Česká zemědělská univerzita v Praze

Provozně ekonomická fakulta

Katedra informačního inženýrství

Diplomová práce

Vývoj mobilní aplikace pro platformu Android na testování WPS zranitelnosti WiFi sítí

Bc. Radek Homola

ČESKÁ ZEMĚDĚLSKÁ UNIVERZITA V PRAZE

Provozně ekonomická fakulta

ZADÁNÍ DIPLOMOVÉ PRÁCE

Bc. Radek Homola

Informatika

Název práce

Vývoj mobilní aplikace pro platformu Android na testování WPS zranitelnosti WiFi sítí

Název anglicky

Development of mobile application for Android platform for testing WPS vulnerability of WiFi networks

---

Cíle práce

Diplomová práce je tematicky zaměřena na problematiku vývoje mobilní aplikace pro platformu Android. Hlavním cílem práce je vývoj funkční aplikace, sloužící k testování zranitelnosti WiFi sítí při odhalení čísla WPS PIN.

Cílem teoretické části práce je vytvořit ucelenou rešerši literárních zdrojů, zaměřených na návrh a vývoj aplikací pro platformu Android, bezpečnost a analýzu standardních zabezpečovacích technologií WiFi sítí, analýzu již existujících obdobně zaměřených aplikací a jejich otestování dynamickou metodou ”černá skříňka”.

Cílem praktické části je návrh objektové struktury a uživatelského rozhraní aplikace, implementace dle návrhu a otestování vzniklého prototypu aplikace.

Metodika

Metodika řešené problematiky diplomové práce je založena na studiu a analýze odborných informačních zdrojů zabývajících se vývojem mobilních aplikací pro platformu Android, a zabezpečením WiFi sítí. Vlastní práce spočívá v rešerši odborné literatury vztahující se k těmto tématům a na to navazující praktické části, týkající se vývoje konkrétní aplikace. Na základě syntézy teoretických poznatků a výsledků praktické části budou formulovány závěry diplomové práce.

Doporučený rozsah práce

60 – 80 stran

Klíčová slova

mobilní aplikace, vývoj, Android, WiFi, protected setup, WPS, PIN, zranitelnost, grafické rozhraní

Doporučené zdroje informací

ALLEN, G. *Android 4 : průvodce programováním mobilních aplikací.* Brno: Computer Press, 2013. ISBN 978-80-251-3782-6.

HEROUT, P. *Učebnice jazyka Java.* České Budějovice: Kopp, 2008. ISBN 978-80-7232-355-5.

LACKO, Ľ. Vývoj aplikací pro Android. Brno: Computer Press, 2015. ISBN 978-80-251-4347-6.

PECINOVSKÝ, R. *Myslíme objektově v jazyku Java : kompletní učebnice pro začátečníky.* Praha: Grada, 2009.

PUŽMANOVÁ, R. Bezpečnost bezdrátové komunikace: Jak zabezpečit Wi-Fi, Bluetooth, GPRS či 3G. vyd. 1. Brno: Computer Press, 2005. ISBN 80-251-0791-4.

VÁVRŮ, Jiří a Miroslav UJBÁNYAI. Programujeme pro Android. Praha: Grada Publishing, 2013. ISBN 978-80-247-4863-4.

Předběžný termín obhajoby

2017/18 LS – PEF

Vedoucí práce

Ing. Dana Vynikarová, Ph.D.

Garantující pracoviště

Katedra informačního inženýrství

Elektronicky schváleno dne 11. 1. 2018

Ing. Martin Pelikán, Ph.D.

Vedoucí katedry

Elektronicky schváleno dne 11. 1. 2018

Ing. Martin Pelikán, Ph.D.

Děkan

V Praze dne 22. 03. 2018

Čestné prohlášení

Prohlašuji, že svou diplomovou práci „Vývoj mobilní aplikace pro platformu Android na testování WPS zranitelnosti WiFi sítí“ jsem vypracoval samostatně pod vedením vedoucího diplomové práce a s použitím odborné literatury a dalších informačních zdrojů, které jsou citovány v práci a uvedeny v seznamu použitých zdrojů na konci práce. Jako autor uvedené diplomové práce dále prohlašuji, že jsem v souvislosti s jejím vytvořením neporušil autorská práva třetích osob.

V Praze dne 28. března 2018 _____

Poděkování

Mé poděkování patří především vedoucí práce paní Ing. Daně Vynikarové, Ph.D. za četné odborné rady a trpělivost během celého vývoje práce. Dále bych rád poděkoval mé rodině a mému nadřízenému Ing. Rostislavu Cendelínovi za podporu a motivaci.

Vývoj mobilní aplikace pro platformu Android na testování WPS zranitelnosti WiFi sítí

Souhrn

Diplomová práce pojednává o procesu vývoje mobilní aplikace pro operační systém Android s výhradním použitím standardních komponent, které tato platforma nabízí. Cílem bylo vytvořit funkční prototyp aplikace, která bude umožňovat základní otestování domácí WiFi sítě na zranitelnost WPS, konkrétně odhalení PIN čísla přístupového bodu (routeru), čímž dojde k připojení do sítě bez znalosti hesla. Cíl se podařilo splnit, avšak z implementace byla z důvodu omezení plynoucích z architektury operačního systému Android vynechána metoda odhalení čísla PIN hrubou silou.

Klíčová slova: vývoj mobilní aplikace, Android, WiFi, protected setup, WPS, PIN, zranitelnost, grafické rozhraní

Development of mobile application for Android platform for testing WPS vulnerability of WiFi networks

Summary

The master's thesis describes the process of mobile application development for operating system Android with usage of standard components, which this platform includes only. The main goal was to develop operational prototype of application, which will enable its user to test WPS vulnerability, specifically disclosuring access point's PIN in his own home WiFi network. That causes the end device to connect to the network without knowing the password. The goal has been met without implementing the brute force functionality because of restrictions according to the architecture of Android operating system.

Keywords: mobile application development, Android, WiFi, protected setup, WPS, PIN, vulnerability, graphical interface

Obsah

Seznam obrázků

Seznam tabulek

1 Úvod

V dnešní době rozmachu moderních informačních technologií neustále přibývá počet bezdrátových WiFi sítí. Zvláště v domácnostech a malých podnicích postupně nahrazují původní sítě metalické. Bezdrátový přenos totiž nabízí uživatelům mnohem větší komfort, zejména z hlediska mobility. Naproti tomu s sebou ovšem přináší nová úskalí a bezpečnostní rizika, která by neměla být podceňována. Je třeba mít na zřeteli, že provoz sítě již neprobíhá pouze v uzavřeném fyzickém médiu přenosového vodiče, jehož vedení zpravidla nepřesahuje území provozovatele sítě. Všesměrově vysílaný signál WiFi sítě lze zachytávat i na místech vně takového území, přičemž ze všech míst v jejím dosahu je na síť možné provádět útoky. Postupem času došlo sice k výraznému vývoji a zlepšení v oblasti zabezpečovacích mechanismů WiFi, aktuálně však stále na vlastníky sítí, zejména méně znalé v oblasti informačních technologií, číhá bezpečnostní slabina v podobě WPS, umožňující mimo jiné připojení k síti pomocí osmimístného čísla namísto běžného hesla.

Motivací pro výběr tématu práce bylo poukázání na tuto často poměrně opomíjenou technologii jako možné úskalí při zabezpečování domácí WiFi sítě, a zároveň možnost převedení teoretických poznatků do implementace reálné aplikace pro mobilní zařízení s dlouhodobě nejrozšířenějším systémem Android tak, aby bylo možné provést základní prověření zranitelnosti WiFi sítě bez nutnosti použití osobního počítače. Aplikace je tak určena všem uživatelům provozujícím domácí WiFi síť, kteří chtějí pomocí mobilního zařízení provést základní prověření WPS zranitelnosti své sítě.

Teoretická část diplomové práce se zaměřuje na problematiku bezpečnosti WiFi sítí a všechny soudobé zabezpečovací mechanismy od WEP po WPA2, přičemž se detailně soustředí právě na technologii WPS, kterou představuje jako potenciální bezpečnostní riziko. Následuje charakteristika komponent operačního systému Android použitých v implementaci praktické části práce a analýza dostupných, obdobně zaměřených aplikací. Praktická část práce je pak věnována návrhu, implementaci a otestování vzniklého prototypu aplikace.

2 Cíl práce a metodika

2.1 Cíl práce

Diplomová práce je tematicky zaměřena na problematiku vývoje mobilní aplikace pro platformu Android. Hlavním cílem práce je vývoj funkční aplikace, sloužící k testování zranitelnosti WiFi sítí při odhalení čísla WPS PIN.

Cílem teoretické části práce je vytvořit ucelenou rešerši literárních zdrojů, zaměřených na návrh a vývoj aplikací pro platformu Android, bezpečnost a analýzu standardních zabezpečovacích technologií WiFi sítí, analýzu již existujících obdobně zaměřených aplikací a jejich otestování dynamickou metodou „černá skříňka“.

Cílem praktické části je návrh objektové struktury a uživatelského rozhraní aplikace, implementace dle návrhu a otestování vzniklého prototypu aplikace.

2.2 Metodika

Metodika řešené problematiky diplomové práce je založena na studiu a analýze odborných informačních zdrojů zabývajících se vývojem mobilních aplikací pro platformu Android, a zabezpečením WiFi sítí. Vlastní práce spočívá v rešerši odborné literatury vztahující se k těmto tématům a na to navazující praktické části, týkající se vývoje konkrétní aplikace. Na základě syntézy teoretických poznatků a výsledků praktické části budou formulovány závěry diplomové práce.

3 Teoretická východiska

3.1 WiFi

Principy komunikace v bezdrátových počítačových lokálních sítích WLAN (*Wireless Local Area Network*) jsou popsány souborem protokolů a standardů s názvem IEEE 802.11. Jak z názvu vyplývá, jedná se o sítě, ve kterých se jako komunikační médium užívá výhradně radiový signál, a rozprostřené spektrum jako prostředek k potlačení šumu. Tyto sítě vysílají signál v bezlicenčním pásmu ISM (*Industrial, Scientic and Medicalband*) určeném pro průmyslové, vědecké a lékařské účely. V tomto pásmu na frekvenci 2.4 GHz pracují také např. zařízení standardu IEEE 802.15 (sítě WPAN, např. technologie Bluetooth), či mikrovlnné trouby. Potlačení šumu je tedy pro správné fungování sítí z pásma ISM nezbytné.

Standardizační dokument 802.11 byl poprvé publikován institutem IEEE v roce 1997, a v roce 1999 rozšířen o další dvě specifikace 802.11a a 802.11b. Zde byl poprvé užit a ustálen pojem „WiFi“ používaný jako obchodní značka zastřešující tuto technologii. V roce 1999 došlo také v americkém Texasu k založení aliance WECA (*Wireless Ethernet Compatibility Alliance*), která byla roku 2002 přejmenována na WiFi Alliance, a která se vývojem standardů pro bezdrátovou komunikaci zabývá dodnes. Hlavními sponzory aliance jsou všichni klíčoví výrobci WiFi zařízení (Apple, Samsung, Cisco, Qualcomm, ...), ale i např. společnosti Microsoft a T-Mobile.

Do dnešní doby bylo vydáno několik inovací tohoto standardu označených písmeny (viz Tabulka 1). Každá zavádí určitý způsob modulace radiového signálu při použití stejného protokolu. Zvyšuje se v čase především deklarovaná maximální rychlost přenosu dat. Některé standardy umožňují přenos i na frekvenci 5 GHz, kde nedochází k interferenci s frekvencí 2.4 GHz. U koncových zařízení dodnes zůstávají nejčastěji podporovanými standardy 802.11b/g/n.

Tabulka 1: Přehled standardů IEEE 802.11 [1]

Označení	**Rok vydání**	**Pásmo [GHz]**	**Max. rychlost [Mbps]**
IEEE 802.11	1997	2.4	2
IEEE 802.11a	1999	5	54
IEEE 802.11b	1999	2.4	11
IEEE 802.11g	2003	2.4	54
IEEE 802.11n	2009	2.4 nebo 5	600
IEEE 802.11y	2008	3.7	54
IEEE 802.11ac	2013	2.4 a 5	1000
IEEE 802.11ad	2012	2.4, 5 a 60	7000
IEEE 802.11ax	v návrhu (2019)	2.4 a 5	10000

3.1.1 Identifikace sítě

Každá WiFi síť je opatřena vlastním SSID (*Service Set Identifier*) – identifikátorem (nikoli však jednoznačným) bezdrátové sítě v podobě textového řetězce znaků ASCII s maximální délkou 32 znaků. Přístupový bod obvykle tento identifikátor periodicky vysílá pro všechna zařízení v okolí, která podle něj mohou síť detekovat a případně se k ní připojit. Vysílán je v tzv. *Beacon rámci* zpravidla 10× – 100× za sekundu. Tento rámec kromě identifikátoru sítě obsahuje i informace o síle jejího signálu, podporovaných frekvencích, zabezpečovacím algoritmu, MAC adresu přístupového bodu apod.

Prakticky všechny dnešní přístupové body však v nastavení obsahují možnost SSID do okolí nevysílat (před veřejností síť skrýt), což lze považovat za jedno z nejefektivnějších opatření pro zvýšení zabezpečení sítě. Na druhou stranu je pak ale nezbytné, aby uživatel při připojování nového zařízení k síti toto SSID znal a zadal ho ručně, jinak je pro zařízení „neviditelná“. Možnost ručního zadání však řada jednodušších koncových zařízení vůbec nenabízí, a proto skrývání SSID lze považovat za nepříliš komfortní opatření.

