VŠB TECHNICKÁ UNIVERZITA OSTRAVA
FAKULTA ELEKTROTECHNIKY A INFORMATIKY

Viaccestná komunikácia v QKD sieťach

Multipath communication in QKD networks

Bc. Patrik Burdiak

Diplomová práce

Vedoucí práce: prof. Ing. Miroslav Vozňák, Ph.D.

Ostrava, 2022

VŠB – Technická univerzita Ostrava
Fakulta elektrotechniky a informatiky
Katedra telekomunikační techniky

Zadání diplomové práce

Student: **Bc. Patrik Burdiak**

Studijní program: N2647 Informační a komunikační technologie

Studijní obor: 2601T013 Telekomunikační technika

Téma: Vícecestná komunikace v QKD sítích
Multipath Communication in QKD Networks

Jazyk vypracování: slovenština

Zásady pro vypracování:

Cílem diplomové práce je navrhnout a implementovat způsob vícecestné komunikace v sítích využívajících kvantovou distribuci klíčů QKD (Quantum Key Distribution). Očekávaným výstupem je rozšíření modulu QKDNetSim v prostředí open source síťového simulátoru NS-3, který bude nově podporovat vícecestnou komunikaci ve veřejném kanále.Vyjednávání přes veřejný kanál má vliv na celkový QKD spoj a výstup této diplomové práce tak umožní prozkoumat efektivitu vícecestné komunikace jako QoS nástroje pro přicházející kvantovou komunikační infrastrukturu.

Vypracování práce bude splňovat následující body zadání:
1. Úvod do distribuce klíčů pomocí principů kvantové mechaniky.
2. NS-3 a jeho využití při síťových simulacích.
3. Návrh vícecestné komunikace a její implementace v QKDNetSim.
4. Experimentální ověření vícecestné komunikace v simulacích.
5. Shrnutí dosažených výsledků.

Seznam doporučené odborné literatury:

[1] M. Mehic, O. Maurhart, S. Rass, D. Komosny, F. Rezac, M. Voznak, Analysis of the Public Channel of Quantum Key Distribution Link (2017) IEEE Journal of Quantum Electronics, Vol. 53, No. 5.
[2] M. Mehic, O. Maurhart, S. Rass, M. Voznak, Implementation of quantum key distribution network simulation module in the network simulator NS-3 (2017) SPRINGER Quantum Information Processing, Vol. 16, No. 10, pp. 253 (23).

Formální náležitosti a rozsah diplomové práce stanoví pokyny pro vypracování zveřejněné na webových stránkách fakulty.

Vedoucí diplomové práce: **prof. Ing. Miroslav Vozňák, Ph.D.**

Konzultant diplomové práce: Ing. Miralem Mehić, Ph.D.

Datum zadání: 01.09.2021

Datum odevzdání: 30.04.2022

prof. Ing. Miroslav Vozňák, Ph.D.	prof. Ing. Jan Platoš, Ph.D.
vedoucí katedry	*děkan fakulty*

Abstrakt

Táto diplomová práca sa zaoberá technológiou Quantum Key Destribution (QKD) a simuláciou tejto technológie pomocou simulátora QKDNetSim. Teoretická časť práce obsahuje úvod do problematiky kvantovej distribúcie kľúčov, ktorá sa nespolieha na matematické problémy ako konvenčne používaná kryptografia ale na zákonitosti kvantovej mechaniky. Praktická časť práce popisuje rozšírenie simulátoru QKDNetSim o viaccestnú komunikáciu a následnú implementáciu. Záver práce sa venuje popisu a experimentálnemu overeniu viaccestnej komunikácie v simuláciách následne zhodnoteniu dosiahnutých výsledkov.

Kľúčové slová

QKD; Viaccestná Komunikácia; Load Balancing; AODV

Abstract

This diploma thesis deals with Quantum Key Destruction (QKD) technology and the simulation of this technology using the QKDNetSim simulator. The theoretical part of the thesis contains an introduction to the issues of quantum key distribution, which does not rely on mathematical problems such as conventionally used cryptography but on the laws of quantum mechanics. The practical part of this thesis describes the extension of the QKDNetSim simulator by multipath communication and subsequent implementation. The conclusion of the work is devoted to the description and experimental verification of multipath communication in simulations and subsequent evaluation of the achieved results.

Keywords

QKD; Multipath Communication; Load Balancing; AODV

Poďakovanie

Na tomto mieste by som rád poďakoval za podporu všetkým, vďaka ktorým mohla táto práca vzniknúť. Vedúcemu diplomovej práce prof. Ing. Miroslavovi Vozňakovi Ph.D. a konzultantovi Ing. Miralemovi Mehićovi Ph.D. za odborné vedenie, konzultácie, trpezlivosť a podnety k zlepšeniu kvality práce. V neposlednom rade by som rád poďakoval mojej rodine a priateľom za podporu.

Obsah

Zoznam použitých skratiek a symbolov

5G	–	Fifth-generation wireless
AES	–	Advanced Encryption Standard
AODV	–	Ad-hoc On-demand Distance Vector
DSDV	–	Destination-Sequenced Distance-Vector
GUI	–	Graphical User Interface
IP	–	Internet Protocol
KME	–	Key Management Entity
LTE	–	Long Term Evolution
MAC	–	Media Access Control Address
MANET	–	Mobile Ad Hoc Network
OSPF	–	Open Shortest Path First
OTP	–	One Time Pad
QKD	–	Quantum Key Destribution
REST-API	–	Representational State Transfer-Application Programming Interface
RSA	–	Rivest-Shamir-Adleman
SAE	–	Secure Application Entity
SNR	–	Signal to Noise Ratio
TCP	–	Transmission Control Protocol
UDP	–	User Datagram Protocol
ZRP	–	Zone Routing Protocol

