpisy sterowników sprzętowych i systemu operacyjnego.

Możesz również mieć do czynienia z technologią Trusted Execution Environment (TEE), na przykład TrustZone w platformach Arm, lub z funkcją bezpiecznego rozruchu Qualcomm Technologies, które weryfikują bezpieczne obrazy rozruchowe.

Blokady

Sprawdzaj, czy urządzenie ma blokadę, i jeżeli tak, czy można ją łatwo ominąć. Weryfikuj, czy wszystkie blokady można otworzyć za pomocą jednego klucza oraz czy każda ma własny klucz. W naszej pracy mieliśmy do czynienia z urządzeniami tego samego producenta, których blokady były całkowicie bezużyteczne, ponieważ wykorzystywały ten sam, łatwy do uzyskania przez każdego klucz. Odkryliśmy, że można było w ten sposób otworzyć obudowy całej linii pomp infuzyjnych i uzyskać dostęp do konfiguracji systemu.

Aby ocenić jakość blokady, oprócz wiedzy o jej typie potrzebny jest zestaw narzędzi ślusarskich. Na przykład zamek zapadkowy otwiera się inaczej niż elektryczny, którego nie można otworzyć ani zamknąć przy braku zasilania.

Zabezpieczenia przed modyfikacjami i wykrywanie modyfikacji

Sprawdzaj, czy urządzenie jest odporne na modyfikacje lub czy może dostarczać dowodów modyfikacji. Jednym ze sposobów jest naklejanie perforowanych etykiet, które po oderwaniu pozostawiają trwałe napisy. Innego rodzaju zabezpieczenia to wlewki, zapięcia, specjalne obudowy zapieczętowane żywicą epoksydową i fizyczne bezpieczniki niszczące tajną zawartość w przypadku demontażu urządzenia.

Mechanizm wykrywający modyfikacje wysyła alarm lub tworzy plik dziennika w razie wykrycia próby naruszenia integralności urządzenia. Szczególnie ważne jest sprawdzanie zabezpieczeń przed modyfikacjami i wykrywanie modyfikacji podczas przeprowadzania testu penetracyjnego systemów IoT w przedsiębiorstwie. Wiele zagrożeń pochodzi od wewnątrz, tj. pracowników (obecnych i byłych) i wykonawców. Zatem dzięki tego rodzaju zabezpieczeniom można wykrywać przypadki celowej modyfikacji urządzeń. Haker miałby problem z demontażem urządzenia odpornego na modyfikacje.

Oprogramowanie układowe

Bezpieczeństwo oprogramowania układowego opiszemy szczegółowo w rozdziale 9., więc nie będziemy się nim tutaj zajmować. Pamiętaj, że uzyskanie nieupoważnionego dostępu do oprogramowania jest nielegalne. Jest to ważne, jeśli zamierzasz opublikować wyniki badań bezpieczeństwa obejmujące dostęp do oprogramowania układowego lub inżynierię wsteczną znajdujących się w nim plików wykonywalnych. Związane z tym kwestie prawne zostały opisane w rozdziale 1., w punkcie „Regulacje dotyczące hakowania IoT".

Interfejsy diagnostyczne

Sprawdzaj, czy urządzenie posiada *interfejsy diagnostyczne, usługowe lub testowe*, których producent mógł używać w celu uproszczenia programowania, produkcji i diagnostyki. Wiele urządzeń wbudowanych posiada tego rodzaju interfejsy. Za ich pomocą można uzyskać natychmiastowy dostęp do konta administratora. W wielu przypadkach, aby zrozumieć funkcjonowanie testowanego urządzenia, otwieraliśmy jego powłokę administracyjną, wykorzystując porty diagnostyczne, ponieważ nie było innej możliwości uzyskania dostępu i zbadania systemu. Aby móc wykonywać tego rodzaju operacje, trzeba zadać sobie trud dokładniejszego poznania szczegółów funkcjonowania protokołów komunikacyjnych wykorzystywanych w interfejsach diagnostycznych, ale uzyskiwane wyniki są tego warte. Najczęściej stosowane interfejsy diagnostyczne to UART, JTAG, SPI oraz I^2C. Opiszemy je w rozdziałach 7. i 8.

Fizyczna odporność

Testuj sprzęt pod kątem wszelkich ograniczeń wynikających z jego fizycznych właściwości, na przykład odporności na *próby rozładowania baterii*, i w efekcie jego skutecznego unieruchomienia, co może się zdarzyć, jeżeli haker przeciąży urządzenie. Pomyśl, jak zagrożone jest życie pacjenta uzależnionego od zaimplantowanego rozrusznika serca. Innym rodzajem testu jest *atak zakłóceniowy*, polegający na umyślnym powodowaniu awarii urządzenia w celu zakwestionowania jego bezpieczeństwa podczas wykonywania krytycznych operacji. Jednym z naszych najbardziej spektakularnych i zaskakujących sukcesów był atak zakłóceniowy na płytę drukowaną urządzenia, w wyniku którego to ataku proces rozruchowy opuścił powłokę administratora. Ponadto spróbuj przeprowadzać inne ataki, na przykład *różnicową analizę mocy*, która ma na celu zmierzenie poboru

energii urządzenia podczas wykonywania operacji kryptograficznych i uzyskanie poufnych informacji.

Znając fizyczne cechy urządzenia, można się domyślić skuteczności innych funkcji bezpieczeństwa. Na przykład małe urządzenie o pojemnej baterii może stosować słaby algorytm szyfrowania komunikacji sieciowej. Moc obliczeniowa wymagana do silniejszego szyfrowania szybciej wyczerpuje baterię, która ma ograniczoną pojemność ze względu na rozmiar urządzenia.

Warstwa sieciowa

Warstwa sieciowa, obejmująca wszystkie komponenty, które bezpośrednio lub pośrednio komunikują się za pomocą standardowych środków, jest zwykle najczęstszym celem ataków. Dlatego ocenę podzielimy na mniejsze części: rekonesans, ataki na protokoły i usługi sieciowe oraz testy protokołów bezprzewodowych.

W tym rozdziale opisujemy wiele testów wykorzystujących sieć, ale niektóre wyróżniliśmy, poświęcając im osobne podrozdziały. Na przykład ocena aplikacji WWW ma własną sekcję ze względu na jej złożoność i liczbę wykonywanych operacji.

Rekonesans

Omówiliśmy już ogólnie kroki pasywnego rekonesansu urządzeń IoT. W tym punkcie skupiamy się na pasywnym i aktywnym rekonesansie sieci. Są to pierwsze kroki przygotowujące do każdego ataku sieciowego. Rekonesans pasywny polega na podsłuchiwaniu ważnych danych przesyłanych przez sieć, natomiast **rekonesans aktywny** wymaga bezpośredniego komunikowania się z urządzeniami.