3.1.2 Proces připojování do sítě

Proces připojování bezdrátového zařízení do WiFi sítě probíhá standardně v pěti dílčích krocích:

- **Skenování.** Zařízení přijímá a analyzuje veřejně dostupné Beacon rámce, tedy informace o všech okolních sítích v dosahu, na jejichž základě bude následně zvolena jedna konkrétní cílová síť. Skenování dostupných sítí klientem může probíhat pasivně, či aktivně:
  - **Pasivní skenování.** Klient postupně na všech jednotlivých frekvenčních kanálech naslouchá Beacon rámcům. Po přijetí je schopen iniciovat proces připojení do sítě. Přijme-li těchto rámců vícero od většího počtu přístupových bodů náležících stejné síti, může se rozhodnout podle parametru síly signálu uvedeného ve všech přijatých rámcích. Na jeho základě pak určí jeden konkrétní bod, ke kterému bude navázáno spojení.
  - **Aktivní skenování.** Pokud je aktivní skenování na připojovaném zařízení zapnuto, klient opět postupně prochází všechny frekvenční kanály, ale místo pasivního naslouchání na ně odesílá tzv. *Probe Request* rámce. Tyto rámce jsou odesílány jako broadcast s cílem aktivně zjistit, jaké sítě aktuálně na daném kanálu vysílají.
- **Spojování.** Klient navazuje komunikaci s přístupovým bodem.
- **Autentizace.** Po úspěšném navázání komunikace klient iniciuje své ověření žádostí zaslanou směrem k přístupovému bodu. Ten ji buď může sám vyhodnotit a připojované zařízení přijmout či zamítnout, nebo žádost postoupit dále autentizačním serverům v případě rozsáhlejších sítí.
- **Asociování.** V případě úspěšné autentizace klienta dojde z jeho strany k odeslání tzv. *Association Request rámce*. Pokud je přijatá odpověď v *Associate Response rámci* od přístupového bodu kladná, dojde k asociaci. Od toho momentu je klient oprávněn skrze síť komunikovat.

- **Reasociování.** K této fázi může dojít pouze u rozlehlejších WiFi sítí (tzv. *ESS, Extended Service Set*), jejichž signál je vysílán pomocí vícero přístupových bodů. Klient se může v průběhu komunikace reasociovat s jiným přístupovým bodem, který mu v danou chvíli začne poskytovat lepší kvalitu signálu, než stávající. [2]

3.2 Zabezpečení WiFi sítí

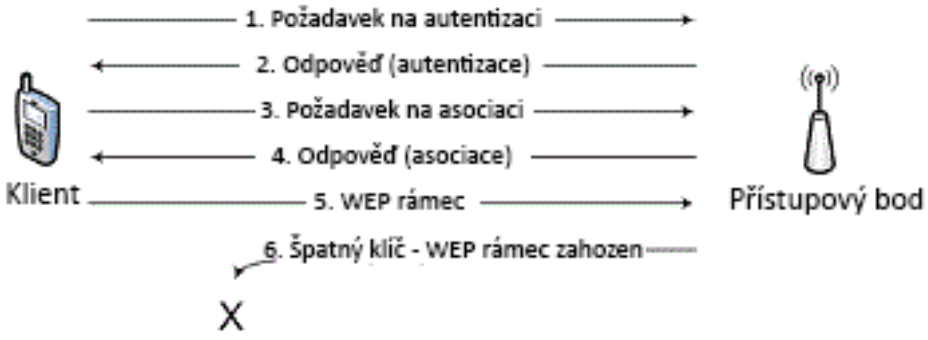
Techniky zabezpečení WiFi sítí lze obecně rozdělit do tří oblastí – autentizace, šifrování a integrita přenosu. Každý uživatel, resp. každé do sítě nově připojované zařízení by mělo být autentizováno, nejčastěji zadáním přístupového hesla. Následně by mělo být uvnitř sítě zajištěno šifrování datových přenosů tak, aby nemohlo dojít k jejich odposlouchávání a integrita, tedy zabezpečení přenášených dat proti modifikaci během přenosu.

3.2.1 WEP

Wired Equivalent Privacy (WEP) je prvním bezpečnostním algoritmem použitým již v prvním vydání standardu IEEE 802.11 v roce 1997. Jeho hlavním cílem bylo, jak už z názvu vyplývá, zajištění důvěrnosti a bezpečnosti přenosu dat srovnatelné s tehdejšími klasickými metalickými sítěmi. Použití WEP bylo jedinou možnou volbou pro zařízení pracující na standardech 802.11a a 802.11b, implementovaný je i na zařízeních 802.11g. V roce 2001 došlo k prolomení tohoto algoritmu.

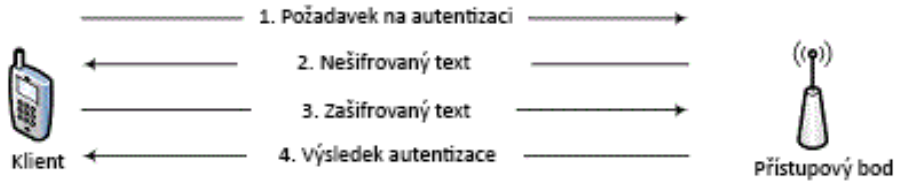
3.2.1.1 Autentizace

Autentizace probíhá dvěma způsoby: otevřeně (tzv. *Open System Authentication*), nebo pomocí sdíleného klíče (tzv. *Shared Key Authentication*). V prvním případě se jedná o jednoduchou žádost o ověření ze strany klienta obsahující jeho identifikátor. Žádost je následována kladnou, či zápornou odpovědí od přístupového bodu. Od momentu přijetí kladné odpovědi se obě zařízení považují za vzájemně ověřené. Ve skutečnosti lze tedy říci, že k žádné reálné autentizaci v podstatě nedochází. Řídící autentizační rámce tohoto protokolu jsou navíc zasílány v podobě prostého textu, takže není pro klienta obtížné zaslat podvržený rámec, kde se může vydávat za klienta jiného. [3]



Obrázek 1: Výměna zpráv během procesu otevřené autentizace [4]

Ve druhém zmíněném případě je žádost o ověření klientem před odesláním zašifrována pomocí tajného klíče. Autentizace proběhne v pořádku, pokud přístupový bod dokáže správně dešifrovat text žádosti. Tímto procesem je tedy možné ověřit pouze zařízení, nikoli však uživatele. Ověřování klienta je navíc jednostranné, tedy nemá možnost jak se ujistit, že se ověřuje vůči autorizovanému přístupovému bodu. [5]



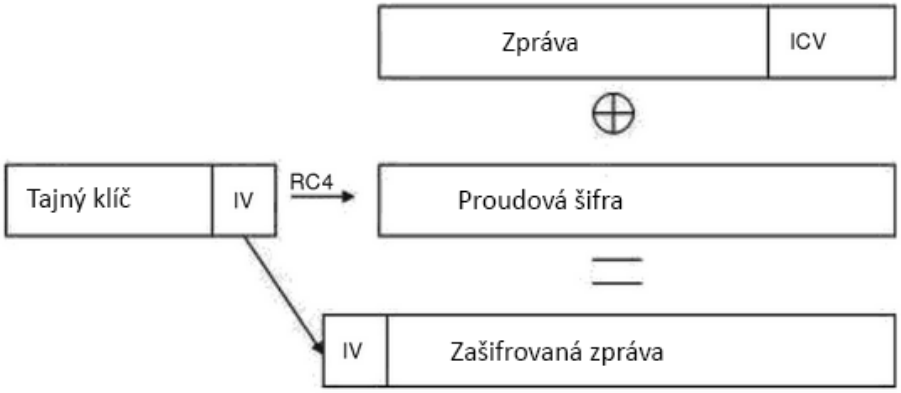
Obrázek 2: Výměna zpráv během procesu autentizace sdíleným klíčem [4]

3.2.1.2 Šifrování a integrita dat

Šifrování WEP je symetrické, používá se stejný algoritmus a RC4 (*Rivest Cipher No. 4*) klíč pro šifrování i dešifrování dat. Ten má délku buď 64 bitů, nebo 128 bitů a skládá se z tzv. *inicializačního vektoru* (vždy 24 bitů) a vlastního tajného klíče (40 bitů nebo 104 bitů).

Inicializační vektor spolu s tajným klíčem tvoří dva vstupy generátoru symetrické proudové šifry (*Stream Cipher*) pro které platí, že inicializační vektor se musí v čase měnit, a naopak tajný klíč zůstává po dobu komunikace neměnný, předem určený správcem sítě. Před vlastním zašifrováním zprávy je z jejího obsahu vytvořen 32 bitů dlouhý kontrolní součet ICV (*Integrity Check Value*), který je následně připojen ke zprávě. Na takto upravené zprávě je poté provedena operace XOR vůči proudové šifře,

čímž vznikne výsledná zašifrovaná zpráva určená k přenosu. Funkce šifrování WEP je znázorněna na obrázku 3. [3]



Obrázek 3: Princip šifrování WEP [6]

Dešifrování zprávy na straně příjemce probíhá analogicky v opačném pořadí. Po získání původního obsahu zprávy je z něj opět vypočítán kontrolní součet, který je následně porovnán se součtem přijatým ve zprávě (od odesílatele). V případě shody jsou data akceptována, v opačném případě odmítnuta. WEP při přenosu šifruje pouze uživatelská data a jejich kontrolní součty, řídící data šifrována nejsou. [6]

3.2.1.3 Zranitelnost

Z důvodu řady bezpečnostních nedokonalostí je WEP poměrně snadno prolomitelný. První úspěšný útok byl popsán v roce 2001. Postupem času bylo sice zabezpečení WEP několikrát vylepšeno (např. WEP+, WEP2), v praxi to ale vždy znamená pouze prodloužení času potřebného k jeho prolomení. Velice náchylný je na útoky hrubou silou a slovníkové útoky, které usilují o prolomení tajného klíče.

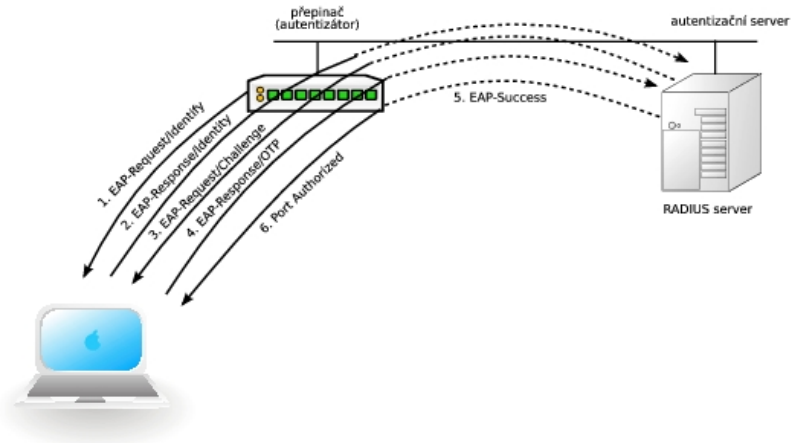
Dalším typem útoků, kterými WEP trpí je tzv. FMS (*Fluhrer, Mantin and Shamir*, dle jmen autorů) útok založený na získání RC4 klíče pomocí odposlouchávání probíhající komunikace v síti. Doba prolomení je tedy u těchto útoků závislá pouze na počtu přijatých dat, tedy na intenzitě komunikace v dané síti, přičemž je však útočník schopen, v případě nízké četnosti přenášených paketů, komunikaci podnítit zasíláním paketů vlastních.

3.2.2 IEEE 802.1x

IEEE 802.1x je standard definující pokročilý autentizační mechanismus při připojování zařízení do lokální sítě, s cílem zamezit komunikaci v síti uživatelům bez dostatečného oprávnění. Vznikl již roku 2001, ale implementovat se začal až v roce 2004. Využití bylo zprvu zamýšleno pouze při přístupu do metalických sítí, záhy bylo však rozšířeno i pro sítě bezdrátové. Standard zapouzdřuje protokol EAP (*Extensible Authentication Protocol*), definovaný v témže roce dokumentem RFC 3748, do ethernetových rámců EAPOL (*EAP Over LAN*). EAP je rozšiřitelný autentizační mechanismus, který umožňuje implementovat různé druhy autentizace, viz dále kapitola 3.2.3.

V mechanismu řízení přístupu k síti 802.1x figurují v základu tři entity, mezi nimiž se proces autentizace odehrává:

- **Suplikant (žadatel).** Žadatele představuje koncová stanice a uživatel připojující se do sítě (žádající o povolení k přístupu).
- **Autentizátor.** V případě bezdrátových sítí se autentizátorem rozumí přístupový bod, který realizuje propouštění či blokaci síťové komunikace suplikanta. Zároveň plní úlohu prostředníka mezi suplikantem a autentizačním serverem.
- **Autentizační server.** Jedná se nejčastěji o server RADIUS (*Remote Authentication Dial In User Service*), který poskytuje autentizační informace autentizátoru. [7]



Obrázek 4: Výměna zpráv během autentizace 802.1x [7]

3.2.2.1 Proces autentizace

Pokud se suplikant (klient) připojuje k autentizátoru a žádá o přístup do sítě, je pro něj blokována veškerý síťový provoz kromě autentizačního EAP protokolu. K odblokování ostatních komunikačních protokolů dojde pochopitelně až ve chvíli úspěšného ověření. Proces (viz Obrázek 4) probíhá následovně:

(i) suplikant naváže spojení s autentizátorem,

(ii) suplikant sestaví a odešle autentizační data o sobě samém autentizátoru,

(iii) autentizátor předá data autentizačnímu serveru,

(iv) na serveru proběhne ověření dat o uživateli,

(v) server předá autentizátoru informaci o výsledku ověření, a ten buď komunikaci suplikantovi odblokuje, či ponechá zablokovanou.