Zoznam obrázkov

Zoznam tabuliek

Kapitola 1

Úvod

Bezpečnosť dát, prenášaných či už verejným internetom alebo vo firemných sieťach, je dôležitým aspektom modernej komunikačnej infraštruktúry. Takáto digitálna bezpečnosť sa dotýka každého jedinca, napríklad pri používaní internetového bankovníctva, sociálnych sieti, cloudových služieb alebo aj vládnych inštitútov, ktoré potrebujú ochrániť svoje citlivé dáta. Dnes používané kryptografické systémy, ktoré využívajú mechaniku verejných kľúčov dokážu byť prelomené pomocou algoritmov, ktoré využívajú výkonnosť kvantových počítačov. V roku 1996 Lov Grover predstavil algoritmus [1], ktorý ovplyvňuje bezpečnosť symetrických šifier. Tento algoritmus poukazuje na to, že aj symetrická šifra dokáže byť prelomená. O rok neskôr, Peter Shor [2], demonštroval svoj algoritmus ktorý poukázal na matematické slabiny ktoré obsahujú algoritmy s využitím verejného kľúča. Pri predstave rozrastajúcej sa infraštruktúry kvantových počítačov vzniká obava o bezpečnosť prenášaných dat. Ako riešenie tohto problému sa ponúka technológia Quantum Key Distribution (QKD).
QKD slúži pre distribúciu binárnych kľúčov, ktoré sa následne využijú pre symetrické šifrovanie. Bezpečnosť symetrickej kryptografie je v podstate nedotknutá. Predpokladá sa, že Shorov algoritmus prelomí kryptografický systém, ktorý používa verejný kľúč v priebehu niekoľkých hodín pomocou jedného kvantového počítača, ktorý pracuje s miliónmi qubitov. V kontraste so Shorovým algoritmom, by prelomenie symetrickej šifry trvalo tomu Groverovmu exponenciálne dlhší čas. [3]. QKD v kombinácií so symetrickým šifrovaním nám teda ponúka bezpečnosť dát v kvantovej ére. QKD technológia je neustále vyvíjaná a vylepšovaná pre čo najjednoduchšiu implementáciu do stávajúcej sieťovej infraštruktúry. Paralelne prebieha aj výskum zo strany vedcov a akademikov ohľadom simulácií QKD sieti [4, 5, 6, 7, 8, 9, 10, 11]. Zároveň prebieha vývoj nových protokolov ako napríklad QSIP [12] a simulátorov ako QKDNetSim [13]. V rámci prípravy na kvantovú éru je potrebné adaptovať používané smerovacie protokoly. Nakoľko ako aj bude vysvetlené v tejto diplomovej práci QKD sieť ma rozličné prevádzkové atribúty od konvenčne používaných sieti. QKD prináša nové paradigma, a to spočíva v tom že sieť pre ustanovenie je separovaná, skladá za z QKD spojov. QKD spoj obsahuje verejný a kvantový kanál. Jedným z odlišných atribútov od konvenčných sieti je kľúčový materiál, ktorý je používaný pre symetrické šifrovanie. Tento parameter je kritický pre

fungovanie siete obsahujúcej QKD uzly. Pokiaľ sa stane kľúčový materiál nedostupný, spoj cez ktorý prúdia šifrované dáta sa stane nedostupnou. Takéto správanie vedie k možným výpadkom v sieti a teda sieť sa stáva nestabilnou. Doteraz používané smerovacie protokoly neberú do úvahy takýto atribút. V tejto práci je navrhnutý a implementovaný smerovací protokol, ktorý využíva všetky dostupné cesty k cieľu (viaccestná komunikácia), využíva navrhnutý load balancing a zohľadňuje vlastnosti QKD siete. Práca je rozdelená do niekoľký kapitol, prvou z nich je úvod do distribúcie kľúčov pomocou kvantovej mechaniky, ktorá vysvetľuje princípy a najpoužívanejšie protokoly QKD, následne NS-3 a jeho využitie pri sieťových simuláciách s popisom QKDNetSim simulátora, návrh viaccestnej komunikácie a jej implementácia v QKDNetSim, experimentálne overenie viaccestnej komunikácie v simuláciách a záver práce sa venuje vyhodnoteniu dosiahnutých výsledkov.

Kapitola 2

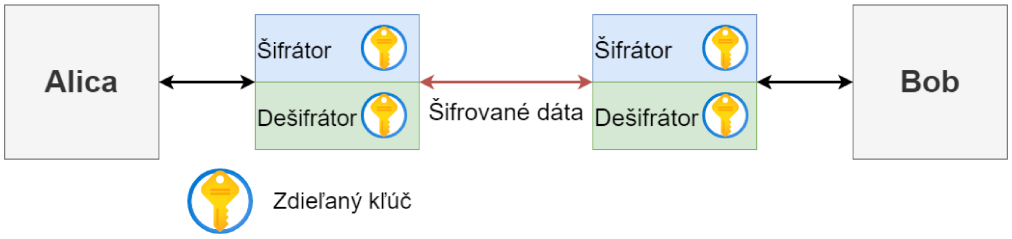
Distribúcia kľúčov pomocou kvantovej mechaniky

Aby sme si mohli vysvetliť myšlienku a základné princípy kvantovej distribúcie kľúčov, je nutné si vysvetliť konvenčne používanú kryptografiu a jej jednotlivé termíny. Kryptografiu môžme vysvetliť ako spôsob zašifrovania komunikácie medzi dvomi stranami. Komunikácia musí byť samozrejme zašifrovaná tak aby tretia strana nemohla dešifrovať danú komunikáciu. Pre dosiahnutie tohto cieľa sú využívané určité algoritmy (tiež nazývané šifry alebo kryptosystémy). Spomenuté algoritmy sú obsiahnuté v kryptografických systémoch, kde esenciálna súčasť je kľúč. Pomocou tohto kľúča a určitých matematických operácii sa kľúč kombinuje s prenášanými dátami. Odveký problém spočíva v distribúcii a ustanovení takéhoto kľúča. Tento krok sa nazýva šifrovanie a výsledkom je kryptogram. Pre dešifrovanie kryptogramu, teda aby sme dostali z kryptogramu pôvodnú stranu je potrebná znalosť kľúča. Myšlienka je teda komunikácia medzi dvomi stranami zostane dôverná a pre tretiu stranu, ktorá daný kľúč pre dešifrovanie nemá, sa táto komunikácia zdá nečitateľná [14].

2.1 Symetrická kryptografia

Symetrická kryptografia využíva šifrovania pomocou rovnakých kľúčov na obidvoch stranách. Táto metóda je rýchlejšia ako asymetrické šifrovanie avšak nastáva problém s distribúciou kľúča. Teda ako tento kľúč rozdistribuovať tak aby zostal tajný pre tretiu stranu. Šifra one time pad (OTP) patrí do tejto kategórie, bola navrhnutá Gilbertom Vernamom z AT&T v roku 1926 [15]. Pri tejto šifre Alica využije vzťah $m_1 \oplus k$ kde m_1 je správa v binárnej podobe , k je náhodne vygenerovaný kľúč a $\oplus$ označuje binárne sčítanie modulo 2. Bob obdrží zašifrovanú spravu, dešifruje ju pomocou vzťahu $s \ominus k = m_1 \oplus k \ominus k = m_1$. Alica aj Bob musia mať samozrejme rovnaký kľúč a dĺžka kľúča musí byť rovnaká ako dĺžka správy. Kľúč môže byť použitý len pre šifrovanie jednej správy. Preto aj nesie názov one time pad. Symetrická šifra je výpočetne efektívnejšia než asymetrická šifra. Na Obr.