W przypadku pojedynczego urządzenia proces jest dość prosty, ponieważ wystarczy sprawdzić tylko jeden adres IP. Jednak w dużym ekosystemie, takim jak inteligentny dom lub system opieki zdrowotnej złożony z urządzeń medycznych, rozpoznanie sieci jest bardziej skomplikowanym zadaniem. Poniżej opisujemy wykrywanie hostów, określanie wersji usług, identyfikowanie systemów operacyjnych i tworzenie mapy sieci.

Wykrywanie hostów

Wykrywanie hostów to proces identyfikowania działających w sieci systemów poprzez ich sondowanie przy użyciu różnych technik, takich jak wysyłanie pakietów *echo-request* (żądanie echa) za pomocą protokołu ICMP (ang. *Internet Control Message Protocol*, internetowy protokół przesyłania komunikatów kontrolnych), skanowanie popularnych portów TCP/UDP, analiza ruchu rozgłoszeniowego oraz wysyłanie zapytań ARP (ang. *Address Resolution Protocol*, protokół odwzorowywania adresów) w segmencie L2. Symbol L2 oznacza drugą warstwę modelu OSI (ang. *Open Systems Interconnection*, łączenie systemów otwartych). Jest to warstwa łącza danych odpowiedzialna za ich przesyłanie między węzłami

znajdującymi się w tym samym segmencie sieci. Opiera się na warstwie fizycznej wykorzystującej zazwyczaj popularny standard komunikacyjny Ethernet.

W przypadku skomplikowanych systemów IoT, na przykład serwerów obsługujących wiele kamer monitoringu zainstalowanych w różnych segmentach sieci, nie można poprzestawać na stosowaniu jednej techniki. Należy je dywersyfikować, aby zwiększyć prawdopodobieństwo ominięcia zapór sieciowych i restrykcyjnych konfiguracji sieci VLAN (ang. *Virtual Local Area Network*, wirtualna sieć lokalna).

Ten krok jest bardzo przydatny podczas przeprowadzania testów penetracyjnych systemów IoT, gdy nie są znane ich adresy IP.

Określanie wersji usług

Po zidentyfikowaniu hostów określ wszystkie uruchomione na nich usługi sieciowe. Zacznij od skanowania portów TCP/UDP. Następnie *odczytaj ich banery* (informacje wysyłane w odpowiedzi na żądanie nawiązania połączenia) i odciski kluczy szyfrujących. W tym celu użyj na przykład narzędzia Amap lub Nmap z opcją `-sV`. Pamiętaj, że niektóre usługi, szczególnie uruchomione na urządzeniach medycznych, są wrażliwe nawet na proste próbkowanie i łatwo ulegają awarii. Mieliśmy do czynienia z systemami IoT, które ulegały awarii i ponownie się uruchamiały tylko dlatego, że skanowaliśmy je za pomocą narzędzia Nmap, aby określić wersję usługi. Skanowanie polega na wysyłaniu specjalnie przygotowanych pakietów w celu uzyskania odpowiedzi od usług, które w normalnych warunkach nie wysyłają żadnych informacji po połączeniu się z nimi. Tego rodzaju pakiety zakłócają stabilność niektórych wrażliwych urządzeń, ponieważ ich usługi nie oczyszczają skutecznie danych wejściowych, co prowadzi do zniekształcenia zawartości pamięci i awarii.

Identyfikowanie systemów operacyjnych

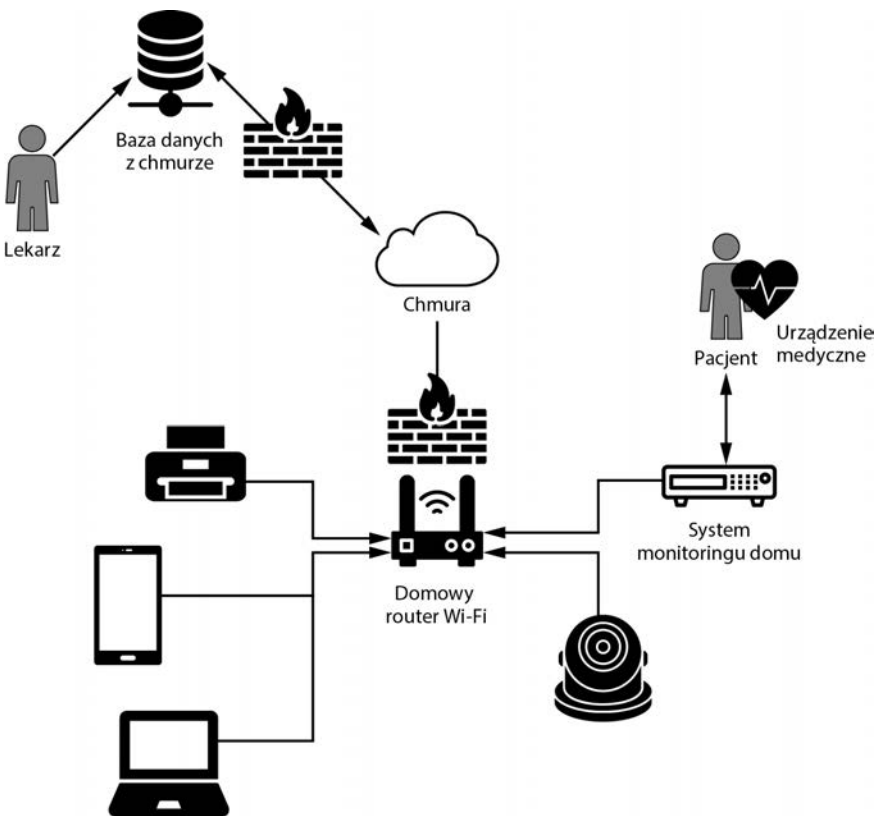
Aby móc eksplorować systemy operacyjne uruchomione na testowanych hostach, musisz je najpierw dokładnie zidentyfikować. Dotyczy to przede wszystkim architektury procesora (x86, x64 lub ARM). Bardzo przydaje się określenie pakietu serwisowego (w systemie Windows) i wersji jądra (w systemach Linux i Unix).