3.2.3 EAP

Extensible Authentication Protocol (EAP) je autentizační framework využívaný zejména v bezdrátových sítích, a také v point-to-point spojeních. Nejedná se přímo o síťový protokol, ale pouze o definici formátů zasílaných zpráv. Konkrétně standard IEEE 802.1x pracuje s pěti variantami EAP, a přijímá je jako oficiální autentizační mechanismy. Celkový počet definovaných a používaných variant EAP v sítích obecně se však pohybuje kolem 40.

3.2.3.1 Autentizační metody

EAP definuje způsob výměny a formát zpráv vyměňovaných mezi suplikantem a autentizátorem během procesu autentizace. Nedefinuje ovšem žádná bezpečnostní opatření a mechanismy pro tento proces – předpokládá, že komunikační kanál je zabezpečený. Z toho důvodu je v rámci 801.1x využíván v pěti variantách, vždy ve spojení s určitým zabezpečovacím mechanismem. [3]

3.2.3.2 EAP–MD5

EAP spolu s hashovací funkcí MD5 byl použit jako vůbec první autentizační mechanismus 802.1x, definovaný ve standardu RFC 2284 pro EAP. Zajišťuje pouze autentizaci suplikanta vůči autentizátoru, nikoli vzájemnou, a proto je náchylný na útoky typu MITM (*Man-In-The-Middle*). Pro ověření musí uživatel vždy zadat heslo, takže je tento mechanismus zranitelný i slovníkovými útoky.

3.2.3.3 EAP–TLS

Bezpečnostní mechanismus *EAP Transport Layer Security* (EAP–TLS), definovaný standardem RFC 5216, patří k nejbezpečnějším autentizačním variantám EAP. Umožňuje vzájemné ověření, tedy i suplikant může prověřit, zda žádá o oprávnění správný přístupový bod. Suplikant i autentizační server jsou opatřeny digitálním certifikátem podepsaným certifikační autoritou. EAP–TLS implementuje infrastrukturu veřejného klíče PKI (*Public Key Infrastructure*). Tím jsou zašifrovány veškeré autentizační zprávy probíhající mezi suplikantem a serverem. Mechanismus tedy klade jistý nárok na účastníky, zejména pak klienty, a to v podobě potřeby přítomnosti certifikátů a režijních nákladů na jejich správu, ale je díky tomu odolný vůči odposlouchávání a útokům MITM. Výhodou je také široká podpora této EAP varianty napříč výrobci síťových zařízení a aplikací.

3.2.3.4 EAP–TTLS

EAP Tunneled Transport Layer Security (EAP–TTLS) je mechanismus vylepšující původní EAP–TLS. Vylepšení spočívá ve zjednodušení implementace a snížení nároků na klienty, kteří už nemusí (avšak stále mohou) být opatřeni certifikátem pro PKI. Pokud nejsou, ověření autorizačního serveru vůči klientovi probíhá stále pomocí certifikátu, zatímco ověření v opačném směru proběhne za pomoci hesla.

3.2.3.5 LEAP

Lightweight EAP (LEAP) byl vyvinut společností Cisco Systems jako proprietární autentizační mechanismus. Příliš rozšířen není, právě z důvodu uzavřenosti, kdy nutným

požadavkem pro jeho použití je provozování celé sítě pouze na Cisco zařízeních. Umožňuje vzájemné ověřování a použití dynamických WEP klíčů. Během vlastní komunikace klienta provádí jeho opětovné ověřování, a v případě úspěchu vždy vygeneruje WEP klíč nový. Funguje tedy na předpokladu, že WEP klíč nebude nikdy odhalen. LEAP však umožňuje rekonfiguraci pro použití TKIP (viz str. 25) namísto dynamických WEP klíčů.

3.2.3.6 PEAP

Protected EAP (PEAP) byl vyvinut ve spolupráci společností Microsoft, Cisco Systems a RSA Security. Umožňuje vzájemné ověřování, přičemž pro ověření autentizačního serveru vůči suplikantovi používá zabezpečený TLS tunel (digitální certifikát), a v opačném směru ověřování dává volnost při výběru některé ze zabezpečovacích metod.

3.2.4 WPA

WiFi Alliance představila technologii *WiFi Protected Access* (WPA) v roce 2002 – krátce po tom, co došlo k prolomení zabezpečení WEP. Definována je v části dokumentu IEEE 802.11i (dodatek ke standardu 802.11). Zlepšuje zejména velmi slabou autentizaci a šifrování statickým klíčem – hlavní nedostatky WEP. Z důvodu potřeby co možná nejrychlejšího nahrazení zastaralé technologie WEP je WPA navrženo tak, aby pracovalo i na hardwaru původně vyrobeném pro WEP, a stačilo tak pro přechod na WPA provést pouze softwarovou aktualizaci. Šifrování přenášených dat je zde proto realizováno také pomocí proudové RC4 šifry.

3.2.4.1 Pracovní režimy

Dle typu koncového uživatele se implementace WPA odlišují ve způsobu distribuce autentizačních klíčů mezi aktivní prvky sítě, a v použité metodě šifrování. Existují dva pracovní režimy: [3]

- **WPA–Personal**. Varianta určená pro menší podniky a domácnosti, tedy tzv. SOHO (*Small Office / Home Office*) segment, kde povětšinou není použit žádný

autentizační server. Využívá předsdílený klíč PSK (*PreShared Key*), proto je také někdy označována jako WPA–PSK. V praxi to znamená, že je síť zabezpečena jedním sdíleným heslem, které zadávají všichni žadatelé o připojení do sítě.

- **WPA–Enterprise**. Varianta určená pro velké podniky, kde se předpokládá přítomnost autentizačního RADIUS serveru. Každý žadatel o připojení k síti je poté ověřován svými vlastními přihlašovacími údaji. Pro autentizaci jsou v tomto režimu využívány metody standardu 802.1x (viz kapitola 3.2.2).

3.2.4.2 Autentizace

WPA pro autentizaci přístupu do sítě využívá prostředků protokolu 802.1x a EAP (viz předchozí kapitoly 3.2.2 a 3.2.3).

3.2.4.3 Vytváření a distribuce šifrovacích klíčů

Po procesu autentizace a asociace připojovaného zařízení musí dojít k sestavení šifrovacích klíčů a jejich rozdistribuování mezi aktivní prvky sítě. Specifikace WPA definuje následující čtyři hlavní typy párových (*Pairwise*, pro unicast komunikaci) a skupinových (*Group*, pro multicast a broadcast komunikaci) klíčů: [9]

- **PMK** – *Pairwise Master Key*. Bereme-li v potaz režim WPA–Personal, pak je shodný s PSK, který je jak na straně suplikanta, tak na straně přístupového bodu vypočítán funkcí pro derivaci klíčů PBKDF2 (*Password–Based Key Derivation Function 2*) z názvu (SSID) a hesla sítě (u WPA–Enterprise je získán od autentizačního serveru).

- **PTK** – *Pairwise Transient Key*. Dočasný šifrovací klíč pro unicast komunikaci, jenž je generován z PMK.

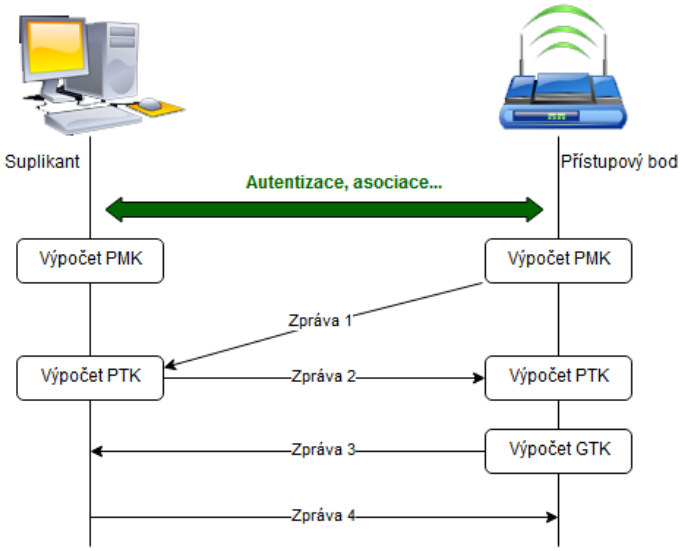
- **GMK** – *Group Master Key*. Číslo náhodně vygenerované přístupovým bodem.

- **GTK** – *Group Transient Key*. Dočasný šifrovací klíč pro multicast a broadcast komunikaci, jenž je generován z GMK tak, že je k němu přidán rozšiřující řetězec znaků (heslo), MAC adresa přístupového bodu a další náhodné číslo. Uvedená

data slouží jako vstupní parametry pro provedení pseudonáhodné funkce HMAC–SHA1 (*Keyed–Hash Message Authentication Code*), jejíž výsledkem je právě klíč GTK.

Vygenerování dočasných klíčů PTK a GTK je provedeno procesem nazývaným čtyřcestný handshake (*Four–Way Handshake*). Během něj dojde mezi suplikantem a přístupovým bodem (autentizátorem) k sekvenční výměně čtyř zpráv: [9]

- **Zpráva 1**: Obě strany mají vypočítán PMK. Přístupový bod svůj výpočet PMK zasílá spolu s náhodně vygenerovaným číslem suplikantovi.
- **Zpráva 2**: Suplikant také vygeneruje náhodné číslo, které spolu se svým výpočtem PMK, MAC adresou přístupového bodu a náhodným číslem přijatým ve zprávě 1 vkládá do pseudonáhodné funkce, čímž vznikne výpočet PTK.
  Následně suplikant zprávou 2 odesílá přístupovému bodu své náhodné číslo, které vygeneroval při výpočtu PTK a přidává navíc MIC (*Message Integrity Code*) vypočítaný z této zprávy.
- **Zpráva 3**: Přístupový bod na základě dat, které sám vypočítal a zná, a náhodného čísla vygenerovaného suplikantem, které obdržel ve zprávě 2 vypočítá obdobným způsobem také klíč PTK. Z vypočítané hodnoty navíc sestaví MIC a porovná ho s MIC zprávy 2. Pokud se MIC shodují, pak se shodují i PTK na obou stranách, a tedy komunikace bude šifrována stejným klíčem.
  Ve zprávě 3 odesílá přístupový bod suplikantovi jím vygenerované GMK a náhodné číslo jako zprávu, která je zašifrována pomocí PTK. Opět k ní ještě sestaví a připojí MIC.
- **Zpráva 4**: Suplikant si dešifruje a uloží přijaté GTK, a zprávou 4 dá přístupovému bodu na vědomí potvrzení celého procesu.



Obrázek 5: Four–Way Handshake

3.2.4.4 Šifrování a integrita dat

IEEE 802.11i definuje nový bezpečnostní protokol TKIP (*Temporal Key Integrity Protocol*) pracující s dynamickými klíči. Zároveň je schopen zajistit bezpečný přenos těchto klíčů nejen při navázání komunikace, ale i v jejím průběhu. Oproti WEP, kde byl použit symetrický šifrovací klíč pro veškerou komunikaci všech účastníků, je u protokolu TKIP každý sítí zasílaný paket zašifrován vlastním unikátním klíčem. Tím je umožněno výrazné zvýšení zabezpečení sítě bez nutnosti hardwarového upgradu vysílacího zařízení.

Pro ověřování integrity dat je u WPA zavedena nová hashovací funkce MAC (*Message Authentication Code*). Tato funkce umožňuje jednak kontrolu, že data nebyla během přenosu nikterak pozměněna, ale zároveň i ověření, že zpráva pochází od opravdového odesílatele. [5]

3.2.4.5 Zranitelnost

Přestože WPA lze oproti WEP stále považovat za mnohem bezpečnější mechanismus, existují způsoby, jak prolomit i tuto ochranu. V případě režimu WPA–Personal potenciálně hrozí prolomení hesla (tj. zjištění PSK) slovníkovým útokem, k čemuž již bylo vyvinuto nespočet automatizovaných aplikací, zejména pro platformu Unix.

Zároveň jsou na internetu volně dostupné stovky slovníků pro tyto aplikace, obsahující slova či celé fráze používané často jako hesla. Některé z nich zahrnují i slova s lokálními a speciálními znaky. Aplikace pak po zvolení WiFi sítě – oběti cyklicky zkouší přihlášení každým záznamem ze slovníku. Obrana je jednoduchá, avšak dodnes stále často zanedbávaná, a to nastavení dostatečně silného hesla (délka řetězce, včetně číslic, včetně speciálních znaků).

Další možný typ útoku je proveditelný napadnutím čtyřcestného handshaku, konkrétně odchycením zpráv, které si zařízení během něj vyměňují. Již po prvních dvou zprávách získá útočník náhodně generované hodnoty, které byly použity pro výpočet PSK (resp. PMK), s jejichž pomocí může začít hádat jednotlivé hodnoty PSK, z nichž algoritmem získá dočasné klíče PTK. Následně pro obě zachycené zprávy vypočítá MIC. Pokud $\mathrm{MIC}_1 \neq \mathrm{MIC}_2$, pak se celý proces opakuje novým odhadem PSK. Po nalezení shody došlo k prolomení zabezpečení WPA, neboť útočník již zná správný dočasný klíč PTK podmíněný heslu. [5]

3.2.5 WPA2

WiFi Protected Access 2 (WPA2) nahrazuje svého předchůdce WPA. Je definován rovněž standardem IEEE 802.11i. Na rozdíl od WPA však tento standard implementuje beze zbytku celý, kdežto WPA implementuje pouze část standardu. Vylepšeny jsou autentizační i šifrovací algoritmy. V současné době (2008) se stále jedná o nejaktuálnější zabezpečovací mechanismus WiFi sítí, považovaný zároveň za nejvíce bezpečný. WiFi Alliance učinila rozhodnutí, že všechna nově vyráběná WiFi zařízení musejí implementovat WPA2 jako výhradní podmínku pro udělení certifikátu a oficiální značky WiFi. WPA2 je standardně implementováno opět ve dvou pracovních režimech WPA2–Personal a WPA2–Enterprise, viz 3.2.4.1.