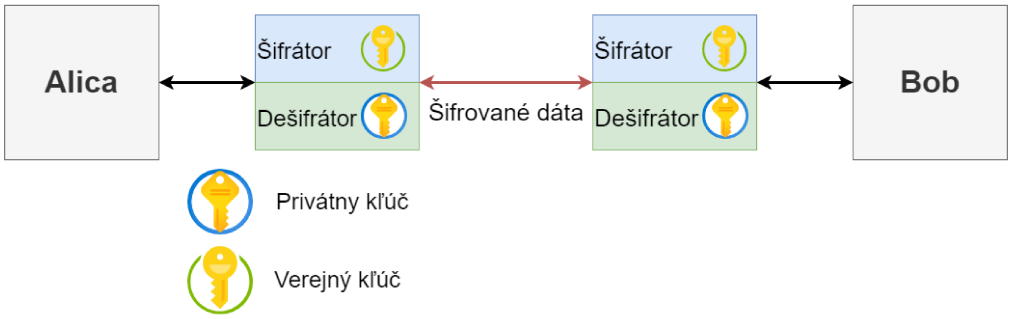
2.1 si môžme všimnúť princíp fungovania symetrických šifier, šifrátor aj dešifrátor zdieľajú totožný kľúč. [14, 16].



Obr. 2.1: Symetrické šifrovanie

2.2 Asymetrická kryptografia

Asymetrická kryptografia tiež známa ako šifrovanie za pomoci verejného kľúča rieši problém distribúcie kľúčov. Pri symetrickej šifre ako už bolo spomenuté vyššie, je šifrované aj dešifrovanie vykonávané pomocou jedného rovnakého kľúča, tu avšak nastáva otázka ako tento kľúč rozšíriť medzi komunikácie strany tak aby to bolo bezpečné. Teda aby tretia strana nemala znalosť tohto kľúča. Asymetrická kryptografia využíva dvojicu kľúčov. Verejný a privátny. Verejný kľúč si dve strany pošlú navzájom medzi sebou navzájom. Privátny si obe strany udržiavajú u seba. Šifruje sa pomocou verejného kľúča protiľahlej strany, avšak dešifrovanie sa vykonáva pomocou vlastného privátneho kľúča. Alica a Bob chcú medzi sebou komunikovať a prenos bude šifrovaný pomocou asymetrickej šifry. Alica a Bob si vymenia svoje verejné kľúče. Alica zašifruje svoje dáta pomocou Bobovho verejného kľúču, akonáhle dáta dorazia k Bobovi ten ich dešifruje pomocou svojho privátneho kľúča. [14, 17] Typickým predstaviteľom asymetrickej kryptografie je RSA šifra pomenovaná po jej tvorcoch Rivest, Shamir, Adleman. Na Obr. 2.2 si môžme všimnúť princíp fungovania asymetrických šifier, šifrovanie je vykonávané pomocou verejného kľúča a dešifrovanie pomocou privátneho kľúča [18].



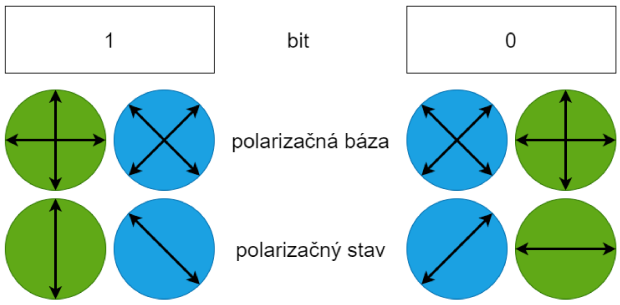
Obr. 2.2: Asymetrické šifrovanie

2.3 Kvantová distribúcia kľúčov

Myšlienka ohľadom kvantovej kryptografie bola prvotne navrhnutá v sedemdesiatych rokoch minulého storočia, autorom bol Stephen Wiesner. Avšak táto oblasť kryptografie získala pozornosť až po vydaní článku od autorov Charles H. Bennetta a Gillesa Brassarda v roku 1984 [19]. V tomto článku predstavili protokol, ktorý teraz poznáme pod názvom BB84. Cieľom kvantovej kryptografie je vykonávať úlohy ktoré nie je možné dosiahnuť s konvenčnou kryptografiou. Tento druh kryptografie využíva princípy kvantovej mechaniky pre dosiahnutie symetrického binárneho kľúča medzi legitimnými stranami ktoré následne využijú tento kľúč pre šifrovanie. Kvantové systémy a technológie súvisia s efektami atómov, elektrónov alebo fotónov, využívajú ich chovanie a charakteristiku. Informácie sú obsiahnuté vo fotónoch ktoré nazývame qubity. Narozdiel od klasického bitu kde hodnota môže mať len dve podoby a to 0 a 1, qubit môže reprezentovať akúkoľvek možnú superpozíciu $|0>$ a $|1>$. To znamená že qubit nepredstavuje hodnotu 0 alebo 1, ale v skutočnosti môže obsahovať akýkoľvek stav zmesí 0 a 1. Meraním je možné určiť pravdepodobnosť určitého stavu qubitu, ale na rozdiel od klasických meraní pri ktorých zostáva bit nedotknutý, meranie qubitu je skôr podobné použitiu filtra, čo núti qubit predpokladať buď stav „0“ alebo „1“, každý s určitou pravdepodobnosťou. To znamená, že meranie donúti qubit prejsť zo stavu superpozície a skolabovať na jednu z týchto dvoch možností. Bezpečnosť ohľadom prenášania qubitov cez optické vlákno je zabezpečená pomocou Heisenbergovho princípu a no-cloning teorému. Heisenbergov princíp hovorí o tom že nie je možné pasívne merať kvantové stavy. Keby tretia strana chcela zachytávať tieto fotóny, nie je možné aby nepozmenila prenášanú informáciu, teda dané fotóny. No-cloning teorém hovorí o tom že nie je možné duplikovať neznámy kvantový stav pri zachovaní pôvodného stavu, teda útok man in the middle je vylúčený. Existujú rôzne QKD protokoly ako BB84 [19], BB92 [20], E91 [21], SSP [22], SAG014 [23]. Protokoly BB84 a BB92 si podrobne rozoberieme pre uvedenie problematiky QKD protokolov [14, 24, 25, 26].

