VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
FACULTY OF INFORMATION TECHNOLOGY

ÚSTAV INFORMAČNÍCH SYSTÉMŮ
DEPARTMENT OF INFORMATION SYSTEMS

ÚTOK NA WIFI SÍŤ S VYUŽITÍM ESP32/8266

WIFI ATTACKS USING ESP32/8266

DIPLOMOVÁ PRÁCE
MASTER'S THESIS

AUTOR PRÁCE — **Bc. RICHARD STEHLÍK**
AUTHOR

VEDOUCÍ PRÁCE — **Ing. JAN PLUSKAL**
SUPERVISOR

BRNO 2021

Vysoké učení technické v Brně
Fakulta informačních technologií

Ústav informačních systémů (UIFS) Akademický rok 2020/2021

Zadání diplomové práce

Student: **Stehlík Richard, Bc.**
Program: Informační technologie
Obor: Bezpečnost informačních technologií
Název: **Útok na WiFi síť s využitím ESP32/8266**
WiFi Attacks Using ESP32/8266
Kategorie: Bezpečnost
Zadání:

1. Nastudujte možnosti platformy ESP-IDF vzhledem k nízko-úrovňové manipulaci s pakety.
2. Nastudujte principy zabezpečení WiFi sítí, zejména pak WPA2.
3. Dle zjištění z bodu 1 a 2 zhodnoťte, které z existujících útoků jsou realizovatelné na platformě ESP32/8266. Po konzultaci s vedoucím vyberte vhodný útok a navrhněte jeho implementaci.
4. Návrh implementujte na vývojových ESP32/8266 kitech dle doporučení vedoucího.
5. Implementaci otestujte v řízeném laboratorním prostředí a vyhodnoťte úspěšnost. Zaměřte se na stabilitu a použitelnost těchto útoků a diskutujte jejich využití v praxi z pohledu útočníka a možnosti obrany.

Literatura:

- Lashkari, A.H., Danesh, M.M.S. and Samadi, B., 2009, August. A survey on wireless security protocols (WEP, WPA and WPA2/802.11 i). In *2009 2nd IEEE International Conference on Computer Science and Information Technology* (pp. 48-52). IEEE.
- Viehböck, S., 2011. Brute forcing wi-fi protected setup. *Wi-Fi Protected Setup*, *9*.
- Inspirujte se open source řešením https://www.varonis.com/blog/hacking-wi-fi-with-the-esp8266/ a https://hackaday.com/2019/09/05/esp8266-and-esp32-wifi-hacked/

Při obhajobě semestrální části projektu je požadováno:

- Body 1, 2 a 3.

Podrobné závazné pokyny pro vypracování práce viz https://www.fit.vut.cz/study/theses/

Vedoucí práce: **Pluskal Jan, Ing.**
Vedoucí ústavu: Kolář Dušan, doc. Dr. Ing.
Datum zadání: 1. listopadu 2020
Datum odevzdání: 19. května 2021
Datum schválení: 26. října 2020

Abstrakt

Tato práce má za cíl zkoumat možnosti čipů ESP32 firmy Espressif a jejich oficiálního vývojového aplikačního rámce Espressif IoT Development Framework s cílem implementace známých útoků na Wi-Fi sítě na této platformě. V práci je navržena implementace deautentizačního útoku ve dvou provedeních, zachycení ustavení WPA/WPA2 klíčů, útoku na PMKID, vytvoření podvrženého přístupového bodu v MitM pozici, útoku silou na WPS PIN a další. Byl navržen a implementován univerzální penetrační nástroj ESP32 Wi-Fi Penetration Tool včetně deautentizačního útoku se zachycením WPA/WPA2 ustavení klíčů. Tento nástroj umožňuje snadnou konfiguraci a spouštění útoků i bez nutnosti většího porozumění problému uživatelem. Výstup této práce otevírá útočníkům nové možnosti vedení útoků na Wi-Fi sítě využitím levných čipů ESP32 s nízkou spotřebou a malými rozměry.

Abstract

The goal of this thesis is an exploration of the possibilities of Espressif's ESP32 chips in combination with Espressif IoT Development Framework with intention of implementing well-known Wi-Fi attacks on this platform. In this work, multiple implementation proposals were done for deauthentication attack in two variants followed by WPA/WPA2 handshake capture, attack on PMKID, creation of rogue MitM access point, or brute-force attack on WPS PIN, and more. A universal penetration tool ESP32 Wi-Fi Penetration Tool was proposed and implemented, including deauthentication attacks with WPA/WPA2 handshake capture. This tool provides an easy way to configure and run malicious Wi-Fi attacks without any domain knowledge required from the user. The outcome of this work opens new attack vectors for the attacker, thanks to cheap, ultra-low powered, and lightweight ESP32 chips.

Klíčová slova

deautentizační útok, esp32, WPA/WPA2 handshake, KRACK, Kr00k, útok na PMKID, Wi-Fi útoky, zranitelnost 802.11

Keywords

802.11 vulnerabilities, attack on PMKID, deauthentication attack, esp32, KRACK, Kr00k, Wi-Fi attacks, WPA/WPA2 handshake

Citace

STEHLÍK, Richard. *Útok na WiFi síť s využitím ESP32/8266*. Brno, 2021. Diplomová práce. Vysoké učení technické v Brně, Fakulta informačních technologií. Vedoucí práce Ing. Jan Pluskal

Útok na WiFi síť s využitím ESP32/8266

Prohlášení

Prohlašuji, že jsem tuto diplomovou práci vypracoval samostatně pod vedením Ing. Jana Pluskala. Uvedl jsem všechny literární prameny, publikace a další zdroje, ze kterých jsem čerpal.

.....
Richard Stehlík
19. května 2021

Poděkování

Děkuji Ing. Janu Pluskalovi za vedení mé práce, za trpělivost, cenné rady a připomínky v průběhu práce a za provedení procesem tvorby uceleného díla.

Obsah

Kapitola 1

Úvod

Bezdrátové sítě Wi-Fi jsou součástí každodenního života většiny uživatelů informačních technologií. Wi-Fi je dnes standardem pro bezdrátovou komunikaci. Certifikaci Wi-Fi k roku 2020 obdrželo více než 50 000 produktů, přičemž v témže roce existovalo přes 18 miliard zařízení podporujících Wi-Fi. Pouze v databázi statistického webu WiGLE je takových sítí přes 700 miliónů z celého světa. Jiné zdroje uvádějí, že přes 600 miliónů Wi-Fi sítí jsou jenom veřejné přístupové body. Celkové množství včetně všech privátních sítí lze tedy předpokládat mnohonásobně větší. Bezdrátové médium přináší řadu nových překážek, které u klasických drátových sítí nebylo třeba řešit. S těmito problémy si standard 802.11 poradil a publikováním nových dodatků jej udržuje stále aktuální. Díky tomu je schopen čelit novým typům útoků, které jsou neustále objevovány.

Přesto jsou slabiny standardu 802.11 často umocňovány neznalostí běžných uživatelů, převážně v domácím prostředí, kde si své privátní sítě spravují samotní členové domácnosti. Nejčastějšími proviněními jsou slabá hesla, snadno prolomitelná slovníkovým útokem silou, nebo nevhodná konfigurace, kde jsou zbytné funkce přístupových bodů ponechány zapnuté. Přestože je aktuálně nejpoužívanější zabezpečení WPA2 považováno za bezpečné, jeho bezpečnost částečně závisí na správné konfiguraci a na použití silného sdíleného klíče. Implementace útoků na tyto zranitelnosti pomocí levných a dostupných čipů upozorňuje na jejich závažnost. S takto dostupnými útoky se stávají běžné Wi-Fi sítě náchylnějšími, jelikož si je může dovolit téměř kdokoliv. Vývojové desky založené na čipech ESP32 firmy Espressif mají potřebné parametry a podporu pro provedení útoku. Motivací pro tuto práci tedy je demonstrování, že platforma ESP32 firmy Espressif má potřebnou podporu pro provedení útoků na Wi-Fi sítě.

Tato práce zkoumá možnosti implementace takových útoků pomocí *Espressif IoT Development Framework (ESP-IDF)* — oficiálního aplikačního rámce pro vývoj aplikací pro platformu ESP32. V práci jsou navrženy implementace jednotlivých útoků. Praktickým výstupem práce je univerzální nástroj *ESP32 Wi-Fi Penetration Tool*[1], který poskytuje „aplikační rámec“ pro implementaci jednotlivých útoků. V rámci této práce je implementován deautentizační útok zahrnutý do tohoto nástroje.

Kapitola 2 shrnuje standard 802.11 a jeho dodatky v kontextu útoků na Wi-Fi sítě. První část kapitoly představuje architekturu běžných Wi-Fi sítí a samotnou komunikaci prvků v síti pomocí rámců. Druhá část se věnuje zabezpečení Wi-Fi sítí se zaměřením na *Wi-Fi Protected Access* a jeho součásti a jednotlivé verze. Kapitola 3 se zaměřuje na známé zranitelnosti standardu 802.11. Popisuje zneužitelné části samotného standardu a existující

[1] https://github.com/risinek/esp32-wifi-penetration-tool

útoky. Kapitola 4 je věnována čipům Espressif. První část kapitoly popisuje čipy ESP32, jejich vlastnosti a varianty, ve kterých jsou k dostání na trhu. Druhá část kapitoly se zaobírá aplikačním rámcem ESP-IDF a jeho aplikačním rozhraním. V této části jsou popsány možnosti ESP-IDF související s popsanými zranitelnostmi a útoky z kapitoly 3. Kapitola 5 je rešerší existujících projektů na čipech ESP8266 a ESP32 souvisejících s implementací útoků na Wi-Fi a shrnutí současného stavu na základě získaných znalostí z předchozích kapitol. Kapitola 6 popisuje návrh řešení implementace univerzálního jednotného nástroje pro spouštění různých útoků na Wi-Fi sítě. Jsou definovány funkční a nefunkční požadavky, možnosti vedení útoků pomocí různých přístupů možných v rámci ESP-IDF a navržena architektura projektu. Ve druhé části kapitoly jsou navrženy přístupy k implementaci útoků z kapitoly 3. Je navržen nový způsob provedení deautentizačního útoku pouze pomocí dostupných funkcí z ESP-IDF využívající specifického chování přístupového bodu definovaného ve standardu 802.11. V kapitole 7 je představena implementace univerzálního nástroje *ESP32 Wi-Fi Penetration Tool*, jeho jednotlivých částí — komponent. Je popsána implementace deautentizačního útoku se dvěma variantami. V první variantě je využito existujícího projektu, jenž umožňuje injektování podvržených deautentizačních rámců. Druhá varianta implementuje nový způsob provedení deautentizačního útoku bez nutnosti injektování rámců. V kapitole 8 je zhodnoceno implementované řešení a možnosti jeho použití. Je zde shrnuto testování implementovaného nástroje ESP32 Wi-Fi Penetration Tool v řízeném laboratorním prostředí. Dále jsou diskutovány možnosti využití implementací útoků na platformě ESP32 z pohledu útočníka a způsoby obrany proti nim.

Kapitola 2

Standard 802.11 z pohledu bezpečnosti

Standard 802.11 byl poprvé publikován organizací *Institute of Electrical and Electronics Engineers (IEEE)* v roce 1997 [26]. Tato organizace standard stále spravuje a stále rozšiřuje novými dodatky. Komerční název *Wi-Fi* pro bezdrátové sítě založené na tomto standardu byl definován v roce 1999 aliancí *Wireless Ethernet Compatibility Alliance*, následně přejmenovanou na *Wi-Fi Alliance* [45]. Tato obchodní nezisková aliance má na starost testování a certifikaci nových produktů implementujících standard 802.11 a opravňuje je k užívání *Wi-Fi Certified* názvu a loga. Označení *Wi-Fi* je tedy dnes běžně známé, a proto dále v této práci lze označení *Wi-Fi síť* chápat jako zaměnitelné s *bezdrátovou sítí standardu 802.11* [23]. Tato kapitola shrnuje podstatné části standardu 802.11 a jeho dodatků vztahujících se k bezpečnosti a známým zranitelnostem.

V první části se kapitola věnuje obecné architektuře bezdrátových sítí a základním stavebním jednotkám přenosu dat — rámcům. Druhá část shrnuje zabezpečení standardu 802.11 a podrobněji rozebírá dnes nejčastěji používaný *Wi-Fi Protected Access* a jeho varianty.

2.1 Architektura Wi-Fi sítí

Definice bezdrátových sítí standardu 802.11 se v mnohém podobá drátové variantě *Ethernet* definované ve standardu 802.3 [18]. Bezdrátové médium ovšem přináší nová úskalí, která v Ethernetu neexistovala. Kupříkladu standard 802.11 i 802.3 řeší vícenásobný přístup k médiu algoritmem *Carrier Sense Multiple Access (CSMA)*, ovšem liší se v řešení kolizí [23, 18]. Zatímco v drátových sítích je možné kolize detekovat (anglicky *collision detection*), a je tedy aplikován algoritmus CSMA/CD [25], u bezdrátových sítí je tato problematika obtížnější a je nutné aplikovat algoritmus, který se kolizím snaží vyhnout (anglicky *collision avoidnance*). Pro bezdrátové Wi-Fi sítě je tedy použit algoritmus CSMA/CA [26, 18]. U Ethernetu není třeba řešit autentizaci (nepočítaje se standardem 802.1x) stanic přímo ve standardu 802.3, jelikož zařízení jsou připojena fyzickou linkou, a je tedy možné zabezpečit přistup k síti fyzicky [18]. Oproti tomu bezdrátové sítě tuto výsadu nemají, a je tedy potřeba tyto problémy řešit standardizací.

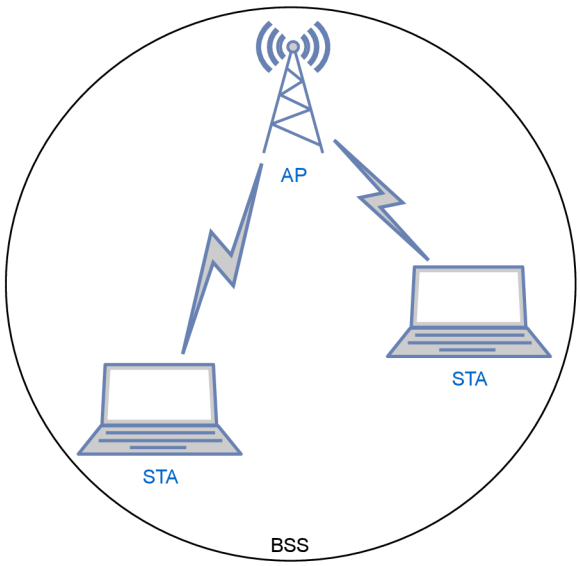
2.1.1 Základní prvky Wi-Fi sítí

V bezdrátových sítích založených na standardu 802.11 se vyskytuje několik prvků, které je vhodné definovat. Pro tyto prvky se v odborné literatuře nebo přímo ve standardu ustálily určité výrazy a zkratky [18]. Ty, které jsou v této práci používaný, jsou v následující části shrnuty.

Stanice (STA) — z anglického *station* — označuje zařízení schopné komunikace pomocí standardu 802.11, které je připojené k existující síti, případně se připojující/odpojující. [26, 18].

Přístupový bod (AP) — z anglického *access point* — je zařízení, které poskytuje služby bezdrátové sítě a slouží jako vstupní brána do sítě pro asociované stanice [26, 18].

Základní obslužný celek (BSS) — z anglického *basic service set* — je základní logický celek skládající se ze stanic a právě jednoho přístupového bodu. Wi-Fi síť složená z BSS se nazývá *infrastrukturní sítí* [18]. Pro identifikaci jednotlivých AP se používá identifikátor *BSSID*, který ve většině případů odpovídá MAC adrese bezdrátového rozhraní AP [26, 18]. Příklad BSS je znázorněn na obrázku 2.1.

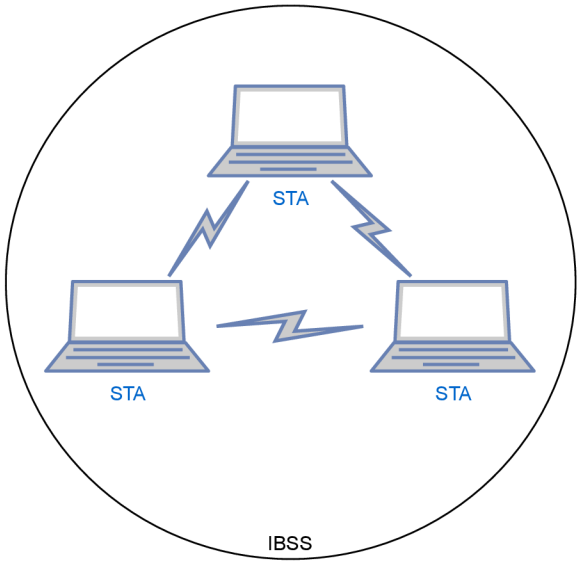


Obrázek 2.1: Příklad základního obslužného celku — BSS. Existuje právě jeden přístupový bod a k němu připojené (asociované) stanice.

Nezávislé BSS (IBSS) — z anglického *independent basic service set* — je obdobou infrastrukturního BSS, ovšem v případě nezávislého BSS v obslužném celku neexistují žádné AP, ale stanice komunikují přímo. Wi-Fi síť složená z IBSS je nazývána *ad-hoc sítí* [26, 18].

Pro ilustraci rozdílu mezi BSS a IBSS lze porovnat obrázky 2.1 a 2.2.

Rozšířený obslužný celek (ESS) — z anglického *extended service set* — je množinou všech propojených BSS (jejich AP) se shodným názvem sítě *SSID*. Wi-Fi síť tedy odpovídá právě této množině ESS, která se skládá z jednoho či více BSS. Identifikátorem ESS je *SSID*, což je zároveň název samotné sítě [26, 18]. Validní je i méně



Obrázek 2.2: Příklad nezávislého základního obslužného celku IBSS. Není přítomen přístupový bod a stanice spolu komunikují přímo.

používaná, přesto přesnější zkratka *ESSID* (z anglického *extended service set identifier*) [35]. Běžně se však používá zkratka *SSID* — včetně textů samotného standardu 802.11 [18]. V této práci bude dále používána právě tato zkratka *SSID*

Příklad sítě je znázorněn na obrázku 2.3.

Distribuční systém (DS) — z anglického *distribution system* — není standardem 802.11 přesně definován, ale určuje vlastnosti, které musí splňovat. Distribuční systém řeší komunikaci mezi BSS v rámci ESS. Musí znát asociace stanic k jednotlivým BSS a řídit přenos dat mezi více BSS vhodným směrováním rámců [18].

Příklad propojení přístupových bodů symbolizující zjednodušený distribuční systém je znázorněn na obrázku 2.3.

2.2 Rámce standardu 802.11

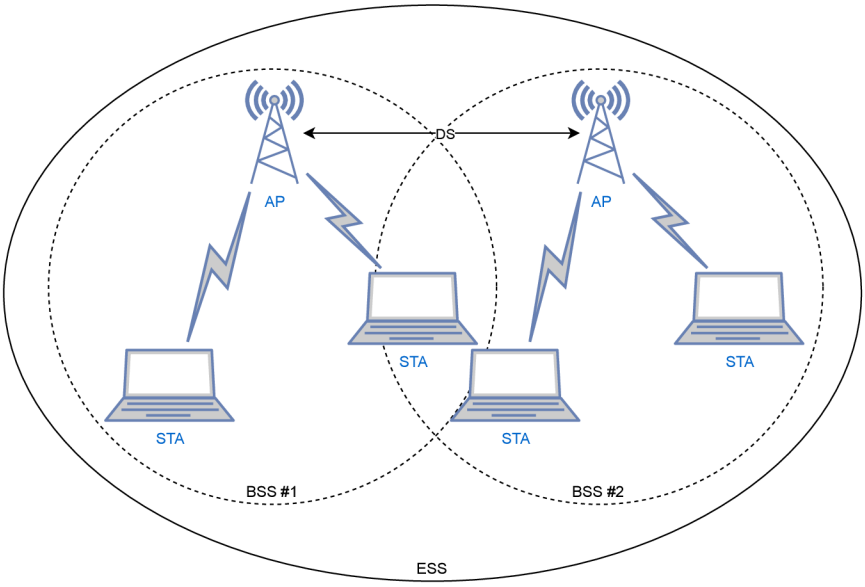
Komunikace mezi prvky uvnitř Wi-Fi sítě probíhá pomocí rámců. Standard 802.11 definuje tři základní typy takovýchto rámců — *kontrolní*, *řídicí* a *datové* [26].

Kontrolní rámce slouží k řízení a domluvě zúčastněných prvků na přístupu k bezdrátovému médiu.

Řídicí rámce slouží ke správě samotné Wi-Fi sítě jako například asociaci STA s AP, jejich autentizaci, deautentizaci a podobně.

Datové rámce slouží k samotnému přenosu užitečných dat protokolů vyšších vrstev, například EAPoL, IP, TCP.

V následujících sekcích jsou tyto rámce a jejich typy blíže popsány. Popis všech jejich podtypů je však záměrně vynechán, neboť jejich znalost není nutná pro pochopení dalších částí práce. Pro pochopení zranitelností a útoků na Wi-Fi sítě popisovaných dále v této práci je však vhodné znát základní princip rámců a formát některých jejich typů.

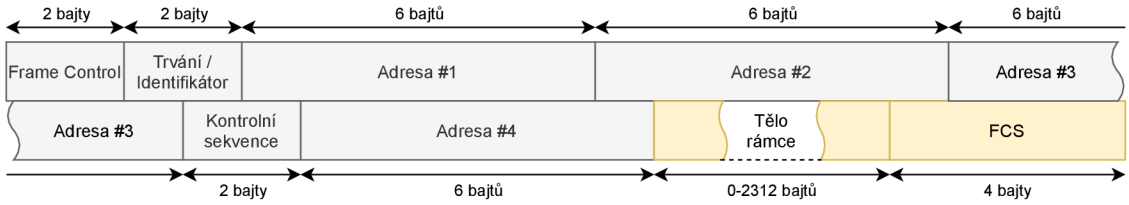


Obrázek 2.3: Příklad sítě Wi-Fi — rozšířeného obslužného celku s distribučním systémem. Síť se skládá z více základních obslužných celků — BSS, jež jsou propojeny distribučním systémem — DS.

2.2.1 Obecný formát rámců

Rámce mají společný formát, kde se podle typu a podtypu rámce mění sémantika a způsob využití jednotlivých polí [26]. Celková délka rámce se liší v závislosti na počtu využitých polí a velikosti samotného těla rámce.

Na obrázku 2.4 je znázorněn obecný tvar rámce a jeho polí.



Obrázek 2.4: Obecný formát rámce. Šedé podbarvení značí prvky, jež jsou součástí MAC hlavičky.

V následující části jsou blíže popsány jednotlivá pole rámce. Definice vychází ze standardu 802.11 [26].

Frame Control Každý rámec začíná dvěma bajty řízení rámce. Tyto bajty nesou informaci, o jaký typ a podtyp rámce jde, a jeho další parametry. Jednotlivé parametry jsou vynechány, jelikož nejsou podstatné pro další části práce.

Trvání Po Frame Control následují 2 bajty trvání (anglicky *duration*). Hodnota tohoto pole udává, po jakou dobu v milisekundách aktuální vysílání obsadí médium. Ostatní prvky v síti monitorují tuhle informaci u všech rámců a upraví ji.

Adresy Každý rámec nese několik adres, jejichž interpretace záleží na typu (a podtypu) rámce. Adresování vychází z konvence užívané v jiných 802 sítích. A tedy využívá 48bi-

tových IEEE MAC identifikátorů. Nejčastěji první adresa je adresa cíle, tedy adresa stanice, které jsou určena data/informace přenášené v rámci. Zdrojová adresa identifikuje stanici, která tento rámec původně vytvořila a odeslala. Dalšími méně častými typy adres jsou adresy odesílatele a příjemce, ale ty se používají ve specifických případech bezdrátového mostu, případně adresování zařízení v Ethernet síti připojené do Wi-Fi [18]. Dále se často v adresovém poli posílá hodnota BSSID, což je identifikátor bezdrátové sítě. V infrastrukturní síti je to MAC adresa bezdrátového rozhraní AP. Ad-hoc sítě generují náhodné BSSID, kde takto vygenerovaný identifikátor je odlišen od klasické MAC adresy nastavením Universal/Local bitu [18].

Kontrolní sekvence Dva bajty kontrolní sekvence se využívají při defragmentaci (první 4 bity), případně rozlišení duplikovaných rámců (zbylých 12 bitů).

Tělo rámce Tělo rámce je ta část, která nese samotná užitečná data u datových rámců, případně další informace v případě management a kontrolních rámců.

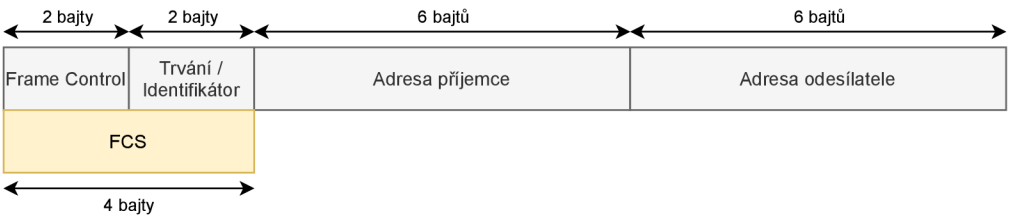
Kontrolní sekvence rámce Frame Check Sequence (FCS) slouží k ověření integrity rámce s využitím CRC (z anglického *cyclic redundancy check*) obdobně jako například u ethernetového protokolu.

2.2.2 Kontrolní rámce

Kontrolní rámce (z anglického výrazu *control frame*) slouží ke správě přístupu stanic a přístupových bodů k bezdrátovému médiu. Existují čtyři typy kontrolních rámců [26]:

- Request to send (RTS, v překladu *žádost o odeslání*)
- Clear to send (CTS, v překladu *volně k odeslání*)
- Acknowledge (ACK, v překladu *potvrzení*)
- Power save poll (PS-Poll, v překladu *dotazování úspory spotřeby*)

Formát kontrolního rámce se liší v závislosti na jeho typu. Příklad formátu kontrolního rámce je ilustrován typem RTS na obrázku 2.5.