Systemy operacyjne można identyfikować na podstawie odpowiedzi hostów na wysłane do nich odpowiednio przygotowane pakiety TCP, UDP i ICMP. Proces ten nosi nazwę „pobierania odcisków palców". Odpowiedzi są różne, ponieważ systemy operacyjne różnią się szczegółami implementacji stosu sieciowego TCP/IP. Na przykład starsze wersje systemu Windows wysyłają odpowiedzi zawierające flagę FIN, inne — RST, a jeszcze inne nie wysyłają odpowiedzi wcale. Przeprowadzając analizę statystyczną odpowiedzi różnych wersji systemów operacyjnych, można utworzyć ich profile i wykorzystać je do identyfikowania systemów w praktyce. Więcej informacji na ten temat jest dostępnych w dokumentacji narzędzia Nmap na stronie *TCIP/IP Fingerprinting Methods Supported by Nmap* („Metody próbkowania TCP/IP dostępne w narzędziu Nmap").

W identyfikowaniu systemów operacyjnych może się przydać skanowanie usług, ponieważ wiele z nich ujawnia tego rodzaju informacje na swoich banerach. Doskonale do takich zastosowań nadaje się narzędzie Nmap. Pamiętaj jednak, że niektóre urządzenia IoT mogą być wrażliwe na tego rodzaju operacje i ulegać awariom.

Tworzenie mapy sieci

Tworzenie mapy (topologii) sieci polega na określaniu połączeń pomiędzy systemami. Ten krok wykonuje się wtedy, gdy trzeba testować cały ekosystem urządzeń znajdujących się w różnych segmentach L3 sieci i komunikujących się między sobą za pośrednictwem routerów i zapór. Symbol L3 oznacza trzecią warstwę modelu OSI. Jest to warstwa sieciowa odpowiedzialna za transmisję i kierowanie pakietów. W tej warstwie działają routery. Mapa sieci testowanych systemów przydaje się do modelowania zagrożeń. Pokazuje, jak wykorzystując luki bezpieczeństwa w łańcuchu hostów, można zaatakować krytyczny zasób. Rysunek 3.4 przedstawia przykładową ogólną mapę sieci.



Rysunek 3.4. Prosta mapa domowej sieci obsługującej implantowane urządzenie monitorujące pacjenta

Przedstawiona przykładowa mapa pokazuje pacjenta używającego urządzenia medycznego komunikującego się z urządzeniem monitorującym dom. Urządzenie to z kolei przesyła przez sieć Wi-Fi dane diagnostyczne do chmury, gdzie lekarz może je okresowo przeglądać w celu wykrycia anomalii.

Ataki na protokoły i usługi sieciowe

Ataki na protokoły i usługi sieciowe dzielą się na następujące etapy: skanowanie luk w bezpieczeństwie, analiza ruchu sieciowego, odwrotna inżynieria protokołów, eksploracja protokołów i usług. Podatności można skanować niezależnie od innych etapów, natomiast pozostałe etapy są ze sobą powiązane.

Skanowanie luk w bezpieczeństwie

Aby poznać zidentyfikowane luki w bezpieczeństwie usług sieciowych, przejrzyj bazę NVD (ang. *National Vulnerability Database*, krajowa baza podatności) lub VulnDB. Niektóre systemy są tak stare, że automatyczne narzędzie skanujące tworzy wielostronicowe raporty. Część luk można wykorzystać zdalnie, bez uprzedniego uwierzytelnienia się. Dobrą praktyką jest stosowanie przynajmniej jednego narzędzia skanującego, wykrywającego najpopularniejsze zagrożenia. Jeżeli znajdziesz poważną lukę, na przykład umożliwiającą uruchamianie zewnętrznego kodu, możesz ją wykorzystać do otwarcia powłoki, co ułatwi Ci dalszą ocenę bezpieczeństwa. Upewnij się, że skanowanie prowadzisz w kontrolowanym środowisku, i uważnie je monitoruj w razie wystąpienia nieprzewidzianych przestojów.

Analiza ruchu sieciowego

Na początku procesu oceny bezpieczeństwa uruchom na dłuższy czas narzędzie rejestrujące ruch sieciowy, na przykład Wireshark lub tcpdump, aby dowiedzieć się, jakie są stosowane protokoły komunikacyjne. Jeśli system IoT składa się z komunikujących się ze sobą komponentów (na przykład kamery monitorującej i serwera albo z pompy infuzyjnej i bazy elektronicznych kart pacjentów), powinieneś zarejestrować przesyłane między nimi pakiety. Typowe ataki, na przykład zatruwanie bufora ARP, są zazwyczaj przeprowadzane w jednym segmencie L3 sieci.

Opisane wyżej narzędzia najlepiej w miarę możliwości uruchamiać bezpośrednio na urządzeniach, aby rejestrować komunikację między lokalnymi procesami (ang. *Inter Process Communication*, IPC). W przypadku urządzeń wbudowanych może to być trudne zadanie, bo zazwyczaj nie dają one takiej możliwości. Nam jednak często udawało się skompilować i zainstalować na przykład narzędzie tcpdump nawet na bardzo restrykcyjnych urządzeniach, na przykład domowych systemach monitoringu rozrusznika serca. Ten przypadek opisujemy w rozdziale 6.

Po zarejestrowaniu reprezentatywnej próbki ruchu sieciowego możesz przystąpić do jego analizy. Sprawdź, czy istnieją niezabezpieczone kanały komunikacyjne (na przykład protokoły przesyłające niezaszyfrowany tekst), protokoły podatne na ataki (takie jak UPnP [ang. *Universal Plug and Play*, uniwersalny system „podłącz i używaj"]) i zastrzeżone protokoły wymagające dokładniejszego zbadania i zastosowania inżynierii odwrotnej (opisanej w następnym punkcie).

Odwrotna inżynieria protokołów

Odwrotnej inżynierii powinieneś poddawać każdy protokół, jaki odkryjesz. Zastrzeżony protokół jest zawsze bronią obosieczną. Niektóre systemy rzeczywiście go wymagają ze względu na wydajność, funkcje, a nawet bezpieczeństwo. Jednak opracowanie funkcjonalnego protokołu jest bardzo skomplikowanym zadaniem. Wiele systemów, którymi się zajmowaliśmy, wykorzystywało protokoły TCP, UDP oraz protokoły wyższych warstw do przesyłania danych XML, JSON lub o innych strukturach. W skomplikowanych systemach, na przykład rozrusznikach serca, mieliśmy do czynienia z zastrzeżonymi protokołami bezprzewodowymi, o których mało było publicznie dostępnych informacji. W takich sytuacjach łatwiej jest badać protokoły pod innym kątem.