2.4 Protokol BB84

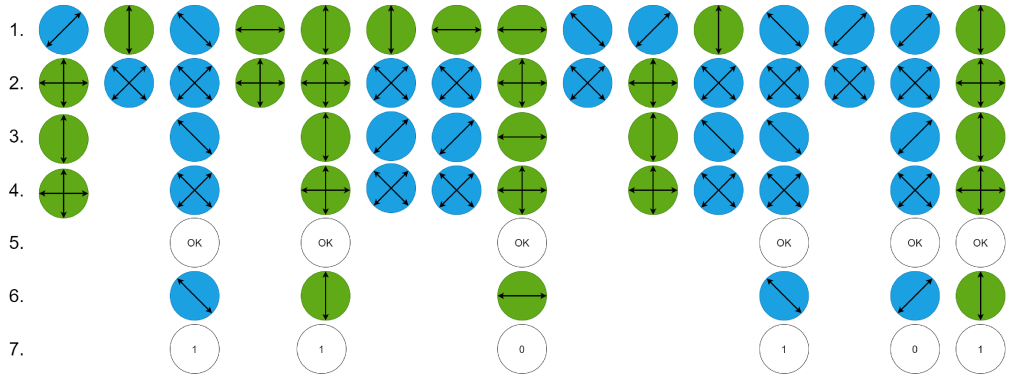
Bennet a Brassard v roku 1984 predstavili prvý protokol pre kvantovú distribúciu kľúčov (Quantum key distribtuion - QKD) nazývaný BB84 [19]. Tento protokol využíva polarizáciu fotónov pre dosiahnutie zdieľaného kľúča, ktorý ako už bolo spomenuté sa využije pre následné šifrovanie. BB84 využíva dve polarizačné bázy a to priamočiaru bázu (+) a diagonálnu bázu (x). V priamočiarej báze ja fotón polarizovaný na stavy $\rightarrow$ alebo $\uparrow$. V diagonálnej báze sú to stavy $\nwarrow$ alebo $\nearrow$. Komunikujúce strany sa musia dohodnúť ktoré stavy budu prezentovať akú bitovú hodnotu [27, 14, 26].



Obr. 2.3: Polarizačné stavy BB84

V konečnom dôsledku máme štyri polarizačné stavy fotónov, každá báza má dva polarizačné stavy. Komunikácia pre výmenu zdieľaného kľúča, sa skladá zo šiestich krokov:

1. Výmena tajného kľúča
2. Sifting (Preosievanie)
3. Odhad chybovosti
4. Oprava chýb
5. Privacy Amplification (Zosilnenie bezpečnosti)



Obr. 2.4: Priebeh komunikácie BB84

Iba prvý krok sa odohráva skrz kvantový kanál, všetky ostatné kroky prebiehajú skrz verejný kanál. Ako môžme vidieť na obrázku 2.4 Alica chce poslať sekvenciu bitov. Alica však vyberie náhodnú polarizáciu zo štyroch možných pre každý bit. Tieto fotóny putujú k Bobovi, ktorý musí zmerať tieto polarizačné stavy. Pre každý prichádzajúci fotón zvoli Bob náhodnú polarizačnú bázu (+ alebo x) na meranie. Bob teda v priemere získava reťazec bitov kde len 50% bitov je správnych. Tento reťazec je nazývaný raw key. Následne pre každý qubit Bob ohlási Alici cez verejný kanál akú polarizačnú bázu predpokladal. Alica mu následne odpovie tak že mu ohlási ktoré meranie boli

správne. Tento proces je nazývaný sifting (preosievanie), pričom vo výsledku obidve strany zdieľajú rovnakú binárnu sekvenciu. [27, 28, 29] Vlastnosti kvantového prenosu nám zabezpečujú že pri pokuse o odpočúvanie jedného fotónu, sa náhodne zmení polarizačný stav daného fotónu. Teda v reálnej situácii, kde môžme predpokladať odpočúvanie na kvantovom kanále, Bobom zvolené bazy pre merania polarizácie môžu byť správne avšak výsledok tohto merania bude nesprávny. Takáto zmena polarizácie môže nastať aj v samotnom optickom vlákne, kvôli jeho nedokonalostiam. Preto musia Alica a Bob pokračovať v komunikácii cez verejný kanál a testovať koreláciu svojich bitov, tento krok dosiahnú tak že zverejnia niektoré bity zo svojich sekvencii. Tieto bity budú následne odstránené zo sekvencie. Na základe odhadovej chybovosti, tiež nazývanej aj Quantum Bit Error Rate (QBER), obidve strany môžu odhaliť prítomnosť tretej strany. Ak by bola hodnota QBERu vyššie než očakávaná celá bitová sekvencia sa zahodí a proces sa začne odznova. V opačnom prípade pokračujú v oprave chýb. V tomto kroku musia obidve strany zosúladiť ich zdieľané bity. Opäť táto komunikácia prebieha skrz verejný kanál, teda komunikácia musí prebiehať v spôsobe takom aby Alica a Bob prezradili čo najmenej informácii o ich zdieľaných kľúčoch. Po oprave chýb Alica a Bob zdieľajú totožnú sekvenciu bitov, ktorá je známa iba im. Za predpokladu že tretie strana teda niekto kto by chcel odpočúvať danú komunikáciu, získal informácie či už z verejného alebo kvantového kanála, prejdú Alica a Bob na krok Privacy Amplification (Zosilnenie bezpečnosti). V tomto kroku sa zmenšia zdieľané kľúče pre snahu o čo najväčšie zmenšenie znalosti zdielaného kľúča treťou stranou [30, 14, 26].