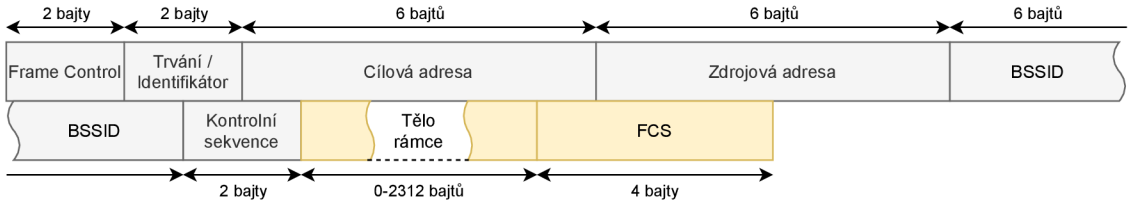
Obrázek 2.5: Příklad formátu kontrolního rámce. Formát kontrolních rámců se liší v závislosti na jejich podtypu. Na obrázku je vyobrazen podtyp RTS. Šedé podbarvení značí MAC hlavičku.

2.2.3 Řídicí rámce

V bezdrátových sítích přibývá režie, která v ethernetových sítích neexistovala [18]. Ve Wi-Fi sítích je potřeba zabezpečit, aby stanice měla možnost zjistit přítomnost a stav sítí v okolí,

aby se mohla autentizovat a asociovat do sítě výměnou informací o podporovaných službách a aby bylo možné spravovat síť samotnou [18]. K tomu slouží právě *řídicí rámce* (z anglického výrazu *management frame*).

Obecný formát řídicích rámců je vyobrazen na obrázku 2.6. Řídicí rámce nevyužívají čtvrté pole pro adresu a za kontrolní sekvencí následuje tělo řídicího rámce. První a druhá adresa odpovídá definici obecného formátu, tedy první adresa je MAC adresa cílového zařízení a druhá adresa je MAC adresa zdrojového zařízení [26]. Třetí adresa zde obsahuje informaci o BSSID, v rámci kterého BSS tato komunikace probíhá. BSSID svým formátem odpovídá délce MAC adresy.

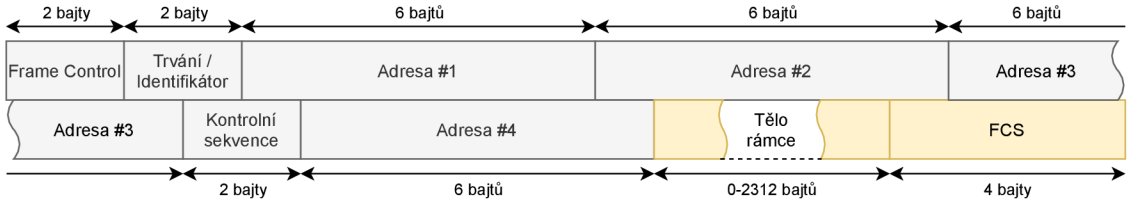


Obrázek 2.6: Formát řídicího rámce. Šedá část označuje prvky, jež jsou součástí MAC hlavičky.

2.2.4 Datové rámce

Datové rámce (z anglického výrazu *data frame*) primárně přenášejí samotná užitečná data protokolů vyšších vrstev, případně mohou vykonávat částečné řídicí funkce v *contention-free sítích*[1] [18].

Formát datových rámců je vyobrazen na obrázku 2.7. Sémantika adres se liší podle hodnot bitů ToDS a FromDS v poli Frame Control. Význam jednotlivých kombinací je znázorněn v tabulce 2.1.



Obrázek 2.7: Obecný formát datového rámce. Šedé podbarvení značí MAC hlavičku. Sémantika jednotlivých adres závisí na parametrech ToDS a FromDS v poli Frame Control.

2.3 Zabezpečení Wi-Fi sítí

V drátových ethernetových sítích je zabezpečení dat proti odposlechu zaručeno fyzickou architekturou sítě [18]. Pro odposlechnutí přenosu v drátových sítích musí útočník většinou získat fyzický přístup ke kabelu nebo ethernetové přípojce [18]. Naopak u bezdrátových sítí je kdokoliv v dostatečné blízkosti aktivní sítě schopen odposlechnout a číst přenášená data. Proto již v první verzi standardu 802.11 bylo definováno zabezpečení, které mělo

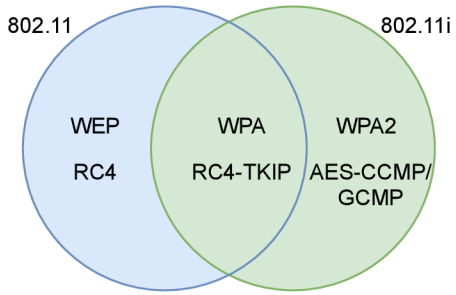
[1] Tyto sítě nejsou téměř vůbec implementovány, a proto nejsou v této práci dále rozebírány. Podrobnosti lze získat například z knihy *802.11 Wireless Networks* od autora *Matthew Gasta* [18].

Tabulka 2.1: Sémantika adres v datových rámcích [26]. CA značí cílovou MAC adresu, ZA značí zdrojovou adresu. V případě komunikace mezi dvěma AP v rámci DS jsou výrazy pozměněny, kde AP značí adresu příjemce a AO adresu odesílatele. Případ, kdy hodnoty ToDS i FromDS jsou rovny nule, značí komunikaci uvnitř ad-hoc sítě (viz IBSS v sekci 2.1.1). V případě, že jsou obě hodnoty rovny 1, jedná se o přenos v rámci DS.

ToDS	**FromDS**	**Adresa #1**	**Adresa #2**	**Adresa #3**	**Adresa #4**
0	0	CA	ZA	BSSID	—
0	1	CA	BSSID	ZA	—
1	0	BSSID	ZA	CA	—
1	1	AP	AO	CA	ZA

odpovídat bezpečnosti drátových sítí — *Wired Equivalent Privacy (WEP)*. To ovšem bylo během krátké doby (v roce 2001) prolomeno a dnes je považováno za nedostatečné [14, 18, 23]. V následujících letech IEEE publikovalo nové dodatky ke standardu 802.11, které představily *Wi-Fi Protected Access* ve třech verzích, jehož druhá verze označovaná jako *WPA2* je dnes nejrozšířenějším typem zabezpečení Wi-Fi [47, 31].

V následující části jsou rozebrána zabezpečení definovaná ve standardu 802.11 a jeho dodatcích a zároveň jsou certifikačními programy Wi-Fi aliance s důrazem na Wi-Fi Protected Access. Vztah mezi zabezpečeními WEP, WPA a WPA2 je znázorněn na obrázku 2.8

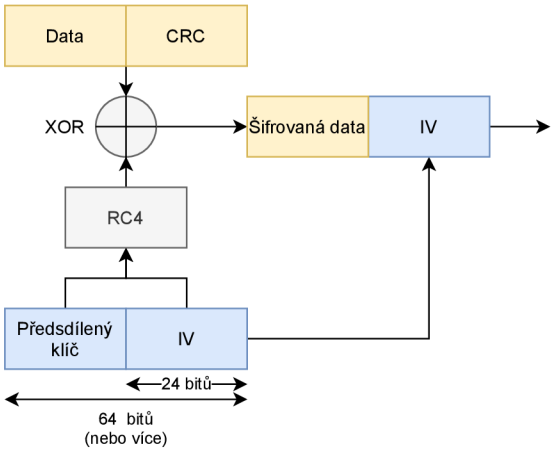


Obrázek 2.8: Znázornění vztahu původního standardu 802.11 a jeho dodatku 802.11i s jednotlivými verzemi zabezpečení WEP, WPA a WPA2. Průnik obou standardů symbolizuje zabezpečení WPA, jež stále využívá princip použitý v zastaralém WEP, ale zároveň ho rozšiřuje o TKIP.

2.3.1 Wired Equivalent Privacy

Wired Equivalent Privacy (WEP) je prvním bezpečnostním algoritmem pro Wi-Fi sítě definovaným v původním standardu 802.11 z roku 1997 [26]. Používá 64bitové nebo 128bitové klíče (existují i varianty s delšími klíči). V 64bitové verzi klíče byla po odečtení 24bitového inicializačního vektoru délka samotného RC4 klíče pouhých 40bitů [26, 2]. Vizualizace zabezpečení WEP je na obrázku 2.9.

Největší známá slabina spočívá v proudové šifře RC4 a velikosti inicializačního vektoru, který má délku 24 bitů. Kryptoanalýza publikovaná v roce 2001 ukázala, že na použitou implementaci šifrování RC4 lze uskutečnit pasivní útok pro získání klíče z odposlechnutých dat [14]. V reakci na tento fakt došlo v roce 2004 k vyřazení WEP z doporučovaných zabezpečení vydáním dodatku 802.11i [27].



Obrázek 2.9: Schéma šifrování použité ve WEP (Wired Equivalent Privacy) zabezpečení. Data v otevřené podobě spolu s jejich kontrolním součtem jsou xorována s výstupem proudové šifry RC4, jež je inicializována předsdíleným klíčem a inicializačním vektorem (IV). Výstupem jsou šifrovaná data a použitý inicializační vektor pro následné dešifrování.

2.3.2 Wi-Fi Protected Access

Po prolomení WEP přišla Wi-Fi aliance s řešením v podobě *Wi-Fi Protected Access (WPA)* a rychle následovanou druhou verzí WPA2 založených na revizích dodatku 802.11i publikovaném v roce 2004 [27]. V roce 2018 Wi-Fi aliance oznámila třetí verzi — WPA3 [21]. Ovšem i v těchto řešeních se stále vyskytují slabiny a stále se objevují nové způsoby útoků na tyto bezpečnostní protokoly [39, 31, 37].

Varianty Personal a Enterprise

Existují dvě varianty WPA — Personal (v překladu *osobní*) a Enterprise (v překladu *firemní*). Jejich názvy napovídají cílovou oblast použití. Hlavní rozdíl spočívá v typu autentizace [36]. V prvním případě Personal se uživatel autentizuje předem známým klíčem — Pre-shared Key (PSK). Z tohoto klíče se poté během čtyřfázové výměny klíčů vygenerují další, které jsou následně použity pro samotné šifrování komunikace.

V Enterprise módu autentizace probíhá podle standardu 802.1X a vyžaduje v síti dostupný autentizační server — například RADIUS [36]. Klíče se generují na základě parametrů v EAPoL rámcích [36].

Typy klíčů používaných WPA

WPA využívá hned několik typů klíčů, ať už pro ustavení dočasných klíčů, zachování integrity rámců, nebo pro samotné šifrování přenosu. Následující definice klíčů vychází ze standardů 802.11i [27].

Pre-Shared Key — PSK — je předsdílený klíč, který musí znát přístupový bod i autentizující se stanice. Jedinou neznámou je v tomto případě heslo. Pro výpočet tohoto klíče je použita kryptografická funkce PBKDF2 (Password-Based Key Derivation Function) druhé verze.

```
PSK = PBKDF2(heslo, SSID, 4096, 256)
```

Pairwise Master Key — PMK —- je klíč, který se nikdy nepřenáší po síti. Jde o hlavní klíč unicastové komunikace, který se pro jednotlivé sezení nemění. Z hlavního klíče jsou derivovány další pro jednotlivá autentizovaná sezení. V režimu *Personal* je PMK roven PSK [27]. Při použití 802.1X EAP se PMK derivuje z AAA klíče — přesněji z jeho prvních 256 bitů [27].

Pairwise Transient Key — PTK — je klíčem vygenerovaným pro aktuální autentizované sezení. Stanice i přístupový bod si během čtyřfázové výměny klíčů sestaví tento PTK. Ilustrace formátu klíče, kde PRF symbolizuje pseudo-náhodnou kryptografickou funkci:

```
PTK = PRF(PMK, ANonce, SNonce, MAC(AP), MAC(STA))
```

Vztah mezi klíči PTK, PMK a PSK je ilustrován na obrázku 2.10. Z PTK se následně derivují tři typy klíčů, které jsou použity pro samotné šifrování unicastové komunikace nebo ověření integrity rámců [11].

EAPoL-Key Confirmation Key — KCK — je klíč sloužící ke kontrole integrity EAPoL-Key rámce. Pomocí tohoto klíče jsou následně generovány kódy integrity zprávy *(Message Integrity Code — MIC)*. Ilustrace formátu klíče:

```
KCK = PTK[0, 128]
```

EAPoL-Key Encryption Key — KEK — je klíč k šifrování pole *Key data* vyskytujícího se v EAPoL-Key rámci. Ilustrace formátu klíče:

```
KEK = PTK[128, 128]
```

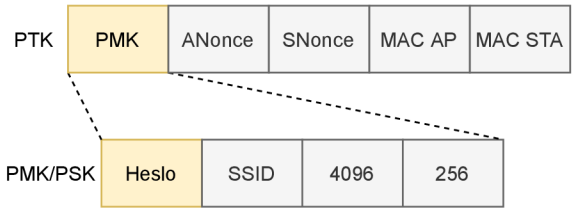
Temporal Key — TK — je klíč pro šifrování datových (nebo i jiných) rámců během autentizované komunikace. Klíč TK začíná na 256. bitu PTK a jeho délka je proměnnou `délka`. Obvyklou délkou klíče je 128 bitů nebo 256 bitů. Ilustrace formátu klíče:

```
TK = PTK[256, délka]
```

Group Master Key — GMK — je klíč obdobných vlastností jako PMK, kde GMK je hlavním klíčem pro multicast a broadcast komunikaci. Tento klíč je generován na straně autentizátoru.

Group Transient Key — GTK — je klíč pro šifrování multicast a broadcast komunikace v síti. Tento klíč je obnovován i během autentizovaného sezení. Při každém odpojení některé ze stanice od sítě se aktualizuje GTK. Klíče GTK se oproti PTK přenášejí po síti, jelikož mohou být již šifrovány právě pomocí PTK (respektive TK derivovaného z PTK). Z GTK se následně derivuje pouze TK. Formát GTK generovaného z GMK je následovný:

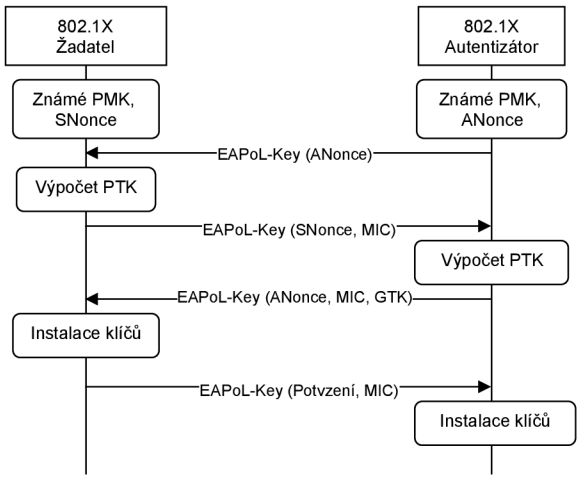
```
GTK = PRF(GMK, MAC(AP), GNonce)
```



Obrázek 2.10: Ilustrace struktury PTK a PMK/PSK klíče a jejich hierarchie. Neznámé hodnoty jsou zvýrazněny oranžovou barvou. Ostatní modře podbarvené hodnoty je možné odposlechnout z komunikace v síti nebo jsou dané přímo standardem 802.11i.

Autentizace

Na počátku komunikace, po asociaci stanice k síti, probíhá čtyřfázové ustavení klíčů pro následné šifrování komunikace [11, 27, 15]. Během této výměny si stanice a přístupový bod vygenerují PTK. Klíč samotný se nikdy nepřenáší přes bezdrátové médium. Obě strany si ho vygenerují na základě přenesených parametrů a PMK, které předem znají. Výměna probíhá pomocí datových rámců protokolem EAPoL typu *EAPoL-Key*. Během výměny si obě strany vymění celkem čtyři zprávy. Během autentizace se autentizující se stanice nazývá *žadatel* (anglicky *supplicant*) a přístupový bod *autentizátor* (anglicky *authenticator*) [23]. V následující části je popsáno ustavení PTK podle jednotlivých zpráv. Pro názornost lze využít obrázku 2.11.



Obrázek 2.11: Průběh ustavení klíčů v zabezpečení WPA/WPA2 [26]

1. AP zahajuje autentizaci zasláním *ANonce* — náhodného (nebo pseudo-náhodného) čísla vygenerovaném AP pro aktuální sezení. Na základě této hodnoty je stanice schopna vygenerovat PTK, jelikož zná všechny parametry pro generování — svoji MAC adresu, MAC adresu AP, PMK, ANonci a SNonci.

2. Stanice odpoví druhou zprávou, ve které zasílá *SNonci* — obdoba ANonce, tentokrát generované stanicí. Od této chvíle přenášené rámce umožňují kontrolu jejich integrity pomocí hodnoty MIC. Ta je spočtena z těla EAPoL rámce pomocí klíče KCK vyderivovaném z PTK.

Po přijetí druhé zprávy přístupovým bodem je AP schopné vygenerovat PTK. Zároveň může ověřit hodnotu MIC, zda nedošlo k porušení integrity EAPoL rámce.

3. Ve třetí zprávě zašle AP GTK, který vygeneruje z GMK, jenž slouží k derivování klíčů pro šifrování multicast a broadcast komunikace v síti. Po přijetí této zprávy žadatel instaluje PTK.

4. Poslední zprávou stanice potvrzuje instalaci dohodnutého klíče a signalizuje AP, že bude používat dohodnutý PTK — instalovat ho.

WPA

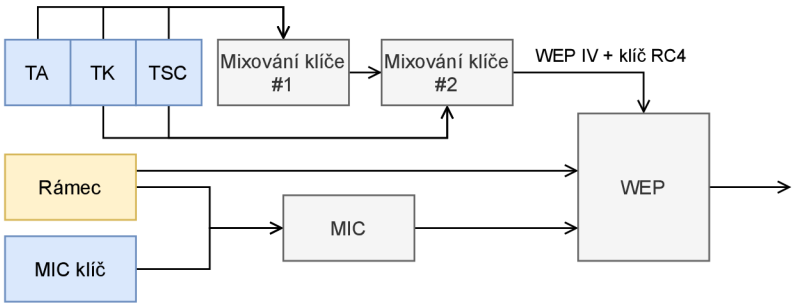
WPA bylo prvním implementovaným zabezpečením z rodiny *Wi-Fi Protected Access*. Je zpětně kompatibilní s hardwarem pro WEP, a bylo tedy možné aktualizací firmwaru nové zabezpečení používat i na starších zařízeních [38]. Kvůli této zpětné kompatibilitě ovšem byly zavedeny kompromisy, které vytvořily slabá místa [38]. Jedním z kompromisů je použití proudové šifry RC4, jelikož pro jiné šifrování by byly nutné změny v hardwaru. Přestože používá nově definovaný protokol Temporal Key Integrity Key (TKIP), je stále možné útočit na slabá místa RC4 [38].

TKIP rozšiřuje zastaralé zabezpečení WEP následujícími body, které mají za cíl zvýšit bezpečnost WEP protokolu [27]:

Integritní kontrola přidáním výpočtu MIC a jeho připojení k rámci.

Ochrana proti útoku přehráním přidáním sekvenčního čítače rámce a povinností příjemce akceptovat pouze rámce s navyšujícím se sekvenčním číslem.

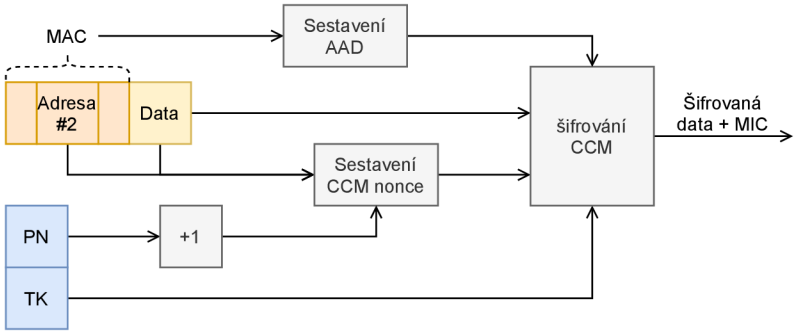
Dvoufázové mixování klíčů zaručuje unikátnost klíče použitého jako klíč RC4 a inicializační vektor WEP IV díky dvojímu použití mixující funkce, jak lze vidět na obrázku 2.12.



Obrázek 2.12: Schéma průběhu šifrování rámce protokolem TKIP. TA značí MAC adresu odesílatele (STA nebo AP), TK dočasný klíč a TSC sekvenční číslo rámce. Pro samotné šifrování je využito zabezpečení WEP popsané v sekci 2.3.1. Fragmentování a některé hodnoty v prvku *rámec* jsou záměrně vynechány pro názornost. Ilustrace je inspirována obrázkem 43c ze standardu 802.11i-2004 [27].

WPA2

WPA2 standard nahrazuje WEP a WPA verze 1. WPA2 bylo definováno ve finálním vydání standardu IEEE 802.11i-2004 [27]. Od roku 2006 je Wi-Fi Aliancí vyžadována implementace WPA2 u všech certifikovaných zařízení [46]. WPA2 odpovídá specifikaci *Robust Security Network (RSN)* popisovaný standardem. Oproti WEP a WPA využívá protokol CCMP[2] založený na šifrování AES[3] [1]. Tento protokol zaručuje důvěrnost, autenticitu a integritu rámce [27]. Průběh šifrování pomocí CCMP je znázorněn na obrázku 2.13.



Obrázek 2.13: Schéma průběhu šifrování rámce protokolem CCMP. PN značí pořadové číslo rámce, AAD značí vybraná pole z MAC hlavičky, pro která je zaručena kontrola integrity. Ilustrace je inspirována obrázkem 43o ze standardu 802.11i-2004 [27].

WPA3

Nejnovější verze WPA3 byla představena v roce 2018. Hlavní změnou oproti předchozím verzím je odstranění PSK a zajištění dopředné bezpečnosti. WPA3 vychází z aktualizovaného standardu 802.11-2016.

WPA3 vynucuje implementaci standardu 802.11w, čímž znemožňuje podvržení řídicích rámců, jako například deautentizační rámce k odpojení stanic od sítě [28].

2.3.3 Wi-Fi Protected Setup

Wi-Fi Protected Setup (WPS) je protokol specifikovaný společností Wi-Fi Alliance v roce 2007 [44] za účelem zjednodušit konfiguraci WPA/WPA2 zabezpečení bez nutnosti znalosti technických detailů. Tento protokol nevychází ze standardu 802.11 ani jeho dodatků. K dnešnímu dni spousta zařízení na trhu jeho podporu implementuje včetně preventivních ochran proti zranitelnostem popisovaných v sekci 3.2.

Při použití WPS uživatel nemusí znát samotné WPA/WPA2 heslo. To je v případě úspěšné WPS autentizace přeneseno pomocí ustaveného dočasného šifrovaného spojení. Odpadá tedy nutnost uživatele znát dlouhé a bezpečné heslo pro WPA/WPA2 autentizaci a zároveň to řeší častý problém s nastavováním slabých hesel neinformovaných uživatelů. WPS autentizace probíhá pomocí jedné z následujících metod:

Stlačení tlačítka (anglicky *Push Button*) metoda Během autentizace pomocí stlačení tlačítka je vyžadován fyzický přístup k WPS autentizátoru (většinou routeru). Po zmáčknutí tlačítka na WPS autentizátoru router akceptuje všechny stanice, které

[2]CCMP — Counter Mode Cipher Block Chaining Message Authentication Code Protocol
[3]AES — Advanced Encryption Standard

projeví zájem o připojení. Doba, po kterou WPS autentizátor akceptuje žádosti stanic, se může lišit.

USB metoda Zřídka používaná metoda, oficiálně již nepodporovaná a netestovaná. Data mezi stanicí a přístupovým bodem se přenášejí fyzicky na USB paměťovém úložišti.

NFC metoda Podobně jako USB metoda ani komunikace pomocí NFC pro předání dat mezi stanicí a přístupovým bodem není často využívána. Její implementace není vyžadována pro získání certifikace WPS.

PIN metoda PIN metoda je z pohledu bezpečnosti stěžejní, jelikož je to jediná metoda, při které není třeba fyzický přístup k WPS autentizátoru či jiné WPS autoritě a zároveň musí být vždy implementována. Během autentizace pomocí WPS PIN se na straně přístupového bodu ověřuje předem známý osmimístný PIN, který musí autentizující se stanice znát předem.

Kapitola 3

Zranitelnosti Wi-Fi sítí a známé útoky

Přestože existuje spousta dodatků k základnímu standardu 802.11, stále v nich existují zranitelnosti, které lze zneužít ve prospěch útočníka. Tyto útoky většinou využívají slabě popsaných částí standardu.

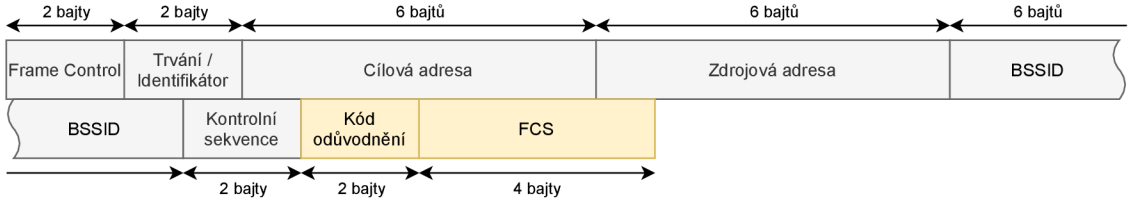
Nejpoužívanějším zabezpečením Wi-Fi sítí je aktuálně *WPA2* [47]. Proto se práce dále zaměřuje převážně na zranitelnosti tohoto zabezpečení. WEP a WPA první verze jsou považovány za nedostatečně bezpečné [41].

3.1 Zneužití deautentizačních rámců

Deautentizační rámce jsou podtypem řídicích rámců [26]. Slouží k ukončení autentizovaného spojení stanice s přístupovým bodem. Jejich zranitelnost spočívá v otevřené, nešifrované formě řídicích rámců [26]. Útočník tedy zná všechny potřebné hodnoty pro vygenerování falešného deautentizačního rámce a může ho zaslat s podvrženou identitou reálné stanice, případně přístupového bodu [33]. Dle definice ve standardu 802.11 je připojené zařízení deautentizováno a pro obnovení komunikace v síti se musí znovu autentizovat [26, 33]. V rámci dodatku 802.11i je navíc definováno, že každá stanice, která je deautentizována, se musí zároveň i disociovat s danou sítí [27]. Samotným opakovaným zasíláním falešných deautentizačních rámců může útočník docílit Denial of Serivce — DoS útoku [21]. Formát deautentizačního rámce odpovídá obecné definici řídicích rámců 2.2.3. Tělem deautentizačního rámce je kód důvodu deautentizace (anglicky *reason code*) [26].

Deautentizační útok

Pro zneužití deautentizačního rámce je stěžejní cílová adresa, zdrojová adresa a BSSID. Všechny tři hodnoty může útočník odposlechnout z komunikace mezi zařízením a přístupovým bodem, jelikož MAC hlavičky rámců nejsou šifrovány [21]. Pro uskutečnění deautentizačního útoku tedy stačí sestavit podvržený rámec s cílovou MAC adresou stanice, zdrojovou MAC adresou přístupového bodu a BSSID. Na kódu důvodu nezáleží, jelikož jediným cílem útočníka je přerušení autentizovaného spojení mezi AP a STA oběti. Formát deautentizačního rámce je znázorněn na obrázku 3.1.