Można na przykład diagnozować usługi komunikujące się ze sterownikami obsługującymi wysyłanie sygnałów radiowych. Nie trzeba analizować zastrzeżonego protokołu bezprzewodowego, tylko sprawdzać, jak jest wykorzystywany w wyższej warstwie. Tę technikę zastosowaliśmy kiedyś, aby uzyskać dostęp do rozrusznika. Użyliśmy narzędzia strace, które podłączyliśmy pod proces komunikujący się ze sterownikiem. Przeglądając dzienniki i pliki PCAP, zidentyfikowaliśmy wykorzystywany kanał komunikacyjny bez analizowania sygnału radiowego czy stosowania czasochłonnych metod, na przykład *transformacji Fouriera* (wykorzystuje się ją w celu rozłożenia sygnału na składowe częstotliwości).

Eksploracja protokołów i usług

Ostatnim etapem ataku sieciowego jest eksploracja protokołów i usług za pomocą specjalnie napisanego programu. Przede wszystkim musisz dokładnie określić warunki eksploracji. Czy jest zawsze powtarzalna? Czy system musi znajdować się w określonym stanie? Czy zapora sieciowa zezwala na wysyłanie lub odbieranie danych? Czy po udanej eksploracji system nadaje się do użytku? Upewnij się, że możesz na te pytania udzielić jednoznacznych odpowiedzi.

Testy protokołów bezprzewodowych

Testowaniu protokołów bezprzewodowych poświęcamy osobny punkt, ponieważ radiowa łączność krótkiego, średniego i dalekiego zasięgu jest w systemach IoT powszechnie stosowana. Ten obszar częściowo pokrywa się z wymienianą w innych źródłach **warstwą percepcyjną**, obejmującą technologie RFID (ang. *Radio Frequency Identification*, identyfikacja za pomocą sygnału radiowego), GPS (ang. *Global Positioning System*, globalny system pozycjonowania) i NFC (ang. *Near Field Communication*, komunikacja na krótkim zasięgu).

Analiza powyższych technologii jest podobna do procesów opisanych wcześniej w punktach „Analiza ruchu sieciowego" i „Odwrotna inżynieria protokołów", dotyczących warstwy sieciowej. Atakowanie i analizowanie protokołów bezprzewodowych wymaga użycia specjalistycznego sprzętu, m.in. chipsetów wstrzykujących dane (na przykład Atheros), modułów Bluetooth (na przykład Ubertooth) oraz narzędzi radiowych definiowanych programowo (na przykład HackRF lub LimeSDR).

Na tym etapie przeprowadza się określone ataki na stosowane protokoły bezprzewodowe. Jeżeli na przykład dany komponent IoT używa sieci Wi-Fi, przeprowadzaj atak asocjacyjny i sprawdzaj, czy jest stosowane szyfrowanie WEP (ang. *Wired Equivalent Privacy*, prywatność równoważna sieciom przewodowym), które łatwo złamać, lub WPA/WPA2 (ang. *Wireless Protected Access*, zabezpieczony dostęp do sieci bezprzewodowej), wykorzystujące słabe poświadczenia. Wkrótce do tej kategorii zapewne dołączy standard WPA3. Najczęściej przeprowadzane ataki na powyższe trzy protokoły będą opisane w rozdziałach 10. – 13. Jeżeli są stosowane niestandardowe protokoły, sprawdzaj, czy jest wykorzystywane uwierzytelnienie stron (również wzajemne) oraz szyfrowanie i kontrola integralności danych. Z naszego doświadczenia wynika, że często brakuje przynajmniej jednego z powyższych zabezpieczeń, nawet w krytycznej infrastrukturze.

Testy aplikacji WWW

Aplikacje WWW, również stosowane w systemach IoT, należą do najprostszych do wykorzystania punktów wejścia do sieci, ponieważ często są dostępne z zewnątrz i mają wiele luk w zabezpieczeniach. Ocenianie bezpieczeństwa aplikacji internetowych to obszerny temat, któremu poświęconych jest wiele publikacji. Skoncentrujemy się na technikach badania aplikacji wykorzystywanych w urządzeniach IoT. Faktem jest, że nie różnią się one zbytnio od typowych aplikacji, ale w ich procesach rozwoju nie są implementowane mechanizmy ochrony oprogramowania, co prowadzi do powstania licznych, typowych luk w bezpieczeństwie. Testowaniu aplikacji WWW jest poświęcona książka *The Web Application Hacker's Handbook* oraz projekty białego wywiadu, m.in. *OWASP Top 10*, *Application Security Verification Standard* (ASVS) i *Testing Guide*.

Tworzenie mapy aplikacji

Tworzenie mapy aplikacji rozpoczynaj od zbadania widocznych, ukrytych i domyślnych treści strony. Identyfikuj punkty wprowadzania danych, ukryte pola i ich parametry. Ten proces możesz przyspieszyć, stosując zautomatyzowane *narzędzia przeszukujące* kolejne strony, ale zawsze powinieneś analizować je również ręcznie. Analizę można przeprowadzać *pasywnie* z wykorzystaniem serwera pośredniczącego (*proxy*), monitorującego treść przeglądaną przez użytkowników, jak również *aktywnie*, wysyłając zapytania na wykryte wcześniej adresy URL i zapytania AJAX zakodowane w skryptach JavaScript.

Stosując typowe nazwy plików i katalogów, można identyfikować *ukryte treści* i adresy nieosiągalne za pośrednictwem dostępnych odnośników. Pamiętaj, że ta metoda jest bardzo „hałaśliwa", ponieważ tego rodzaju zapytania generują bardzo duży ruch w sieci. Na przykład średniej wielkości lista typowych nazw plików i katalogów stosowana w narzędziu DirBuster zawiera 220 560 pozycji. Oznacza to, że w celu znalezienia potencjalnych ukrytych adresów URL narzędzie to wysyła do wybranej witryny 220 560 zapytań HTTP. Nie należy jednak pomijać tego kroku, szczególnie jeżeli ocenę przeprowadza się w kontrolowanym środowisku. W analizowanych przez nas aplikacjach WWW na urządzeniach IoT często znajdowaliśmy bardzo ciekawe, zazwyczaj niezabezpieczone adresy URL. Na przykład w popularnej kamerze monitorującej znaleźliśmy ukryty adres umożliwiający pobieranie obrazów bez uwierzytelnienia. Przy jego użyciu haker mógł z zewnątrz widzieć wszystko, co monitorowała kamera!

Ważne jest również identyfikowanie punktów umożliwiających użytkownikom wprowadzanie danych. Źródłem większości zagrożeń jest możliwość wprowadzania niezaufanych danych przez nieuwierzytelnionych użytkowników z zewnątrz. Tego rodzaju punkty wykorzystuje się do przeprowadzania testów zakłóceniowych (zautomatyzowanego wprowadzania losowych, niepoprawnych danych) oraz do wstrzykiwania zapytań.