2.5 Protokol B92

V roku 1992, Benner vytvoril protokol B92 [20], v myšlienke zjednodušiť pôvodný protokol BB84. Tento protokol spočíva vo využívaní iba dvoch polarizácii fotónov namiesto štyroch ako to je pri BB84. Majme opäť situáciu kedy chce Alica zdieľať tajný kľúč s Bobom. V prvom kroku Alica musí vytvoriť náhodnú binárnu sekvenciu $S_A = [001110]$ o určitej dĺžke Q.

$$|\varphi\rangle = \begin{cases} |0\rangle & ak\ S_A[i] = 0 \\ |1\rangle & ak\ S_A[i] = 1 \end{cases} \tag{2.1}$$

Z rovnice 2.1, kde i je ukazovateľ na prvok v binárnej sekvencii, môžme vidieť že 1 a 0 majú rozdielne polarizácie fotónov, a používajú sa len dve polarizácie. Alica teda pošle polarizované fotóny Bobovi. Bob nemá žiadnu informáciu o tom ako vyzerá Alicina binárna sekvencia, takže si vytvorí svoju náhodnú sekvenciu $S_B = [011011]$, na túto sekvenciu však musí uplatniť rovnaký spôsob polarizácie ako Alica (2.1). Čiže na prvok $i = 0$ použije bázu, ktorá zodpovedá polarizácii pre 0. Avšak v situácii keď Bob nezvolí správnu bázu pre meranie polarizácie fotónu, nezaznamená nič. Po ukončení meraní, Bob informuje Alicu o tom, akú bázu použil pre jednotlivé bity. Bob nezverejní bitové hodnoty ale len použitú bázu pre daný bit. Následne Alica a Bob zahodia bity pre ktoré Bob

použil nesprávnu bázu. B92 obsahuje identické "post-processing"kroky (Sifting, Odhad chybovosti, Oprava chýb, Privacy Amplification) ako vyššie zmienený BB84 [13, 31].

2.6 DV-QKD a CV-QKD

V tejto podkapitole si predstavíme dve základné triedy QKD systémov.

2.6.1 CV-QKD

CV-QKD alebo *Continues Variable QKD*, predstavuje jednu z dvoch základných tried QKD. CV-QKD systémy sú založené na homodynnej detekcii [32, 33]. Informácia o zdieľanom kľúči nie je prenášaná jedným fotónom, ale je prenášaná pomocou vlnových vlastností svetla a využitím multi fotónových kvantových stavov. Už koncom 90-tych rokov [34], vzišla prvá myšlienka ohľadom CV-QKD protokolov. Neskôr Grosshans a Grangier navrhli koherentný stavovo vyvážený homodynnný detekčný protokol [35]. Toto riešenie, teraz známe ako protokol GG02, predpokladá že Alica náhodne moduluje Gaussov lúč (moduluje sa fáza aj amplitúda s Gaussovými náhodnými číslami) a pošle ho Bobovi, ktorý zmeria buď fázu alebo amplitúdu lúča a informuje Alicu, ktoré meranie vykonal. Alice a Bob potom majú dve korelované množiny gaussovských premenných, z ktorých môžu extrahovať tajný kľúč. Výhodou CV-QKD systémov je vyššia efektivita a možnosť integrácie s existujúcimi systémami optického prenosu dát. V roku 2013 [36], bol vykonaný experiment kde cez 25 km optické vlákno bola dosiahnutá rýchlosť generácie kľúčového materiálu 10 kb/s pomocou GG02 protokolu. Maximálna možná dĺžka optického vlákna avšak mohla byť až 80,5 km ale pri tejto dĺžke nastal rapídny pokles rýchlosti generovanie kľúčového materiálu a to na niekolko stoviek bitov za sekundu.

2.6.2 DV-QKD

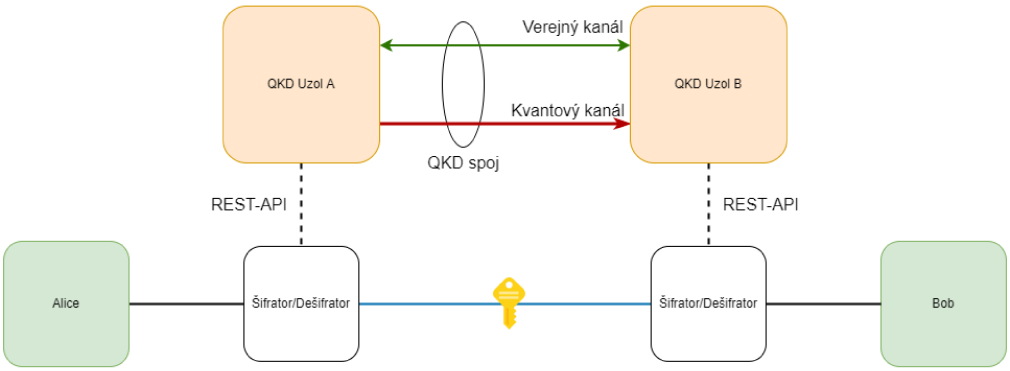
DV-QKD alebo *Discrete Variable QKD*, používa optické detektory identifikujúce jednotlivé fotóny, ide o diskrétne meranie vo forme áno či nie. Prekonanie veľkej vzdialenosti vyžaduje kvantové opakovače, ktoré sú avšak stále vo vývoji, preto táto metóda je veľmi náročná pre implementáciu. Distribúcia kľúčového materiálu závisí na detekcii jedného fotónu. Výsledkom prenosu a analýzy je diskrétna hodnota, teda či je fotón detegovaný alebo nie. Tu môžme vidieť rozdiel medzi CV-QKD a DV-QKD, pomocou CV-QKD môžme dosiahnúť vyššie prenosové rýchlosti pri šírení kľúčového materiálu, pretože využíva multi fotónové kvantové stavy a informácia sa neprenáša len jedným fotónom. Navyše CV-QKD je menej finančne náročné na implementáciu než DV-QKD [37, 38].