Obrázek 3.1: Formát deautentizačního rámce.

Obrana proti zneužití deautentizačních rámců

Dodatek 802.11w definuje šifrování některých řídicích rámců využitím klíčů, které jsou během komunikace již k dispozici [28]. Tím je útočníkovi znemožněno podvržení deautentizačních rámců, které jsou tímto dodatkem šifrovány.

Problém skrytého uzlu

Wi-Fi sítě uplatňují protokol CSMA/CA (Carrier Sense Multiple Access/Collision Avoidance) [26]. Na základě tohoto protokolu zařízení, která chtějí vysílat, nejprve naslouchají, zda v síti probíhá komunikace mezi některými z připojených zařízení. Pokud žádnou komunikaci nedetekují po určitý čas, předpokládají, že je médium volné a zahájí komunikaci. Problém v bezdrátové síti nastává v případě, že z pohledu zařízení iniciujících komunikaci existuje v síti skrytý uzel — jiné zařízení v síti, které není v dosahu vysílajícího zařízení [18]. Pokud obě zařízení začnou vysílat ve stejnou chvíli, dojde ke kolizi.

Standard 802.11 tento případ řeší volitelným rozšířením sítě o rámce *žádost o odeslání* (RTS, z anglického *request to send*) a *volno k odeslání* (CTS, z anglického *clear to send*) [26]. Rámec RTS zasílá stanice, která detekuje volné médium a chce zahájit komunikaci s jiným zařízením. Pokud cílová stanice nebo přístupový bod nejsou obsazeny a mají kapacitu pro zahájení komunikace, odpoví rámcem CTS. Všechny RTS/CTS rámce přijímají všechny připojené stanice v síti. Okolní stanice po obdržení RTS rámce přeruší vysílání. Stejně tak učiní všechny stanice, které následně obdrží rámec CTS. Formát těchto kontrolních rámců je znázorněn na obrázku 2.5.

Útok pomocí RTS/CTS zpráv

Kontrolní rámce nejsou šifrovány [26]. Pro útočníka není tedy problém sestavit vlastní zprávu s cílovou MAC adresou stanice, která je připojená do některé z okolních sítí, aniž by musel mít do sítě sám přístup. Potom může útočník opakovaně zasílat RTS zprávy s maximální hodnotou NAV například s AP jako svým cílem [32]. AP má ve svém dosahu všechny připojené zařízení, a odesláním rámce CTS je tedy umlčí. Tím je útočník schopen aktivně blokovat přístup ostatním zařízením k médiu. Jde tedy o útok typu Denial of Service [32].

3.2 Útok na WPA/WPA2 odposlechnutím počátečního ustavení klíčů

Zabezpečení WPA/WPA2 spoléhá na bezpečné ustavení klíčů pro šifrování přenášených dat na počátku komunikace. Ustavení klíčů ovšem probíhá pomocí nešifrovaných datových rámců protokolem EAPoL, a je tedy možné tuto komunikaci odposlechnout a dále s ní pracovat [27, 3]. Jednou z překážek pro útočníka může být, že toto ustavení klíčů probíhá

pouze při připojení zařízení do sítě a následně se po dobu připojení neopakuje. Útočník tedy musí čekat, dokud k této vyměně nedojde samovolně bez jeho přičinění. Tuto překážku však může eliminovat zneužitím deautentizačních rámců popsaném v sekci 3.1. Tím donutí zařízení přerušit autentizované spojení a pro další komunikaci musí zařízení znovu ustavit klíče s přístupovým bodem. Navíc dle standardu 802.11 se stanice, jež byla deautentizována, pokusí znovu autentizovat. Útočník tak může předpokládat, že zařízení ustavení klíčů zopakuje automaticky během krátké chvíle po jeho deautentizaci.

Získáním alespoň dvou zpráv z ustavení klíčů je útočník schopen útokem silou získat PMK a PTK. Se znalostí PTK je útočník schopen dešifrovat komunikaci zařízení oběti (během daného autentizovaného sezení), aniž by měl sám přístup do sítě [4]. Získáním PMK naopak v případě sítí v režimu *Personal* získá útočník heslo k síti — PSK — a může se následně do sítě autentizovat [1].

Prolomení odposlechnutého ustavení klíčů

Z odposlechnutého ustavení klíčů je možné útokem silou sestavit PTK, ze kterého jsou následně derivovány klíče pro samotné zabezpečení rámců — šifrováním, kontrolními součty atp. Formát PTK je popsán v sekci 2.3.2. Všechny hodnoty, kromě PMK jsou útočníkovi známy z odposlechnutého handshake. Prolomení probíhá hádáním PMK a opakovaným sestavováním PTK. Pro ověření, zda vygenerovaný PTK je tím hledaným, lze vyderivovat *Key Confirmation Key (KCK)* z PTK a následně pomocí něj spočítat MIC druhé zprávy ustavení klíčů přenášející SNonci a porovnat s MIC v originálním rámci [4]. Pokud se MIC shodují, jde o hledané PTK. V opačném případě pokračuje v hledání. Vzhledem k tomu, že WPA2 vyžaduje minimálně osmiznakovou bezpečnostní frázi, její prolomení útokem silou je časově a výpočetně náročné. Proto se k prolomení fráze používá slovníkový útok cílící na použití slabých hesel.

3.3 Útok na PMKID

Jedním z novějších útoků popsaném v roce 2018 je útok na PMKID. Využívá volitelného rozšíření RSN Information Element v EAPoL rámcích z dodatku 802.11i [37]. Jedinou podmínkou je, aby AP mělo aktivovanou službu roaming [37]. Tvar PMKID lze popsat následovně:

```
PMKID = HMAC-SHA1(PMK, MAC(AP), MAC(STA))
```

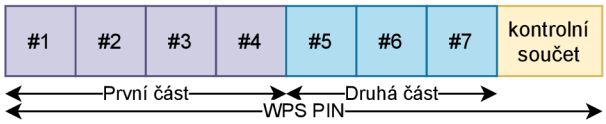
Výhodou tohoto útoku je, že není nutné odposlechnout čtyřfázovou WPA/WPA2 autentizaci. To tedy znamená, že pro úspěšný útok nemusí být v síti žádné připojené zařízení [37], což je podstatný rozdíl oproti klasickému útoku na ustavení klíčů WPA/WPA2.

3.4 Útok silou na WPS PIN

Přestože základní princip protokolu WPS při použití metody PIN lze považovat za bezpečný, jelikož osmimístný číselný PIN kód vytváří 10^8 možných kombinací, samotná implementace ověření zadaného PIN tuto hodnotu mnohonásobně snižuje, a vytváří tak zranitelné místo pro celý protokol [40]. Dle specifikace WPS je během ověřování zadaného osmimístného PIN tento PIN rozdělen na dvě části stejné velikosti. V první fázi tedy *registrující* strana ověřuje validitu prvních čtyř číslic. V případě neúspěchu je protistraně odeslán zamítající zpráva *NACK*. Pokud je první fáze úspěšná, postoupí registrující strana k validování druhé

části PIN. Ta je ovšem dále rozdělena na tři číslice, které jsou součástí PIN, a poslední osmou číslici, která je kontrolním součtem předcházejících sedmi číslic. [44]. Toto rozdělení PIN je ilustrováno obrázkem 3.2.

Pro útok silou to tedy znamená postup ve dvou fázích, kdy v prvním fázi je hledána číselná kombinace v prostoru 10^4 a v druhé fázi z prostoru 10^3. Dohromady jde tedy o $10^4 + 10^3 = 11000$ kombinací. Za předpokladu, že registrující strana neimplementuje žádné ochranné mechanismy, lze takový prostor projít během 4 hodin [40].



Obrázek 3.2: Ilustrace formátu osmimístného WPS PIN. Ten je během validace rozdělen na tři části, které jsou validovány sekvenčně — první část zahrnuje první čtyři číslice PIN, druhá část pouze tři číslice a poslední pozice je hodnota kontrolního součtu předchozích dvou částí.

3.5 Útok KRACK

KRACK (zkratka anglického *Key Reinstallation Attack*), poprvé popsaný v roce 2017, využívá nejasně specifikovaných částí standardu 802.11i [39]. Podle tohoto standardu po obdržení třetí zprávy WPA/WPA2 ustavení klíčů stanicí má stanice nainstalovat klíč vygenerovaný po přijetí první zprávy od přístupového bodu. Na obrázku 2.11 lze vidět tuto událost v kontextu celého ustavení klíčů. Podmínka této definice je ovšem splněna, i pokud stanice obdrží třetí zprávu v době, kdy už má ustavené klíče a probíhá šifrovaná komunikace. Poté dojde k přeinstalování stejného klíče a zároveň k obnově kryptografických parametrů [39].

Jelikož zprávy posílané během ustavení klíčů implementují ochrany proti útoku přehráním a podvržení rámce výpočtem *Message Integrity Code (MIC)*, čítačem opakování a noncí, nelze tento útok provést sestavením podvrženého rámce (jelikož by nebylo možné bez znalosti WPA/WPA2 klíčů spočítat MIC). Pro uskutečnění KRACK útoku je tedy nutné vytvořit AP v pozici *Man in the Middle*. To pro bezdrátové sítě Wi-Fi znamená spustit podvržené AP simulující reálné AP cílové sítě, ale na jiném kanálu. Poté pokud se některá stanice snaží komunikovat s pravým AP, rámce pro a od stanice budou přenášeny přes podvržené AP. Toho lze využít a pozdržet třetí zprávu handshake.

Ve starších verzích operačního systému Android a linuxového nástroje *wpa-supplicant* byl tento útok znásoben chybnou implementací, kdy klíč byl vynulován a poté při přeinstalování došlo k použití kryptograficky slabého klíče složeného ze samých nul.

3.6 Zranitelnost Kr00k

Jedna z nejnovějších publikovaných zranitelností týkajících se Wi-Fi sítí je zranitelnost nazvaná *Kr00k*, která byla zveřejněna v roce 2019. Popisuje zranitelnost objevenou v čipech Broadcom a Cypress [31]. Ta spočívá v chybném návrhu těchto čipů, respektive v implementaci jejich odesílací vyrovnávací paměti. Při disociaci stanice od sítě dojde k vymazání uloženého klíče pro aktuální autentizované sezení. Tím dochází k šifrování zbylých přenášených dat slabým kryptografickým klíčem složeným ze samých nul (podobně jako

u KRACK). Pokud jsou v té době v odesílací vyrovnávací paměti ještě rámce k odeslání, pravě tyto rámce jsou šifrovány slabým klíčem.

Kapitola 4

Čipy Espressif Systems

Čipy ESP32 jsou levné a snadno dostupné systémy na čipu — *SoC* (z anglického výrazu *System on Chip*) s nízkou spotřebou energie, integrovaným bezdrátovým rozhraním Wi-Fi a Bluetooth, vyvíjené společností Espressif Systems. Čipy jsou obvykle integrovány v modulech a vývojových deskách. Moduly jsou certifikovány Wi-Fi certifikací [42, 43].

První část kapitoly se věnuje architektuře čipů a variantám, ve kterých jsou běžně distribuovány — samotné čipy, moduly a vývojové desky. Následně je popsána specifikace čipů ESP32. Druhá část kapitoly popisuje oficiální vývojový aplikační rámec *Espressif IoT Development Framework* pro programování čipů ESP32. Závěr kapitoly se zaměřuje na možnosti programování a práce s bezdrátovým rozhraním Wi-Fi s ohledem na dříve diskutované zranitelnosti standardu 802.11 a možné implementace útoků na těchto čipech.

4.1 Obecná charakteristika čipů Espressif

Čipy Espressif jsou běžně dostupné ve třech variantách [24, 7]. První variantou jsou samotné čipy SoC. Tato varianta předpokládá, že zákazník si zařídí všechny potřebné certifikace a integrace sám. Další dvě varianty jsou popsány v následující části. Na obrázku 4.1 jsou znázorněny všechny tři varianty.

Moduly Moduly integrují samotný čip a některé základní, klíčové komponenty jako například krystalový oscilátor nebo anténový obvod. Moduly jsou připravené pro integraci do produktů a jsou schválené FCC a Wi-Fi aliancí [24, 42, 43].

Vývojové desky Moduly se následně běžně integrují na vývojové desky. Tyto desky usnadňují prototypování a obsahují všechny potřebné periferie pro naprogramování čipu — většinou USB rozhraní. Existuje mnoho variant s různými kombinacemi periferií podle cílového užití [9]. Příkladem může být *ESP32-DevKitC* považovaný za vstupní model [9].

4.2 Specifikace ESP32

ESP32 je systém s dvoujádrovým 32bitovým procesorem harvardské architektury Xtensa LX6 [5]. Procesor může být taktován až na 240MHz s výkonem až 600 DMIPS [13]. Implementuje TCP/IP a 802.11b/g/n Wi-Fi MAC protokol [7]. Wi-Fi rozhraní je tedy schopno komunikovat pouze v pásmu 2,4 Ghz (802.11ac) [7]. Wi-Fi podporuje režim STA a SoftAP

Obrázek 4.1: Varianty distribuce Espressif čipů. Zprava — samostatný SoC *ESP32-D0W0-V3*, modul *ESP32-WROOM-32* s integrovanou PCB anténou, vývojová deska *ESP32 DevKitC* s modulem *ESP32-WROVER* a USB rozhraním. Použité obrázky jsou převzaty z webové prezentace Espressif Systems [10].

(Softwarový AP). Díky podpoře čtyř virtuálních Wi-Fi rozhraní umožňuje ESP32 simultánně provozovat režim STA, SoftAP a promiskuitní režim [7]. ESP32 je schopen vysílat výkonem až 20.5 dBm, přičemž vysílací výkon je softwarově nastavitelný [7].

4.3 Espressif IoT Development Framework

Espressif IoT Development Framework (ESP-IDF) je oficiálním aplikační rámcem pro vývoj aplikací pro platformu ESP32 a ESP32-S [6]. Využívá programovací jazyk C a C++. Umožňuje práci s čipem na nízké úrovni.

4.3.1 Systém sestavení programu

Systém sestavení programu (anglicky *build system*) je v případě ESP-IDF nadstavbou nad existujícím systémem *CMake* [6]. Pro zachování zpětné kompatibility je stále podporováno i sestavování programu pomocí staršího systému *Makefile*, v aktuálních verzích již nahrazeného právě CMake [6].

ESP-IDF poskytuje v rámci projektu systém komponent. Komponenty jsou samostatné části projektu, které jsou během sestavování programu zkompilovány do statických knihoven a následně linkovány s programem [6]. Některé komponenty jsou přímo součástí ESP-IDF, jiné lze dodat externě. V rámci projektu jsou umístěny v adresáři `components`, ve kterých mají vlastní podadresář obsahující zdrojové soubory, konfigurační soubor pro CMake `CMakeLists.txt` a konfigurační soubor samotné komponenty.

Výjimkou je speciální hlavní komponenta `main`, která se musí nacházet v každém projektu v kořenovém adresáři projektu. Sestavovací systém přistupuje k této komponentě odlišně. Oproti ostatním komponentám nemusí mít hlavní komponenta specifikované závislosti na jiných komponentách, ale automaticky se předpokládá závislost na všech komponentách v rámci projektu [6]. V hlavní komponentě je očekávána funkce `app_main`, která je volána při spouštění programu, a samotný kód projektu spojuje všechny komponenty ve funkční celek. Na obrázku 4.2 je znázorněna obecná struktura projektu s komponentami.

```
- myProject/
        - CMakeLists.txt
        - sdkconfig
        - components/ - component1/ - CMakeLists.txt
                                    - Kconfig
                                    - src1.c
                      - component2/ - CMakeLists.txt
                                    - Kconfig
                                    - src1.c
                                    - include/ - component2.h
        - main/       - CMakeLists.txt
                      - src1.c
                      - src2.c

        - build/
```

Obrázek 4.2: Příklad obecné struktury projektu ESP-IDF se dvěma komponentami. Převzato z dokumentace ESP-IDF [6].

ESP-IDF také poskytuje nástroj `idf.py` pro usnadnění správy projektů [6]. Nástroj zjednodušuje proces sestavování programu příkazem `idf.py build`, nahrávání (anglicky *flashing*) binárních souborů na čip příkazem `idf.py flash` a monitorování sériové linky příkazem `idf.py monitor`. Příkazů poskytuje více, ale není cílem je zde všechny popisovat. Jejich seznam lze získat příkazem `idf.py help`, případně v dokumentaci ESP-IDF [6].

Příkaz `idf.py build` spustí proces sestavení programu a jeho výstupem jsou binární soubory ve formátu *Executable and Linkable Format (ELF)* samotného programu a zavaděče systému — `bootloader.elf`. Tento zavaděč automaticky spustí program po spuštění systému. Následně příkazem `idf.py flash` jsou tyto soubory nahrány do paměti čipu Espressif. Po nahrání je čip restartován (dle instrukcí v konfiguračním souboru projektu `sdkconfig`) a je spuštěn program. Pro čtení výstupu programu se lze připojit přes sériovou linku příkazem `idf.py monitor`.

4.3.2 Knihovna smyčky událostí

Události slouží jako komunikační prostředek mezi komponenty v systému. Komponenta může definovat události, na které jiná komponenta může reagovat spuštěním kódu. V ESP-IDF existují dva typy smyček — uživatelská a výchozí. Výchozí smyčka je speciální variantou smyček událostí převážně pro systémové události, například Wi-Fi [6]. Je ale možné do této smyčky posílat i vlastní události a ušetřit tak režii vytváření nové smyčky, uchovávání jejích handlerů atp. Je to zároveň i doporučený postup pro ušetření místa v paměti [6]. Pro vytvoření výchozí smyčky je třeba nejdříve volat funkci `esp_event_loop_create_default()`. Kterákoliv komponenta potřebuje reagovat na události v této smyčce, může si voláním funkce `esp_event_handler_instance_register()` registrovat vlastní funkci, která je vyvolána po přijetí události. Naopak komponenta, která potřebuje událost zaslat do výchozí smyčky událostí, musí volat funkci `esp_event_post()`. K samotné události může připojit i data, se kterými může příjemce dále pracovat.

4.3.3 Knihovny Wi-Fi zásobníku

Knihovny Wi-Fi zásobníku *WiFi Stack Libraries* jsou oproti ostatním knihovnám/komponentám ESP-IDF uzavřené. Je tedy nutné spoléhat se na veřejné síťové aplikační rozhraní poskytované ESP-IDF. Mezi jinými ESP-IDF Wi-Fi aplikační rozhraní poskytuje odposlouchávání paketů, odesílání 802.11 rámců v omezené míře, STA a SoftAP režim atd [6, 7, 30]. Důvod, proč *WiFi Stack Libraries* nejsou otevřené, nebyl oficiálně oznámen. Jedním z důvodů udávaným vývojáři Espressif na oficiálním fóru repozitáře obsahující *WiFi Stack libraries* je pomalý refaktoring starého kódu a problém s oddělením částí kódu, které mohou být zveřejněny, od částí spadajících pod intelektuální vlastnictví jako například knihovna PHY [20]. Dalším důvodem může být snaha vývojářů zabránit zneužití některých funkcí umožnujících odesílání libovolných rámců. Dle komentáře jednoho z vývojářů na oficiálním fóru ESP32[1] byla funkce umožňující volné odesílání rámců odebrána z ESP8266 SDK právě kvůli potenciálnímu zneužití k deautentizačním útokům, spamování beaconů atp. [19]

Analýza 802.11 rámců

Pro analýzu přenosu dat v okolí je možné přepnout bezdrátové rozhraní ESP32 do promiskuitního režimu. Tento režim se nastavuje funkcí `esp_wifi_set_promiscuous()`. V tomto režimu ESP32 Wi-Fi rozhraní odchytává všechny řídicí, datové i kontrolní rámce. Výjimkou jsou chybové rámce, například CRC chybový rámec [6]. Analýza rámců lze spustit samostatně nebo simultánně s režimem STA, SoftAP nebo kombinací STA a SoftAP.

Aplikace může definovat filtry takto analyzovaných rámců, což vede k optimalizaci výkonu. Slouží k tomu funkce `esp_wifi_set_promiscuous_filter()` pro řídicí a datové rámce a `esp_wifi_set_promiscuous_ctrl_filter()` pro kontrolní rámce. Bez definovaných filtrů jsou zpracovávány pouze datové a řídicí rámce. Kontrolní rámce jsou ve výchozím nastavení ignorovány [6]. Jakmile Wi-Fi rozhraní odchytí paket, volá funkci zpětného volání `wifi_promiscuous_cb_t`. Tato funkce zpětného volání je zpracovávána ve Wi-Fi úkolu. V případě náročného zpracování rámců je vhodné přesunout zpracování do aplikačního úkolu a neblokovat tak Wi-Fi úkol [6].

Odesílání 802.11 rámců

Pomocí funkce `esp_wifi_80211_tx()` je možné odeslat některé typy 802.11 rámců. Jsou to beacon, probe request/response a action rámce a všechny non-QoS data rámce [6]. Jak již bylo zmíněno v úvodu této sekce, oficiální důvod pro omezení určitých typů rámců neexistuje. Existují pouze spekulace na základě odpovědí vývojářů na různých fórech [19, 20].

Manipulace MAC adresy bezdrátového rozhraní

ESP-IDF umožňuje libovolně měnit MAC adresu bezdrátového rozhraní pro STA i SoftAP pomocí funkce aplikačního rozhraní `esp_wifi_set_mac()`. Tuto změnu je třeba provést před samotným spuštěním Wi-Fi rozhraní. Nová hodnota musí splňovat všechny formální parametry MAC adresy. První bit nesmí být 1, jinak je adresa považována za nevalidní [6].

[1] https://esp32.com

Kapitola 5

Zhodnocení současného stavu a existujících řešení

Útoky na Wi-Fi jsou stále aktivní hrozbou, a jak dokazují nově definované útoky [31, 39, 37] z posledních let, stále nebyly nalezeny a popsány všechny zranitelnosti standardu 802.11. Čipy Espressif byly již v minulosti používány pro implementace útoků na Wi-Fi sítě. Na čipu ESP8266 — předchůdci ESP32 — vznikaly projekty implementující útoky využívající možnosti injektování rámců do probíhající komunikace, jež zpočátku nebylo v oficiálním SDK nijak omezeno. Ovšem tyto útoky byly převážně typu odepření služby, jelikož na čipech ESP8266 je omezena velikost vyrovnávací paměti udržující odposlechnuté rámce. Naopak čipy ESP32 nejsou velikostí vyrovnávací paměti limitovány a je tedy možné v promiskuitním režimu odposlouchávat kompletní komunikaci. ESP32 je však záměrně omezeno v možnostech injektování rámců v uzavřených knihovnách Wi-Fi Stack Libraries.

Existujícím řešením se věnuje první část kapitoly. V druhé části jsou diskutovány alternativní možnosti implementací obdobných útoků v rámci ESP-IDF, případně některé další zatím neimplementované útoky.

5.1 Existující řešení

Přestože existují pokusy o útoky s čipy ESP32, většinou jde pouze o demonstrace možností, jak obejít blokující opatření v knihovnách Wi-Fi Stack Libraries, znemožňující odesílání určitých rámců (včetně deautentizačních), nebo naopak pouze o pasivní monitorování komunikace v okolí, či útoky typu odepření služeb. K dnešnímu dni však není známo řešení, které by některý z popsaných útoků v kapitole 3 kompletně implementovalo na některém z Espressif čipů. Následující sekce shrnuje veřejné projekty implementující útoky na Wi-Fi na ESP8266 a ESP32 čipech. Nejde však o kompletní seznam projektů zabývajících se tematikou útoků na Wi-Fi, ale pouze výběrem aktivních projektů souvisejících s tématem této práce.

5.1.1 ESP8266 Deauther

Projekt *SpacehuhnTech/esp8266_deauther*[1] je jedním s nejaktivnějších projektů implementující deautentizační útok (popsaný v sekci 3.1) na čipu ESP8266. Hlavní vlastností ESP8266 Deauther je schopnost zasílat deautentizační rámce a tím znemožnit obětem v síti využívat

[1] https://github.com/SpacehuhnTech/esp8266_deauther

jejích služeb. Mimo to tento projekt implementuje útok zahlcení Wi-Fi bezdrátové médium *beacon* nebo *probe* rámci. Cílem projektu je ukázat, že útoky Wi-Fi sítě jsou realizovatelné i s levným a snadno dostupným hardwarem. Tento projekt se ovšem zaměřuje pouze na *Denial of Service* (odepření služby) útoky.

Jako další logické rozšíření tohoto projektu by mohlo být získání WPA/WPA2 autentizačních zpráv. To je ovšem limitováno možnostmi čipu ESP8266. V promiskuitním režimu není možné číst celé datové rámce kvůli omezení velikosti záchytné vyrovnávací paměti, která je omezena na 112 bajtů [8].

5.1.2 ESP32 802.11 Freedom Output

Jelikož do ESP-IDF verze 3.1 nebylo k dispozici aplikační rozhraní pro odesílání vlastních rámců, vznikl projekt *ESP32 802.11 Freedom Output*[2]. Autor projektu reverzním inženýrstvím částečně dekompiloval knihovnu `libnet80211.a`, jež je součástí WiFi Stack Libraries. Na základě informací, které při dekompilování získal, byl schopen použít funkci `ieee80211_freedom_output()`, která byla používána interně k odeslání rámců. Přestože autor projektu v dokumentaci zmiňuje možnost odesílání deautentizačních rámců [12], podle dostupných informací[3] tomu tak není a limitace odesílání deautentizačních (a některých jiných) rámců již existovala ve starších verzích knihoven Wi-Fi Stack Libraries.

V roce 2017 byla do veřejného aplikačního rozhraní ESP-IDF přidána funkce pro odesílání rámců (s určitými limitacemi)[4] `esp_wifi_80211_tx` a tento projekt se stal neaktuálním.

5.1.3 ESP32 Deauther

Projekt *GANESH-ICMC/esp32-deauther*[5] je demonstrací možnosti obejití blokujících mechanismů uvnitř uzavřených knihoven Wi-Fi Stack Libraries. Je z části inspirován projektem ESP8266 Deauther (viz sekce 5.1.1) a na příkladu deautentizačních rámců ukazuje možnost odesílání libovolných rámců i na platformě ESP32 a ESP-IDF.