Kontrolki klienckie

Eksploracji można poddać *kontrolki klienckie*, czyli wszystko, co obsługuje przeglądarka, aplikacja stacjonarna lub przenośna. Są to m.in. ukryte pola, ciasteczka, aplety Java, skrypty JavaScript oraz obiekty AJAX, ASP.NET ViewState, ActiveX, Flash i Silverlight. W wielu badanych przez nas aplikacjach użytkownicy byli uwierzytelniani po stronie klienckiej. Haker mógłby ten mechanizm łatwo ominąć, ponieważ byłby w stanie kontrolować wszystko, co się dzieje po stronie klienta. Wykorzystywane były skrypty JavaScript i pliki *.jar*, *.swf* i *.xap*, które haker mógłby zdekompilować i odpowiednio zmodyfikować, aby móc wykonywać swoje operacje.

Uwierzytelnianie użytkowników

Szukaj luk w bezpieczeństwie mechanizmu uwierzytelniania użytkowników aplikacji. Powszechnie wiadomo, że ogromna liczba systemów IoT zawiera wstępnie skonfigurowane, słabe poświadczenia, które zazwyczaj nie są zmieniane. Można je znaleźć w instrukcjach obsługi i materiałach w internecie, jak również po prostu odgadnąć. Często podczas testowania systemów IoT spotykaliśmy poświadczenia typu *admin-admin*, *a-a* (tak, tak, zarówno nazwa użytkownika, jak i hasło to „a") lub całkowity ich brak. Aby złamać niestandardowe hasło, należy przeprowadzić *atak słownikowy* na wszystkie adresy uwierzytelniające. Wykorzystuje się w tym celu narzędzie, które automatycznie stosuje najpopularniejsze słowa lub typowe hasła. Niemal wszystkie raporty oceny bezpieczeństwa, jakie napisaliśmy, zawierały wniosek „brak ochrony przed atakami brutalnej siły". Urządzenia IoT często mają ograniczone zasoby sprzętowe i nie są w stanie działać pod zwiększonym obciążeniem, jak na przykład aplikacje w chmurze.

Zwracaj również uwagę na niezabezpieczoną transmisję poświadczeń, wykorzystującą zazwyczaj domyślny protokół HTTP bez przełączania na HTTPS. Badaj funkcje „zapomniałem hasła" lub „przypomnij hasło". *Wyliczaj nazwy użytkowników* (odgaduj je lub twórz listę rzeczywistych nazw). Szukaj sytuacji, w których po nieudanym uwierzytelnieniu z jakiegoś powodu można uzyskać dostęp do aplikacji.

Zarządzanie sesjami

Sesja w aplikacji WWW jest sekwencją transakcji HTTP skojarzonych z jednym użytkownikiem. Zarządzanie sesją, czyli śledzenie tych transakcji HTTP, może być skomplikowane, dlatego badaj je pod kątem błędów. Sprawdzaj, czy tokeny są przewidywalne, czy są przesyłane w niezabezpieczony sposób lub ujawniane w dziennikach. Może się również okazać, że sesje nie wygasają odpowiednio szybko, można je przejmować lub są podatne na atak CSRF (ang. *Cross Site Request Forgery*, fałszywe żądania z innych witryn), polegający na manipulowaniu sesjami uwierzytelnionych użytkowników w celu wykonania szkodliwych operacji.

Kontrola dostępu i autoryzacja

Sprawdzaj, czy witryna prawidłowo kontroluje dostęp. *Definiowanie uprawnień na poziomie użytkowników*, czyli nadawanie im różnych praw dostępu do danych i funkcji, jest typowe dla urządzeń IoT, w szczególności skomplikowanych systemów medycznych. Jest to tzw. kontrola RBAC (ang. *role-based access control*, kontrola dostępu oparta na rolach). Na przykład lekarz może mieć szeroki dostęp do bazy elektronicznych kart pacjentów, a pielęgniarka ograniczony tylko do ich odczytywania. Podobnie systemy kamer monitorujących mają przynajmniej po dwa konta: administratora, z uprawnieniami do zmieniania ustawień, oraz mniej uprzywilejowane konto operatora, dające jedynie wgląd w obrazy z kamer. Oba rodzaje systemów, aby mogły poprawnie funkcjonować, muszą kontrolować dostęp. Widzieliśmy systemy, w których można było wykonywać zabronione operacje za pomocą nieuprzywilejowanych kont, po prostu wysyłając odpowiednie żądania HTTP na odpowiednie adresy URL. Ta technika nosi nazwę **wymuszonego przeglądania**. Jeżeli system obsługuje wiele kont użytkowników, sprawdzaj zakresy wszystkich uprawnień. Na przykład czy za pomocą konta gościa można użyć funkcjonalności przeznaczonej tylko dla administratora? Czy można zyskać dostęp do interfejsu API obsługiwanego przez inną upoważnioną do tego celu platformę?

Weryfikacja danych wejściowych

Sprawdzaj, czy aplikacja weryfikuje i oczyszcza dane wprowadzane przez użytkownika we wszystkich punktach. Jest to bardzo ważne, ponieważ najpopularniejszą luką w bezpieczeństwie aplikacji WWW jest możliwość wstrzykiwania danych. Haker może ją wykorzystać do wprowadzenia do aplikacji własnego kodu (patrz lista *OWASP Top 10* najpopularniejszych luk). Testowanie stosowanego w aplikacji mechanizmu weryfikacji danych wejściowych może być czasochłonnym

procesem, obejmującym wstrzykiwanie wszelkiego rodzaju danych (w tym zapytań SQL i poleceń systemu operacyjnego) oraz uruchamianie skryptów XSS (ang. *Cross-Site Scripting*, skrypty międzydomenowe) i XEE (ang. *XML External Entity*, zewnętrzna jednostka XML).

Błędy w algorytmie

Szukaj podatności na ataki wynikających z błędów w algorytmie. Jest to szczególnie ważne, jeżeli aplikacja realizuje wieloetapowe procesy, w których operacje muszą być wykonywane w ściśle określonej kolejności. Jeżeli w wyniku zmiany tego porządku aplikacja zacznie działać w nieprzewidziany lub niepożądany sposób, to oznacza, że w jej algorytmie jest błąd. Często wykrywanie tego rodzaju błędów jest ręcznym procesem, wymagającym znajomości kontekstu aplikacji i branży, dla której jest przeznaczona.