2.7 Architektúra QKD sieti

V predchádzajúcich kapitolách sme si ozrejmili konvenčne používanú kryptografiu aj kvantovú kryptografiu a jej využitie. V tejto kapitole bude popísaná samotná QKD technológia, vrátane toho ako má spoj medzi dvomi QKD uzlami vyzerať

2.7.1 QKD spoj

QKD spoj tvoria dva kanály, jeden z nich je kanál kvantový, ktorý slúži pre výmenu kľúčov ktoré sú zakódované v určitých vlastnostiach fotónov, ako napríklad polarizácia. Verejný kanál slúži pre verifikáciu vymenených kľúčov medzi dvoma stranami a cez verejný kanál je taktiež možný prenos šifrovaných dat. Kombinácia týchto dvoch kanálov nám tvorí QKD spojenie, pre výmenu kľúčov. V porovnaní s konvenčnými spojeniami dvoch uzlov sieti tu môžme nájsť niekoľko výhod aj nevýhod. Ako nevýhodu môžme uviesť vlastnosti samotného kvantového kanálu, kvantový kanál má obmedzenú rýchlosť generovania kľúčov. Na druhú stranu samotná technológia QKD nám ponúka bezpečnú výmenu kľúčov na základe kvantovej fyziky. Kvantový kanál môže byť vytvorený len medzi dvomi uzlami v spôsobe point-to-point a do určitej vzdialenosti kvôli absorpcii a rozptylu polarizovaných fotónov. Vzdialenosť záleží od kvality optického vlákna, optického zdroja, detektora a QKD protokolu, ktorý je použitý. Uvažujme že QKD systém vytvára útlm 20 dB a útlm vlákna je 0.2 dB/km tak potom maximálna dĺžka kvantového kanálu je 100 km. Možná vzdialenosť pre kvantový kanál je to 100 km, kde rýchlosť tvorby kľúčov sa pohybuje od desiatok po stovky kbps. V prípade navýšenia dosahu kvantovej linky je zaujímavá technológia twin field (dvojité pole), táto technológia ponúka až dvojnásobný dosah no doposiaľ ešte nebola použitá komerčne [39, 40, 9, 41].



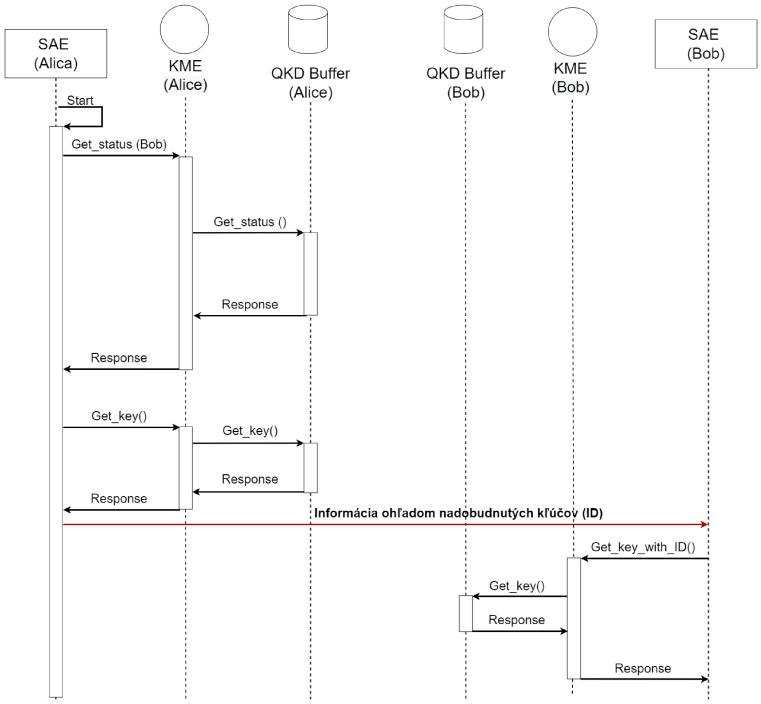
Obr. 2.5: QKD spoj

Znázornenie QKD spoju si môžme všimnúť na 2.5. Kvantový kanál avšak nemusí tvoriť len optické vlákno ale aj voľný priestor alebo laserový kanál vedený pod hladinou vody. V [42] autori vytvorili QKD platformu, ktorá bola založená na BB84 protokole. Dosiahli rýchlosť tvorenie kľúčov 563.54 kbps a QBER bol menší než 1%. Spoj cez voľný priestor sa môže zdať ako najviac vhodné

riešenie, však spoj takéhoto typu má niekoľko nevýhod a to že potrebuje priamu svetelnú dráhu, dobré atmosférické podmienky a prijateľný pomer signálu k šumu (SNR). Tieto všetky nevýhody výrazne obmedzujú možnú dobu používania takéhoto spoju. Avšak výsledky dosiahnuté v experimentoch [43, 44] sú sľubné a poukazujú na schopnosť aplikovať QKD v rámci satelitných spojov [45].

2.7.2 Štandard ETSI 014

Na Obr. 2.5 si môžme všimnúť že medzi šifrátormi/dešifrátormi a QKD uzlami prebieha komunikácia pomocou štandardu ETSI 014. Štandard ETSI 014 [46] definuje REST-API (Representational State Transfer-Application Programming Interface) pre komunikáciu medzi QKD uzlom ktorý obsahuje KME (Key Management Entity) a SAE (Secure Application Entity). SAE si môžme predstaviť ako službu ktorú obsahuje šifrátor/dešifrátor a slúži pre komunikáciu s QKD uzlom. REST-API používa HTTPS protokol, kľúče sú prenášane v JSON formáte spolu s identifikátorom kľúčov (ID). Identifikátor kľúča slúži ako ukazovateľ do úložiska kľúčov v QKD uzle. Takýto identifikátor si šifrátori/dešifrátori medzi sebou vymenia. Týmto krokom sa dosiahne to že pre šifrovanie a dešifrovanie bude použitý kľúč s rovnakým ID.



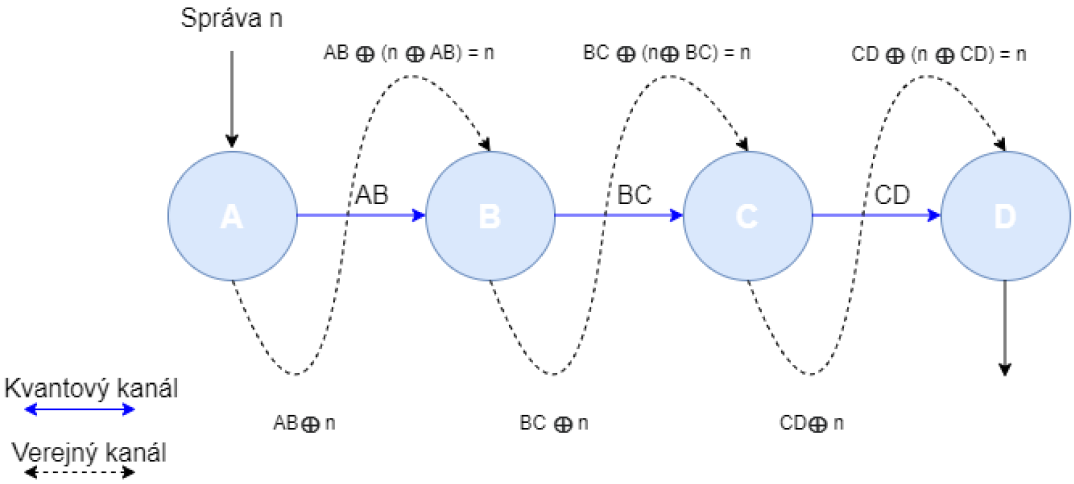
Obr. 2.6: Príklad komunikácie pri ETSI 014