Autoři projektu použili nástroj *Ghidra*[6] pro reverzní inženýrství vyvíjené americkou organizací NSA, čímž byli schopni částečně dekompilovat knihovny Wi-Fi Stack Libraries [34]. Při zkoumání takto dekompilovaných knihoven objevili volání oné blokující funkce `ieee80211_raw_frame_sanity_check`. Definici této funkce následně změnili ve vlastním programu a předáním parametru `-z muldefs` kompilačnímu linkeru zajistili přepsání samotné funkce [16]. Tento parametr umožňuje ignorovat chybové hlášení během kompilace a dovolí zkompilovat program i přesto, že existuje více definic pro jednu funkci. Linker poté použije první definici funkce [16], což v případě kompilace komponenty ESP-IDF je definice právě z tohoto modulu. Díky tomu je možné z funkce `ieee80211_raw_frame_sanity_check` vždy vrátit hodnotu, jež povolí odeslání rámce z vyrovnávací paměti. Přestože se projekt nazývá „Deauther“, ve skutečnosti umožňuje odesílat libovolné rámce, nejen deautentizační.

Limitací může být fakt, že tento projekt vyžaduje použití nástroje *make* pro kompilování celého projektu ESP-IDF. Ten je však již od verze 4.0 považován za zastaralý způsob a je doporučeno použití kompilačního nástroje *cmake* [6].

[2] https://github.com/Jeija/esp32free80211

[3] https://github.com/Jeija/esp32free80211/issues/6

[4] Změna v repozitáři, během které byla přidána podpora odesílání rámců z vyrovnávací paměti — https://github.com/espressif/esp-idf/commit/846b848bfc987389336fe18b229651983dfa783a

[5] https://github.com/GANESH-ICMC/esp32-deauther

[6] https://ghidra-sre.org/

5.1.4 ESP32 WiFi Hash Monster

Další projekt *G4lile0/ESP32-WiFi-Hash-Monster*[7] se naopak zaměřuje na odchytávání EAPoL rámců a PMKID a ukládá je na přítomnou SD kartu. Jde tedy o pasivní analýzu rámců v okolí ESP32 čipu bez implementace aktivní části útoku. Implementace dle slov autora vychází z 90 % z jiného projektu *spacehuhn/PacketMonitor32*[8] [17], který implementuje obecný nástroj pro analýzu 802.11 komunikace v okolí. Jelikož je implementace vázána na vývojovou desku *M5Stack FIRE*, využívající její displej a hardwarové ovládací prvky, je využití tohoto projektu značně limitováno. Další limitací může být implementace v aplikačním rámci Arduino IDE.

5.1.5 ESP32 Marauder

Podobným projektem je *justcallmekoko/ESP32Marauder*[9], který implementuje podobnou sadu útoků jako projekt ESP8266 Deauther. Nástroj poskytuje možnost odchytávání různých typů rámců jako třeba deautentizační, probe a beacon rámce i zprávy WPA/WPA2 autentizace — podobně jako projekt ESP32 WiFi Hash Monster. V rámci tohoto nástroje jsou implementovány i některé útoky injektováním rámců obdobně jako u projektu ESP8266, které však nepodléhají blokování v knihovnách Wi-Fi Stack Libraries. Jsou tedy opět umožněny jen některé útoky odepření služby. Limitací využití tohoto nástroje může opět být vyžadování připojeného displeje a hardwarových ovládacích prvků.

5.2 Zhodnocení současného stavu

Dle zjištění představených v kapitole 4 je zřejmé, že poskytované funkce umožňují implementaci většiny známých útoků díky možnosti měnit adresu MAC pro Wi-Fi rozhraní, injektovat/odesílat vlastní rámce a monitorovat tok dat díky promiskuitnímu režimu. Injektování je však úmyslně omezeno vývojáři Espressif, kdy rámce s potenciální možností zneužití není možné odesílat. Jde právě třeba o deautentizační rámce. Tuto limitaci odstraňuje existující řešení v podobě projektu *esp32-deauther*, který umožňuje odesílání libovolných rámců přepsáním funkce používané k blokování uvnitř uzavřených knihoven Wi-Fi Stack Libraries. Na základě těchto poznatků je zjevné, že pomocí ESP-IDF a platformy ESP32 samotné lze implementovat běžně známé útoky diskutované v kapitole 3.

5.3 Výběr vhodného čipu

Hlavním cílem této práce je ukázat, že je možné uskutečnit útoky na Wi-Fi sítě pomocí čipů Espressif a vývojového aplikačního rámce *Espressif IoT Development Framework*. Tento aplikační rámec je primárně určený pro čipy ESP32 a ESP32-C. Zároveň tyto čipy řady ESP32 jsou považovány za nástupce čipů ESP8266 [30]. Pro potřeby této práce je nejvhodnější variantou zvolit některou z dostupných vývojových desek integrující ESP32 modul umožnující programování čipu přes USB rozhraní. Základní a běžně dostupnou vývojovou deskou je *ESP32-DevKitC* [9]. Tato deska integruje (mimo jiné) modul *ESP32-WROOM-32E* nebo *ESP32-WROOM-32UE*. První zmiňovaný modul integruje PCB anténu (anténu

[7] https://github.com/G4lile0/ESP32-WiFi-Hash-Monster
[8] https://github.com/spacehuhn/PacketMonitor32/
[9] https://github.com/justcallmekoko/ESP32Marauder

integrovanou do obvodu), 4MB SPI flash paměť a další méně podstatné proprietární obvody. Druhý modul namísto PCB antény integruje IPEX rozhraní, umožnující připojit externí anténu [10]. Oba zmiňované moduly integrují třetí verzi dvoujádrového čipu *ESP32-D0WD-V3* s parametry 520KB SRAM a 448KB ROM [10]. Vývojová deska s modulem *ESP32-WROOM-32E* je dostačující pro vývoj a demonstraci systému v laboratorních podmínkách.

Kapitola 6

Návrh řešení

Z předchozích kapitol je zřejmé, že na platformě ESP32 jsou běžně známé Wi-Fi útoky implementovatelné. Proto jsem se na základě tohoto poznatku rozhodl vytvořit univerzální nástroj pro penetrační testování Wi-Fi sítí, který sjednocuje různé typy útoků a jejich variant. Toto řešení může usnadnit přidání implementací dalších útoků. Pro účely demonstrace funkcionality jsem zvolil deautentizační útoky následované odchytáváním WPA/WPA2 ustavení klíčů a PMKID. V této kapitole se věnuji návrhu tohoto nástroje a jednotlivých útoků popsaných v kapitole 3.

6.1 Požadavky na systém

Před návrhem implementace je vhodné definovat požadavky na výsledný systém. Jsou definovány následující funkční požadavky na systém. Dále jsou definovány nefunkční požadavky na systém pro efektivní použití v reálném prostředí.

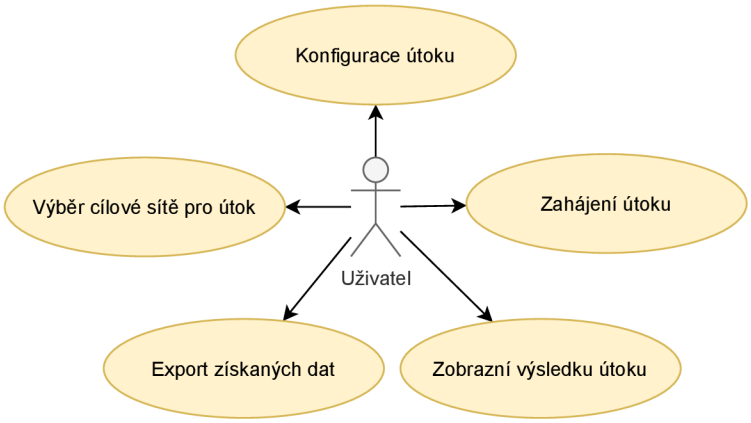
Požadavky na uživatelské rozhraní

Je žádoucí, aby měl uživatel možnost útok konfigurovat, získat informace o úspěšnosti útoku a exportovat případná získaná data. Uživatelské interakce se systémem jsou ilustrovány v diagramu případu užití na obrázku 6.1. Jednotlivé případy jsou v následující části blíže specifikovány.

Výběr cílové sítě pro útok Uživatel musí mít možnost zvolit síť, proti které bude následně útok veden. Uživateli je nabídnut seznam sítí v okolí, identifikovaných podle SSID, a uživatel má možnost vybrat jednu síť.

Konfigurace útoku Uživatel musí mít možnost zvolit typ útoku, který má být proveden. V případě dostupnosti více způsobů, kterými lze takový útok vykonat, musí mít uživatel možnost vybrat jednu z poskytovaných metod.

Zobrazení výsledku útoku Po skončení útoku musí mít uživatel možnost získat informace o výsledku útoku. V případě úspěšného útoku jsou uživateli zprostředkovány informace o úspěšnosti útoku (v závislosti na konfiguraci). V případě neúspěšného útoku musí uživatel získat zpětnou vazbu o důvodu selhání či neúspěchu útoku. Například v případě analýzy rámců a odposlouchávání WPA autentizace nemusela být v době monitorování sítě aktivní žádná stanice, nebo nebylo možné deautentizovat stanice.



Obrázek 6.1: Diagram případu užití z pohledu uživatele. Systém má za cíl klást na uživatele minimální nároky a měl by být co nejjednodušší na konfiguraci.

Export získaných dat Po úspěšném útoku je nutné uživateli zprostředkovat data získaná během útoku pro jejich další zpracování. Odposlechnuté rámce, jež byly předmětem zájmu během útoku, musí být k dispozici ve standardním formátu PCAP. Pro usnadnění následného zpracování získaných dat je v případě předpokládaného použití útoku silou vhodné poskytnout data ve formátu přijímaném nástrojem *Hashcat*.

6.1.1 Požadavky na automatizované provedení útoků

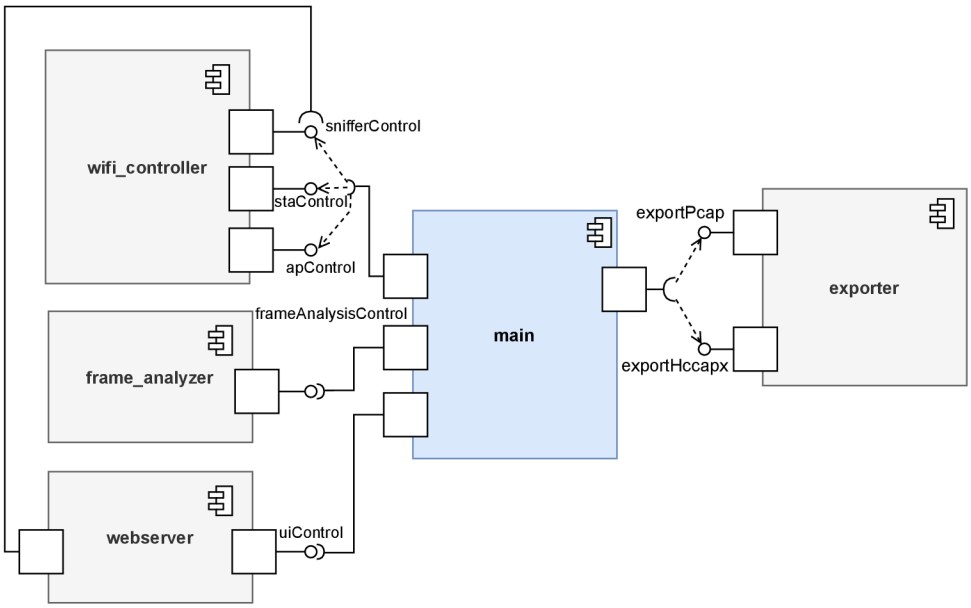
Jedním z nejzásadnějších nefunkčních požadavků je důraz na automatizované provedení útoku bez nutnosti interakce s uživatelem (kromě počáteční konfigurace).

6.2 Návrh architektury

Celý systém se skládá z několika samostatných logických celků. K jejich rozdělení v projektu poslouží systém komponent poskytovaný aplikačním rámcem ESP-IDF. Lze tedy uvažovat pět komponent (včetně speciální hlavní komponenty `main`) znázorněných na obrázku 6.2:

- Hlavní komponenta
- Analyzátor rámců
- Ovladač Wi-Fi
- Webový server
- Export dat

Výhodou rozdělení do komponent je jejich znovupoužitelnost a snadnější rozšíření projektu o nové části, případně využití jednotlivých komponent v jiných projektech. Například komponenta pro export dat může být užitečná i v jiných projektech. Další výhodou může být možnost vyměnit komponentu za jinou (například `webserver`) za předpokladu, že implementuje stejné funkce rozhraní, odpovídající původní komponentě.



Obrázek 6.2: Návrh rozdělení komponent a jejich rozhraní, pomocí kterých spolu komunikují. Smyčky událostí jsou součástí navržených rozhraní.

6.2.1 Uživatelské rozhraní

Přestože útok může být do jisté míry automatizován, stále je žádoucí, aby měl uživatel možnost útok konfigurovat a po úspěšném útoku získat potřebná data pro následný útok silou mimo platformu ESP32.

Před samotným útokem nebo po jeho skončení je tedy možné vytvořit přístupový bod na ESP32, ke kterému se může uživatel připojit. Po připojení může AP poskytnout webový server a na něm dostupné uživatelské prostředí pro konfiguraci a export dat. Implementaci webového serveru a zobrazení webové prezentace poskytne komponenta `webserver`. Sekvenční diagram znázorňující stavy uživatelského prostředí je vyobrazen na obrázku 6.3.

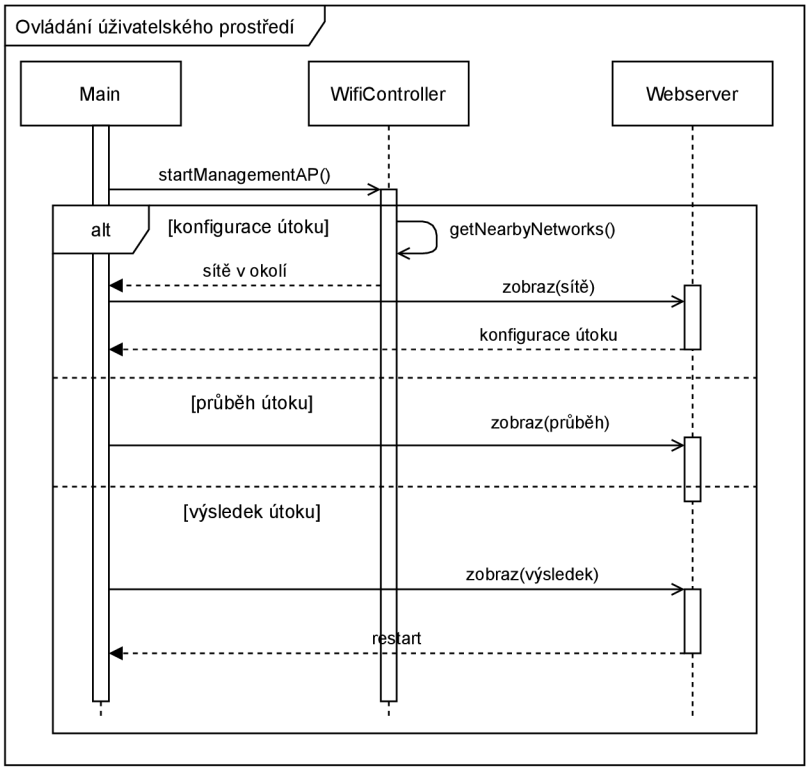
6.3 Návrh útoku k získání PMKID

Pro provedení útoku k získání PMKID lze využít nativní funkcionalitu režimu stanice a iniciovat autentizaci s cílovým AP. Výměnu EAPoL paketů pro ustavení klíčů vždy zahajuje AP, které posílá stanici náhodně vygenerované číslo — *ANonce* — v rámci EAPoL–Key. Součást tohoto rámce ovšem může být i *RSN Information element*, který obsahuje hledané *PMKID*. Diagram znázorňující postup získání PMKID je vyobrazen na obrázku A.1.

Pokud je PMKID nalezeno, je uloženo v paměti a následně zobrazeno uživateli pomocí uživatelského prostředí definovaného v sekci 6.2.1.

6.4 Návrh deautentizačního útoku

Existují dva způsoby, jak odeslat deautentizační rámec na čipu ESP32. První předpokládá využití funkce z uzavřené knihovny Wi-Fi Stack Libraries, ovšem naráží na limitaci zavedenou přímo v těchto knihovnách. Druhá metoda využívá nativní podpory vytvoření vlastního AP s libovolnou MAC adresou. Obě možnosti předpokládají předchozí získání MAC adresy



Obrázek 6.3: Průběh zobrazení uživatelského prostředí pomocí *webserver* komponenty. Komponenta *wifi_controller* vytváří přístupový bod, ke kterému se uživatel může připojit. Po připojení je mu poskytnuta webová stránka, na které má možnost konfigurovat útok, případně získat výsledky posledního útoku.

AP. MAC adresa AP je známá již z počátku, kdy uživatel vybírá cílovou síť pro útok. V následující části sekce jsou oba způsoby blíže popsány.

6.4.1 Metoda rozšířením projektu ESP32 Deauther

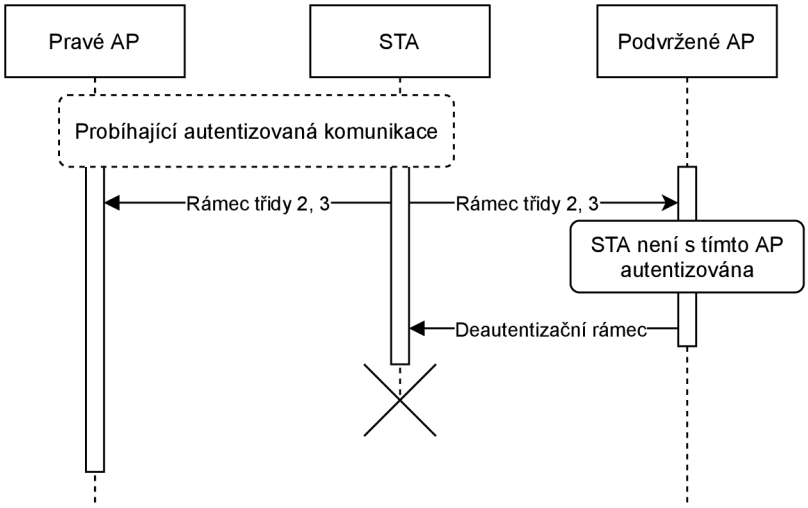
Projekt *esp32-deauther* implementuje řešení obejití kontroly typu rámce a umožňuje jeho následné odeslání. Lze tedy sestavit podvržený deautentizační rámec s vhodně vyplněnými adresami stanice a přístupového bodu.

Výhodou tohoto přístupu může být kontrola nad odesíláním deautentizačních rámců, počtu zaslaných rámců a jejich kadence. Nevýhodou je spolehnutí se na obcházení opatření uvnitř uzavřené knihovny, kterou mohou autoři kdykoliv změnit a tím znefunkčnit tento přístup. Následný průběh interakce komponent pro zaslání deautentizačních rámců je ilustrován na obrázku A.2.

Pro realizaci této varianty je navržena komponenta `DeauthInjector`. V rámci této komponenty je opakovaně zasílán deautentizační rámec sestavený z MAC adresy AP cílové sítě. Ostatní hodnoty jsou známy. Deautentizační rámce jsou zasílány uvnitř úkolu, který je aktivován v pravidelném časovém intervalu. Vhodný časový interval je v rámci sekund, aby nedocházelo k zahlcení sítě deautentizačními rámci. Úkol zasílající deautentizační rámce je vykonáván ve smyčce, dokud není ukončen hlavní komponentou `main`, případně překročením časového limitu definovaném při spuštění této komponenty.

6.4.2 Metoda vytvořením podvrženého AP

Druhou variantou je využít možnosti vytvořit na ESP32 vlastní přístupový bod. V kombinaci s možností nastavení libovolné MAC adresy pro bezdrátové rozhraní lze dosáhnout vytvoření kopie existujícího AP — *podvrženého AP*. Je tedy třeba nastavit MAC adresu ESP32 bezdrátového rozhraní na MAC adresu existujícího AP v síti. Poté jakmile je takovéto podvržené AP spuštěno v blízkosti existujícího pravého AP a některá z autentizovaných stanic odešle rámec třídy 2 nebo 3 přístupovému bodu, tuto komunikaci zachytí pravé AP, ale i podvržené AP.



Obrázek 6.4: Obecný příklad využití podvrženého AP k deautentizaci stanic v síti. V případě, že je vytvořeno podvržené AP na stejném kanálu, se stejnou MAC adresou a parametry pravé sítě, podvržené AP bude odesílat deautentizační rámce, kdykoliv od některé stanice obdrží rámec třídy 2 nebo 3 odeslaný pravému AP. Toho je dosáhnuto využitím definice ve standardu 802.11i, kdy odpoví deautentizačním rámcem, pokud AP obdrží rámec třídy 2 nebo 3 předpokládající existující autentizované spojení [27].

Z pohledu podvržené AP ovšem neexistuje žádná autentizovaná stanice, a proto odpoví deautentizačním rámcem. Deautentizační rámec zaslaný podvrženým AP obdrží stanice, která se pokoušela o komunikaci, a z jejího pohledu nemá možnost detekovat, že jde o rámec z podvrženého AP. Stanice se tedy odpojí od sítě. Průběh útoku pomocí podvrženého AP je ilustrován na obrázcích 6.4 a A.3.

Výhodou tohoto přístupu je, že nevyužívá žádných interních funkcí Wi-Fi knihovny. Nevýhodou může být nemožnost kontroly zasílaných rámců tímto duplikovaným přístupovým bodem.

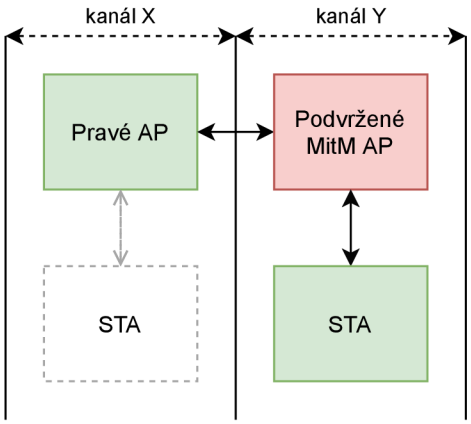
6.5 Návrh Man-in-the-Middle přístupového bodu

Pro některé typy útoků je třeba využít *Man-in-the-Middle (MitM)* podvrženého přístupového bodu, díky čemuž útočník získá částečnou schopnost ovlivnit komunikaci mezi stanicí a pravým přístupovým bodem. U bezdrátového přístupového média standardu 802.11 lze dosáhnout takovéto pozice využitím izolace kanálů. Jelikož stanice i přístupový bod musí vysílat a přijímat na stejném kanálu, je možné vytvořit podvržené AP na jiném kanálu,

než je pravé AP, a stanici přesvědčit, aby komunikovala právě na tomto kanálu. Toho lze docílit díky běžnému způsobu, kterým stanice volí přístupový bod — podle síly signálu.

K vytvoření takto podvrženého AP nelze využít nativní funkcionalitu AP režimu jako u deautentizačního útoku v sekci 6.4. Je třeba simulovat existenci AP pomocí odesílání vlastních rámců pro signalizaci existence takového AP na jiném kanálu. Ke změně kanálu lze využít funkci `esp_wifi_set_channel`. V momentě, kdy stanice zvolí ke komunikaci kanál podvrženého AP, je potřeba monitorovat veškerou komunikaci této stanice a přeposílat ji na původním kanálu pravého AP. To stejné je třeba i opačně — veškerou komunikaci od pravého AP k stanici oběti přeposílat na kanálu podvrženého AP. Tento proces je znázorněn na obrázku 6.5. K monitorování komunikace lze přepnout Wi-Fi rozhraní do promiskuitního režimu funkcí `esp_wifi_set_promiscuous` a následné přeposlání zachycených rámců vykonat pomocí funkce `esp_wifi_80211_tx`. Vzhledem k tomu, že mezi přeposílanými rámci mohou být i takové, které ESP-IDF úmyslně blokují, je pro MitM AP vhodné využít existujícího projektu *esp32-deauther*. Zároveň pro vynucení nové autentizace stanice tak, aby zvolila MitM AP je možné využít deautentizačního útoku. Nedostatkem tohoto řešení může být problém s přepínáním kanálu, jelikož Wi-Fi rozhraní na platformě ESP32 může vysílat a přijímat pouze na jediném kanálu. Je tedy nutné pravidelně mezi kanály přeskakovat, což může vést ke ztrátě některých rámců při záchytu.

Příkladem útoku vyžadujícího AP v pozici MitM může být útok KRACK popsaný v sekci 3.5. Během toho lze pozdržet přeposlání třetí zprávy handshake a odeslat ji později po zahájení autentizované komunikace.

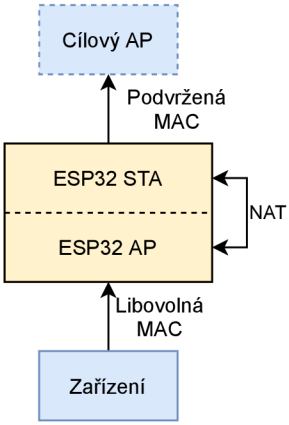


Obrázek 6.5: Ilustrace Man-in-the-Middle pozice podvrženého přístupového bodu. Využívá izolace komunikace mezi jednotlivými kanály, kdy stanice volí AP dle jeho parametrů — většinou podle síly signálu a poskytovaného zabezpečení. Pokud tedy podvržené MitM AP získá důvěru stanice, má poté kontrolu nad veškerou komunikací mezi pravým AP a stanicí.

6.6 Návrh postupu k obejití filtrování MAC adres

Přestože filtrování MAC adres na přístupovém bodu není považováno za zabezpečení jako takové, je občas využíváno jako další vrstva pro zpomalení nebo odrazení útočníka. Přestože dnes má spousta zařízení možnost měnit adresu MAC pro bezdrátové rozhraní Wi-Fi, existují i taková, která tuto možnost neposkytují. Hlavním příkladem mohou být chytré telefony se systémy Android nebo iOS, případně zařízení s vlastním neveřejným firmwarem,

jako například tiskárny. ESP32 lze v tomto případě využít jako prostředníka, který se díky schopnosti fungovat v režimu AP+STA a možnosti měnit adresu MAC pomocí funkce `esp_wifi_set_mac` dokáže připojit k síti a zároveň vytvořit neomezený přístupový bod, ke kterému se mohou připojit zařízení s jakoukoliv adresou MAC. Pro zpřístupnění cílové sítě lze využít komponenty *lwip* a implementace překladu síťových adres (anglicky *NAT*). Toto řešení je ilustrováno na obrázku 6.6. Tento přístup předpokládá znalost hesla k cílové síti, a tedy možnosti použití funkce `esp_wifi_connect` k připojení ESP32 k dané síti.