Serwer aplikacyjny

Sprawdzaj, czy serwer, na którym działa aplikacja, jest bezpieczny. Zainstalowanie bezpiecznej aplikacji na niezabezpieczonym serwerze przeczy zasadom bezpieczeństwa. Aby przetestować bezpieczeństwo serwera, użyj skanera wykrywającego błędy i powszechnie znane luki. Sprawdzaj odporność serwera na ataki deserializacyjne, skuteczność zapory aplikacji internetowych i błędy w konfiguracji, takie jak listy katalogów, domyślna zawartość i metody HTTP. Identyfikuj luki w szyfrowaniu SSL/TLS, na przykład słabe algorytmy, samopodpisane certyfikaty.

Przegląd konfiguracji hosta

Przeglądanie konfiguracji hosta polega na ocenianiu bezpieczeństwa systemu po uzyskaniu do niego lokalnego dostępu. Przykładem jest przegląd konta lokalnego użytkownika serwera Windows, będącego komponentem systemu IoT. Po uzyskaniu dostępu ocenia się m.in. konta użytkowników, połączenia zdalnej pomocy technicznej, kontrolę dostępu do systemu plików, udostępniane usługi sieciowe, zabezpieczenie konfiguracji.

Konta użytkowników

Sprawdzaj, czy konta użytkowników są skonfigurowane w bezpieczny sposób, tj. czy istnieją konta domyślne i czy zasady dostępu są skuteczne. Zasady te obejmują rejestrowanie *historii haseł* (czy i kiedy można ponownie stosować stare hasła), terminy *ich wygaśnięcia* (tj. jak często system wymaga od użytkowników zmian haseł) i *mechanizmy blokowania* kont (tj. po ilu nieudanych próbach zalogowania konto jest blokowane). Jeżeli urządzenie IoT znajduje się w firmowej sieci, sprawdzaj zasady jednolitości kont. Jeżeli na przykład zasada bezpieczeństwa w organizacji wymaga, aby użytkownicy zmieniali hasła co sześć miesięcy,

sprawdzaj, czy dotyczy ona wszystkich kont. Idealnym rozwiązaniem jest integracja systemu z usługą Active Directory lub LDAP. Dzięki temu zasady można centralnie egzekwować za pomocą serwera.

Ten etap testów może wydawać się prozaiczny, ale jest jednym z najważniejszych. Hakerzy bardzo często wykorzystują nieobjęte centralnym zarządzaniem, niewłaściwie skonfigurowane i zapomniane konta. Często znajdowaliśmy lokalne konta, do których hasła były takie same jak nazwy użytkowników i do tego nie wygasały.

Siła haseł

Sprawdzaj bezpieczeństwo haseł. Hasła muszą być silne, aby haker nie mógł ich złamać za pomocą zautomatyzowanego narzędzia. Weryfikuj zasady złożoności haseł zdefiniowane w systemie Windows w zasadach grupowych lub lokalnych, a w systemie Linux — w module PAM (ang. *Pluggable Authentication Module*, dołączany moduł uwierzytelniający). Pamiętaj jednak, że zasady te nie mogą zakłócać procesów w organizacji. Przeanalizujmy następujący przykład: zasada bezpieczeństwa zdefiniowana w systemie chirurgicznym wymaga, aby hasło składało się z 16 znaków, a ponadto konto jest blokowane po trzech nieudanych próbach zalogowania. Jest to recepta na katastrofę, ponieważ lekarz lub pielęgniarka w krytycznej sytuacji mogą nie być w stanie uwierzytelnić się w systemie. Gdy znaczenie mają sekundy i stawką jest ludzkie życie, zasady bezpieczeństwa nie mogą wywoływać negatywnych skutków.

Uprawnienia kont

Sprawdzaj, czy konta i usługi są skonfigurowane zgodnie z *zasadą minimalnych uprawnień*, tj. dają dostęp wyłącznie do tych zasobów, które są potrzebne. Często mieliśmy do czynienia z błędnie skonfigurowanym oprogramowaniem, bez precyzyjnie zdefiniowanych uprawnień. Na przykład główny proces nie zwracał rozszerzonych uprawnień, gdy już ich nie potrzebował, albo system pozwalał na uruchamianie różnych procesów na tym samym koncie. Te procesy zazwyczaj wymagały dostępu tylko do ograniczonych zasobów, zatem ich uprawnienia były zbyt szerokie. Gdyby haker przejął nad nimi kontrolę, mógłby uzyskać dostęp do całego systemu. Nierzadko widzieliśmy proste usługi dzienników działające z uprawnieniami systemowymi lub administracyjnymi. Niemal we wszystkich naszych raportach umieszczaliśmy wniosek „usługi o zbyt szerokich uprawnieniach".

W systemie Windows można rozwiązać ten problem za pomocą *zarządzanych kont usług*, które pozwalają izolować konta domenowe używane przez krytyczne aplikacje i automatyzować zarządzanie poświadczeniami. W systemach Linux są stosowane mechanizmy bezpieczeństwa, takie jak funkcjonalności (*capabilities*), seccomp (umieszczanie wywołań systemowych na białej liście), SELinux i AppArmor, przy użyciu których można ograniczać uprawnienia procesów i wzmacniać bezpieczeństwo systemu. Oprócz tego w zarządzaniu kontami mogą być pomocne rozwiązania takie jak Kerberos, OpenLDAP i FreeIPA.

Poziom poprawek

Sprawdzaj, czy system operacyjny, aplikacje i zewnętrzne biblioteki są aktualne i czy jest aktywny proces ich aktualizowania. Poprawki są ważne, skomplikowane i zazwyczaj niezrozumiałe. Testowanie przestarzałego oprogramowania może wydawać się rutynowym zadaniem (które zwykle można zautomatyzować za pomocą narzędzi do skanowania podatności), ale prawie nigdzie nie są stosowane w pełni aktualne systemy. Aby wyszukiwać otwarte komponenty z rozpoznanymi lukami w zabezpieczeniach, stosuj *narzędzia do analizy oprogramowania*, które automatycznie sprawdzają zewnętrzne kody pod kątem brakujących poprawek. Aby wykrywać brakujące poprawki systemu operacyjnego, korzystaj z wiarygodnych skanerów podatności lub rób to ręcznie. Upewniaj się, że dostawca urządzenia IoT serwisuje system Windows lub Linux. Przekonasz się, że często tak nie jest.