Majme SAE Alica a SAE Bob ako na Obr. 2.6. SAE Alice sa najskôr spýta svojho KME pomocou funkcie *Get_Status(Bob)* na status QKD medzi pamäte (buffer) kde sú uložené kľúče. Funkcia vracia informácie o priradenej KME entite, veľkosti kľúča, ktorý je možné doručiť do SAE, počte

kľúčov uložených v bufferi a o maximálnom počte kľúčov, ktoré je možné doručiť v jednej odpovedi. Následne Alica požiada o blok kľúčov (alebo len o jeden kľúč) pomocou funkcie *Get_Key()*, ako parameter tejto funkcie je počet, veľkosť kľúčov a SAE ID. SAE ID slúži ako identifikátor SAE. Akonáhle Alica obdrží blok kľúčov aj s príslušným ID, toto ID prepošle Bobovi. Bob následne požiada svoje KME o blok kľúčov s príslušným ID od Alice pomocou funkcie *Get_Key_with_ID()*. Z teórie QKD vieme že pre daný QKD spoj, obe komunikujúce strany majú rovnaký QKD buffer, teda obsahujú rovnaké kľúče. Teda vo výsledku Alica a Bob majú rovnaké kľúče pre šifrovanie a dešifrovanie.

2.7.3 QKD sieť

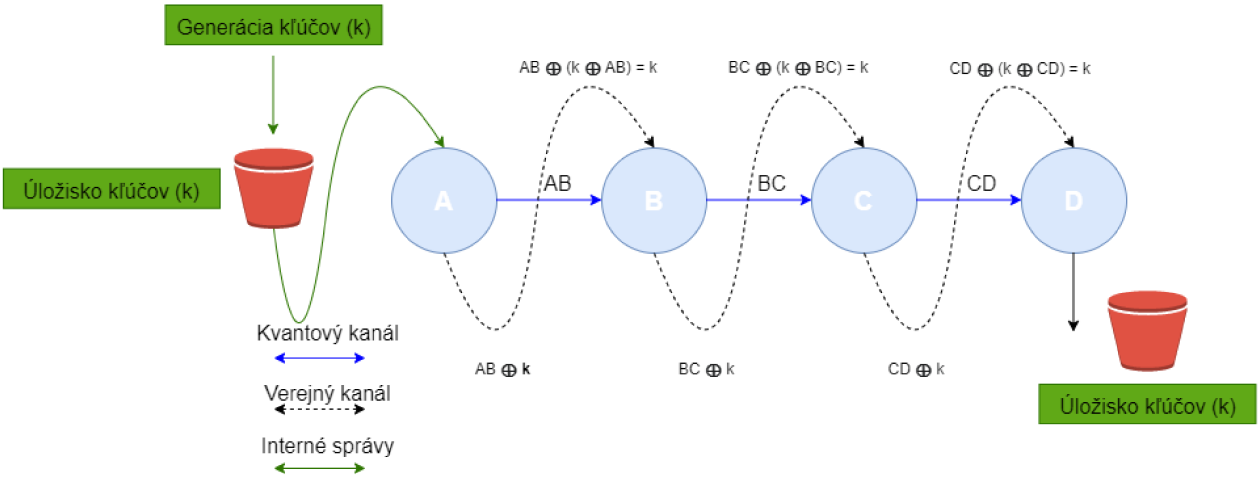
QKD sieť je tvorená statickými uzlami, ktoré reprezentujú bezpečný prístupový bod v danej sieti. Majme správu n ktorú chceme poslať z určitého uzla v QKD sieti na iný QKD uzol v sieti. V tomto prípade môžme využiť dva spôsoby ako preniesť správu n skrz danú sieť. Jedným spôsobom je prístup *Trusted-relay*.
Na obrázku 2.7 si môžme všimnúť príklad komunikácie v tomto režime. Správu n chceme poslať z uzla A na uzol D. Dva susedné uzly zdieľajú rovnaké kľúče. To znamená že uzol B má medzipamäť (alebo buffer) pre kľúče, ktoré zdieľa s uzlom A a ďalšiu medzipamäť pre kľúče, ktoré sú zdieľané s uzlom C. Správa n je teda najskôr zašifrovaná kľúčom AB, následne je táto šifrovaná správa poslaná na uzol B kde je dešifrovaná kľúčom AB. Uzol B následne zašifruje správu pomocou kľúča BC a následne je správa poslaná na uzol C. Takýmto spôsobom správa dorazí až na uzol D, kde sa na výstupe objaví pôvodná správa n. Nevýhoda tohto prístupu je vyššia spotreba kľúčov.



Obr. 2.7: Komunikácia v režime Trusted-Relay

Druhý spôsob je metóda *Key-Relay*. Obr. 2.8 znázorňuje priebeh tejto metódy. Namiesto toho aby sa správa n šifrovala a dešifrovala na každom uzle v rámci cesty z počiatočného uzlu na konečný

uzol, tak sa spôsobom *Trusted-Relay* prenesie počiatočný kľúč k. Teda uzol A zašifruje správu n pomocou kľúča k, následne sa zašifrovaná správa pošle cez verejný kanál priamo uzlu D. Vo výsledku je správa zašifrovaná len na uzle A a dešifrovaná na uzle D.