Obrázek 6.6: Ilustrace možnosti obejití filtrování MAC adres na straně cílového AP a poskytnutí přístupu do sítě zařízením s libovonou adresou MAC. Je zde využito funkcionality překladu síťových adres (NAT) a režimu AP+STA, kdy ESP32 v režimu STA je připojeno k cílové stanici s podvrženou MAC adresou některé z povolených stanic a v režimu AP, které je libovolně konfigurovatelné.

Kapitola 7

Implementace nástroje ESP32 Wi-Fi Penetration Tool

Na základě návrhu univerzálního nástroje pro spouštění útoků na čipech ESP32 byl implementován nástroj *ESP32 Wi-Fi Penetration Tool*. Tento nástroj implementovaný v jazyce C, poskytuje jednotné uživatelské i vývojářské rozhraní pro spouštění a implementaci útoků na Wi-Fi sítě s použitím aplikačního rámce ESP-IDF. Jeho cílem je umožnit vést útoky s co nejmenší potřebou konfigurace a vysokou mírou úspěšnosti. Cílí na nenápadnou obsluhu uživatelem pomocí řídicího přístupového bodu umožňujícího vzdálenou konfiguraci například pomocí chytrého telefonu.

V této kapitole se budu zabývat vlastní implementací navrženého nástroje z kapitoly 6. Jsou představeny jednotlivé komponenty, komunikace mezi nimi a uživatelské prostředí. Následně jsou popsány implementace dvou variant deautentizačních útoků a způsob zpracování odchytávaných rámců pro útoky silou na externích strojích.

7.1 Koncept

Nástroj *ESP32 Wi-Fi Penetration Tool* byl implementován s cílem snadného doplnění nových typů útoků a znovupoužitelností v dalších projektech. K tomu bylo využito systému komponent poskytovaných systémem sestavení programu popsaným v sekci 4.3.1. Sdílené úkony z různých typů útoků byly implementovány v jednotlivých komponentách. Komunikace pro asynchronní události probíhá pomocí nativní komponenty smyčky událostí `esp_event`. Ta zároveň umožňuje i lepší izolaci komponent, jelikož komponenty nemusí přímo předávat data jiným, ale mohou využít právě smyčku událostí. Nástroj si dále klade za cíl být snadno ovladatelný i pro uživatele s minimální znalostí problému. Útoky jsou tedy z velké části automatizovány a po uživateli je vyžadována pouze počáteční konfigurace útoku.

Jednotlivé komponenty vždy obsahují veřejné aplikační rozhraní v hlavičkových souborech uvnitř `interface` adresáře. To umožňuje integrovat komponentu do projektu nezávisle na implementaci ostatních komponent.

Implementace v jazyce C je dokumentována pomocí nástroje Doxygen. Tento nástroj definuje standardní formát pro zápis dokumentačních anotací. Jde o nejrozšířenější způsob dokumentace zdrojových kódů jazyka C [22].

7.2 Architektura

Architektura nástroje je značně ovlivněna systémem komponent, jež ESP-IDF poskytuje (detailněji rozebrán v sekci 4.3.1). Implementace je rozdělena do sedmi komponent, které (kromě hlavní komponenty) cílí na co největší znovupoužitelnost. Každá komponenta tedy poskytuje aplikační rozhraní v podobě hlavičkových souborů v adresáři `interface` s funkcemi a strukturami dostupnými z ostatních komponent. Struktura nástroje vypadá následovně, kde jednotlivé komponenty jsou dále rozebrány detailněji:

- `main`
- `components/frame_analyzer`
- `components/hccapx_serializer`
- `components/pcap_serializer`
- `components/webserver`
- `components/wifi_controller`
- `components/wsl_bypasser`

7.2.1 Hlavní komponenta

Po úspěšném bootu je předáno řízení samotného programu voláním funkce `app_main()` uvnitř hlavní `main` komponenty. Ta inicializuje Wi-Fi rozhraní a spustí přístupový bod pro konfiguraci — *Management AP* a řízení programu přenechá webovému serveru — komponentě nazvané `webserver`.

Attack wrapper

Jelikož útoky mají často společné implementační části, je implementován tzv. *„attack wrapper“*, který tyto části řeší pro všechny útoky. Při přidávání nových typů nebo metod útoků není tedy nutné zabývat se znovu touto implementací. Tento wrapper tedy řeší hlavně předání konfigurace útoku z webového serveru do logiky samotného útoku. Během útoku udržuje jeho aktuální stav, který je k dispozici webovému serveru, a po skončení útoku mu předává veškeré zjištěné informace. Za další také vždy řeší časový limit útoku (timeout), kdy při vypršení tohoto limitu volá příslušné ukončující funkce v závislosti na typu útoku. Tento wrapper tedy nastavuje a udržuje aktuální stav celého stavového automatu ve struktuře `attack_status_t`.

7.2.2 Komponenta webového serveru

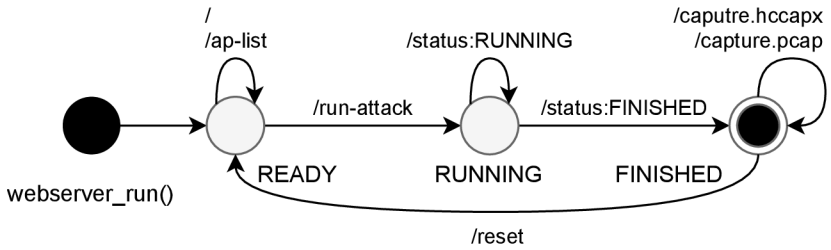
Webový server poskytuje na standardním HTTP portu `80` webové rozhraní pro interakci s uživatelem. K implementaci využívá nativní komponenty `esp_http_server`. Tento webový server poskytuje soubor několika koncových bodů (z anglického *endpoint*), pomocí kterých lze ovládat samotný nástroj. Kořenový koncový bod poskytuje uživateli možnost konfigurace útoku — výběr cílové sítě, typ a metodu útoku a časový limit. Skenování okolních sítí, respektive přístupových bodů zajišťuje koncový bod `/ap-list`, který vrátí seznam přístupových bodů v okolí. Pro zadání konfigurace slouží `POST` koncový bod `/run-attack`, který po validaci zaslané konfigurace spustí daný útok.

Komunikace mezi webovým serverem a klientem je bezstavová — klient nezná aktuální stav a zobrazuje data podle odpovědi na dotaz. Pro ověření aktuálního stavu slouží `GET` koncový bod `/status`. Bezstavová komunikace je nutná, jelikož během útoku může dojít k přerušení autentizovaného spojení s přístupovým bodem (například způsobené restartováním Wi-Fi rozhraní) a klient by přišel o aktuální stav.

Pro návrat do iniciálního konfiguračního stavu slouží `HEAD` koncový bod `/reset`, který vrátí hodnoty do původního stavu.

Ve stavu ukončeného útoku jsou v závislosti na typu útoku k dispozici koncové body `capture.hccapx` a `capture.pcap`, které poskytují zformátovaný výstup útoku v binárním kódu, připravené pro další zpracování externími nástroji — například Wireshark nebo Hashcat.

Přechody stavy nástroje ve vztahu s koncovými body je ilustrován konečným automatem na obrázku 7.1.



Obrázek 7.1: Konečný automat znázorňující změny vnitřního stavu nástroje v závislosti na volaných koncových bodech.

Webový klient

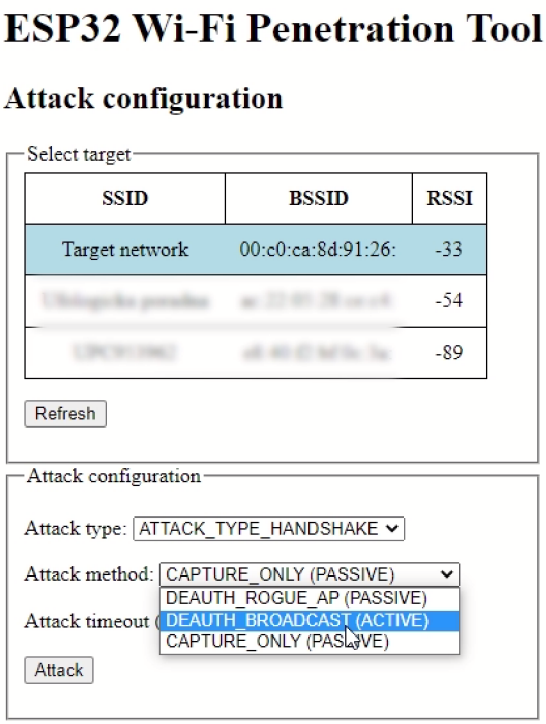
Webové uživatelské prostředí využívá základní prvky HTML5 bez složitějších úprav. Záměrem je snížit velikost webového klienta na co nejmenší paměťový prostor, aby mohl být uložen v RAM. V aktuálním provedení je tedy celá webová aplikace uložena přímo v paměti ESP32 ve statické konstantě. Pro snazší editaci a úpravy byl vytvořen pomocný nástroj `convert_html_to_header_file.sh`, který z běžného HTML souboru vygeneruje hlavičkový soubor jazyka C s jedinou konstantou obsahující celý HTML soubor jako textový řetězec. Na obrázku 7.2 je vyobrazeno uživatelské prostředí ve stavu *READY*.

Po zahájení útoku klient opětovně dotazuje webový server na ESP32 a čeká na koncový stav *FINISHED* nebo *TIMEOUT*. Během tohoto dotazování je uživateli zobrazen zbývající čas do konce časového limitu specifikovaného v konfiguraci útoku. Po přechodu do koncového stavu zobrazí výsledek útoku. Pokud byl klient během útoku odpojen od *Management AP*, je informován o nutnosti znovu se připojit. Jde o očekávaný stav, jelikož pokud ESP32 k útoku potřebuje manipulovat s Wi-Fi rozhraním, dochází k jeho restartování, což může vést k odpojení připojených stanic.

Přechody mezi stavy nástroje jsou ilustrovány konečným automatem na obrázku 7.3.

7.2.3 Komponenta pro export ve formátu HCCAPX

Komponenta `hccapx_serializer` implementuje serializaci dat získaných při útoku na ustavení WPA/WPA2 klíčů. Jednotlivé zprávy ze zachyceného ustavení klíčů jsou zformátovány do formátu HCCAPX. Tento formát je používán nástrojem Hashcat pro obnovu hesla ze



Obrázek 7.2: Na snímku je zachyceno uživatelské prostředí ve stavu *READY*, kdy má uživatel možnost vybrat cílovou síť/přístupový bod a konfigurovat útok, který chce spustit.

zachycené WPA/WPA2 autentizace. Formát je implementován dle oficiální dokumentace HCCAPX[1]. Tato dokumentace však není v některých bodech jednoznačná a úplná, a bylo tedy třeba během implementace reverzním inženýrstvím tato místa v dokumentaci upřesnit. Z dokumentace není jasné, že v přiloženém EAPoL rámci má být vynulována hodnota MIC.

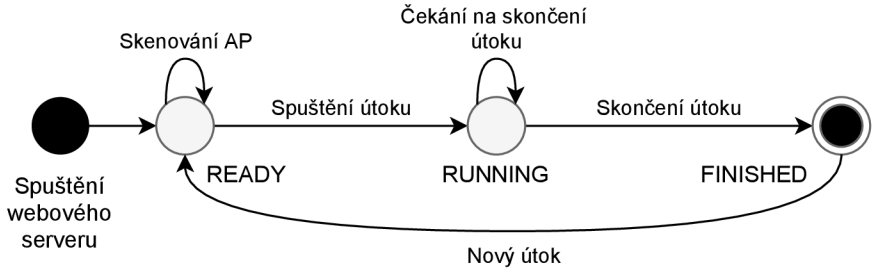
7.2.4 Komponenta pro export ve formátu PCAP

Komponenta `pcap_serializer` implementuje serializaci zachycených rámců do souboru formátu PCAP. Lze kdykoliv inicializovat nový PCAP soubor, který je následně uchováván ve strukturách v paměti. Struktury vycházejí z definice *Libpcap* od Wireshark[2]. Množství uložených rámců záleží pouze na velikosti dostupné paměti. Jednotlivé rámce jsou přidávány do souboru voláním funkce `pcap_serializer_append_frame`, jsou k nim přidána metadata a zařazeny na konec souboru pomocí realokováním paměti — voláním funkce `realloc()`.

7.2.5 Komponenta analýzy rámců

Komponenta analýzy rámců zpracovává odchycené rámce po přepnutí Wi-Fi rozhraní do promiskuitního módu ve `wifi_controller` komponentě. Rámce procházejí filtrací, která je předem nakonfigurována dle typu útoku. V první fázi se filtrují pouze rámce, které pocházejí z komunikace mezi cílovým přístupovým bodem a některou připojenou stanicí podle jejího

[1] https://hashcat.net/wiki/doku.php?id=hccapx
[2] https://wiki.wireshark.org/Development/LibpcapFileFormat



Obrázek 7.3: Končený automat ilustrující změny vnitřního stavu systému v závislosti na událostech.

BSSID. Dále se analýza větví dle typu hledaných dat. Pro získání PMKID anebo zpráv WPA/WPA2 ustavení klíčů jsou bodem zájmu datové rámce přenášející pakety EAPoL. V případě kladného výsledku analýzy, tedy v případě, že byl nalezen hledaný rámec, jsou zpracovaná data předána zpět do smyčky událostí typu `FRAME_ANALYZER_EVENTS`.

Tato komponenta dále poskytuje aplikační rozhraní pro syntaktickou analýzu (neboli parsování) rámců. Například pro získání struktury obsahující EAPoL paket lze volat funkci `parse_eapol_packet()`. Příklad rozkladu EAPoL-Key rámce je ilustrován na obrázku 7.4.

7.2.6 Komponenta ovládání Wi-Fi rozhraní

Vzhledem k faktu, že většina operací s Wi-Fi rozhraním je pro útoky společná, či minimálně podobná, jsou operace s ním seskupeny v komponentě `wifi_controller`. Ta převážně vytváří zjednodušené aplikační rozhraní nad nativní komponentou `esp_wifi`. Poskytuje aplikační rozhraní pro vytváření přístupových bodů, připojování ESP32 do okolních sítí, přepnutí Wi-Fi rozhraní do promiskuitního módu, nastavování MAC adres a skenování sítí v okolí.

7.2.7 Komponenta pro obejití restrikcí ESP-IDF pro odeslání libovolných rámců 802.11

Komponenta `wsl_bypasser` vycházející z projektu *GANESH-UMC/esp32-deauther* využívá postupu pro přepsání definice blokující funkce v ESP-IDF. Způsob, jakým je tohle obejití řešeno, je popsán v sekci 5.1.3. Tato komponenta poté umožňuje odeslání libovolného obsahu odesílací vyrovnávací paměti pomocí oficiálně dokumentované funkce `esp_wifi_80211_tx`.

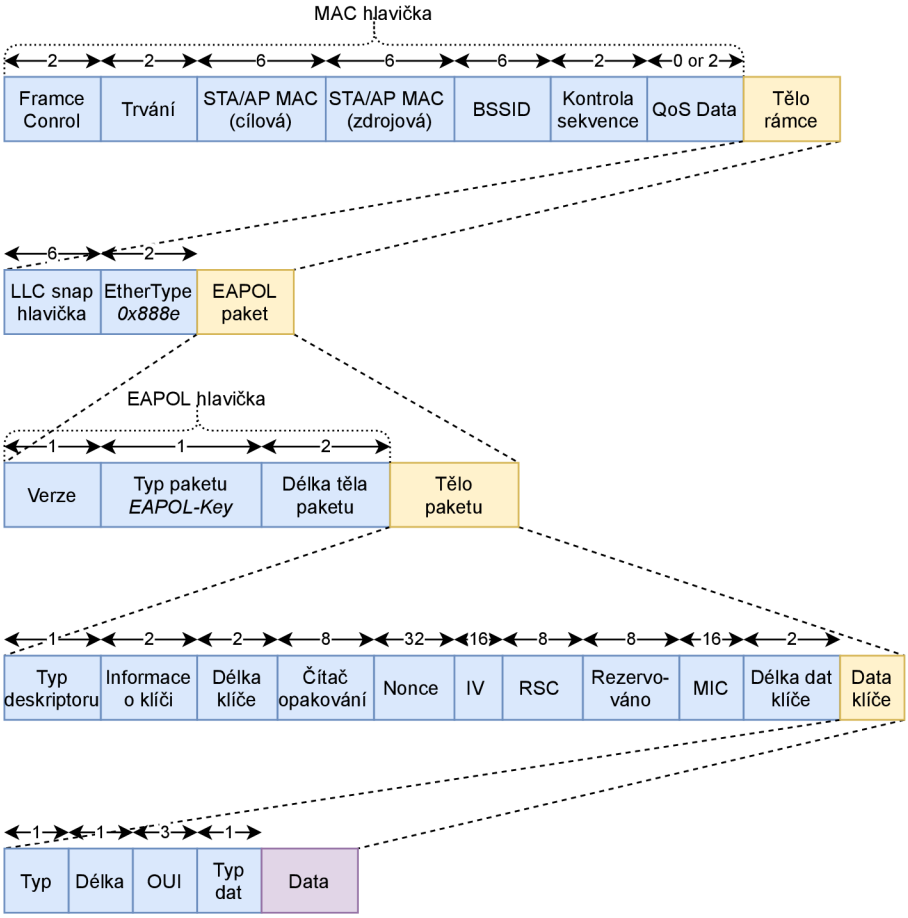
Úprava projektu ESP32 Deauther

Jelikož projekt ESP32 Deauther vyžaduje použití kompilačního nástroje *make*, který v aktuální ESP-IDF verzi 4.1 již není oficiálně podporován a je doporučeno použití nástroje *cmake*, je nutné provést změny ve způsobu kompilace tohoto prijektu (respektive z něj odvozené komponenty `wsl_bypasser`). Do souboru `CMakeLists.txt` v komponentě `wsl_bypasser` jsou přidány instrukce pro linkování knihoven příkazem:

```
target_link_libraries(${COMPONENT_LIB} -Wl,-zmuldefs)
```

Zároveň byla změna navržena i do originálního repozitáře projektu[3].

[3] https://github.com/GANESH-ICMC/esp32-deauther/pull/14



Obrázek 7.4: Ilustrace struktury rámce a zanoření dat. Na obrázku je znázorněn EAPoL-Key rámec.

7.3 Deautentizační útok injektováním rámců

První variantou pro implementaci deautentizačního útoku je odesílání broadcast deautentizačních rámců. Vzhledem k tomu, že ESP-IDF blokuje odesílání takových rámců, je využito komponenty `wsl_bypasser`, jež vychází z projektu *GANESH-ICMC/esp32-deauther*. Je vygenerován deautentizační rámec s broadcast cílovou MAC adresou, zdrojovou MAC adresou a BSSID cílového přístupového bodu a kódem zdůvodnění `0x2` — `INVALID_AUTHENTICATION` informující stanici, že předchozí autentizace již není validní. Tento kód byl vybrán, jelikož jde o obecný důvod, který by stanice neměly ignorovat. Tyto rámce jsou odesílány v pravidelném parametrizovaném intervalu. Toto časování je zajištěno nativní komponentou `esp_timer`. Příklad výstupu tohoto typu útoku v uživatelském prostředí je zachycen na obrázku 7.5.

7.4 Deautentizační útok spuštěním duplikovaného přístupového bodu

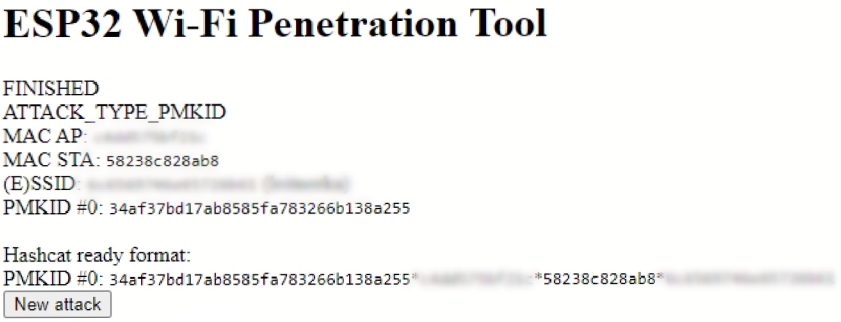
Druhá varianta deautentizačního útoku nevyžaduje možnost injektování rámců. To umožňuje implementaci útoku čistě pomocí aplikačního rozhraní poskytovaného ESP-IDF. Bě-

Obrázek 7.5: Snímek uživatelského prostředí ve stavu *FINISHED*, kdy je uživateli zobrazen výsledek útoku na WPA/WPA2 ustavení klíčů a jsou mu poskytnuta získaná data ke stažení.

hem této metody je vytvořeno duplikované podvržené AP. K vytvoření takového AP je změněna MAC adresa Wi-Fi rozhraní na shodnou adresu s cílovým AP využitím metody `esp_wifi_set_mac`. Poté je běžným způsobem nastartován přístupový bod kopírující ESSID, kanál a autentizační mód (například WPA2-PSK). SoftAP implementované v komponentě `esp_wifi` se drží standardu 802.11. V případě, že takto podvržené AP obdrží rámec třídy 2 nebo 3, automaticky odpoví deautentizačním rámcem s cílovou adresou komunikující stanice. Na obrázku 7.5 je ilustrován příklad výsledku deautentizačního útoku v uživatelském prostředí.

7.5 Získání PMKID

Pro získání PMKID od cílového přístupového bodu ESP32 zahájí WPA/WPA2 autentizaci v režimu STA. Zároveň je přepnuto Wi-Fi rozhraní do promiskuitního módu a jsou analyzovány rámce s BSSID cílového AP. Probíhá syntaktická analýza rámců a hledají se EAPoL pakety. V případě nalezení EAPoL paketu se dále ověřuje, zda jde o EAPoL-Key paket. V EAPoL-Key paketu se může nacházet hledané PMKID jako součást pole *key data*. Těchto PMKID může být 0 až n, a jsou tedy ukládány do lineárního seznamu struktur `pmkid_item_t`. Pokud bylo tedy alespoň jedno PMKID nalezeno, je útok ukončen a stav útoku je přepnut do stavu *FINISHED*. Samotná PMKID jsou uložena v paměti bez dalších metadat, jelikož jejich délka je vždy stejná — 16 bajtů. Příklad výsledku útoku na PMKID je zachycen na obrázku 7.6.



Obrázek 7.6: Na snímku je zachyceno uživatelské prostředí opět ve stavu *FINISHED* a s výsledkem útoku na PMKID. Uživateli jsou zobrazeny parametry potřebné pro provedení útoku silou. Ty mu jsou zároveň poskytnuty ve zformátovaném tvaru pro další zpracování nástrojem Hashcat.

Kapitola 8

Zhodnocení implementace a testování

Tato kapitola se věnuje testování implementovaného nástroje *ESP32 Wi-Fi Penetration Tool* navrženého v kapitole 6 a implementovaného v kapitole 7. Jsou zde shrnuta zařízení použitá pro simulaci reálných sítí — jak stanice a přístupové body, tak i jednotlivé distribuce platformy ESP32. Jsou definovány testovací scénáře a následně diskutovány jejich výsledky. Ve druhé části kapitoly jsou diskutovány možnosti využití a reálného nasazení tohoto nástroje z pohledu útočníka.

8.1 Zařízení použitá pro testování

Pro účely testování nástroje byla vybrána zařízení různých typů, která se běžně vyskytují v reálném prostředí. Vybraná zařízení poskytující funkce přístupového bodu, která byla použita během testování, jsou shrnuta v tabulce 8.1. Seznam stanic, které byly během testování připojeny k testovacím Wi-Fi sítím, je v tabulce 8.2. Následuje shrnutí použitého hardwaru s čipy ESP32 během testování:

ESP32-DEVKITC-32E — oficiální vývojová deska od Espressif s integrovaným modulem ESP32-WROOM-32E s čipem ESP32-D0WD-V3. Tento modul zahrnuje integrovanou anténu.

ESP32-DEVKITC-32UE — oficiální vývojová deska od Espressif s integrovaným modulem ESP32-WROOM-32UE s čipem ESP32-D0WD-V3. Tento modul neobsahuje integrovanou anténu a poskytuje IPEX konektor pro připojení externí antény.

ESP32-WROOM-32E — samostatný modul s čipem ESP32-D0WD-V3.

IoT ESP-WROOM-32 2.4GHz Dual-Mode WiFi+Bluetooth rev.1 — neoficiální vývojová deska integrující modul.

8.2 Testované oblasti

Pro testování funkcionality a využitelnosti nástroje ESP32 Wi-Fi Penetration Tool byly nejprve definovány následující oblasti. Průběh a výsledky jednotlivých testovaných oblastí jsou

Tabulka 8.1: Seznam přístupových bodů, které byly během testování použity.

Modelové označení zařízení	**Popis**
COMPAL CH7465LG	Používaný ISP UPC/Vodafone
TECHNICOLOR TC7200	Používaný ISP UPC/Vodafone
Asus RT-N12+	Běžně dostupný SOHO router
Hak5 WiFi Pineapple NANO	Nástroj pro penetrační testování

Tabulka 8.2: Seznam zařízení, jež byla použita k testování úspěšnosti deautentizačních útoků. Tato zařízení byla v době útoku připojena k cílovému přístupovému bodu a probíhala aktivní komunikace.

Modelové označení zařízení	**Operační systém**	**Typ zařízení**
Motorola Moto G8	Android 10	Chytrý telefon
Xiaomi Redmi Note 8	Android 9	Chytrý telefon
Samsung Galaxy Tab 4 (SM-T335)	Android 5.1.1	Chytrý telefon
MSI GE70-2PE Apache Pro	Windows 10 Education (1909)	Laptop
HP LaserJet Pro M102w	Vlastní firmware	Tiskárna

rozebrány v následujících sekcích. Důraz je kladen na testování samotného útoku na autentizaci WPA/WPA2 zahrnující deautentizační útok. Pro následné vyhodnocení použitelnosti byla též subjektem testování charakteristika chování čipu ESP32 během útoků.

- Ověření úspěšnosti deautentizačních útoků na vybraná zařízení s běžným nastavením AP
- Funkcionalita nástroje při napájení ESP32 z různých zdrojů — baterie, USB
- Úspěšnost deautentizačních útoků v závislosti na připojených stanicích k síti
- Úspěšnost zachycení komunikace během WPA/WPA2 autentizace
- Ověření exportu dat do souborů formátu PCAP a HCCAPX
- Průměrná spotřeba elektrické energie během aktivního útoku
- Ověření použitelnosti uživatelského prostředí
- Dostupnost řídicího přístupového bodu před/během/po provedení útoku.