Instalowanie poprawek to prawdziwa zmora w dziedzinie bezpieczeństwa informacji, zwłaszcza w świecie IoT. Wynika to głównie z faktu, że urządzenia wbudowane zazwyczaj mają zapisane na stałe skomplikowane oprogramowanie układowe, którego aktualizowanie z natury jest trudnym zadaniem. Ponadto regularne instalowanie poprawek w niektórych systemach, na przykład bankomatach, może być zbyt kosztowną praktyką z powodu kosztów *przestoju* (czasu, w którym klienci nie mają dostępu do systemu) oraz dużego nakładu pracy. W przypadku bardziej specjalistycznych systemów, takich jak urządzenia medyczne, sprzedawca przed udostępnieniem nowej poprawki musi przeprowadzać rygorystyczne testy. Na pewno nie chciałbyś, aby z powodu błędu w obliczeniach zmiennoprzecinkowych w najnowszej poprawce analizator krwi pokazał dodatni wynik testu na zapalenie wątroby, prawda? A co z poprawkami oprogramowania rozrusznika serca? Aktualizacja może być kwestią życia lub śmierci pacjentów, chyba że każdego będzie się wzywać do gabinetu lekarskiego w celu instalacji poprawek.

W naszej pracy często widzieliśmy zewnętrzne oprogramowanie bez zainstalowanych poprawek stosowane z aktualnymi, podstawowymi komponentami. Typowym przykładem jest system Windows z oprogramowaniem Java, Adobe, a nawet Wireshark. W systemie Linux powszechne jest stosowanie starych wersji oprogramowania OpenSSL. Czasami jakieś programy są instalowane bez żadnego powodu. W takim przypadku najlepiej je usunąć, zamiast opracowywać proces ich aktualizacji. Na przykład do czego może być potrzebny program Adobe Flash na serwerze komunikującym się z aparatem USG?

Zdalne utrzymanie

Sprawdzaj zabezpieczenia połączeń przeznczonych do zdalnego utrzymania i obsługi urządzenia. Często w celu zainstalowania poprawek w danym urządzeniu nie wysyła się go do dostawcy, tylko daje jego personelowi technicznemu zdalny dostęp. Haker może wykorzystać taką funkcjonalność jako tylne wejście do systemu i uzyskać do niego administracyjny dostęp. W większości przypadków takie połączenia nie są zabezpieczone. Przypomnij sobie atak Target, w którym hakerzy dostali się do głównej sieci sklepu za pośrednictwem firmy obsługującej system wentylacji.

Dostawcy urządzeń IoT mogą zdalnie instalować poprawki, ponieważ może to być jedyny skuteczny sposób ich szybkiego dostarczania. Niektóre urządzenia są tak delikatne i skomplikowane, że użytkownicy nie mają odwagi, aby je samodzielnie aktualizować. Zawsze istnieje pewne prawdopodobieństwo, że w wyniku takiego procesu urządzenie zostanie uszkodzone. Co się stanie, gdy urządzenie ulegnie awarii, a będzie pilnie potrzebne (na przykład tomograf komputerowy w szpitalu lub bardzo ważny czujnik temperatury w elektrowni)?

Ważne jest ocenianie nie tylko oprogramowania do zdalnego utrzymywania urządzenia (najlepiej poprzez inżynierię odwrotną jego plików binarnych) i jego kanału komunikacyjnego, ale także całego procesu. Czy dostęp jest możliwy w trybie 24/7? Czy jest stosowane uwierzytelnienie dwuskładnikowe? Czy wykonywane operacje są rejestrowane?

Kontrola dostępu do systemu plików

Sprawdzaj, czy opisana wcześniej w tym rozdziale zasada minimalnych uprawnień dotyczy również plików i katalogów. Często zdarza się, że użytkownik o wąskich uprawnieniach może zapisywać i odczytywać krytyczne katalogi i pliki (na przykład pliki wykonywalne usług), co daje mu możliwość rozszerzenia uprawnień. Czy użytkownik inny niż administrator faktycznie potrzebuje uprawnień do zapisu plików w katalogu *C:\Program Files*? Czy wszyscy użytkownicy muszą mieć dostęp do katalogu */root*? Kiedyś badaliśmy urządzenie wykorzystujące ponad pięć różnych skryptów uruchomieniowych, które mogli modyfikować użytkownicy nieadministratorzy. W efekcie haker posiadający lokalny dostęp mógł uruchamiać własne programy i przejmować kontrolę nad całym systemem.

Szyfrowanie danych

Sprawdzaj, czy poufne dane są szyfrowane. Zaczynaj od najważniejszych, na przykład *zdrowotnych i osobowych*. Dane zdrowotne zawierają informacje o stanie zdrowia pacjentów, świadczeniach medycznych i opłacanych składkach, natomiast dane osobowe mogą być wykorzystane do identyfikowania osób. Sprawdzaj konfigurację systemu pod kątem stosowanych algorytmów kryptograficznych. Czy w przypadku kradzieży dysku dane mogłyby zostać odczytane? Czy szyfrowany jest cały dysk, baza danych albo innego rodzaju informacje? Jak bezpieczny jest stosowany algorytm kryptograficzny?

Błędy w konfiguracji serwera

Błędnie skonfigurowane usługi mogą nie być bezpieczne. Zdarzają się na przykład serwery FTP posiadające domyślne konta gości, które haker może wykorzystywać do nawiązywania anonimowych połączeń, a następnie odczytywania i zapisywania określonych katalogów. Kiedyś napotkaliśmy program Oracle Enterprise Manager posiadający uprawnienia systemowe, do którego dostęp był możliwy z domyślnymi poświadczeniami. Haker mógł zmienić zapisane procedury Javy i za ich pomocą korzystać z poleceń systemu operacyjnego. Wykorzystując tę lukę, mógł poprzez sieć przejąć kontrolę nad całym systemem.

Testy aplikacji przenośnych i chmurowych

Testuj bezpieczeństwo każdej aplikacji przenośnej związanej z systemem IoT. Dzisiaj programiści tworzą aplikacje na systemy Android i iOS do wszystkiego, nawet rozruszników serca! Więcej o bezpieczeństwie aplikacji przenośnych dowiesz się w rozdziale 14. Zapoznaj się z następującymi stronami projektu *OWASP*: *Mobile Top 10* (10 najważniejszych zagrożeń aplikacji przenośnych), *Mobile Security Testing Guide* (przewodnik po testach bezpieczeństwa aplikacji przenośnych) oraz *Application Security Verification Standard* (standard weryfikacji bezpieczeństwa aplikacji).

Badając ostatnio pewną aplikację, odkryliśmy, że wysyłała ona dane medyczne pacjentów do chmury bez wiedzy korzystających z niej lekarzy i pielęgniarek. Nie jest to wprawdzie błąd techniczny, ale istotne naruszenie poufności danych, o którym użytkownicy powinni wiedzieć.