3.2.5.1 Autentizace

WPA2 pro autentizaci a asociaci přístupového bodu a suplikanta (žadatele) využívá, stejně jako předchůdce WPA, prostředků protokolu 802.1x a EAP (viz kapitoly 3.2.2 a 3.2.3).

3.2.5.2 Vytváření a distribuce šifrovacích klíčů

Vytváření a distribuce klíčů probíhá u WPA2 stejně jako u předchůdce WPA. Opět se pro šifrování komunikace používají dočasné klíče PTK (pro unicast) a GTK (pro multicast, broadcast) získané a distribuované procesem Four–Way Handshake.

3.2.5.3 Šifrování a integrita dat

WPA2 upouští od využití proudové šifry RC4, a dává na výběr novou, blokovou šifru AES (*Advanced Encryption Standard*). Stále je však možné šifrování realizovat pomocí méně bezpečného protokolu TKIP (viz str. 25).

Implementace blokové AES šifry je u WPA2 zapouzdřena do CCMP (*Counter Mode CBC–MAC Protocol*) protokolu. Pokud bude název protokolu postupně rozebrán, pak režim Counter Mode (CTR) rozděluje odesílanou zprávu na bloky pevné délky. Využívá se při tom čítač, jehož hodnota je inicializována na náhodnou hodnotu a s průchodem přes každý další blok původní zprávy je inkrementována. Pro každý blok poté platí, že jeho výsledná šifrovaná podoba je rovna jeho původnímu obsahu XOR hodnota čítače příslušného bloku zašifrovaná pomocí AES. Analogicky pak u příjemce probíhá dešifrování jednotlivých bloků přijaté zprávy.

Režim Cipher Block Chaining Message Authentication Code (CBC–MAC) poskytuje mechanismus pro ověření integrity přenášených dat pomocí integritního kódu zprávy MAC (*Message Integrity Code*). CBC nejprve vygeneruje náhodné číslo, mezi nímž a prvním blokem zprávy provede XOR. Výsledek zašifruje pomocí AES, čímž je zašifrován první blok. Poté algoritmus iteruje přes následující bloky, jejichž zašifrovaná podoba se vypočítá stejně jako v případě prvního bloku s rozdílem, že jedním ze vstupů operace XOR je namísto náhodného čísla hodnota zašifrovaného předchozího bloku. Tímto provázáním je dosaženo vzájemné závislosti po sobě jdoucích bloků. Zpráva složená z prve vygenerovaného náhodného čísla a zašifrovaných bloků je následně odeslána příjemci, který opět analogickým postupem, iterací přes všechny bloky může ověřit, zda zpráva nebyla během přenosu pozměněna. [5]

Spojením výše uvedených režimů vzniká CCM protokol specifikovaný v dokumentu RFC 3610.

3.2.5.4 Zranitelnost

Zabezpečení WPA2 zůstávalo neprolomené až do roku 2017, kdy byla objevena chyba v jeho implementaci. Bezpečnostní slabina, resp. útok na ni byl pojmenován KRACK (*Key Reinstallation Attack*) a cílí opět na čtyřcestný handshake. Bylo totiž objeveno, že mechanismus umožňuje přístupovému bodu např. pro případ ztráty paketů po cestě vícekrát odeslat zprávu 3. KRACK záměrně vynucuje opakované odesílání zprávy 3, což umožňuje podstrčit již použitý klíč, přenastavit další údaje o komunikaci, nastrčit falešnou WiFi síť a odposlouchávat komunikaci. Není tímto způsobem sice možné zjistit textový řetězec hesla sítě, ovšem objevení této chyby nabývá na významu tím, že se týká všech zařízení implementujících WPA2. [10] Nejzákladnější ochranou je v tomto případě používání šifrovaného protokolu HTTPS, namísto HTTP. Výrobci WiFi zařízení však poměrně rychle zareagovali a vydali aktualizace firmwarů, které tuto chybu opravují. Více detailních informací přímo od autorů KRACK útoku je dostupných na dedikované webové stránce [11].

3.3 WPS

WiFi Protected Setup (WPS) je standard vyvinutý za účelem zjednodušit koncovým, zejména domácím uživatelům připojení nových zařízení do zabezpečené WiFi sítě, a to bez nutnosti znalosti jakýchkoli informací o jejich síti, a zadávání přístupového hesla. Jeho autorem je WiFi Alliance, která jej oficiálně představila na začátku roku 2007, tehdy ještě pod oficiálním názvem WSC (*WiFi Simple Configuration*). [12] WPS je běžně na zařízeních implementováno čtyřmi různými metodami, z nichž pro získání certifikátu WPS musí dle standardu povinně podporovat PBC metodu, a zařízení disponující displejem a klávesnicí pak i metodu PIN:

3.3.1 Pracovní režimy

3.3.1.1 PIN

Osmimístné číslo PIN (*Personal Identification Number*) je z výroby přednastaveno na přístupovém bodu, a zpravidla také natištěno na štítku vně zařízení. Spárování proběhne tak, že uživatel zjistí toto číslo, a zadá ho na svém připojovaném zařízení. Pokud se zadávané číslo shoduje s číslem nastaveným na přístupovém bodu, dojde k připojení nového zařízení do sítě.

Během ověřování je s číslem PIN pracováno po částech, konkrétně je jeho struktura dělena na dvě poloviny. Poslední, osmou číslicí je vždy kontrolní součet sedmi předcházejících, viz následující tabulka:

Tabulka 2: Struktura čísla WPS PIN

1	**2**	**3**	**4**	**5**	**6**	**7**	**0**
První polovina (PIN1)				Druhá polovina (PIN2)			Kontrolní součet

Algoritmus pro výpočet i následné ověření kontrolního součtu je uveden v oficiální specifikaci. Pro názornost je níže uvedena funkce počítající kontrolní součet v syntaxi jazyku Java, jejíž vstupním parametrem je sedmimístné číslo a návratovou hodnotou číslice 0 – 9.

```
int computeChecksum(long pin) {
    long accum = 0;
    pin *= 10;
    accum += 3 * ((pin / 10000000) % 10);
    accum += 1 * ((pin / 1000000) % 10);
    accum += 3 * ((pin / 100000) % 10);
    accum += 1 * ((pin / 10000) % 10);
    accum += 3 * ((pin / 1000) % 10);
    accum += 1 * ((pin / 100) % 10);
    accum += 3 * ((pin / 10) % 10);
    int digit = (accum % 10);
    return (10 - digit) % 10;
}
```

Příklad 1: Funkce pro výpočet kontrolního součtu [12]

3.3.1.2 PBC

Přístupové body podporující WPS jsou opatřeny tzv. PBC hardwarovým tlačítkem (*Push Button Configuration*), příslušně označeným oficiálním symbolem technologie. V případě stisku (na některých zařízeních dlouhého, příp. několikanásobného) se aktivuje na určitou dobu (většinou jedna minuta) režim naslouchání WPS komunikaci. Pokud uživatel WPS tlačítko před uplynutím doby stiskne i na druhém zařízení (na koncových zařízeních jako počítače, mobilní telefony apod. se jedná o tlačítko softwarové), je koncové zařízení připojeno do sítě.

3.3.1.3 NFC

Využití NFC (*Near Field Communication*) při připojování zařízení do sítě předpokládá podporu této technologie jak na straně přístupového bodu, tak u připojovaného zařízení. Samotný proces pak probíhá tak, že na klientském zařízení (zpravidla mobilní telefon či tablet) je aktivováno NFC, a následně je přiloženo k přístupovému bodu. Dojde ke spárování a připojení k síti. Možnost využití technologie NFC byla do standardu WPS přidána teprve v roce 2014.

3.3.1.4 USB

Ve všech předešlých případech probíhá při připojování určitá výměna konfiguračních dat mezi klientem a přístupovým bodem, nutných pro spárování a zahájení

vlastní komunikace přes WiFi síť. Nejinak je tomu v případě využití USB, kdy se tato data na jednom zařízení uloží na přenosné USB médium (např. flash disk), pomocí nějž jsou fyzicky přenesena na druhé zařízení. Tím opět dojde ke spárování a připojení klientského zařízení. V praxi tato metoda nikdy nebyla příliš využívána, WiFi Alliance ji označila za zastaralou, a ani její podpora není podmínkou pro získání WiFi certifikace (stejně jako v případě technologie NFC).

Pro úplnost je třeba zmínit, že získání oficiálního WPS certifikátu zařízení od WiFi Alliance nikterak neovlivňuje získání významnějšího certifikátu značky WiFi, a není pro něj podmínkou. Jinými slovy, certifikované WiFi zařízení nemusí technologii WPS vůbec podporovat.

3.3.2 WPS protokol

Zařízení účastnící se procesu spárování a připojení pomocí WPS protokolu jsou dle standardu dělena na následující entity:

- **Enrollee** – žadatel; klientské zařízení žádající o přístup do sítě.
- **AP** – Access Point; přístupový bod sítě splňující specifikaci IEEE 802.11.
- **Registrar** – registrátor; entita s kompetencí rozhodovat o povolení či odepření přístupu. Představuje tak prostředníka v komunikaci mezi žadatelem a AP. Většinou je implementována přímo jako logická součást AP, může však být realizována jako samostatné zařízení. [12]

Při připojování žadatele do sítě pomocí WPS protokolu nejprve proběhne standardní autentizace a asociace. V případě připojování pomocí zadání čísla PIN je průběh výměny EAP zpráv mezi žadatelem a AP (uvažujme, že AP a registrátor představují jedno zařízení tak, jak je tomu u všech domácích routerů) následující:

- **EAPOL–Start**. Žadatel touto zprávou navazuje spojení s AP a odesílá informaci, že bude chtít být připojen pomocí WPS.
- **EAP–Request/Identity**. Odpověď AP žadateli s výzvou, aby zaslal svůj identifikátor.

- **EAP–Response/Identity**. Žadatel zasílá identifikátor (v praxi dle standardu pouze ve znění „WFA–SimpleConfig–Enrollee–1–0“). Po přijetí této zprávy AP je inicializováno EAP spojení.

- **Zpráva M1, M2, M3**. Pomocí těchto zpráv vyměněných mezi žadatelem a AP je inicializován WPS protokol.

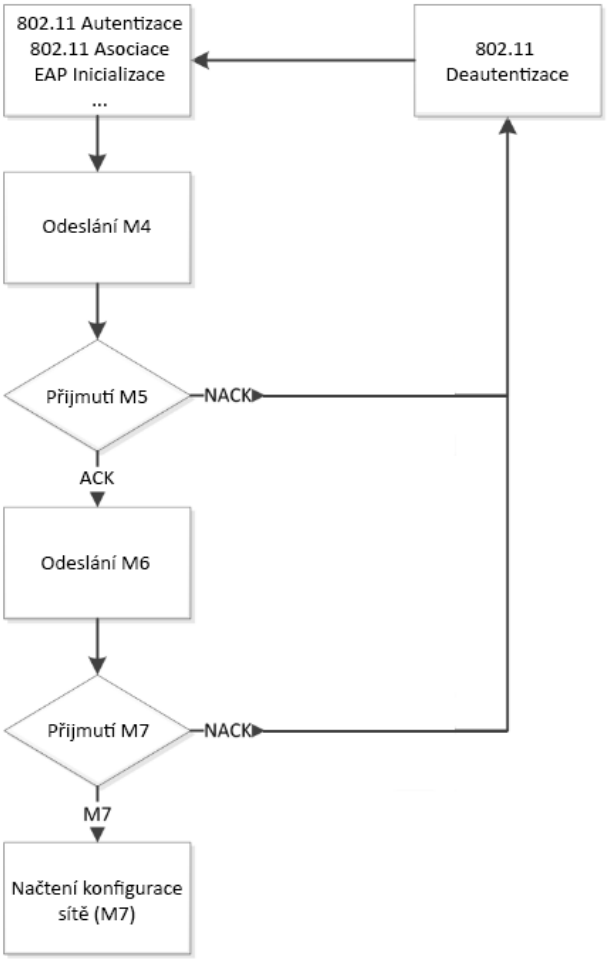
- **Zpráva M4**. Ve zprávě M4 odesílá žadatel PIN, kterým se chce připojit k síti.

- **Zpráva M5**. AP vrací odpověď na zaslaný PIN, resp. na jeho první polovinu (PIN1, viz Tabulka 2). Odpověď může nabývat buď hodnoty ACK (*Acknowledge*), nebo NACK (*Negative Acknowledge*). Pokud žadatel obdrží jako odpověď NACK znamená to pro něj, že PIN1 je nesprávný. Žadatele v takovém případě AP deautentizuje a proces se vrací na začátek. Obdrží-li žadatel jako odpověď ACK, může si PIN1 uchovat a pro zbytek pokusů ponechat konstantní, jelikož ACK přijatý ve zprávě M5 je pro něj naopak signál, že je PIN1 správný. Proces pak pokračuje dále.

- **Zpráva M6**. Žadatel opět zasílá zkoušený PIN tentokrát s předpokladem, že PIN1 je správný.