8.3 Ověření použitelnosti uživatelského prostředí

Nástroj byl otestován na rozlišeních 1920x1080 v desktopových prohlížečích i v mobilním prohlížeči. Byl použit chytrý telefon Motorola Moto G8+ s rozlišením 1920x1080 a mobilní aplikací webového prohlížeče Chrome. Dále k testování desktopové varianty byl použit laptop MSI GE70 2PE Apache Pro s rozlišením 1920x1080 a webovým prohlížečem

Chrome. Nástroj nevykazoval známky nekompatibility a díky minimalistickému uživatelskému prostředí nenastávaly problémy s rozdílnou velikostí použitého displeje. Výstupem tohoto testování jsou některá doporučení ke zlepšení nástroje, shrnuté v následujícím výčtu:

Možnost přerušení útoku ve stavu *RUNNING* — Při probíhajícím útoku by uživatel měl mít možnost útok předčasně ukončit a nástroj by měl přejít do koncového stavu *FINISHED*, stejně jako tomu je při vypršení časového limitu útoku.

Možnost ukončit útok předčasně ve stavu *RUNNING* — Uživatel by měl mít možnost útok ukončit během jeho běhu. Oproti přerušení by v případě ukončení přešel nástroj ze stavu *RUNNING* do počátečního stavu *READY*. Chování by tedy odpovídalo předčasnému přerušení útoku, následované resetováním vnitřního stavu.

Indikace průběhu útoku — Kromě uběhlého času není během útoku indikován průběh útoku — například počet zachycených paketů, počet zachycených PMKID a dalších parametrů. Indikace průběhu je vhodným předpokladem pro implementaci přerušení útoku, jelikož uživatel by měl znát aktuální stav, aby se mohl rozhodnout, zda útok má ještě pokračovat, nebo ho může předčasně přerušit.

Možnost exportu zformátovaného PMKID pro Hashcat — Přestože během útoku na PMKID je výsledek poskytován v textové formě vypsané ve webovém prohlížeči ve formátu akceptovaného nástrojem Hashcat, vzniká zde inkonzistence s výstupem útoků na WPA/WPA2 ustavení klíčů, které poskytují zformátovaná data jako soubory ke stažení. Taková možnost odstraňuje nutnost dalšího zpracování dat před použitím externích nástrojů, jako například Hashcat nebo Wireshark. Je tedy vhodné doplnit export PMKID zformátovaného zápisu pro nástroj Hashcat a umožnit stažení tohoto souboru.

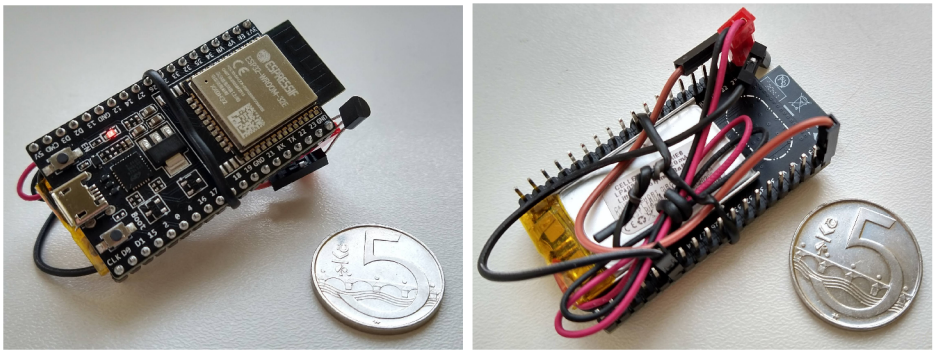
8.4 Průměrná spotřeba elektrické energie během aktivního útoku

Experimentálně byla změřena spotřeba elektrické energie během provádění implementovaných útoků. Při deautentizačních útocích a během analýzy rámců se odběr proudu pohybuje mezi 90 až 110 mA. Za předpokladu připojení čipu na běžně dostupné Li-ion baterie (akumulátory) 18650 s kapacitou 2 600 mAh umožňuje takováto spotřeba zhruba 24 hodin aktivního deautentizačního útoku. Lze předpokládat, že spotřeba se nebude měnit ani při jiných typech útoků. Je to zároveň dost pro útok silou na WPS PIN zabezpečení, které průměrně trvá 4 hodiny [40].

V případě cílení na co nejmenší váhu a rozměr lze využít menších akumulátorů. Na obrázku 8.1 byl pro demonstrační účely zapojen ESP32 k Li-Pol baterii o kapacitě 220 mA, která postačuje k dvouhodinovému útoku.

8.5 Úspěšnost útoků

Během testování implementovaných útoků bylo ověřeno, že čip ESP32 je schopen zachytit všechny rámce WPA/WPA2 autentizace. Testy byly provedeny proti přístupovým bodům v tabulce 8.1. Všechny přístupové body měly aktivní zabezpečení WPA2-PSK a žádné ochrany nebyly aktivní. Úspěšnost deautentizačních útoků závisí na implementaci stanice, jelikož některá zařízení záměrně ignorují deautentizační rámce s broadcast MAC cílovou



Obrázek 8.1: Na snímcích zachycená konfigurace vážící 17 g demonstruje výhody ESP32 svou velikostí, váhou a nízkou spotřebou elektrické energie oproti alternativním řešením. Vyobrazená konfigurace se skládá z vývojové desky ESP32 DevKitC s integrovaným modulem ESP32-VROOM-32E, Li-Pol akumulátoru s kapacitou 220 mA a regulátorem napětí na 3,3 V.

adresou. Ze zařízení, jež byla součástí testování, se jednalo hlavně o zařízení s operačním systémem Windows 10. Kombinací implementovaných variant deautentizačních útoků — cílených deautentizačních rámců specifickým stanicím a deautentizačních rámců odesílaných s broadcastovou cílovou adresou — bylo dosaženo 100% úspěšnosti deautentizačního útoku testovaných zařízení z tabulky 8.2.

8.6 Použitelnost z pohledu uživatele

Nástroj lze naprogramovat na čip ESP32 pomocí jednoduchého nástroje `esptool.py` implementovaného v jazyce Python, jenž je součástí kolekce nástrojů pro vývoj v aplikačním rámci ESP-IDF, nebo pomocí samostatného programu s grafickým rozhraním pro Windows[1]. Mimo tento jednorázový úkon nástroj nevyžaduje od uživatele téměř žádnou doménovou znalost. Útoky na sítě jsou automatizovány a uživatel pouze vybírá cílovou síť (respektive přístupový bod) a typ útoku. Typ útoku může být přednastaven, a uživatel tedy nemusí znát jejich rozdíly. Implementací systému koncových bodů (z anglického *endpoint*) pro ovládání nástroje je umožněna implementace alternativních klientů, například v podobě nativní mobilní aplikace, či integrace nástroje do pokročilejšího automatizačního aplikačního rámce.

8.7 Využitelnost z pohledu útočníka

Nástroj ESP32 Wi-Fi Penetration Tool implementovaný v této práci otevírá teoretickému útočníku nové útočné scénáře a možnosti, jak se vyhnout odhalení okolím. V této sekci jsou představeny scénáře a možnosti, které mohou být považovány za klady implementovaného nástroje.

[1] https://www.espressif.com/en/support/download/othertools

Přístup k jinak nedostupným sítím

Díky nízké váze modulu ESP32-VROOM-32E 3 g a jeho malé velikosti lze ESP32 připevnit například na dron, což útočníkovi otevírá nové útočné vektory. Lze například překonat fyzické zábrany a zabezpečení a útočit na sítě běžně nedostupné — v nepřístupných patrech výškových budov nebo v objektech zabezpečených před přístupem osob.

Vzdálené ovládání bez přímého spojení útočníka s čipem

Díky využití řídicího přístupového bodu pro ovládání nástroje není nutné mít k čipu fyzický přístup v době provádění útoku. To umožňuje čip s nástrojem například uschovat na skrytém místě v okolí cílového AP a následně ho vzdáleně ovládat. Lze se tím vyhnout přímému spojení s použitým čipem, a tedy spojení s útokem samotným, což může mít za následek například nemožnost zahájení trestního stíhaní pro nedůvodnost [29]. Nemenší výhodou může být i fakt, že nástroj je ovladatelný pomocí jakéhokoli zařízení, které je schopno připojit se k přístupovému bodu Wi-Fi a komunikovat pomocí HTTP. Při použití nástroje ve veřejných prostorech je tedy méně nápadné například jeho ovládání pomocí chytrého telefonu, který je dnes běžnou součástí každodenního života a jeho používání nevzbuzuje pozornost.

Variabilita zapojení modulu ESP32

Různé možnosti zapojení a použití nástroje jsou zachyceny na obrázku B. Napájením čipu z chytrého telefonu pomocí funkce OTG lze čip schovat do kapsy, což v dnešní době hojného využívání přenosných záložních zdrojů působí nenápadně. Při připojení čipu k baterii nebo akumulátoru lze vytvořit samostatně fungující jednotku, která není vázána na klientské zařízení, a tedy útočník se může volně pohybovat po okolí, aniž by měl čip s nástrojem přímo u sebe. Na obrázku B je též možné porovnat velikost vývojové desky se samostatným čipem. Pro zvýšení operačního dosahu nástroje lze použít čip s IPEX konektorem, který umožňuje připojení externí antény. Tato variabilita zapojení vývojové desky či modulu ESP32 samotného umožňuje útočníkovi použití stejného nástroje v různých podmínkách v závislosti na aktuálních potřebách.

Odepření služeb nedostupných sítí

Jelikož deautentizační útok je sám o sobě útokem typu odepření služeb, lze implementovaný nástroj využít i pro uskutečnění útoků s cílem odepření služeb Wi-Fi sítě. Díky parametrům popsaných výše v předchozích bodech, lze tento útok uskutečnit i v místech, kde by se útočník běžně nedostal — například v hlídaných kancelářích výškových budov. Vzhledem k nízké ceně modulů ESP32 může útočník rozmístit větší množství čipů v okolí sítě, které do cílového prostoru dopraví například již zmíněným dronem. Tím může dočasně blokovat přístup mobilních zařízení pracovníků k Wi-Fi síti a zamezit jim tak ve výkonu práce, což vytváří finanční ztrátu na straně zaměstnavatele či provozovatele. Jak bylo ukázáno v sekci 8.4, při napájení baterií s kapacitou 2 600 mAh je možné útok vést nepřetržitě téměř 24 hodin. Útočník tak může například vyžadovat výkupné, aby útok ukončil předčasně. Útočníkem nemusí v tomto případě být pouze osoba motivovaná ziskem, ale například aktivisté požadující reakci od provozovatele cílové sítě.

8.8 Obrana proti útokům

Účinnou obranou proti deautentizačním útokům je konfigurace šifrování řídicích rámců na přístupovém bodu definovaných ve standardu 802.11w [28]. Pro detekci použití nástroje ESP32 Wi-Fi Penetration Tool lze kontrolovat komunikaci v okolí přístupového bodu a hledat komunikaci řídicího přístupového bodu a stanice — zařízení, jež útočník používá pro ovládání nástroje. Obecnou obranou proti útokům pomocí čipů ESP32 je komunikace pouze v pásmu 5Ghz, jelikož platforma ESP32 v tomto pásmu nedokáže komunikovat [7], a síť je tedy imunní proti těmto útokům. Dalším vhodným opatřením je mít k dispozici alternativní síť, například drátovou síť Ethernet, která může zabránit kompletnímu výpadku připojení v případě aktivního útoku typu odepření služeb.

8.9 Integrace s externími nástroji pro útok silou

Jelikož ESP32 nemá dostatečný výkon pro uskutečnění efektivního útoku silou, je vhodné tento útok provést na externím stroji s použitím nástrojů pro obnovu hesel. Tomu napomáhá právě implementace exportu do formátů pro nástroj Hashcat a zjednodušuje celý proces. Lze využít cloudových služeb, jako je AWS EC2[2] nebo Google poskytující podíl na jejich výpočetním výkonu. Tato část je nejnákladnější, jelikož tyto cloudové služby jsou placené. Je možné též využít i nástroje *FitCrack 2*[3] vyvíjeného výzkumnou skupinou NES@FIT[4] pro distribuovanou obnovu dat, využívající právě nástroje Hashcat.

[2] https://aws.amazon.com/ec2/
[3] https://fitcrack.fit.vutbr.cz/
[4] https://www.fit.vut.cz/research/group/nes@fit

Kapitola 9

Závěr

Cílem diplomové práce bylo prozkoumat možnosti implementace známých útoků na Wi-Fi sítě na platformě ESP32 pomocí oficiálního vývojového aplikačního rámce Espressif IoT Development Framework (ESP-IDF) a vybraný útok navrhnout, implementovat a vyhodnotit jeho úspěšnost. Tento záměr byl splněn implementací deautentizačního útoku. V práci byly shrnuty aktuálně známé útoky na Wi-Fi sítě a zranitelnosti standardu 802.11 a jeho dodatků. Byl představen souhrn využitelných funkcí a komponent ESP-IDF a diskutovány limitace v kontextu práce s Wi-Fi rozhraním. Byly shrnuty aktuálně existující řešení a projekty zabývající se tematikou implementace útoků na Wi-Fi na platformách ESP32 a ESP8266. Následně byly navrženy implementace útoků pomocí ESP-IDF s využitím projektu *esp32-deauther*. V práci bylo ukázáno, že díky možnostem ESP-IDF, jako jsou změna MAC adresy Wi-Fi rozhraní, konfigurace a provozování přístupových bodů a stanic nebo přepínání rozhraní do promiskuitního módu v kombinaci s možností obejití blokujících opatření ve *Wi-Fi Stack Libraries*, umožňuje implementaci útoků popsaných v úvodní části práce.

Práce byla dále rozšířena o implementaci univerzálního nástroje *ESP32 Wi-Fi Penetration Tool* vytvářejícího jednotné prostředí pro spouštění útoků na platformě ESP32, který zároveň umožňuje snadnou rozšiřitelnost dalšími typy útoků a znovupoužitelnost již implementovaných komponent. Byl představen způsob uskutečnění deautentizačního útoku i v případě nemožnosti injektování upravených 802.11 rámců využitím definice chování přístupového bodu dle standardu 802.11 a oficiálně dostupného ESP-IDF aplikačního rozhraní bez nutnosti dalších modifikací. Tento fakt poukazuje na zbytečnost snah vývojářů Espressif o blokování odesílání určitých typů rámců právě jako prevenci před zneužitím jejich výrobků k provádění deautentizačních a jiných útoků.

Výsledek práce dále poukazuje na nutnost věnovat zabezpečení sítí pozornost, ať už jde o velké korporátní sítě, nebo naopak o malé domácí sítě. Využití levných, dostupných čipů ESP32 s nízkou spotřebou umožňuje komukoliv s minimem znalostí oboru spouštět relativně nebezpečné útoky na sítě ve svém okolí. Díky malým rozměrům modulů ESP32 se útočníkovi otevírají nové možnosti vedení útoků na běžně nedostupné sítě zabezpečené například nepřístupností prostoru v jejich blízkosti. Útočník může osadit malý dron tímto lehkým a malým modulem ESP32 a pomocí něj překonat fyzické překážky. Možnost napájení čipu bateriemi a vzdálené ovládání nástroje přes řídicí přístupový bod zase útočníkovi umožňuje skryté použití na veřejnosti bez vzbuzení větší pozornosti. Vzdálené ovládání taktéž útočníkovi poskytuje určitou ochranu proti případným právním krokům při podezření z neoprávněného proniknutí do cizí sítě.

Práce mi dala hlubší porozumění Wi-Fi sítí a standardu 802.11. Naučil jsem se pracovat s čipy ESP32 a jejich Wi-Fi rozhraním. Zároveň jsem díky této práci získal nové zkušenosti s vývojem open-source projektů. V práci bych rád pokračoval dalším rozvíjením nástroje *ESP32 Wi-Fi Penetration Tool* implementací dalších útoků, jejichž implementace byly v této práci navrženy, a pokusil se o vytvoření fungující open-source komunity kolem tohoto nástroje. Z toho důvodu jsem projekt publikoval ve službě GitHub[1] pod MIT licencí. Dále by bylo možné výstup práce rozvíjet integrací s cloudovými nástroji pro obnovu hesel jako například *FitCrack* vyvíjený výzkumnou skupinou NES@FIT. Tento projekt implementuje distribuovaný systém využívající právě nástroj *Hashcat*, pro který již v této práci existuje podpora — export zachycených dat ve formátech připravených pro další zpracování tímto nástrojem.

Práce byla prezentována na studentské konferenci Excel@FIT 2021 s identifikačním číslem 48[2]. Článek z této konference je v příloze E. Jeho součástí je také demonstrační video představující funkcionalitu nástroje ESP32 Wi-Fi Penetration Tool, dostupné ve službě YouTube[3]. Na přiloženém CD se nachází mimo jiné repozitář Git se zdrojovými kódy nástroje, jehož obsah je částečně shrnut v příloze C. V příloze D se nachází postup pro zprovoznění nástroje.

[1] https://github.com/risinek/esp32-wifi-penetration-tool
[2] http://excel.fit.vutbr.cz/submissions/2021/048/48.pdf
[3] https://www.youtube.com/watch?v=9I3BxRu86GE

Literatura

[1] Asante, M. a Akomea Agyin, K. Analysis of Security Vulnerabilities in Wifi Protected Access Pre Shared Key (WPA PSK/ WPA2 PSK). *International Research Journal of Engineering and Technology (IRJET)*. 1. vyd. Kumasi, State, Ghana: [b.n.]. 2019, sv. 6, č. 1, s. 537 – 545. ISSN 2395-0072.

[2] Brown, B. 802.11: the security differences between b and i. *IEEE Potentials*. 1. vyd. IEEE. 2003, sv. 22, č. 4, s. 23–27. DOI: 10.1109/MP.2003.1238689. ISSN 0278-6648.

[3] Dalabaev, S., Quanfu, S., Qinghua, L., Zhuping, H., Shilian, Y. et al. 4-way handshake attack analysis and improvement in 802.11i. *2013 Cross Strait Quad-Regional Radio Science and Wireless Technology Conference*. 1. vyd. IEEE. 2013, č. 6657453, s. 455–458. DOI: 10.1109/CSQRWC.2013.6657453.

[4] Daniel, M. *WPA password cracking: Parallel Processing on the Cell BE*. Aalborg, DK, 2009. Master's thesis. Aalborg University, Institute for Electronic Systems.

[5] Espressif Systems. *ESP32: Technical Reference Manual*. 2020. Dostupné z: https://www.espressif.com/sites/default/files/documentation/esp32_technical_reference_manual_en.pdf.

[6] Espressif Systems. *ESP32: ESP-IDF Programming Guide*. 2020. Dostupné z: https://www.espressif.com/sites/default/files/documentation/esp32_technical_reference_manual_en.pdf.

[7] Espressif Systems. *ESP32 Series: Datasheet*. 2020. Dostupné z: https://www.espressif.com/sites/default/files/documentation/esp32_datasheet_en.pdf.

[8] Espressif Systems. *ESP8266EX: Datasheet*. 6.6 (2020.10). 2020. Dostupné z: https://www.espressif.com/sites/default/files/documentation/0a-esp8266ex_datasheet_en.pdf.

[9] Espressif Systems. *Development Boards*. 2021 [cit. 12. 1. 2021]. Dostupné z: https://www.espressif.com/en/products/devkits.

[10] Espressif Systems. *Products*. 2021 [cit. 13. 1. 2021]. Dostupné z: https://www.espressif.com/en/products.

[11] eTutorials.org. *Details of Key Derivation for WPA*. 2013 [cit. 12. 1. 2021]. Dostupné z: http://etutorials.org/Networking/802.11+security.+wi-fi+protected+access+and+802.11i/Part+II+The+Design+of+Wi-Fi+Security/Chapter+10.+WPA+and+RSN+Key+Hierarchy/Details+of+Key+Derivation+for+WPA/.

[12] Euchner, F. *ESP32 802.11 Freedom Output: README* [online]. GitHub, 2017. Dostupné z: https://github.com/Jeija/esp32free80211/blob/c940b169f86f4ce805df6af962396c9fd27fc235/README.md.

[13] Explore Embedded. *Overview of ESP32 features. What do they practically mean?* 2017 [cit. 12. 1. 2021]. Dostupné z: https://www.exploreembedded.com/wiki/Overview_of_ESP32_features._What_do_they_practically_mean%3F.

[14] Fluhrer, S., Mantin, I. a Shamir, A. Weaknesses in the key scheduling algorithm of RC4. In: Cisco Systems. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics).* San Jose: Springer Verlag, 2001, sv. 2259, s. 1–24. ISBN 9783540430667.

[15] Frankel, S., Eydt, B., Owens, L. a Scarfone, K. *NIST Special Publication 800-97, Establishing Wireless Robust Security Networks.* 1. vyd. únor 2007. 162 s. ISBN 9781495291807.

[16] Free Software Foundation, Inc. *Ld(1): Linux man page.* 2009 [cit. 12. 1. 2021]. Dostupné z: https://linux.die.net/man/1/ld.

[17] G4lile0. *ESP32 WiFi Hash Monster: README* [online]. GitHub, 2020. Dostupné z: https://github.com/G4lile0/ESP32-WiFi-Hash-Monster/blob/6174524e8391f0e84e31e454599f186b91428de7/README.md.

[18] Gast, M. *802.11 wireless networks: the definitive guide.* 1st ed. Beijing: O'Reilly, 2002. ISBN 0-596-00183-5.

[19] Gratton, A. *Esp_wifi_internal.* 2016 [cit. 18. 1. 2021]. Odpověd ve vláknu. Dostupné z: https://www.esp32.com/viewtopic.php?t=586.

[20] Grokhotkov, I. a Domburg, J. *Please open source this library: Issue #2* [online]. GitHub, 2016 [cit. 18. 1. 2021]. Odpovědi ve vlákně. Dostupné z: https://github.com/espressif/esp32-wifi-lib/issues/2.

[21] Hayajneh, T. a Kohlios, C. P. A Comprehensive Attack Flow Model and Security Analysis for Wi-Fi and WPA3. *Electronics (Basel).* 1. vyd. MDPI AG. 2018, sv. 7, č. 11, s. 284. DOI: 10.3390/electronics7110284.

[22] Heesch, D. van. *Doxygen* [online]. 2021 [cit. 15. 5. 2021]. Dostupné z: https://www.doxygen.nl/index.html.

[23] Holt, A. a Huang, C.-Y. *802.11 Wireless Networks.* 1. vyd. London: Springer London, 2010. ISBN 978-1-84996-274-2.

[24] HÜBSCHMANN, I. *ESP32 for IoT: A Complete Guide.* Srpen 2020 [cit. 12. 1. 2021]. Dostupné z: https://www.nabto.com/guide-to-iot-esp-32/.

[25] Institute of Electrical and Electronics Engineers. *802.3-1985 - IEEE Standards for Local Area Networks: Carrier Sense Multiple Access With Collision Detection (CSMA/CD) Access Method and Physical Layer Specifications.* Piscataway, USA: IEEE, 1984. 802.3.

[26] Institute of Electrical and Electronics Engineers. *IEEE Standard for Information Technology- Telecommunications and Information Exchange Between Systems- Local and Metropolitan Area Networks- Specific Requirements- Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications*. 1999 Ed. New York, USA: IEEE, 2003.

[27] Institute of Electrical and Electronics Engineers. *IEEE Standard for information technology-Telecommunications and information exchange between systems-Local and metropolitan area networks-Specific requirements-Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) specifications*. 2004 Ed. New York, USA: IEEE, 2004. 802.11i.

[28] Institute of Electrical and Electronics Engineers. *IEEE Standard for Information technology - Telecommunications and information exchange between systems - Local and metropolitan area networks - Specific requirements. Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications Amendment 4*. 2009 Ed. USA: IEEE, 2009. 802.11w.

[29] Jelínek, J. a kol. *Trestní zákoník a trestní řád s poznámkami a judikaturou*. 6. vyd. Praha: Leges, 2016. 1280 s. ISBN 978-80-7502-395-7.

[30] Kolban, N. *Kolban's book on ESP32* [online]. 1. vyd. 2018. Dostupné z: https://leanpub.com/kolban-ESP32.

[31] Lipovský, R., Svorenčík a Čermák, M. KR00K - CVE-2019-15126: Serious vulnerability deep inside your Wi-Fi encryption. *We live security*. 1. vyd. 2020, s. 9, [cit. 18. 1. 2021]. Dostupné z: https://www.welivesecurity.com/wp-content/uploads/2020/02/ESET_Kr00k.pdf.

[32] Nagarjun, P. M. D., Kumar, V. A., Kumar, C. A. a Ravi, A. Simulation and analysis of RTS/CTS DoS attack variants in 802.11 networks. *2013 International Conference on Pattern Recognition, Informatics and Mobile Engineering*. 1. vyd. IEEE. 2013, č. 6496483, s. 258–263. DOI: 10.1109/ICPRIME.2013.6496483.

[33] Nguyen, T., Nguyen, D., Tran, B., Vu, H. a Mittal, N. A Lightweight Solution for Defending Against Deauthentication/Disassociation Attacks on 802.11 Networks. In: IEEE. *2008 Proceedings of 17th International Conference on Computer Communications and Networks*. Hochiminh City, Vietnam: IEEE, 2008, s. 1–6. DOI: 10.1109/ICCCN.2008.ECP.51. ISBN 9781424423897.

[34] Núñez, H. H. L. *Where did ieee80211_raw_frame_sanity_check came from?: Issue #9* [online]. GitHub, 2020. Dostupné z: https://github.com/GANESH-ICMC/esp32-deauther/issues/9.

[35] Peterka, J. *Báječný svět počítačových sítí: Část XXV: Architektura Wi-Fi sítí*. 2007 [cit. 12. 1. 2021]. Dostupné z: https://www.earchiv.cz/b07/b0500001.php3.

[36] SecureW2. *Simplifying WPA2-Enterprise and 802.1x*. Březen 2021 [cit. 12. 1. 2021]. Dostupné z: https://www.securew2.com/solutions/wpa2-enterprise-and-802-1x-simplified/.

[37] Steube, J. *New attack on WPA/WPA2 using PMKID*. Srpen 2018 [cit. 12. 1. 2021]. Dostupné z: https://hashcat.net/forum/thread-7717-post-41427.html.

[38] Vanhoef, M. a Piessens, F. Practical verification of WPA-TKIP vulnerabilities. In: Association for Computing Machinery. *ASIA CCS '13: Proceedings of the 8th ACM SIGSAC symposium on Information, computer and communications security*. 1. vyd. Květen 2013, s. 427–436. DOI: 10.1145/2484313.2484368. ISBN 978-1-4503-1767-2.

[39] Vanhoef, M. a Piessens, F. Key Reinstallation Attacks: Forcing Nonce Reuse in WPA2. In: Association for Computing Machinery. *CCS '17: Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*. New York: Association for Computing Machinery, 2017, s. 1313–1328. DOI: 3133956.3134027. ISBN 9781450349468.