Oceniaj również bezpieczeństwo wszystkich chmurowych komponentów systemu IoT. Badaj interakcje między chmurą a komponentami IoT. Szczególną uwagę zwracaj na wewnętrzne interfejsy API i implementacje na platformach chmurowych, m.in. AWS, Azure i Google Cloud Platform. Przekonasz się, jak wiele z nich stosuje niezabezpieczone bezpośrednie odwołania do obiektów (ang. *Insecure Direct Object References*, IDOR), za pomocą których każdy, kto zna odpowiedni adres URL, może uzyskać dostęp do poufnych danych. W ten sposób można na przykład odczytywać obiekty danych umieszczone w pojemniku S3 w usłudze AWS.

Wiele zadań związanych z testowaniem chmury obejmuje ocenianie bezpieczeństwa aplikacji przenośnych i internetowych. W pierwszym przypadku powodem jest to, że aplikacja kliencka wykorzystująca interfejs API działa w systemie Android lub iOS. W drugim natomiast liczne komponenty chmurowe to usługi internetowe. Sprawdzaj wszelkie chmurowe połączenia służące do zdalnego utrzymywania i serwisowania aplikacji, o czym pisaliśmy w podrozdziale „Przegląd konfiguracji hosta".

W naszej pracy widzieliśmy wiele luk związanych z chmurą: zakodowane na stałe tokeny, klucze API zapisane w aplikacjach mobilnych i w plikach binarnych oprogramowania układowego, brak zaufanych certyfikatów TLS oraz publicznie dostępne usługi intranetowe (na przykład nieuwierzytelniony serwer Redis lub usługa metadanych) spowodowane błędem w konfiguracji. Pamiętaj, że na przeprowadzenie jakichkolwiek testów usług chmurowych musisz mieć pozwolenie od ich właściciela.

Podsumowanie

Niektórzy z nas służyli w wojskowych departamentach cyberobrony. Nauczyliśmy się tam, że dogłębna analiza jest jednym z najważniejszych aspektów bezpieczeństwa informacji. Ważne jest stosowanie ustalonych metodyk testowania, aby nie przeoczyć niektórych oczywistych przypadków. Łatwo jest zignorować pewne zagrożenia tylko dlatego, że wydają się błahe lub oczywiste.

W tym rozdziale została przedstawiona metodyka oceniania bezpieczeństwa systemów IoT. Opisaliśmy pasywny rekonesans, a następnie podzieliliśmy na mniejsze segmenty warstwy: fizyczną, sieciową, aplikacji internetowych, hosta, aplikacji przenośnych i chmury.

Pamiętaj, że omówione w tym rozdziale warstwy koncepcyjne nie są bezwzględnie obowiązujące. Pomiędzy dwiema lub więcej warstwami może znajdować się kilka innych, zachodzących na siebie warstw. Na przykład atak mający na celu wyczerpanie baterii może po części należeć do warstwy fizycznej, ponieważ bateria jest sprzętem. Może również należeć do warstwy sieciowej, ponieważ atak jest przeprowadzany z wykorzystaniem protokołu komunikacji bezprzewodowej. Lista ocenianych komponentów też nie jest wyczerpująca, dlatego w szczególnych przypadkach skorzystaj z dodatkowych materiałów.


PROGRAM
PARTNERSKI
GRUPY HELION
1. ZAREJESTRUJ SIĘ
2. PREZENTUJ KSIĄŻKI
3. ZBIERAJ PROWIZJĘ
Zmień swoją stronę WWW w działający bankomat!
Dowiedz się więcej i dołącz już dzisiaj!
http://program-partnerski.helion.pl
GRUPA
Helion

IoT. Które urządzenie zhakujesz najpierw?

Konsekwencje udanego ataku na IoT mogą być niezwykle poważne. Zagrożenie dotyczy przecież wszystkiego, co działa pod kontrolą komputera. Mogą to być samochód, rozrusznik serca, zamki w inteligentnym domu czy też system sterujący maszynami w fabryce. Zabezpieczanie i testowanie bezpieczeństwa urządzeń internetu rzeczy jest jednak specyficzną dziedziną. Przesądza o tym jego charakter, ale także budowa i ekonomika produkcji urządzeń IoT. Inżynier bezpieczeństwa internetu rzeczy musi więc przyswoić zupełnie inne metody pracy i pokonywać inne problemy.

Ta książka jest praktycznym przewodnikiem po technikach atakowania internetu rzeczy. Dzięki niej dowiesz się, w jaki sposób testować systemy, urządzenia i protokoły i jak ograniczać ryzyko. Zawarto tutaj przegląd typowych zagrożeń i opisano sposoby ich modelowania. Omówiono również metodykę testowania bezpieczeństwa i pasywnego rekonesansu, a także zasady oceny zabezpieczeń wszystkich warstw systemów IoT. Zaprezentowano techniki ataków polegających na przeskakiwaniu między sieciami VLAN, łamaniu uwierzytelnień w protokole MQTT, zakłócaniu usługi mDNS czy zniekształcaniu komunikatów WS-Discovery. W trakcie lektury nauczysz się hakować sprzęt i transmisję radiową, poznasz też metodykę ataków na wbudowane urządzenia IoT i systemy RFID.

W książce:

- skanowanie usługi DICOM
- hakowanie mikrokontrolerów
- inżynieria wsteczna oprogramowania układowego
- analiza aplikacji mobilnych
- zakłócanie pracy czytnika NFC
- hakowanie urządzeń inteligentnego domu

FOTIOS CHANTZIS zajmuje się zabezpieczeniami sieci i systemów sztucznej inteligencji. Od 2009 roku jest członkiem rdzenia zespołu programistów narzędzia Nmap. Występował na renomowanych konferencjach poświęconych bezpieczeństwu informatycznemu.

IOANNIS STAIS jest starszym analitykiem bezpieczeństwa IT. Zajmuje się prowadzeniem kontrolowanych ataków hakerskich. Interesuje się rozwojem algorytmów uczenia maszynowego, a także zagrożeniami aplikacji mobilnych i internetowych.

Współautorami tej książki są również **Paulino Calderon**, **Evangelos Deirmentzoglou** i **Beau Woods,** uznani praktycy w dziedzinie bezpieczeństwa informatycznego i autorzy cenionych publikacji w tym zakresie. Z ich doświadczenia skorzystało już wiele poważnych przedsiębiorstw i instytucji.

Helion
helion.pl
HELION SA
ul. Kościuszki 1c
44-100 Gliwice
tel.: 32 230 98 63
helion@helion.pl

ISBN 978-83-283-8339-5

INFORMATYKA W NAJLEPSZYM WYDANIU
Cena: 89,00 zł
no starch press