- **Zpráva M7**. AP opět vrací odpověď s informací o správnosti PIN2. Je-li odpověď NACK, žadatel je deautentizován a proces se vrací na začátek. V opačném případě v této zprávě AP zasílá žadateli informace o konfiguraci sítě, mimo jiné včetně přístupového hesla (PSK) pro WPA/2. [12]

Pro názornost je průběh tohoto procesu znázorněn na str. 33 vývojovým diagramem.



Obrázek 6: Vývojový diagram procesu výměny EAP zpráv

3.3.3 Zranitelnost

Možnost změny čísla PIN v administračním rozhraní sítě je provozovateli často opomíjena, což z WPS činí bezpečnostní slabinu, jelikož výrobci přístupových bodů často na svá zařízení přednastavují stejný výchozí PIN. Otevřené databáze výchozích PIN čísel jednotlivých nejběžnějších výrobců jsou pak volně dostupné na internetu.

Ovšem také samotný princip fungování a způsob implementace WPS protokolu pro připojování pomocí čísla PIN nahrává potenciálním útočníkům. Jak bylo uvedeno výše, žadatel během zkoušení různých kombinací PIN získává od AP konkrétní informace o správnosti či nesprávnosti jednotlivých polovin čísla. Tím se razantně snižuje množství číselných kombinací a pokusů potřebných k odhalení správné kombinace.

Pro demonstraci uveďme následující srovnání. Pokud by žadatel od AP dostá-

val pouze souhrnnou informaci o správnosti všech osmi zkoušených číslic, bylo by na odhalení správné kombinace zapotřebí 10^7 (= 10 000 000) pokusů za předpokladu, že osmá číslice je vždy kontrolní součet. Dělením čísla PIN na dvě poloviny a ověřováním každé poloviny zvlášť se počet potřebných pokusů snižuje na $10^4 + 10^3$ (= 11 000). [13]

I pokud bychom však nebrali v potaz toto zjednodušení, pak číslo PIN je útokem hrubou silou mnohem snáze odhalitelné, než vlastní heslo sítě. PIN může vždy obsahovat pouze číslice, kdežto součástí hesla mohou být i alfanumerické a speciální znaky, s nimiž je odhalení hesla pomocí hrubé síly či slovníkového útoku prakticky nemožné. Navíc PIN představuje číslo konstantně osmimístné, naproti tomu heslo je textový řetězec s délkou 8 – 64 znaků.

Nutno však podotknout, že výrobci routerů po několika letech začali brát v potaz technologii WPS jako značné bezpečnostní riziko, a začali do firmwarů nově vyráběných zařízení implementovat různé druhy ochrany. U některých modelů se jedná o ochranu pomocí omezeného počtu pokusů pro připojení číslem PIN, kdy po několika neúspěšných pokusech dojde buď k dočasnému (v řádu minut), či trvalému (do opětovného ručního zapnutí správcem sítě) vypnutí WPS funkcionality. Jiní výrobci se rozhodli čelit této bezpečnostní slabině cestou úplného vypnutí WPS v továrním nastavení. V případě potřeby pak musí být WPS na routeru zapnuto ručně, což je však u těchto zařízení umožněno pouze na určitou omezenou dobu.

3.4 Operační systém Android

Teoretická východiska problematiky vývoje mobilní aplikace pro operační systém Android, a operačního systému samotného jsou přehledně uvedena v [14] z roku 2016. Ve zmíněné práci jsou popsány charakteristiky architektury OS Android, vývojové prostředí Android Studio, obecná struktura projektu mobilní aplikace a základní programové a grafické komponenty dostupné z Android SDK (*Software Development Kit*). V této kapitole tedy budou uvedeny pouze skutečnosti a rozdíly, ke kterým od r. 2016 do dnešní doby (2018) v systému došlo a komponenty, které byly využity při implementaci praktické části práce.

3.4.1 Verze

Od roku 2016 byly společností Google zveřejněny následující nové verze OS Android:

Android 7.0
- „Nougat“, 08/2016, API level 24
- podpora práce ve vícero oknech (rozdělení obrazovky na dvě nezávislé poloviny)
- vylepšený systém notifikací
- podpora virtuální reality

Android 7.1
- 10/2016, API level 25
- zlepšení dotekové odezvy displeje
- noční režim

Android 8.0
- „Oreo“, 08/2017, API level 26
- zásadní optimalizace výkonu a výdrže baterie
- automatické předvyplňování formulářů
- pro některá zařízení podpora WiFi Aware (nová technologie umožňující automatické decentralizované propojení WiFi zařízení v určitém vzdálenostním okruhu)

Android 8.1
- 12/2017, API level 27
- projekt „Android Go“ pro low–end zařízení

3.4.2 Grafické rozhraní – Constraint Layout

Hlavní novinkou v oblasti designu grafického uživatelského rozhraní (*GUI*) je kontejner „Constraint Layout“, který je v rámci Android SDK distribuován jako podpůrná (support) knihovna v přídavném balíčku `android.support.constraint`. Tento kontejner umožňuje návrh a tvorbu komplexních layoutů (rozložení grafických komponent) bez nutnosti hlubšího vnořování a kombinování vícero kontejnerů do sebe. Charakteristikou se poměrně podobá kontejneru Relative Layout (detailně popsán v [14]) a to zejména tím, že poloha každé obsažené komponenty je definována relativně vůči okolním komponentám anebo vůči rodičovskému kontejneru. [15]

Zavedení Constraint Layoutu si klade za cíl zvýšení responzivnosti GUI s minimalizací znovupoužívání grafických návrhů. V neposlední řadě je jeho výhodou také silná podpora ze strany Layout editoru v Android Studiu, kde mohou být rozložení GUI postavená na Constraint Layoutu plně tvořená v grafickém režimu (tedy metodou „drag–and–drop“), nikoli přímou editací XML zápisu.

3.4.3 WifiManager

Třída `android.net.wifi.WifiManager` poskytuje aplikacím komplexní aplikační rozhraní (*API*) pro obsluhování WiFi konektivity. Instance je získána z kontextu aplikace jako systémová služba, tedy voláním funkce `Context.getSystemService(String)` s parametrem `Context.WIFI_SERVICE`. Umožňuje aplikacím skenovat dostupné WiFi sítě, připojovat se k nim, odpojovat, nebo např. ovládat funkci hotspot. [17]

V praktické části práce bude využita zejména metoda `startScan()` a funkce `getScanResults()`. Prvně jmenovaná metoda zahajuje skenování okolních sítí, druhá jmenovaná funkce asynchronně vrací výsledek v podobě seznamu instancí objektu `ScanResult` (viz 3.4.4). Dále bude užito metody `setWifiEnabled(boolean enabled)`, která na zařízení zapíná či vypíná WiFi adaptér, a metoda `disconnect()`, která v případě, že je zařízení aktuálně připojeno k WiFi síti zajistí jeho odpojení.

Další, pro funkčnost aplikace nezbytnou metodou bude metoda, která inicializuje pokus o připojení pomocí WPS `startWps(WpsInfo config, WpsCallback listener)`, kde první parametr – objekt `WpsInfo` (viz 3.4.5) konfiguruje připojení, a druhý parametr – objekt `WpsCallback` (viz 3.4.6) detekuje výsledek pokusu o připojení.

3.4.4 ScanResult

Každá instance třídy `android.net.wifi.ScanResult`, obdržená jako výsledek skenování sítí obsahuje všechny základní informace o jednom konkrétním zjištěném přístupovém bodu, jako např. SSID sítě, BSSID (tj. MAC adresa přístupového bodu), informaci o změřené síle signálu, frekvenci a textový řetězec `capabilities`, jehož obsahem je charakteristika autentizační a šifrovací technologie použité v dané síti spolu s indikátorem, zda je v síti aktivní funkce WPS. [18]

3.4.5 WpsInfo

Třída `android.net.wifi.WpsInfo` slouží ke konfiguraci parametrů nového pokusu o připojení pomocí WPS protokolu. Prostřednictvím proměnné `setup` je definována metoda připojení (PIN, PBC, ...), proměnnou `BSSID` je předána MAC adresa cílového přístupového bodu, a v případě připojování pomocí PIN je předáno proměnnou `pin` i samotné testované číslo. [19]

3.4.6 WifiManager.WpsCallback

Rozhraní `WifiManager.WpsCallback` slouží pro detekci a zpracování výsledků pokusů o připojení protokolem WPS. Definuje tři stavové metody, které jsou spuštěny postupně po volání metody `startWps`: [20]

(i) `onStarted(String pin)` – nastane bezprostředně po úspěšné inicializaci procesu WPS připojení,

(ii) `onSucceeded()` – nastane v případě úspěšného připojení, tedy detekuje správnost PIN čísla,

(iii) `onFailed(int reason)` – nastane v případě neúspěšného připojení, kdy parametr `reason` udává důvod, proč došlo k selhání. Třídou `WifiManager` definované důvody mohou následující:

   a) `WPS_TIMED_OUT` – došlo k vypršení časového limitu na připojení,

   b) `WPS_AUTH_FAILURE` – zadán nesprávný PIN.

3.4.7 AsyncTask

Generická třída `android.os.AsyncTask<Params, Progress, Result>` slouží k asynchronnímu provádění úloh na pozadí, které mohou být časově náročné. Nedochází tak k blokování hlavního vlákna aplikace (tj. „zamrzání“ aplikace), které obsluhuje především uživatelské rozhraní. Zároveň umožňuje GUI snadno aktualizovat, bez nutnosti přímé práce s vlákny.

Při jejím rozšiřování jsou určeny a mezi znaky `< >` uvedeny tři datové typy definující postupně typ:

1. vstupního parametru (`Params`), se kterým bude úloha spuštěna,
2. jednotky, kterou bude reportován postup úlohy (`Progress`),
3. výsledné navrácené hodnoty (`Result`).

AsyncTask implementuje čtyři kroky provádění úlohy, které jsou systémem volány vždy automaticky v níže uvedeném pořadí. Při rozšiřování třídy je nezbytné přepsat alespoň `doInBackground(Params...)`. Ostatní kroky mohou být přepsány volitelně v případě potřeby:

- `onPreExecute()` – voláno před vlastním spuštěním úlohy.
- `doInBackground(Params...)` – tělo asynchronní úlohy, tedy kód prováděný na pozadí. Přijímá vstupní parametr typu `Params` v podobě varargs, tedy parametrem může být:
  - jedna samostatná hodnota,
  - vícero samostatných hodnot oddělených čárkou,
  - pole hodnot. [21]

  K hodnotám parametru je následně uvnitř těla úlohy přistupováno jako k hodnotám standardního pole v Javě. [23]
- `onProgressUpdate(Progress...)` – proběhne, je-li v těle úlohy zavolána metoda `publishProgress(Progress...)`, a zpravidla zajišťuje aktualizaci uživatelského

rozhraní na základě přijaté hodnoty `Progress` udávající, o kolik jednotek se změnil průběh úlohy.

- `onPostExecute(Result)` – voláno bezprostředně po dokončení asynchronní činnosti. Prostřednictvím parametru typu `Result` je předán výsledek úlohy.

Po vytvoření nové instance asynchronní úlohy je tato spuštěna zavoláním metody `execute(Params...)`, čímž zároveň dojde k předání vstupních parametrů úloze. Může být také kdykoli zrušena zavoláním `cancel(boolean)`. Parametr metody určuje, zda má ke zrušení dojít okamžitě, nebo, je-li právě spuštěna až po dokončení její činnosti. [22]

3.5 Analýza a testování existujících aplikací

V oficiálním úložišti Android aplikací Google Play je v současné době dostupných cca patnáct řešení pro prověřování zranitelnosti WPS. Pro účely testování bylo vybráno pět produktů různých vývojářů s nejvyšším počtem stažení a nejlepším uživatelským hodnocením. Při testování byly sledován zejména faktor úspěšnosti připojení. Všechny testované aplikace jsou dostupné ke stažení zdarma. Jejich otestování proběhlo metodou „černá skříňka“, jelikož žádná z aplikací není poskytována jako open–source (tj. nejsou veřejně dostupné zdrojové kódy).

Aplikace byly testovány na vybraném vzorku pěti WiFi přístupových bodů určených pro SOHO segment od různých běžných výrobců. Všechny zvolené přístupové body byly před začátkem testování uvedeny do továrního nastavení tak, aby byla nasimulována typická situace, kdy je WiFi síť prostřednictvím WPS nejzranitelnější, tedy když provozovatel sítě přístupový bod zprovozní, a dále již pak nevěnuje pozornost jeho konfiguraci. Na všech zvolených přístupových bodech byla funkce WPS PIN v továrním nastavení aktivní.