[40] Viehböck, S. Brute forcing Wi-Fi Protected Setup: When poor design meets poor implementation. 3. vyd. Prosinec 2011.

[41] Vilius, K., Liu, L., Panneerselvam, J. a Stimpson, T. A Critical Analysis of the Efficiencies of Emerging Wireless Security Standards Against Network Attacks. *2015 International Conference on Intelligent Networking and Collaborative Systems*. 1. vyd. IEEE. 2015, č. 56, s. 472–477. DOI: 10.1109/INCoS.2015.56.

[42] Wi-Fi Alliance. *Wi-Fi CERTIFIED Interoperability Certificate: WFA97858*. Wi-Fi Alliance, 2020. Dostupné z: https://www.espressif.com/en/support/documents/certificates.

[43] Wi-Fi Alliance. *Wi-Fi CERTIFIED Interoperability Certificate: WFA1011278*. Wi-Fi Alliance, 2020. Dostupné z: https://www.espressif.com/en/support/documents/certificates.

[44] Wi-Fi Alliance. *Wi-Fi Protected Setup Specification*. 1.0h. 2006. Dostupné z: https://www.wi-fi.org/discover-wi-fi/wi-fi-protected-setup.

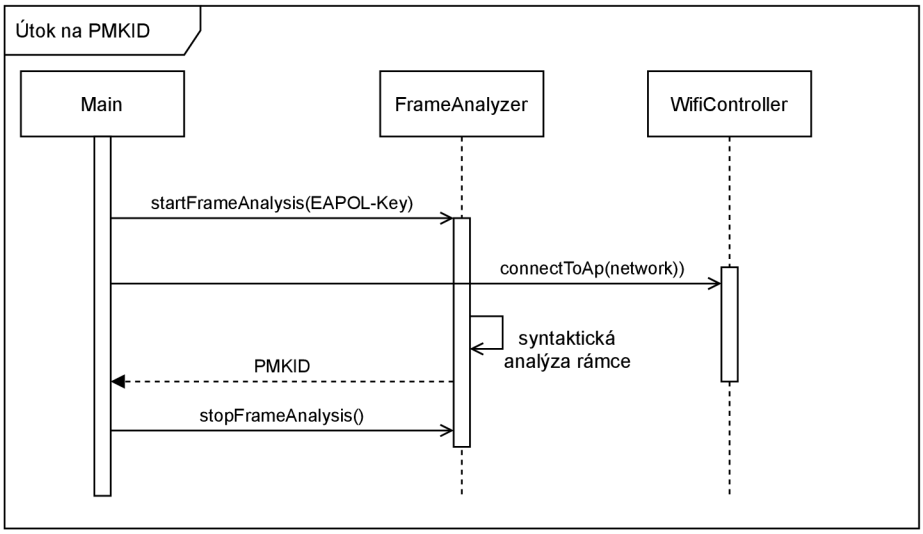
[45] Wi-Fi Alliance. History. *Wi-Fi Alliance: Who Are We* [online]. 2019 [cit. 12. 1. 2021]. Dostupné z: https://www.wi-fi.org/who-we-are/history.

[46] Wi-Fi Alliance. *WPA2™ Security Now Mandatory for Wi-Fi CERTIFIED™ Products*. Austin, Texas, USA: [b.n.], 13. března 2006 [cit. 12. 1. 2021]. Dostupné z: https://www.wi-fi.org/news-events/newsroom/wpa2-security-now-mandatory-for-wi-fi-certified-products.

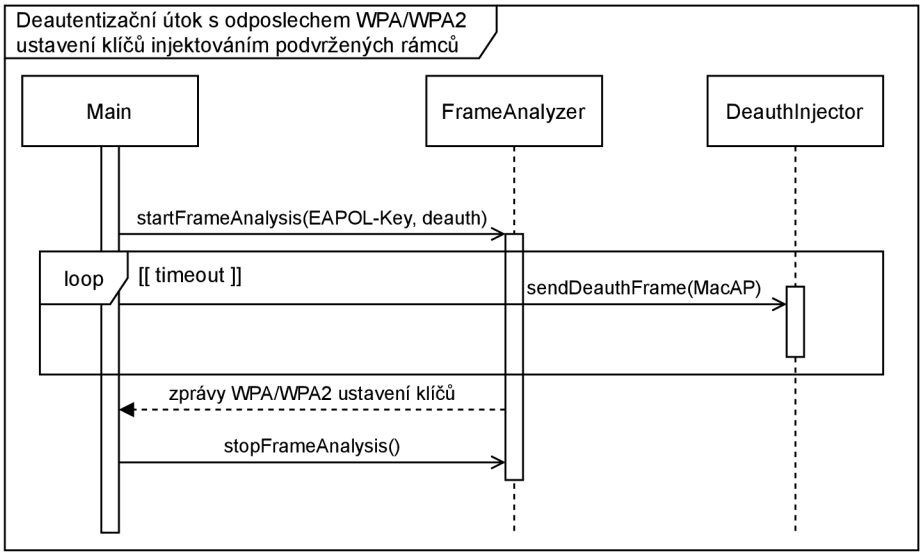
[47] WiGLE. Stats. *WiGLE* [online]. 2021 [cit. 12. 1. 2021]. Dostupné z: https://wigle.net/stats.

Příloha A

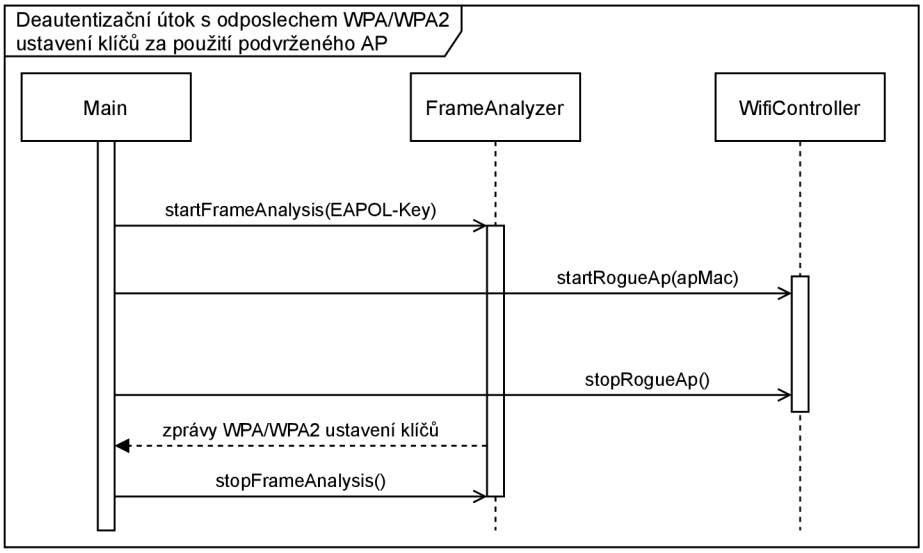
Návrh interakcí komponent



Obrázek A.1: Interakce komponent během útoku na PMKID. Hlavní komponenta aktivuje analýzu rámců zaměřenou na hledání EAPoL-Key rámců a v nich obsažených identifikátorů PMKID. Poté pomocí komponenty WifiController zahájí připojení se k cílové síti, což vyvolá ze strany AP zahájení autentizace a odeslání první zprávy ustavení klíčů. Rámec s touto zprávou je analyzován FrameAnalyzer komponentou a o výsledku je informována hlavní komponenta Main.



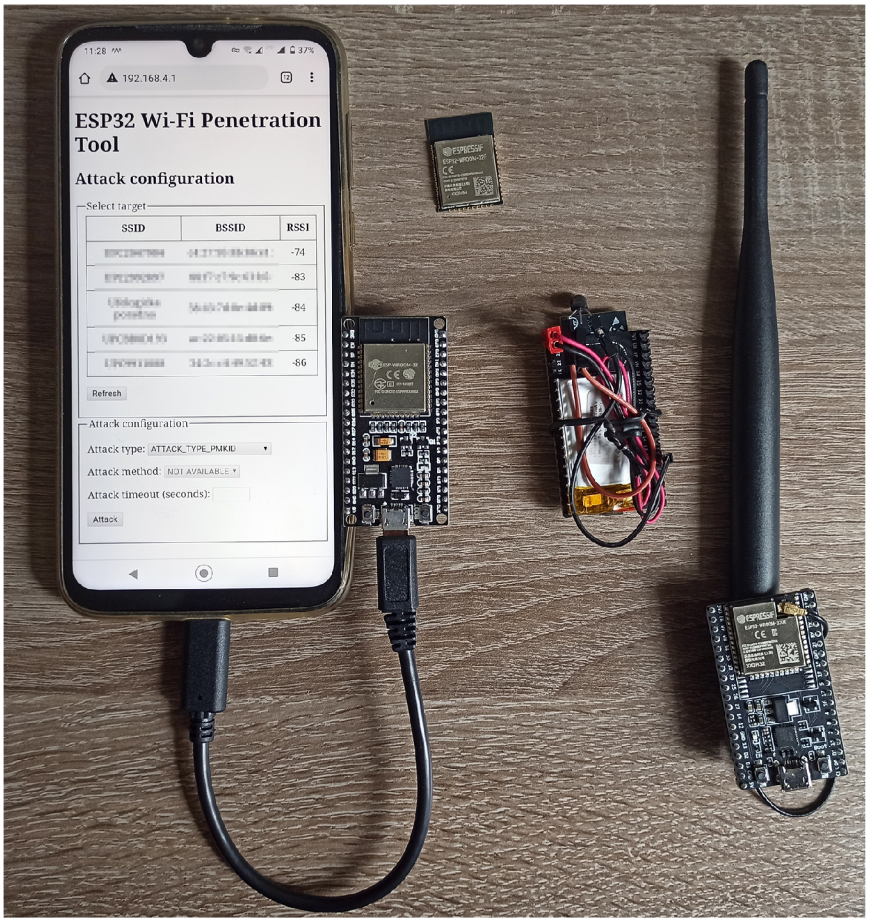
Obrázek A.2: Interakce komponent během deautentizačního útoku injektováním podvržených rámců. Hlavní komponenta aktivuje analýzu EAPoL-Key a deautentizačních rámců pomocí komponenty FrameAnalyzer. Následně pomocí komponenty DeauthInjector, jež implementuje obejití blokujícího opatření v knihovnách Wi-Fi Stack Libraries, odesílá podvržené deautentizační rámce v pravidelných intervalech do vypršení časového limitu.



Obrázek A.3: Interakce komponent při deautentizačním útoku pomocí podvrženého AP. Hlavní komponenta aktivuje analýzu rámců zaměřenou na EAPoL-Key rámce. Následně pomocí komponenty WifiController nastaví parametry Wi-Fi rozhraní podle cílového AP a vytvoří podvržené duplikované AP. Po skončení útoku je podvržené AP zastaveno a zachycené rámce z komponenty FrameAnalyser jsou předány zpět hlavní komponentě.

Příloha B

Možnosti zapojení ESP32



Obrázek B.1: Na obrázku jsou zachyceny různé varianty zapojení a použití ESP32 s nástrojem ESP32 Wi-Fi Penetration Tool. Zleva je vývojová deska ESP32 napájená z chytrého telefonu pomocí OTG funkce a zároveň je zde zachyceno uživatelské prostředí na mobilním telefonu. Uprostřed se nachází vývojová deska ESP32-DEVKITC-32E připojená k Li-Pol baterii. Vpravo je zachycena vývojová deska ESP32-DEVKITC-32UE s připojenou externí anténou pomocí I-PEX konektoru. Pro porovnání velikosti je uprostřed nahoře přiložen i samostatný modul ESP32-WROOM-32E.

Příloha C

Obsah přiloženého paměťového média

- Git repozitář obsahující zdrojové kódy nástroje ESP32 Wi-Fi Penetration Tool — `esp32-wifi-penetration-tool/`
  - Binární soubory k naprogramování nástroje na čip ESP32
    - `build/esp32-wifi-penetration-tool.bin`
    - `build/bootloader/bootloader.bin`
    - `build/partition_table/partition-table.bin`
  - Dokumentace — `README.md` soubory ve složkách jednotlivých komponent
  - Reference aplikačního rozhraní vygenerovaná nástrojem Doxygen — `doc/api/html/index.html`
- Zdrojové kódy textové části práce v LaTeX— `latex/`
- Text této práce v PDF formátu — `DP_WiFi_Attacks_Using_ESP32_8266.pdf`
- Článek z konference Excel@FIT 2021 — `2021_ExcelFIT_ESP32WiFiAttacks.pdf`
- Demonstrační video představující použití nástroje ESP32 Wi-Fi Penetration Tool — `esp32-wifi-penetration-tool-demo.mp4`

Příloha D

Manuál

Předpokladem pro naprogramování nástroje *ESP32 Wi-Fi Penetration Tool* je funkční prostředí ESP-IDF[1], nebo alespoň funkční programovací nástroj `esptool.py`[2]. Následuje postup pro naprogramování a spuštění nástroje:

1. Naprogramovat binární soubory v adresáři `build/` příkazem `idf.py flash` (spuštěného v kořenovém adresáři) nebo použitím nástroje `esptool.py`
2. Připojit ESP32 ke zdroji
3. Po startu se automaticky spustí řídicí přístupový bod *Management AP*
4. Připojit se k *Management AP*. Heslo je `mgmtadmin`
5. Na připojené stanici otevřít v prohlížeči adresu `192.168.4.1`
6. Vybrat cílovou síť pro útok
7. Vybrat typ útoku — DoS, získání WPA/WPA2 ustavení klíčů nebo PMKID
8. Zvolit metodu útoku, pokud je dispozici
9. Postupovat dle instrukcí nástroje

[1] https://docs.espressif.com/projects/esp-idf/en/latest/esp32/get-started/index.html
[2] https://github.com/espressif/esptool

Příloha E

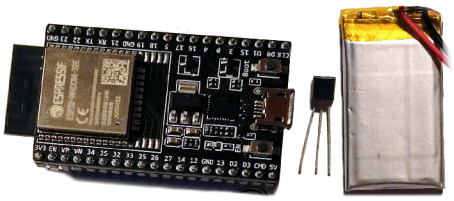
Článek publikovaný na konferenci Excel@FIT 2021

BRNO UNIVERSITY OF TECHNOLOGY
FACULTY OF INFORMATION TECHNOLOGY

Excel@FIT 2021
http://excel.fit.vutbr.cz

Wi-Fi attacks using ESP32

Richard Stehlík*



Abstract

This work explores possibilities of Espressif's ESP32 SoCs in combination with its official development framework ESP-IDF in terms of implementing well-known Wi-Fi attacks on them. Using ESP32 for such attacks may allow attackers to scale their malicious intentions more easily and cut cost and complexities of Wi-Fi attack executions to minimum. Being low powered device also opens ways to minimize size of necessary hardware for Wi-Fi attacks and can easily operate on battery while maintaining a low weight.

Proposed solution presented in this work covers attacks on WPA/WPA2 authentication and their variations like station deauthentication, WPS PIN brute-force attack or PMKID capture. An universal Wi-Fi penetration tool for ESP32 was introduced, that provides easy way to implement new attacks and their variants in the future. It shows how these attacks can be implemented purely by using ESP-IDF's public API or by bypassing closed source Wi-Fi Stack Libraries that have incorporated protection against misusing ESP32 for sending forged frames.

The outcome supports the need to mitigate some vulnerabilities in currently widely used Wi-Fi security features and give them more attention with higher priority.

Keywords: ESP32 — ESP-IDF — Wi-Fi attacks

Supplementary Material: Demonstration Video — Git Repository

*xstehl16@stud.fit.vutbr.cz, *Faculty of Information Technology, Brno University of Technology*

1. Introduction

While Wi-Fi networks and its underlying 802.11 standard is widespread all around the world and today its usage is even more supported by raise of IoT devices and demand for portability, it still has its flaws in original design that were not yet fully mitigated. Besides some of the longtime well-known attacks like deauthentication attack that are exploiting parts of original 802.11 standard, new attacks are appearing even now, more than 20 years later after initial 802.11 standard release back in 1997. Some of them are exploiting newly found vulnerabilities like *KRACK* attack described in 2017 [1] or describing vulnerabilities themselves like *KR00K* first disclosed in 2020 [2]. There are also new methods being discovered that simplifies previously described attacks like *attack on PMKID* that greatly reduces requirements for classic handshake capture and brute force attack to crack network passphrase to gain access to target network or session Pairwise Transient Key to decrypt captured communication [3, 4].

Most of these attacks are already implemented and integrated in tools that usually rely on services provided by underlying operating system. One of the mostly referred tool for this kind of attacks is *aircrack-ng*[1] that operates on Windows, Linux, OS X and other UNIX based operation systems. They also require some insight into the topic and require specific hard-

[1] https://www.aircrack-ng.org/

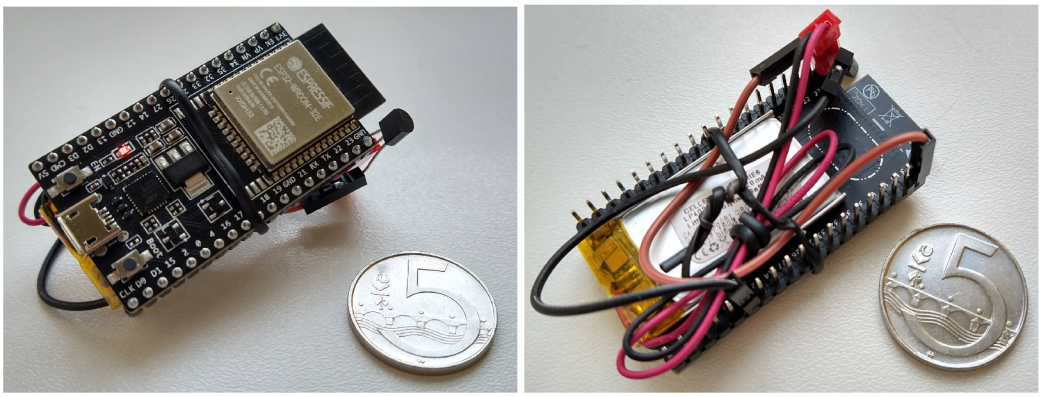


Figure 1. An example of *ESP32 DevKitC* board connected to a Li-Pol accumulator that altogether weights around 17 grams. Czech 5-koruna coin included for scale. This demonstrates how using ESP platform can be useful in terms of downsizing hardware requirements. It can be easily downsized further by using ESP32 module directly, voltage regulator and smaller accumulator to a really small piece of hardware that is easy to hide or transportable for example with small remotely controlled drone.

ware like Wi-Fi adapter with promiscuous mode and frame injection capability. Even though most of these systems are able to run on small computers like Raspberry Pi, in order to go further with reducing hardware size, weight and price needed for the attack, these operating systems are usually a limitation by themselves. By reducing hardware size and price, it can unleash new attack vectors that can lead to attacks in places, that for example were considered secure by physical measures. It can also simplify the overall usage for users by preprogrammed chips that then work in a plug and play way. Microcontrollers manufactured by Espressif Systems with integrated Wi-Fi interface are perfect adepts to be used for Wi-Fi penetration considering also their well documented Espressif IoT Developement Framework (ESP-IDF). ESP32 is currently their latest and most evolved MCU they produce. Figure 1 demonstrates one of the many possibile ways how to wire and operate ESP32 in a compact way.

Motivation behind this work is to explore possibilities of ESP-IDF and demonstrate that ESP32 platform is capable of executing malicious Wi-Fi attacks just by utilising ESP-IDF itself. Executing attacks on ESP32 can be an efficient alternative to existing solutions running on *Raspberry Pi Zero W* or similar micro-computers. Direct comparison with Raspberry Pi Zero W, that can be considered a closest competitor to ESP32, is presented in Table 1. As ESP32 requires low power and allows some low level configurations with Wi-Fi interface on MAC layer, it creates space for experimentation and may open new possibilities in attacker's scenarios.

This work explores capabilities of ESP-IDF and propose ways how to implement different well-known Wi-Fi attacks. It also takes advantage of existing projects, that already solved some of the obstacles introduced by ESP-IDF design and developer decisions and builds complex attacks on top of them. Practical outcome of this work is an universal Wi-Fi penetration tool that is easily extensible by adding new attack types and their methods and also includes some of the proposed attacks implementations to prove this concept.

1.1 Contribution

The main contribution of this work is proposal of implementations of various widely known Wi-Fi attacks using ESP-IDF and ESP32 platform. New approach to execute deauthentication attack by creating cloned rogue access point exploiting native behaviour defined in 802.11i standard is proposed that allows deauthentication attack even without frame injection capability. Some of these implementations are then realised by implementing a new tool to consolidate various attacks into one place that makes executing them simple alongside with easing addition of new attacks and their variants. To demonstrate usability, deauthentication attacks for denial of service attacks and for WPA/WPA2 handshake capture were included in the tool.

2. Common Wi-Fi attacks

Even though 802.11 standard was first introduced more than 20 years ago and is being actively amended by IEEE organisation, there are still vulnerabilities deep in its design that are hard to mitigate without reworking whole concept. These vulnerabilities are drawing attention of various attackers with all kinds of intentions. Attacks and exploits taking advantage of these vulnerabilities are being well described and are implemented in various forms. This section briefly describes common Wi-Fi attacks that have a potential to be implemented on ESP32 platform.

Table 1. Comparison of ESP32 with Raspberry Pi Zero W

	ESP32	Raspberry Pi Zero W
Monitor mode	native	requires custom firmware [5]
Frame injection	limited, can be unblocked	requires custom firmware [5]
Average current draw	∼100mA	∼150mA [6]
Boot current draw	<100mA	up to 200mA [6]
Voltage	3.3V or 5V	5V [6]
OS	—	e.g. Kali Linux
Weight	3.5g (module)	9g [6]
Dimensions	25.2x18x2.8mm (module)	65mm x 30mm x 5.4mm [6]
Price	∼80 CZK (module)	∼300 CZK
Other caveats		requires SD card often out of stock

2.1 Deauthentication attack

Deauthentication attack exploits a behaviour described in 802.11 standard, which allows access point (also referred to as *AP* in this work) or station (*STA*) to interrupt authenticated session by sending deauthentication management frame to its counterpart. If the station or access point receives this kind of frame, it will stop further communication with opposite side until STA authenticates itself again. In original version of 802.11 standard management frames are not encrypted nor they are authenticated [7], so potential attacker within the reach of target network can forge deauthentication frames and broadcast them to his vicinity. Due to lack of authentication or encryption of these frames, stations and access points cannot differentiate between forged and genuine frames and will assume they are valid — resulting into deauthenticating themselves from the network.

Even though this exploit was addressed in *802.11w* amendment in 2009 by forcing encryption of subset of management frames including deauthentication ones [8], it is not widely used and most of now-days APs does not have this feature enabled by default or even not available at all [9]. Hence this type of attack is one of the most used and is often a precondition of many other attacks.

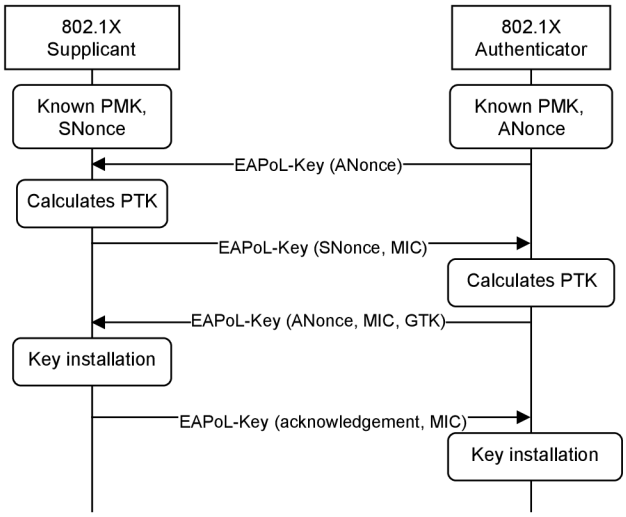


Figure 2. Simplified sequence diagram of WPA/WPA2 handshake exchange where both sides exchange four messages with parameters for calculating same Pairwise Transient Key that is then used for frame encryption. From the sequence diagram it's clear that only secret here is Pairwise Master Key that is used to calculate PTK and is not being transmitted over network. Hence PMK is main target of handshake capture and following brute-force attack

2.2 WPA handshake brute-force attack

Due to a way how session keys for authenticated sessions are being established in WPA/WPA2, it's possible to brute force network passphrase from the initial handshake exchange if captured. During handshake both involved parties are exchanging parameters from which they later calculate same *Pairwise Transient Key — PTK* in format `PTK = PRF (PMK, ANonce, SNonce, MAC(AA), MAC(SA))`. These parameters are random AP nonce, random STA nonce and both their MAC addresses as can be seen on figure 2. All of these parameters are not encrypted and can be easily eavesdropped. The unknown part here is *Pairwise Master Key — PMK*, that both sides know in advance and is never transmitted over network. But as *PMK* format is as following — `PMK = PBKDF2(Passphrase, SSID, 4096, 256)`, the only really unknown part is the network's *passphrase*. This is visualized on Figure 3. Once STA or AP has calculated *PTK*, it starts calculating *Message Integrity Code – MIC* using the *PTK* over subsequent messages and includes it in the message itself. Having all these information from captured handshake, attacker can brute-force the *passphrase* by assembling *PMK* with guessed passphrase, then calculating *PTK* that is then

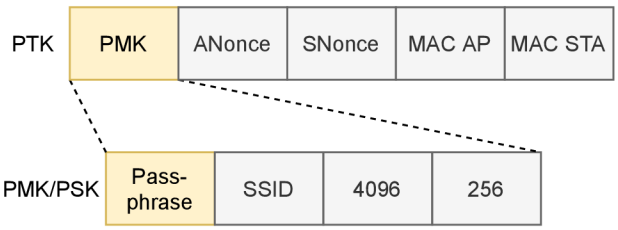


Figure 3. Hierarchy of keys used in WPA authentication. Blue color visualises fields, that are known to attacker as they are transmitted over wireless medium without any encryption. Orange color visualises the secrets, that are unknown to attacker.

used to calculate *MIC* and finally comparing it to actual *MIC* captured from original handshake message. If they match, the passphrase was found. This kind of brute-force attack can be automated by using for example password recovery tool *Hashcat*[2].

As this type of attack require some genuine station to join network at the time when the capture is happening, it's usually combined with deauthentication attack. This, in combination with an automatic rejoin feature that tries to connect device again to the network if it was disconnected, speeds up whole process of obtaining WPA/WPA2 handshake from target network.