3.5.1 Testované aplikace

Níže je uveden výčet vybraných pěti aplikací (názvy uvedeny přesně tak, jak jsou prezentovány v Google Play) se základními charakteristikami:

WiFi Warden (WPS Connect)

- **vývojář:** Ramtin Ardeshiri
- **hodnocení:** 4.5 / 5
- **počet stažení:** 5 mil.+
- **funkcionalita:** vygenerování možných čísel PIN a pokus o připojení pomocí nich, připojení pomocí uživatelem zadaného čísla PIN či pomocí zadaného hesla
- aplikace poskytuje podrobnější analýzu zabezpečení WiFi sítí, informace o síle signálu a informace o výrobci přístupového bodu

WPSApp

- **vývojář:** TheMauSoft
- **hodnocení:** 4.4 / 5
- **počet stažení:** 10 mil.+
- **funkcionalita:** vygenerování možných čísel PIN a pokus o připojení pomocí nich, připojení pomocí uživatelem zadaného čísla PIN či pomocí zadaného hesla
- aplikace jako další funkci nabízí skenování zařízení v síti, do které je aktuálně připojena

WIFI WPS WPA TESTER

- **vývojář:** Sangiorgi Srl
- **hodnocení:** 4.3 / 5
- **počet stažení:** 10 mil.+
- **funkcionalita:** zobrazuje pouze WiFi sítě s aktivním WPS, vygenerování možných čísel PIN a pokus o připojení pomocí nich, připojení pomocí uživatelem zadaného čísla PIN, možnost odhalení čísla PIN hrubou silou

AndroDumpper (WPS Connect)

- **vývojář:** Osama Abukmail
- **hodnocení:** 4.3 / 5
- **počet stažení:** 10 mil.+
- **funkcionalita:** ve výchozím nastavení zobrazuje pouze WiFi sítě s aktivním WPS, připojení pomocí uživatelem zadaného čísla PIN nebo pomocí série vygenerovaných PIN čísel (nejsou však uživateli zobrazeny)

WPS Connect

- **vývojář:** FroX
- **hodnocení:** 4.1 / 5
- **počet stažení:** 10 mil.+
- **funkcionalita:** vygenerování možných čísel PIN a pokus o připojení pomocí nich, připojení pomocí uživatelem zadaného čísla PIN
- další funkčností aplikace je možnost zobrazit v zařízení uložená hesla známých WiFi sítí (vyžaduje však oprávnění root)

3.5.2 Testované přístupové body

Následuje výčet pěti WiFi přístupových bodů (routerů), které byly použity pro testování jednotlivých uvedených aplikací, s jejich základními charakteristikami v továrním nastavení:

Tenda W311R+

- **rok výroby:** 2010
- **WPS:** přednastaven jednoduchý PIN 12345670

TP–Link TD–W8961NB

- **rok výroby:** 2011
- **WPS:** PIN 37494063

ASUS RT–N12 D1

- **rok výroby:** 2017
- **WPS:** funkce se po špatně zadaném čísle PIN deaktivuje

Huawei LTE CPE B310

- **rok výroby:** 2016
- **WPS:** funkce se po špatně zadaném čísle PIN deaktivuje

Netis WF2419

- **rok výroby:** 2016
- **WPS:** PIN 42854715

3.5.3 Praktický test

Níže je uvedena tabulka s přehledem úspěšnosti jednotlivých testovaných aplikací vůči jednotlivým routerům. Symbol „X“ značí úspěšnost odhalení čísla PIN jednou z aplikací nabízených variant.

Tabulka 3: Výsledky praktického testu aplikací

	Tenda W311R+	**TP–Link TD–W8961NB**	**ASUS RT–N12 D1**	**Huawei LTE CPE B310**	**Netis WF2419**
WiFi Warden (WPS Connect)	X	X			X
WPSApp	X				
WIFI WPS WPA TESTER	X				
AndroDumpper (WPS Connect)	X	X			
WPS Connect	X	X			

3.5.4 Zhodnocení

Jak je z tabulky patrné, u prvního zvoleného routeru Tenda W311R+ byla zjištěna stoprocentní úspěšnost z důvodu triviálnosti přednastaveného PIN čísla. Dále byla u novějších routerů ASUS RT–N12 D1 a Huawei LTE CPE B310 zjištěna ochrana neoprávněného průniku pomocí automatické deaktivace funkčnosti WPS, u obou modelů shodně trvající až do ručního opětovného zapnutí v administračním rozhraní. Žádná z testovaných aplikací tedy vůči těmto dvěma zařízením neuspěla.

Při porovnání mobilních aplikací s aplikacemi určenými pro osobní počítače (aplikace „Reaver“ pro operační systém Linux a aplikace „Dumpper“ pro Windows) není nikdy dosaženo tak vysoké výkonnosti a efektivity. Jednak procesory mobilních zařízení (telefonů a tabletů) stále nejsou tolik výkonné, a jednak s ohledem na architekturu mobilního operačního systému Android zde není možné s WPS protokolem pracovat na nízké úrovni jednotlivých zpráv, tak jak je tomu u desktopových platforem. Z toho důvodu je implementace funkce odhalení PIN čísla hrubou silou sice realizovatelná, avšak její běh vyžaduje řádově vyšší časovou náročnost, než je zapotřebí

u osobních počítačů, jelikož na platformě Android není možné využít známých nedostatků protokolu uvedených v 3.3.2.

Jako nejúspěšnější a nejefektivnější aplikace z testu vyšla **WiFi Warden (WPS Connect)**, jejíž úspěšnost byla u 3 z 5 vybraných routerů. Tato aplikace je současně na Google Play uživateli nejlépe ohodnocenou.

4 Vlastní práce

4.1 Zadání

Vyvinutý prototyp nové aplikace bude umět skenovat okolní WiFi sítě a přehledně uživateli zobrazit jejich seznam s označením sítí, které mají aktivní technologii WPS. V případě, že zařízení v době skenování bude mít konektivitu do internetu (mobilní data), aplikace ke každému nalezenému záznamu sítě stáhne a zobrazí název výrobce přístupového bodu. Po klepnutí na síť s aktivním WPS si bude uživatel moci zvolit, zda se chce připojit pomocí jím zadaného čísla PIN, nebo pomocí aplikací vygenerovaných variant. V případě zvolení druhé možnosti bude z MAC adresy přístupového bodu použitými algoritmy vygenerován seznam možných PIN čísel, z nichž uživatel bude tyto moci testovat buď jednotlivě, ručně na základě své volby, nebo sekvenčně všechny nabízené možnosti.

Po vybrání některé z variant připojení aplikace začne navazovat spojení s použitím WPS protokolu. Uživatel o tom bude současně informován, a stejně tak bude následně informován o výsledku testu (v případě sekvenčního testu všech nabízených PIN čísel až po úplném skončení procesu, pakliže v průběhu nedojde k odhalení čísla PIN nebo ke ztracení spojení).

4.2 Návrh struktury aplikace

Vnitřní objektová struktura aplikace bude sestávat z následujících tříd a jedné hlavní aktivity včetně rozložení grafického rozhraní.

4.2.1 Třídy

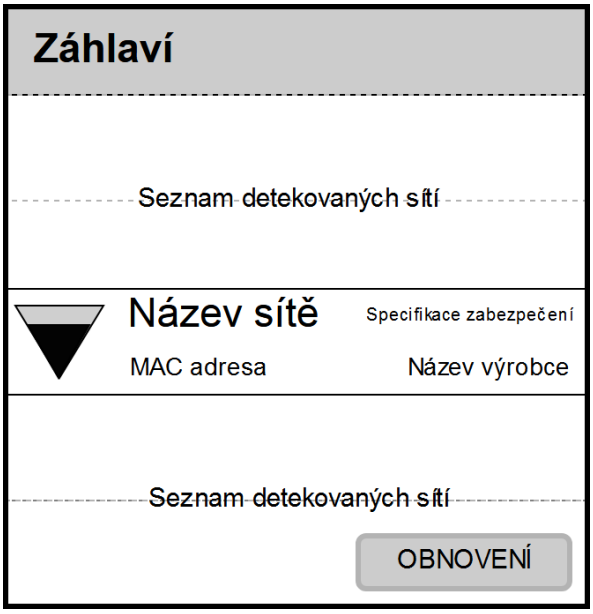
- **PinGenerator.** Třída zajišťující generování možných čísel PIN na základě zadané MAC adresy.
- **WpsConnector.** Třída implementující vnitřní funkčnost WPS protokolu.
- **ScanResultsAdapter.** Třída adaptující jednotlivé záznamy o nalezených WiFi sítích na grafické položky v seznamu.

- **VendorTask.** Asynchronní úloha, která v případě existence konektivity do internetu zjistí název výrobce přístupového bodu na základě MAC adresy.
- **PinListDialog.** Dialogové okno zobrazující seznam vygenerovaných PIN čísel ke zvolené síti.
- **CustomPinDialog.** Dialogové okno pro ruční zadání PIN čísla.
- **PinTryDialog.** Dialogové okno zobrazující průběh testu jednotlivých PIN čísel.
- **ScanningDialog.** Dialogové okno informující uživatele o průběhu skenování dostupných WiFi sítí.

4.2.2 Aktivity

- **MainActivity.** Základní aktivita, která bude uživateli zobrazena bezprostředně po spuštění aplikace. Zastřešuje veškerou funkčnost a poskytuje uživateli GUI.

  Níže na obrázku 7 je znázorněn návrh zamýšleného rozložení uživatelského rozhraní, kde hlavní obrazovce aplikace dominuje seznam detekovaných WiFi sítí. U každé z nich je přehledně zobrazena informace o názvu sítě, síle signálu, MAC adrese a výrobci přístupového bodu a bližší specifikace zabezpečení v síti (autentizační a šifrovací mechanismy). Aktivita pak bude prostřednictvím každého z těchto řádků interaktivně reagovat na klepnutí, jakožto akci výběru konkrétní WiFi sítě uživatelem. Do pravého dolního rohu aplikace je umístěno dobře viditelné tlačítko pro obnovení seznamu, tedy opětovné spuštění skenování sítí.



Obrázek 7: Návrh grafického uživatelského rozhraní hlavní aktivity.

4.3 Implementace struktury aplikace

Výsledná aplikace, vyvinutá ve vývojovém prostředí „Android Studio“, je určena pro verze systému Android 5.0 „Marshmallow“ (API 21) a vyšší. Na předchozích verzích systému je funkčnost WPS implementována pouze v omezené míře, a navíc je pro ni vyžadováno oprávnění root, což může přinášet jistá bezpečnostní rizika. Dle oficiálních statistik společnosti Google aktuálního zastoupení jednotlivých verzí systému na trhu však i přesto aplikace cílí a bude spustitelná na 71.3 % všech zařízení (aktuální stav k březnu 2018).

Aplikace ke své správné funkčnosti vyžaduje přidělení oprávnění přístupu k přibližné poloze zařízení (`ACCESS_COARSE_LOCATION`, od verze Android 6.0 a vyšších nezbytné pro skenování okolních sítí). Dále jsou požadována oprávnění pro přístup k informacím o zjištěných WiFi sítích (`ACCESS_WIFI_STATE`) a pro změnu konektivity (připojování a odpojování) WiFi adaptéru zařízení (`CHANGE_WIFI_STATE`). Pro proces otestování všech vygenerovaných PIN čísel je z důvodu vyšší časové náročnosti požadováno oprávnění pro zamezení přechodu zařízení do režimu spánku (`WAKE_LOCK`). Zbytně pro svou hlavní funkčnost si aplikace žádá oprávnění pro přístup do internetu (`INTERNET`) pro možnost stažení názvů výrobců detekovaných přístupových bodů.

4.3.1 Třídy

- **PinGenerator.** Třída implementuje návrhový vzor „Singleton“. [23] Přístup k její instanci je prováděn voláním `PinGenerator.getInstance()`. Veřejně poskytuje jedinou funkci `generatePins(String)`, jejíž parametrem je MAC adresa přístupového bodu (BSSID). Uvedená funkce nad zadanou MAC adresou provede několik algoritmů založených zejména na bitových operacích, jejichž výsledkem (a také návratovou hodnotou funkce) je pole `String[]` obsahující možné varianty čísel PIN pro daný přístupový bod. Algoritmy funkce byly čerpány převážně z [24]. PIN čísla jsou vrácena v podobě textových řetězců proto, že i vnitřní objekty systému Android pro obsluhu WPS protokolu pracují s položkami PIN jako s řetězci znaků.

- **WpsConnector.** Třída WpsConnector zastřešuje aplikací používané prostředky pro WPS komunikaci. Zároveň však i vytváří a aktualizuje dialogové okno třídy `PinTryDialog`, které uživatele informuje o průběhu testu. Po inicializaci instance je volána metoda `setConfig(String, String)`, jejíž parametry jsou MAC adresa přístupového bodu a PIN, který bude v daném případě testován. Vlastní proces pokusu o připojení pomocí WPS je následně proveden zavoláním metody `connect(WiFiManager.WpsCallback)`, která dle dříve zadaných dat BSSID a PIN vytvoří nový objekt `WpsInfo`, zobrazí dialogové okno s indikátorem průběhu testu a současně iniciuje WPS spojení s přístupovým bodem. Pro obsluhu výsledku procesu je použit objekt parametru `WpsCallback` předaný z hlavní aktivity.

```
public void connect(WifiManager.WpsCallback callback) {
    final WpsInfo config = new WpsInfo();
    config.setup = WpsInfo.KEYPAD; //připojení metodou PIN
    config.BSSID = bssid;
    config.pin = pin;
    dialog.setMessage(context.getString(R.string.trying,
        config.pin));
    dialog.show();
    WifiManager wifi = (WifiManager)context.getApplicationContext()
        .getSystemService(Context.WIFI_SERVICE);
    wifi.startWps(config, callback);
}
```

Příklad 2: Sestavení objektu WpsInfo pro inicializaci WPS připojení

- **ScanResultsAdapter.** Tato třída představuje adaptér mezi daty a grafickým rozhraním aplikace (rozšiřuje třídu `android.widget.BaseAdapter`). Po ukončení skenování okolních sítí obdrží její instance seznam objektů `ScanResult`. Tyto objekty seřadí v seznamu sestupně na základě porovnání jejich vlastnosti `level` indikující sílu signálu dané nalezené sítě tak, aby uživateli byly sítě zobrazeny dle síly signálu od nejvyšší po nejmenší.

```
Collections.sort(this.data, new Comparator<ScanResult>() {

    @Override
    public int compare(ScanResult result1, ScanResult result2) {
        Integer level1 = WifiManager.calculateSignalLevel(
            result1.level, 100);
        Integer level2 = WifiManager.calculateSignalLevel(
            result2.level, 100);
        return level2.compareTo(level1);
    }
});
```

Příklad 3: Seřazení sítí podle síly signálu postupným porovnáváním

Sestavení grafických rozhraní jednotlivých řádků listu poté třída provádí v přepsané metodě `getView()`. Pomocí volání `LayoutInflater.inflate()` načte rozložení grafických komponent z definice list_item.xml. Do získaných komponent vloží příslušná data (SSID sítě, BSSID a údaje o zabezpečení sítě). Zároveň spustí úlohu `VendorTask` (viz níže) a její výsledek (název výrobce) vloží do příslušného textového pole. Dle síly signálu sítě načte z prostředků aplikace odpovídající ikonu bezdrátového připojení a vloží jí do obrázkového pole. Pokud v síti není zjištěna aktivní funkce WPS, načte ikonu se zámkem pro grafické odlišení od sítí, ve kterých WPS aktivní je. Vektorové ikony byly do aplikace importovány ze systémových prostředků pomocí pluginu „Android Drawable Importer“ pro Android Studio, volně dostupného z [25].