2.3 PMKID capture and brute-force attack

Relatively new attack described in 2018 [3] aims on capturing *PMKID* from access point. The format of PMKID is as following `PMKID = HMAC-SHA1(PMK, "PMK Name", MAC_AP, MAC_STA)`. Again, the only unknown secret here is the *PMK*. The *PMKID* is usually being send by APs with roaming feature enabled in the first message of WPA handshake. Hence it can be brute-forced by checking calculated PMKIDs against the original one similarly to the brute-force attack on PMK in classic WPA/WPA2 handshake brute-force attack described in section 2.2. In contrast to classic WPA/WPA2 handshake capture and brute force attack, this approach does not require any STA to be active on the network.

2.4 WPS PIN attack

Another common attack that is still applicable on many networks is taking advantage of the poor design of *Wi-Fi Protected Setup* [10]. WPS was introduced by Wi-Fi Alliance in 2007 to ease setting up WPA2 secured networks and connecting new stations for non-technical users. When station is successfully authenticated, AP transmits WPA/WPA2 passphrase to station so it can then automatically proceed with WPA/WPA2 authentication as if user would provide passphrase manually. Besides other methods, it allows authentication by PIN code that is provided by AP and is the only method that does not require physical access to some authentication authority (for example physical push button or NFC reader) [11].

[2] https://hashcat.net/hashcat/

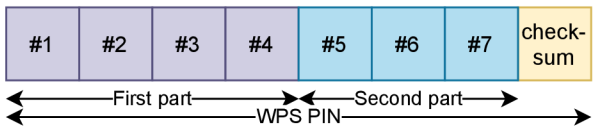


Figure 4. WPS PIN consists of eight digits, where the last digit is a checksum of the first seven digits. During WPS authentication where AP is in Enrollee role, AP is validating PIN in two phases. First it checks first four digits and only if they are correct, it moves forward to validate next three digits. Checksum is calculated over the first seven digits.

Although WPS PIN method can be considered secure in theory, the implementation of it, especially the PIN verification phase makes it vulnerable to brute-force attack. By design the eight-digits long PIN code is split in half. In a first authentication phase, AP validates first four digits and if they are not correct, authentication fails with NACK message from AP. When the first four digits are correct, it proceeds with validation of second half. Second half of PIN is split even further — last digit is checksum of first seven digits of the PIN. This division is visualised on Figure 4.

This validation flow drastically reduces complexity of brute-force attack as instead of 10^8 possible combination (if the full length of the PIN is not predictable and validation is done in a single run) it now requires only $10^4 + 10^3 = 11000$ attempts to exhaust whole space of possible combinations, making this type of attack very efficient. If AP is not protected against this attack by some cool-down period after given number of authentication failures, it may take up to approximately 4 hours to find the correct PIN [10].

2.5 KRACK attack

KRACK (which is an abbreviation of *Key Reinstallation Attack*), first described in 2017, exploits behaviour of stations after receiving third message of WPA handshake [1]. By design of 802.11i standard which is underlying protocol for WPA/WPA2 standard, when station receives third handshake message, it should install PTK calculated after receiving first handshake message from AP [12] — this can be seen as *Key installation* event on Figure 2. This event also resets session parameters like packet nonce or replay counter [1]. Due to presence of *Message Integrity Code* and packet nonce and replay counter in handshake messages, they

cannot be simply captured and replayed later. Man in the Middle (MitM) attack has to be setup to postpone third handshake message from AP with valid relay counter and MIC for later re-transmission to actual STA. At the time of this vulnerability disclosure, the vulnerability severity was multiplied by a bad design of Linux's `wpa_supplicant` in versions 2.4 and 2.5 used in Linux and Android distributions. This implementation caused reinstallation of all-zero key instead of the original PTK generated after first handshake message [1]. As a result, all following communication was encrypted by cryptographically weak all-zero key. This issue was addressed since version 2.6 of `wpa_supplicant` and hence only outdated devices may still be prone to this kind of attack [1]. There is still a space for cryptoanalysis considering reused parameters are being used, but those are out of scope of this work.

2.6 Kr00k attack

One of the most recent vulnerabilities discovered in Wi-Fi networks called *Kr00k* and disclosed in 2019, takes advantage of vulnerability in Broadcom and Cypress Wi-Fi chips design [2]. These chips implement transmit buffer from which some frames may be transmitted encrypted by all-zero key [2]. This occurs, when the vulnerable station is disassociated from AP and clears it's encryption key before all frames are transmitted from transmit buffer [2]. This is not an attack on its own, but if the disassociation is forced intentionally by for example deauthentication attack and frames encrypted by all-zero key are captured, it may target on specific victim on the network and can be considered an attack. Although this was patched on most of the affected chips, it can still be a possible attack vector on some outdated devices.

3. Possibilities of ESP-IDF

ESP-IDF provides powerful API for controlling embedded Wi-Fi interface. In this section the actual implementations of attacks briefly introduced in section 2 are proposed and for some variations are discussed. It also discusses existing projects that may be utilised for attack implementations. This section also covers ESP-IDF limitations that creates some unnecessary obstacles to implement these types of attacks.

3.1 Limitations

Main limitation for working with Wi-Fi interface on ESP32 are closed source *Wi-Fi Stack Libraries*[3]. These libraries incorporate a blocking mechanism that prevents sending arbitrary frames of specific types like deauthentication frames due to various undisclosed reasons. Based on posts from Espressif's employees on official Espressif forum, one of the reasons may be a potential risk of abusing this functionality for deauthentication attacks [13, 14].

Another limitation may be the fact, that ESP32 does not have sufficient power to run offline brute-force attack on captured WPA handshake or PMKID in efficient time. To compensate this limitation, further processing of captured handshake can be reduced by providing captured data directly in a format that is consumable by some third party password recovery tool. For example it can be HCCAPX[4] format, that can be directly passed to well-known password recovery tool *Hashcat*.

3.2 Existing solutions

The idea of utilising Espressif's micro-controllers for Wi-Fi attacks is not new and it was already done on ESP8266, which is predecessor of ESP32. Even though it's different platform incompatible with ESP32, it may still be beneficial to observe their implementation and take advantage of their findings while implementing similar attacks on ESP32 platform.

One of the most activate and well-maintained Wi-Fi attack project on ESP8266 platform is Spacehuhn Tech's *ESP8266 Deauther*[5]. This project implements various attacks, mostly by sending forged frames like deauthentication, probe or beacon frames. This project is limited by ESP8266 platform itself, as it is able to capture frames in promiscuous mode only up to 112 bytes and strips the rest [15]. Hence it cannot implement proper handshake capture.

First try to get better insight into Wi-Fi Stack Libraries on ESP32 was demonstrated in project by Jeija *esp32free80211*[6]. Author of this project partially decompiled Wi-Fi Stack Libraries and was able to figure out a function, that could send raw frames from buffer. Even though in project documentation it's being said that it may allow sending deauthentication packet, it was probably never possible as the blocking mechanism was still in place by Wi-Fi Stack Libraries. Espressif's developers also made limited sending function `esp_wifi_80211_tx` publicly available in ESP-IDF, hence this project is not maintained anymore.

[3] https://github.com/espressif/esp32-wifi-lib

[4] https://hashcat.net/wiki/doku.php?id=hccapx

[5] https://github.com/SpacehuhnTech/esp8266_deauther

[6] https://github.com/Jeija/esp32free80211

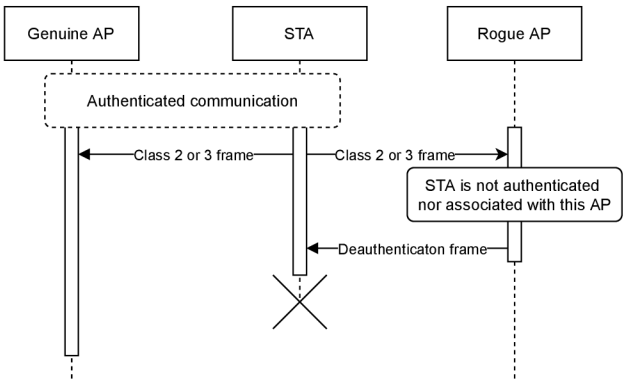


Figure 5. By utilising ESP-IDF option to change MAC address of interface used for AP, duplicated AP can be created that will act as common AP responding to stations that are sending frames to them. If unauthenticated (from AP point of view) station sends class 2 or class 3 frame with matching MAC address (BSSID), AP responds with deauthentication frame and appropriate reason as designed in 802.11 standard. For station receiving this deauthentication frame it's impossible to differentiate between genuine AP and rogue AP and accepts it as valid one disconnecting itself from network.

More recent project from GANESH ICMC/USP organization *esp32-deauther*[7] published in 2019 demonstrated a working solution that bypasses the blocking mechanism in Wi-Fi Stack Libraries by overriding function definition at compilation time. They were able to decompile Wi-Fi Stack Libraries and get a declaration of function that checks validity of frame in transmit buffer — `ieee80211_raw_frame_sanity_check`. Then by using linker flag `-z muldefs` [16] during compilation this method can be simply overridden and always return value `0`, that allows transmission of frame in transmit buffer. Despite the name of the project, this solution actually allows sending not only deauthentication frames, but anything stored in transmit buffer without further validation by Wi-Fi Stack Libraries.

3.3 Deauthentication using ESP-IDF libraries without modifications

To not rely on bypassing Wi-Fi Stack Libraries blocking mechanism by overriding guarding function explained in section 3.2, it's possible to utilise features provided by ESP-IDF only. Native behaviour of access points can be used to send deauthentication frames to stations that try to communicate with genuine AP within it's a range. This is demonstrated in Figure 5.

The main feature that supports this approach is an option to change MAC address of Wi-Fi interface of ESP32. If MAC address of target AP is obtained (it is usually BSSID that is present in most of the frames going through network), it can be configured to ESP32's Wi-Fi interface. If then also SSID is obtained, for example by using built in scanning functionality of `esp_wifi` component, it's possible to create exact copy of targeted AP. Considering an ongoing authenticated communication between targeted AP and some authenticated station, whenever ESP32 AP receives frame from this STA of class 2 or 3, it will automatically respond with deauthentication frame [12]. This is because from ESP32 AP point of view this station is not authenticated and have to authenticate first.

Deauhentication attack can be used for example to exploit Kr00k vulnerability or many other attacks where deauthentication of station is a prerequisite. This approach can be also used on various other systems, where Wi-Fi network interface has no frame injection capability. This approach requires only MAC spoofing option.

3.4 Capturing PMKID

Another possible attack that utilises ESP-IDF only without any modification is PMKID attack. To capture PMKIDs, only the first message of handshake has to be captured. This can be triggered by connecting to target AP with ESP32 in station mode. WPA handshake is always initiated by AP, so if AP has PMKIDs available, it will send them in the first message of the handshake without any prior authentication from the station.

3.5 WPS PIN brute-force attack

Running WPS PIN attack requires working WPS Registrar logic to be implemented on ESP32. In WPS Registrar mode, STA is authenticating itself using pre-shared PIN code provided by AP [11]. Unfortunately ESP-IDF provides only WPS Enrolee mode, hence to run WPS PIN attack, WPS Registrar mode has to be implemented first. This can be done by utilising *esp32-deauther* project introduced in section 3.2, that unblocks sending arbitrary 802.11 frames. It can be used to send all messages until the first part of PIN is confirmed by AP and then proceed with second part and finally get the WPA credentials. Even though the official documentation does not mention WPS Registrar mode in WPA supplicant component, source code exists for this mode in `wpa_supplicant` component[8] and can be used to implement WPS Registrar mode.

[7] https://github.com/GANESH-ICMC/esp32-deauther

[8] https://github.com/espressif/esp-idf/blob/master/components/wpa_supplicant/src/wps/wps_registrar.c

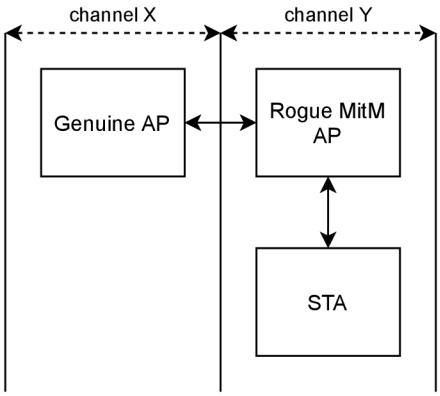


Figure 6. Visualisation of rogue AP in Man in the Middle position. Rogue AP fully duplicates genuine AP using MAC spoofing and using same SSID but operating on different channel. If some STA tries to connect to rogue AP, for example because it has better signal strength, rogue AP is able to manipulate and relay communication between genuine AP and this STA.

3.6 Man in the Middle AP

Wi-Fi interface channel of ESP32 can be switched by using ESP-IDF's function `esp_wifi_set_channel`. In combination with spoofing MAC address of genuine AP, this allows creation of rogue AP clone on different channel. By utilising *esp32-deauther* project mentioned in section 3.2, it's possible to forward and manipulate whole communication between genuine STA and AP. MitM position is demonstrated on Figure 6.

MitM AP can be used for example for KRACK attack described in 2.5. It can be used to postpone transmission of third handshake message that will cause key re-installation event later on STA.

3.7 Bypassing MAC filtering

Even though MAC filtering is not considered a proper security method alone, it's often used as another layer of security creating additional obstacle for potential attacker trying to break into the target network. Although changing MAC address of Wi-Fi interface is commonly available on laptops, it's often not possible on mobile devices. ESP32 can be used to provide an open AP giving any station nearby access to the target network even if they doesn't support changing of their Wi-Fi interface MAC address. ESP-IDF provides a function `esp_wifi_set_mac` that allows changing MAC address of interface. By taking advantage of promiscuous mode, automatic capture of connected STA MAC can be implemented and assuming the target network passphrase is already known, it can connect to network by spoofing captured MAC address.

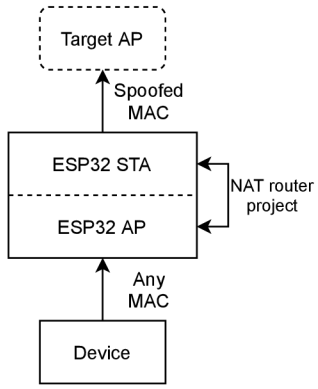


Figure 7. Visualisation of ESP32 configuration used to bypass MAC filtering on AP and open access into target network to devices without MAC changing capability. This approach assumes that the passphrase for target network is already known.

Deauthentication attack proposed in section 3.3 can be used to disconnect genuine stations of which MAC address is being used by ESP32. To setup an AP that then provides access to the target network, existing project *ESP32 NAT Router*[9] can be utilised. This configuration is visualised on Figure 7.

Direct comparison of what modifications different attacks are dependent on can be seen in Table 2.

4. ESP32 Wi-Fi Penetration Tool project

As it came out of the proposed implementations in section 3, various different attacks can be implemented on ESP32. This fact led to an idea to create a universal tool that would wrap shared logic and ease addition of new attack types and their methods to create a comprehensive but lightweight solution for Wi-Fi penetration testing. In this section I will introduce *ESP32 Wi-Fi Penetration Tool* that realizes this idea.

To demonstrate how attacks proposed in section 3 can be actually executed on ESP32, deauthentication attack was picked for that purpose and were implemented using both proposed methods. First method is by utilising *esp32-deauther* project that allows sending arbitrary frames including deauthentication ones. Second method is by creating duplicated rogue AP exploiting native behaviour standardised in 802.11i, where AP should automatically send deauthentication frames whenever it receives class 2 or 3 frame from unauthenticated station.

4.1 Project structure

The project is build on top of the components that ESP-IDF provides and recommends to use [17]. This way,

[9] https://github.com/martin-ger/esp32_nat_router

Table 2. Attack dependencies

Attack	Dependent on
Deauth	ESP-IDF only (using rogue AP) or WSL bypassing (forging deauth frames)
WPS PIN bruteforce	WPS Registrar mode
KRACK	MitM AP and WSL bypassing
Kr00k	Deauth attack
MAC filtering bypasser	ESP-IDF only (can utilise esp32-nat-router)

shared parts like Wi-Fi controller or frame analyzer are being implemented in standalone components that can be easily reused either directly in this project or even in other project by simply copying them and reusing them out of the box. ESP32 Wi-Fi Penetration Tool project consists of following components:

Wi-Fi controller Component that handles all Wi-Fi interface related operations and provides simplified interface. It's used to initialize Wi-Fi interface, to control access point and station configurations, to switch Wi-Fi interface into promiscuous mode and other similar operations.

Frame analyzer Main purpose of this component is to parse frames captured by Wi-Fi controller and do an analysis of these frames. For example it does parsing of PMKIDs from EAPOL packets, detected encrypted frames, filter frames by BSSID and passes the results into event pool. It also provides a parsing functionality for other components like HCCAPX serializer.

WSL Bypasser This component is used to unblock raw frame transmission by overriding blocking function in Wi-Fi Stack Libraries. This component is based on an existing project *ESP32-deauther*.

Webserver Webserver component provides an UI to control the tool itself and allows configuration of available attacks. It's build on top of ESP-IDFs *HTTP server* component. Sample of user interface is shown on Figure 8.

PCAP and HCCAPX serializers These two small components fromat captured frames into a common formats like PCAP for further analysis in Wireshark or other tools or HCCAPX for direct use with password recovery tool *Hashcat*.

Main The main component groups all the attack types and variations alongside with a universal attack handler, that takes care of timeouts, configurations and other shared operations.

4.2 Attack wrapper

The attack logic itself lives inside Main component. It handles shared attack operations like attack timeout and abort calls. It obtains configuration via Webserver component from user input in JavaScript powered web client and executes appropriate attacks and their methods based on this configuration. When the attack finishes, it returns results of the attack to the client that displays them to user.

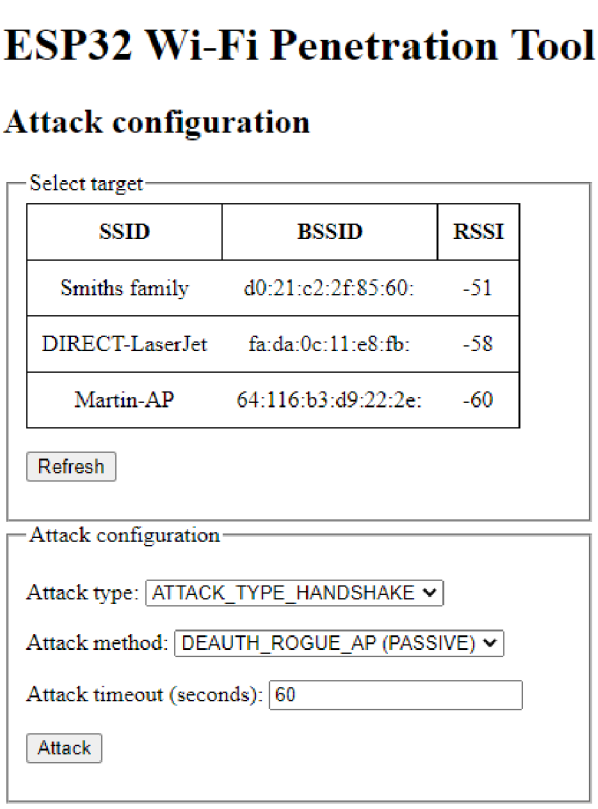


Figure 8. User interface for universal ESP32 Wi-Fi penetration tool that shows scanned APs in ESP32 surrounding and allows configuration of the attack itself by choosing attack type, method and timeout.

When adding new attack type or method, all the programmer have to do is add this new attack into enumeration of available attack types and methods in main component and in JavaScript client. Then it's a matter of calling appropriate functions from components that do required operations and implementing what is not already included.

4.3 General use-case scenario

To execute available attacks on surrounding APs with this tool, when user powers the ESP32, management AP is automatically started when ESP32 boot completes. User then can connect to this management AP using for example his cellphone. He's provided with a web application, that allows him to pick a target AP in his vicinity and choose attack type he wants to execute.

When the automated attack finishes, web application shows a result of this attack to user, from where he can for example download PCAP or HCCAPX files or see a reason for attack failure. This depends on specific attack type implementation.

4.4 Evaluation

ESP32 is not sufficient to run brute-force attacks so this has to be done on external machine with reasonable power. However this is not efficient even on standard laptops or computers. Brute-force cracking can be outsourced to some cloud service like AWS EC2 P3 instances[10].

ESP32 with ESP32 Wi-Fi Penetration Tool can reliably run deauthentication attacks. Some devices may ignore broadcast deauthentication frames or be protected against some types of attacks. It can be solved by combining different approaches when attacking against given network which is supported by this tool by its design. There was not detected any significant frame loss. Handshakes and PMKIDs were always captured.

Flashing this project onto ESP32 can be done by running simple binary included in ESP-IDF toolchain or by using standalone flashing tool with graphical user interface[11]. It can also be distributed pre-flashed, ready to work out of the box. In both cases it doesn't require almost any domain knowledge from the user. The attacks themselves are automated and everything the user has to do is choosing target AP/network and the type of attack he wants to run.

ESP32 as an ultra-low power platform can be efficiently powered by batteries, or smartphone which makes the solution portable and inconspicuous. Being less noticeable in the public by controlling the attacks using for example a smartphone can open new attack vectors to the attacker. Also thanks to the small dimensions and light weight of the ESP32-WROOM module alone, it makes it easily attachable to a small drone by which attacker can reach otherwise physically unreachable networks[12].

By experimental measurement while executing attacks implemented in Wi-Fi Penetration Tool, ESP32 consumes from 90 to 110 mA of power. Considering two standard Li-ion batteries 18650 with capacity of 2600 mAh, this allows approximately 24 hours[13] of active attack. That is enough to run long running online brute-force WPS attack proposed in section 3.5 taking up to 4 hours[10]. For practical example, on Figure 1, I have used a Li-Pol accumulator with capacity of 220 mAh, that is able to provide enough power for about two hours of active usage.

5. Conclusions

In this paper I have explored possibilities of portability of common Wi-Fi attacks to ESP32 platform. It was demonstrated that attacks mentioned in this work are possible to be implemented and executed on ESP32 thanks to capabilities like changing Wi-Fi interface MAC address, configuring access points and stations, promiscuous mode with an optional bypass of Wi-Fi Stack Libraries. An universal ESP32 Wi-Fi Penetration Tool was introduced which acts as a wrapper for further attack implementations.

Outcome of this work supports arguments why current Wi-Fi vulnerabilities has to be taken seriously as they are easily executable on cheap and accessible hardware. Considering that resources for Wi-Fi attacks may be really lightweight both in terms of price, weight and size it allows anyone with minimal knowledge about the topic — often referred to as *script kiddie* — to run potentially harmful attacks against Wi-Fi networks in their vicinity. It may raise awareness of how easily some of the attacks can be implemented and why one should not rely on obsolete 802.11 implementations and outdated security features and should upgrade their hardware (routers, cellphones and other Wi-Fi capable devices) and software to include latest security measures like WPA3 or 802.11w amendment.

As an demonstration of utilising official ESP-IDF framework for Wi-Fi attack implementations I have implemented deauthentication attack with handshake frames capture by creating rogue access point that duplicates genuine one and rely on native AP behaviour defined in 802.11 standard. I have also implemented other variants of these attack by assembling and sending deauthentication frames directly from code and an PMKID capture attack by simply connecting to

[10] https://aws.amazon.com/ec2/instance-types/p3/

[11] Windows OS application — https://www.espressif.com/en/support/download/other-tools

[12] e.g., networks in some buildings behind fence or in higher floors in office buildings that cannot be accessed by public

[13] excluding step-up voltage regulator power consumption

target AP and parsing PMKIDs from first handshake message.

This work provides a tool for further extensions and experimentation with Wi-Fi attacks on cheap and low powered ESP32 SoC. It shows that attacks can be done on small hardware powered with small battery which opens new ways how the attackers can execute attacks. Components structure in the project provides reusability and may simplify further researches of new attacks. Next steps may be implementation of WPS attack or use ESP32 to bypass MAC address filtering on AP.

ESP32 Wi-Fi Penetration Tool was published as an open-source project on GitHub[14]. The repository includes also more detailed description of the components and overall functionality available also in form of GitHub Pages[15].

Acknowledgements

I would like to thank my supervisor Ing. Jan Pluskal for his valuable inputs during my work.

References

[1] Mathy Vanhoef and Frank Piessens. Key reinstallation attacks: Forcing nonce reuse in wpa2. In *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, CCS '17, page 1313–1328, New York, NY, USA, 2017. Association for Computing Machinery.

[2] Róbert Lipovský, Svorenčík, and Miloš Čermák. Kr00k - cve-2019-15126. *We live security*, page 9, 2020.

[3] New attack on wpa/wpa2 using pmkid. online, https://hashcat.net/forum/thread-7717-post-41427.html#pid41427.

[4] Michael Asante and Kwabena Akomea-Agyin. Analysis of security vulnerabilities in wifi protected access pre shared key (wpa psk/ wpa2 psk). *International Research Journal of Engineering and Technology (IRJET)*, 6(1):537 – 545.

[5] Kody. Enable monitor mode & packet injection on the raspberry pi. online, https://null-byte.wonderhowto.com/how-to/enable-monitor-mode-packet-injection-raspberry-pi-0189378/.

[6] RASPBERRY PI FOUNDATION. Faqs. online, https://www.raspberrypi.org/documentation/faqs.

[7] *IEEE Standard for Information Technology-Telecommunications and Information Exchange Between Systems- Local and Metropolitan Area Networks- Specific Requirements- Part 11*. New York, USA, 1999 ed. edition, 2003.

[8] *IEEE Standard for Information technology - Telecommunications and information exchange between systems - Local and metropolitan area networks - Specific requirements. Part 11*. USA, 2009 ed. edition, 2009. 802.11w.

[9] Hans Matthé. 802.11w – does it work? online, https://www.boostyourwifi.be/2018/07/04/802-11w-does-it-work/.

[10] Stefan Viehböck. Brute forcing wi-fi protected setup. 2011.

[11] *Wi-Fi Protected Setup Specification*, 1.0h edition, 2006. online, https://www.wi-fi.org/discover-wi-fi/wi-fi-protected-setup.

[12] *IEEE Standard for information technology-Telecommunications and information exchange between systems-Local and metropolitan area networks-Specific requirements-Part 11*. New York, USA, 2004 ed. edition, 2004. 802.11i.

[13] Angus Gratton. esp_wifi_internal. online, https://www.esp32.com/viewtopic.php?t=586.

[14] Ivan Grokhotkov and Jeroen Domburg. Please open source this library. online, https://github.com/espressif/esp32-wifi-lib/issues/2.

[15] *ESP8266EX*, 6.6 (2020.10) edition, 2020. online, https://www.espressif.com/sites/default/files/documentation/0a-esp8266ex_datasheet_en.pdf.

[16] *ld(1) - Linux man page*. online, https://linux.die.net/man/1/ld.

[17] *ESP32*, 2020. online, https://www.espressif.com/sites/default/files/documentation/esp32_technical_reference_manual_en.pdf.

[14] https://github.com/risinek/esp32-wifi-penetration-tool

[15] https://risinek.github.io/esp32-wifi-penetration-tool/