- **VendorTask.** Asynchronní úloha rozšiřující třídu `android.os.AsyncTask`. Je spouštěna po dokončení skenování sítí a jejím účelem je, v případě existence konektivity zařízení do internetu, stažení názvů výrobců detekovaných přístupových bodů. Tyto názvy jsou získávány z otevřeného API, dostupného z [26].

Při vytváření nové instance přijímá konstruktor třídy dva parametry: referenci na hlavní aktivitu pro udržení kontextu, a referenci na textové pole, do nějž bude po dokončení úlohy vložen řetězec s názvem výrobce. Vlastní proces stažení názvu výrobce je implementován v těle úlohy `doInBackground(String...)`, který jako vstupní parametr přijímá textový řetězec s MAC adresou, a následně vrací řetězec s názvem výrobce. Ten je po skončení úlohy vložen do textového pole v metodě `onPostExecute(String)`.

```
@Override
protected String doInBackground(String... strings) {
    String vendor = "";
    try {
        URLConnection conn = new URL(String.format(
            "http://macvendors.co/api/%s/csv", strings[0]))
            .openConnection();
        conn.connect();
        InputStreamReader isr = new InputStreamReader(conn
            .getInputStream());
        BufferedReader br = new BufferedReader(isr);
        vendor = br.readLine().split(",")[0].replace("\"", "");
        if(vendor.equals("no result")) vendor = "";
        isr.close();
        br.close();
    } catch (IOException e) {
        e.printStackTrace();
    }
    return vendor;
}
```

Příklad 4: Získání názvu výrobce přístupového bodu z internetu

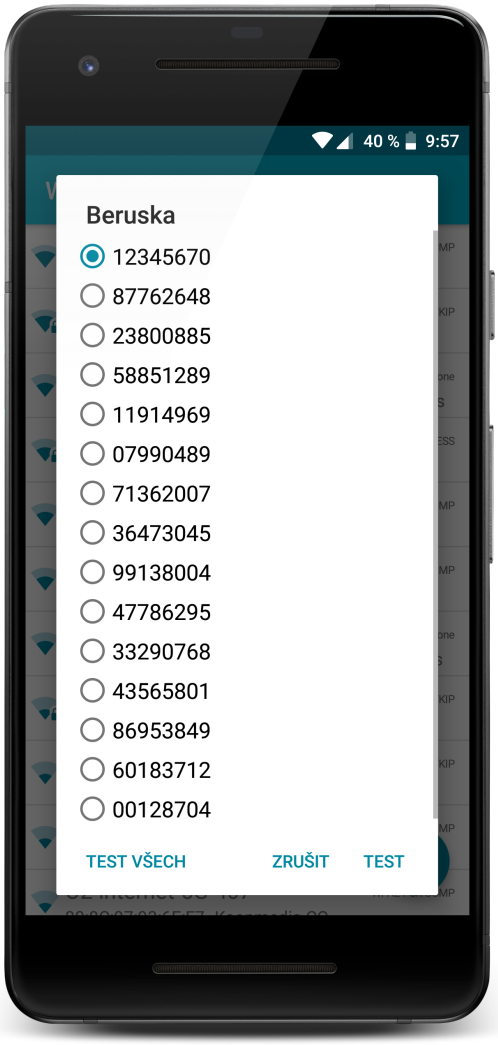
- **PinListDialog.** Objekt dialogového okna je inicializován konstruktorem s parametry `Context` pro udržení kontextu aktivity, a `ScanResult` obsahující informace o zvolené WiFi síti. Dojde zde k zavolání funkce `generatePins(String)` objektu `PinGenerator`, jíž je předána MAC adresa přístupového bodu zvolené sítě. Funkcí vrácené pole textových řetězců obsahuje vygenerovaná možná PIN čísla. Z každého tohoto řetězce je vytvořena grafická komponenta `RadioButton` (přepínací tlačítko), která je přidána do kořenového kontejneru `RadioGroup` sdružující tato tlačítka. Ten je následně zabalen do kontejneru `ScrollView`, čímž je

zajištěna možnost posuvu seznamu, pokud se všechna PIN čísla do dialogového okna vertikálně nevejdou. Obalující kontejner je konečně nastaven jako obsah dialogového okna metodou `setView(View)`.

```
protected PinListDialog(Context context, ScanResult network) {
    super(context);
    setTitle(network.SSID);
    ScrollView view = new ScrollView(context);
    rg = new RadioGroup(context);
    pins = PinGenerator.getInstance().generatePins(network.BSSID);
    for(String pin : pins) {
        RadioButton rb = new RadioButton(context);
        rb.setTextSize(18);
        rb.setText(pin);
        rg.setPadding(40, 14, 40, 14);
        rg.addView(rb, new RadioGroup.LayoutParams(ViewGroup
            .LayoutParams.MATCH_PARENT, ViewGroup
            .LayoutParams.MATCH_PARENT));
    }
    view.setScrollbarFadingEnabled(false);
    view.addView(rg);
    setView(view);
    ((RadioButton)rg.getChildAt(0)).setChecked(true);
}
```

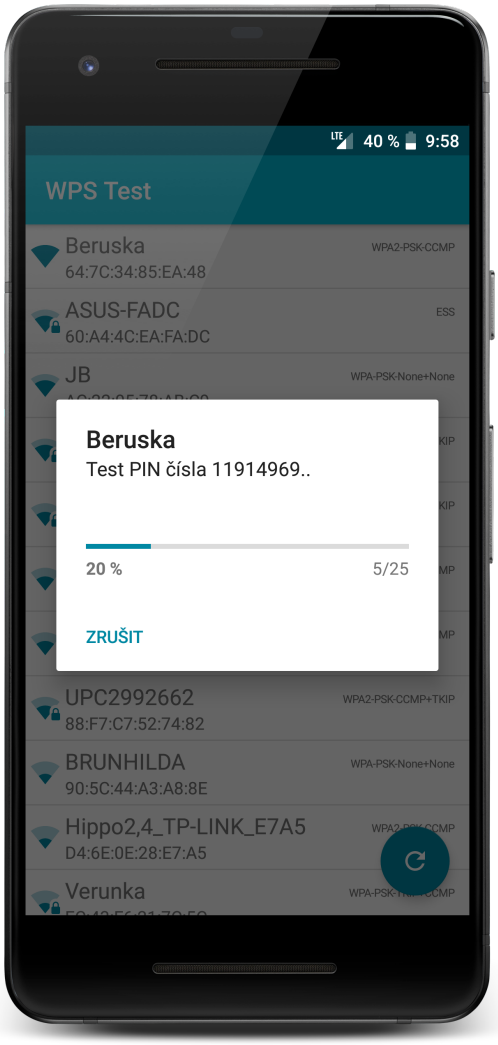
Příklad 5: Vygenerování PIN čísel a vytvoření přepínacích tlačítek

- **CustomPinDialog.** V konstruktoru dialogového okna je jako grafický obsah nastavena komponenta `android.widget.EditText` – editovatelné textové pole. Voláním `setInputType(InputType.TYPE_CLASS_NUMBER)` je omezen akceptovaný formát vstupu pouze na číslice. Tím je automaticky přizpůsobena i zobrazovaná klávesnice při aktivaci pole – nabízí pouze číselné hodnoty.



Obrázek 8: Dialogové okno se seznamem vygenerovaných PIN čísel

- **PinTryDialog.** Dialogové okno rozšiřující třídu `android.app.ProgressDialog`, tedy třídu dialogového okna pro zobrazení průběhu či vývoje nějaké operace. Konstruktor přijímá parametr `Context` pro udržení kontextu aktivity, a parametr `int max`, který udává počet PIN čísel, které se budou testovat. Tento počet je nastaven jako maximální hodnota indikátoru (progress baru), který bude v okně zobrazen. Voláním `setProgressStyle(ProgressDialog.STYLE_HORIZONTAL)` dojde k aktivaci právě onoho progress baru, jinak `ProgressDialog` implicitně zobrazuje pouze animované kolečko (spinner) s textovým popiskem, znázorňující probíhající operaci. Následně bude hodnota progress baru navyšována voláním metody tohoto objektu `incrementProgressBy(int)` z hlavní aktivity, a to po každém vyzkoušeném PIN čísle.



Obrázek 9: Dialogové okno zobrazující průběh testování jednotlivých PIN čísel

- **ScanningDialog.** Dialogové okno rozšiřuje třídu `android.app.ProgressDialog`. Oproti předchozímu dialogovému oknu stejného typu je u něj voláním metody `setIndeterminate(true)` v konstruktoru zaručeno, že se v okně namísto indikátoru průběhu zobrazí pouze animované kolečko s textovou informací, že právě probíhá skenování.

4.3.2 Aktivity

- **MainActivity.** Hlavní aktivita aplikace byla vytvořena z předdefinovaného vzoru Basic Activity, což je prázdná aktivita, jejímž jediným obsahem je grafická komponenta `android.support.design.widget.FloatingActionButton` – výrazné kruhové tlačítko ve spodní části obrazovky. Toto tlačítko bude v souladu s původ-

ním návrhem grafického rozložení aktivity použito pro spouštění skenování sítí.

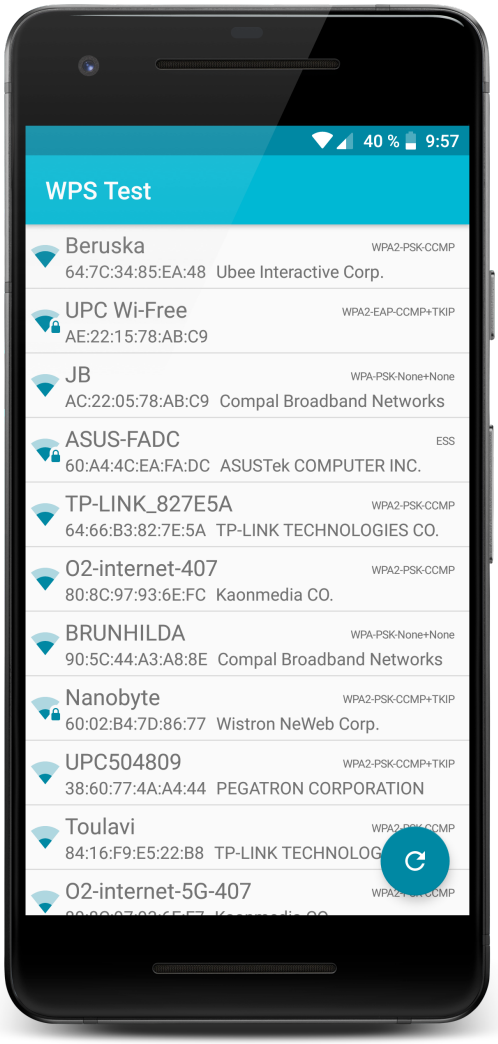
V Layout editoru byla v rozložení grafického rozhraní vytvořena nová komponenta `android.widget.ListView` – seznam pro zobrazování nalezených sítí a k němu textové pole, které je automaticky zobrazováno vždy, když v seznamu není žádná položka. To je nastaveno metodou `setEmptyView(View)` objektu `ListView`. Účelem textového pole je informovat uživatele o nutnosti spustit skenování sítí pomocí tlačítka.

Dle životního cyklu aktivity je při jejím startování nejprve spuštěna metoda `onCreate(Bundle)`. [14] Pomocí ní je v této aktivitě provedena inicializace výše popsaných grafických komponent, vytvoření a uložení referencí na systémové objekty `WifiManager` a `LocationManager`. Dále dojde k vytvoření instance třídy `android.content.BroadcastReceiver` (bližší specifikace viz [14]), která naslouchá jediné systémové události `SCAN_RESULTS_AVAILABLE_ACTION` – událost dokončení skenování sítí. Výsledek skenování je poté přijat jako `List` objektů `ScanResult` zavoláním funkce `getScanResults()` objektu `WifiManager`. Seznam detekovaných sítí je předán do nově vytvořené instance třídy `ScanResultsAdapter`, a instance samotná je nastavena komponentě `ListView` jako adaptér zavoláním `setAdapter(ListAdapter)`. Tím se v grafickém rozhraní aktivity zobrazí jednotlivé sítě.

Pokud je aplikace spuštěna na systému Android verze 6.0 a vyšší, je dále během spouštění aktivity ověřeno, a příp. vyžádáno přidělení oprávnění pro přístup k přibližné poloze zařízení. Oprávnění je u vyšších verzí systému nezbytné pro skenování sítí. Následně je komponentě seznamu sítí `ListView` prostřednictvím volání jeho metody `setOnItemClickListener(OnItemClickListener)` nastaven nově vytvořený posluchač události kliknutí na položku seznamu. Při vyvolání této události dojde nejdříve k prověření, zda v dané síti byla detekována aktivní funkce WPS. Pokud ne, zobrazí se uživateli pouze informace o této skutečnosti. Pokud je WPS ve zvolené síti aktivní, je zobrazeno dialogové okno s dotazem, zda bude testován ručně zadaný PIN, či zda bude vygenerován soubor možných PIN čísel pomocí algoritmů. Volbou první možnosti je uživateli zobrazeno dialogové

okno `CustomPinDialog`, volbou druhé možnosti je vytvořen a zobrazen dialog `PinListDialog`.

Ve všech případech je další postup obdobný: pro navázání WPS spojení se vytvoří instance třídy `WpsConnector`, které je předáno BSSID přístupového bodu a PIN (v případě testu celého vygenerovaného souboru první z nich). Následně je na tomto objektu zavolána metoda `connect(WpsCallback)` s nově vytvořeným objektem typu `WpsCallback`. Jeho stavovými metodami `onSucceeded()`, resp. `onFailed(int)` je uživatel obeznámen o výsledku testu. V případě testu celého souboru vygenerovaných PIN čísel je při neúspěšném pokusu aktualizována instance `WpsConnector` dalším PIN číslem v pořadí, až dokud nedojde k úspěšnému odhalení správného PIN, či nebudou otestovány všechny aplikací nabízené.



Obrázek 10: Hlavní aktivita aplikace se seznamem zjištěných WiFi sítí

4.4 Test aplikace

Vzniklá aplikace byla testována za stejných podmínek na stejných pěti routerech jako již existující aplikace v kapitole 3.5. Níže je uvedena tabulka s přehledem úspěšnosti vůči jednotlivým routerům. Symbol „X“ značí úspěšnost odhalení čísla PIN jednou z aplikací nabízených variant.

Tabulka 4: Výsledky praktického testu prototypu nové aplikace

	Tenda W311R+	**TP–Link TD–W8961NB**	**ASUS RT–N12 D1**	**Huawei LTE CPE B310**	**Netis WF2419**
Prototyp aplikace	X	X			X

5 Zhodnocení výsledků

Vzniklý prototyp nové aplikace při testování dosáhl stejné úspěšnosti, jako aktuálně uživateli nejlépe hodnocená aplikace na Google Play s názvem „WiFi Warden (WPS Connect)“. Úspěšnost odhalení byla shodně u 3 z 5 routerů. Je však třeba brát v potaz, že výrobci routerů na tuto zranitelnost začínají reagovat a zavádět různá bezpečnostní opatření, tudíž pokud by byly pro testování k dispozici novější routery, mohla by být úspěšnost nižší.

Výhodou vyvinuté aplikace může být oproti již existujícím zejména počet implementovaných algoritmů pro vygenerování PIN čísel z MAC adresy. Všechny existující testované aplikace nabízí k vyzkoušení pouze cca 5 – 10 možností, prototyp nové aplikace bezmála 30. Tím u routerů nezabezpečených proti této zranitelnost výrazně stoupá pravděpodobnost úspěchu.

V rámci praktické části práce se nepodařilo zrealizovat původní záměr implementace metody odhalení čísla PIN pomocí hrubé síly, a přiblížení se tak efektivitě a výkonnosti aplikací stejného zaměření pro desktopové operační systémy Linux a Windows. Nerealizovatelnost této funkčnosti plyne z omezení architektury platformy Android. V průběhu sestavování rešerše o komponentách pro obsluhu WPS protokolu obsažených v Android SDK bylo zjištěno, že systém obsluhuje jednotlivé zprávy protokolu interně a aplikaci se pouze dostane informace, zda bylo testované číslo PIN pro připojení správné, nebo ne. Tím je u OS Android znemožněno využití největší bezpečnostní slabiny technologie WPS.

Aplikace byla vyvinuta na zařízení Samsung Galaxy S5 s verzí Android 6.0. Testovací provoz prototypu aplikace však prokázal, že funguje bez problémů i na zařízeních Lenovo A2010 s verzí Android 5.1, Xiaomi Redmi 4X s OS Android 7.0 a dalších. Všechny textové popisky jsou v aplikaci ve dvou jazykových mutacích, a to česky a anglicky. V angličtině aplikace komunikuje vždy když je v systému nastaven kterýkoli jiný jazyk než čeština.

6 Závěr

Na nápad vývoje mobilní aplikace pro testování WPS zranitelnosti WiFi sítí přivedla autora práce předchozí osobní zkušenost se stejným typem aplikace pro platformu Windows na osobních počítačích. Před začátkem prací na návrhu a vývoji prototypu aplikace byla sestavena ucelená rešerše literárních zdrojů týkajících se problematiky zabezpečení bezdrátových WiFi sítí se zaměřením na technologii WPS z pohledu potenciálního bezpečnostního rizika. Dále byly charakterizovány základní programové komponenty pro obsluhu WPS protokolu poskytované v rámci standardního Android SDK. Následně byla provedena analýza existujících řešení dostupných pro platformu Android v oficiálním úložišti Google Play. Důležitým poznatkem při studiu teoretických podkladů se stal fakt, že mobilní aplikace jsou odstíněny od práce s vlastními zprávami WPS protokolu, čímž je značně snížena jejich efektivita oproti aplikacím desktopovým. Bylo zjištěno, že srovnatelné efektivity by bylo možné dosáhnout na adekvátně modifikované instalaci vlastního systému Android včetně nainstalovaného root oprávnění, což je však nad rámec této práce, a navíc při reálném užívání takto modifikovaného systému je značně narušeno zabezpečení mobilního zařízení.

I přes tyto poznatky se však podařilo vyvinout funkční aplikaci s intuitivním uživatelským rozhraním, jejíž cílovou skupinou by měli být lidé provozující v domácích či malopodnikových podmínkách svou vlastní bezdrátovou WiFi síť. Aplikace skenuje okolní WiFi sítě, zobrazuje jejich seznam s přehledným vyznačením sítí, ve kterých je aktivovaná funkce WPS. Po uživatelově výběru jedné z těchto sítí vygeneruje několika algoritmy na základě MAC adresy přístupového bodu soubor možných čísel PIN, které nabídne uživateli. Ten se pak může pokusit pomocí těchto variant připojit. Samozřejmě je obeznámen o výsledku každého z pokusů, ať už pozitivním, či negativním. Dále má uživatel možnost vyzkoušet jím zadaný PIN, čímž je suplována funkčnost, která v minulosti byla integrovaná přímo v systému, avšak později z něj byla odstraněna.

Vyvinutá aplikace by měla v reálném světě fungovat opravdu pouze jako tester zabezpečení domácí WiFi sítě, vlastněné a provozované uživatelem aplikace. Vzhledem k jejímu zaměření by však mohla být potenciálně zneužita pro neoprávněný vnik do cizí sítě, což je dle platné legislativy České republiky klasifikováno jako trestný čin

neoprávněného přístupu k počítačovému systému a nosiči informací dle § 230 odst. 1, 2 trestního zákoníku. Uživatel proto při prvním spuštění aplikace souhlasí s klauzulí, že aplikaci bude používat výhradně pro testování své vlastní WiFi sítě. Pokud se aplikaci podaří odhalit číslo PIN, provozovateli sítě se důrazně doporučuje zaujmout některé bezpečností opatření, z nichž nejúčinnější je vždy úplná deaktivace WPS (příp. alespoň přepnutí na režim PBC, kde je pro připojení do sítě zapotřebí fyzický kontakt s routerem). WPS technologie zpravidla za svým původním účelem, za jakým byla navržena, není uživateli sítě používána na denní bázi.

7 Seznam použitých zdrojů

[1] SUNDAR, S. *The Theory of Wi-Fi Evolution and IEEE 802.11 Selection* [online]. [cit. 2017-12-02]. Dostupné z: http://www.net-ctrl.com/2016/07/14/the-theory-of-wi-fi-evolution-and-ieee-802-11-selection/

[2] BARTOLIC, I. How WLAN Works – Your Best WLAN Tutorial: 5 Stages in the WiFi Network. *TheBestWirelessInternet* [online]. [cit. 2017-12-12]. Dostupné z: http://thebestwirelessinternet.com/how-wlan-works.html

[3] PUŽMANOVÁ, R. *Bezpečnost bezdrátové komunikace*. Brno: Computer Press, 2005. ISBN 80-251-0791-4.

[4] BARTOLIC, I. Wireless Authentication and How to Ensure the Best Wireless Internet Security. *TheBestWirelessInternet* [online]. [cit. 2017-12-12]. Dostupné z: http://thebestwirelessinternet.com/wireless-authentication.html

[5] JELÍNEK, M. *Bezpečnost bezdrátových počítačových sítí.* Brno, 2010. Diplomová práce. Vysoké učení technické v Brně. Vedoucí práce Radek Doležel.

[6] OSTERHAGE, W. *Wireless Security*. Florida, USA: CRC Press, 2016. ISBN 978-1578087686.

[7] ROHLEDER, D. a Václav LORENC. 802.1x – autentizace v počítačových sítích. *Zpravodaj ÚVT MU*. Brno, 2008. ISSN 1212-0901.

[8] KWAN, P. *White paper: 802.1x authentication & extensible authentication protocol (EAP)* [online]. 2003 [cit. 2017-12-28]. Dostupné z: https://www.scribd.com/document/6613012/Wp-8021x-Authentication-Eap

[9] NOVÁK, M. Odposlouchávání a prolamování Wi-Fi sítí zabezpečených pomocí WPA2. *Root.cz* [online]. [cit. 2018-02-07]. Dostupné z: https://www.root.cz/clanky/odposlouchavani-a-prolamovani-wi-fi-siti-zabezpecenych-pomoci-wpa2/

[10] ŠURKALA, M. KRACK: bezpečnostní díra ve WPA2, Wi-Fi není bezpečná. *Svět Hardware* [online]. 16.10.2017 [cit. 2018-02-09]. Dostupné z: https://www.svethardware.cz/krack-bezpecnostni-dira-ve-wpa2-wi-fi-neni-bezpecna/45342

[11] VANHOEF, M. *Key Reinstallation Attacks: Breaking WPA2 by forcing nonce reuse* [online]. KU Leuven, 2017 [cit. 2018-02-09]. Dostupné z: https://www.krackattacks.com/

[12] Wi-Fi Simple Configuration Technical Specification. *Wi-Fi Alliance* [online]. 2014 [cit. 2018-01-02]. Dostupné z: https://www.wi-fi.org/download.php?file=/sites/default/files/private/Wi-Fi_Simple_Configuration_Technical_Specification_v2.0.5.pdf

[13] VIEHBÖCK, S. *Brute forcing Wi-Fi Protected Setup* [online]. 2011 [cit. 2018-01-22]. Dostupné z: https://sviehb.files.wordpress.com/2011/12/viehboeck_wps.pdf

[14] HOMOLA, R. *Vývoj mobilní aplikace pro platformu Android*. Praha, 2016. Bakalářská práce. Česká zemědělská univerzita v Praze. Vedoucí práce Pavel Šimek.

[15] Build a Responsive UI with ConstraintLayout. *Android Developers* [online]. [cit. 2018-02-18]. Dostupné z: https://developer.android.com/training/constraint-layout/index.html

[16] ConstraintLayout. *Android Developers* [online]. [cit. 2018-02-18]. Dostupné z: https://developer.android.com/reference/android/support/constraint/ConstraintLayout.html

[17] WifiManager. *Android Developers* [online]. [cit. 2018-02-18]. Dostupné z: https://developer.android.com/reference/android/net/wifi/WifiManager.html

[18] ScanResult. *Android Developers* [online]. [cit. 2018-02-18]. Dostupné z: https://developer.android.com/reference/android/net/wifi/ScanResult.html

[19] WpsInfo. *Android Developers* [online]. [cit. 2018-02-18]. Dostupné z: https://developer.android.com/reference/android/net/wifi/WpsInfo.html

[20] WifiManager.WpsCallback. *Android Developers* [online]. [cit. 2018-02-18]. Dostupné z: https://developer.android.com/reference/android/net/wifi/WifiManager.WpsCallback.html

[21] Varargs. *JDK^{TM} 5.0 Documentation* [online]. 2010 [cit. 2018-02-18]. Dostupné z: https://docs.oracle.com/javase/1.5.0/docs/guide/language/varargs.html

[22] AsyncTask. *Android Developers* [online]. [cit. 2018-02-18]. Dostupné z: https://developer.android.com/reference/android/os/AsyncTask.html

[23] PECINOVSKÝ, R. *Myslíme objektově v jazyku Java – kompletní učebnice pro začátečníky*. Praha: Grada, 2009. ISBN 978-80-247-2653-3.

[24] WPS PIN generator. *3WiFi* [online]. [cit. 2018-03-12]. Dostupné z: https://3wifi.stascorp.com/wpspin

[25] Android Drawable Importer. *JetBrains Plugin Repository* [online]. [cit. 2018-03-12]. Dostupné z: https://plugins.jetbrains.com/plugin/7658-android-drawable-importer

[26] Mac address lookup API. *Mac Vendors* [online]. [cit. 2018-03-12]. Dostupné z: http://macvendors.co/api

8 Přílohy

Přílohou této diplomové práce je CD s kompletním zdrojovým kódem vyvinutého prototypu aplikace včetně snímků obrazovek a apk souboru, pomocí nějž může být aplikace nainstalována na zařízení s operačním systémem Android.