Obr. 2.8: Komunikácia v režime Key-Relay

2.7.4 Hierarchia QKD siete

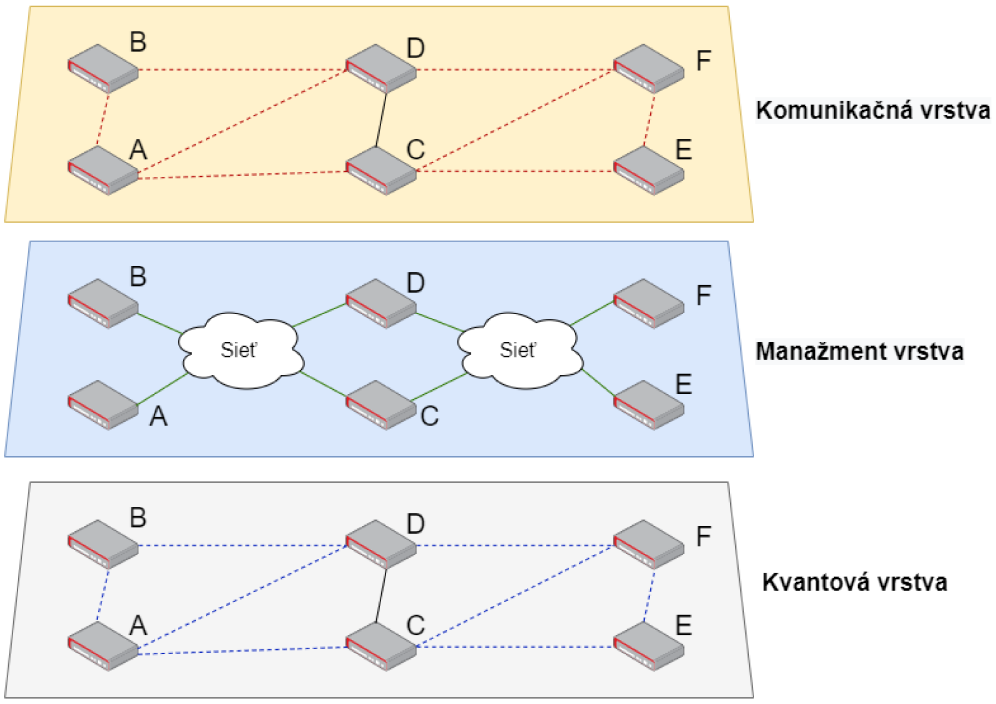
Hierarchiu QKD siete môžme rozdeliť na tri vrstvy.

- Kvantová vrstva, na tejto vrstve prebieha výmena kľúčov. Na tejto vrstve môžme nájsť QKD protokoly ako napríklad BB84 alebo B92.

- Manažmentová vrstva, táto vrstva slúži na overenie a správu vymenených kľúčov. Na tejto vrstve môžme nájsť REST-API podľa štandardu ETSI-014.

- Komunikačná vrstva, tu prebieha výmena šifrovaných dát, dáta sú šifrované pomocou dosiahnutých kľúčov. Typickými zastáncami tejto vrstvy sú smerovacie protokoly ako napríklad OSPF alebo DSDV.

Táto práca sa sústredí najmä na komunikačnú vrstvu, pretože táto vrstva ovplyvňuje smerovanie v QKD sieti. Na Obr. 2.9 je graficky zobrazená hierarchia QKD siete.



Obr. 2.9: QKD sieťová hierarchia

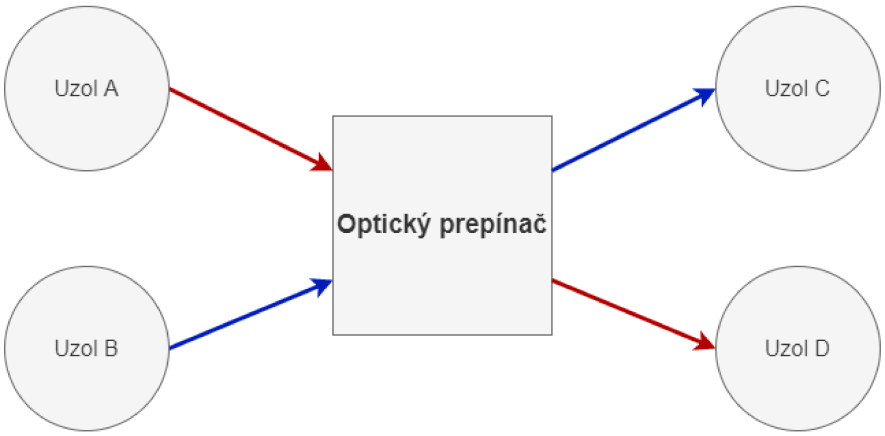
2.7.5 Druhy QKD sieti

Druhy QKD sieti môžme rozdeliť do dvoch kategórii:

- Prepínané QKD siete
- Trusted node QKD siete

2.7.5.1 Prepínané QKD siete

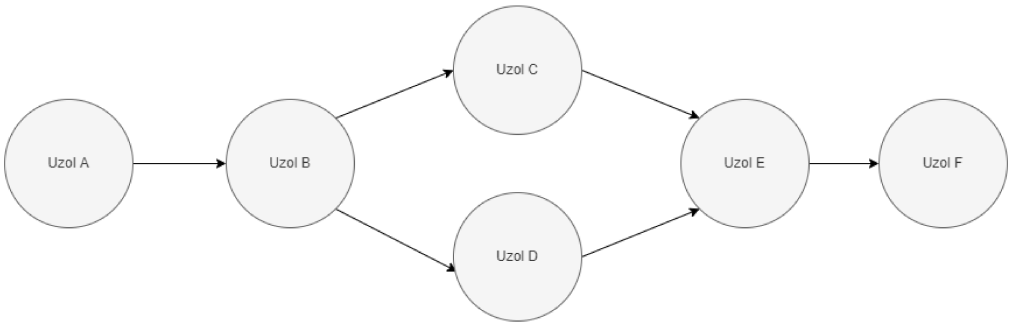
Switched QKD siete alebo aj prepínané QKD siete, takáto sieť sa skladá z uzlov, ktoré sú pripojené len k plne optickej sieti. Základným prvkom takejto siete je opticky switch ktorého funkciou je vytvárať point-to-point prepojenia medzi dvomi uzlami QKD siete. Takéto riešenie môže byť použité len na metropolitné územia pretože takýto optický prepínač pridá niekoľko dB útlmu v rámci optickej trasy medzi dvoma uzlami. Taktiež pre vybudovanie takejto infraštruktúry je potrebné vytvoriť dedikovanú optickú sieť, ktorá bude vyhradená len pre kvantové kanály čo nie je vždy ekonomicky výhodné. Prvá prepínaná QKD sieť bola DARPA QKD [13, 47].



Obr. 2.10: Prepínaná QKD sieť

2.7.5.2 Trusted node QKD siete

Trusted node QKD sieť alebo QKD sieť tvorená dôveryhodnými uzlami, takáto sieť je komponovaná iba QKD uzlami, kde všetky uzly sú chránené a dôveryhodné. Point-to-point komunikácia medzi dvoma uzlami ponúka identické kľúče pre uzly, čo zabezpečuje bezpečnú komunikáciu. Kvantové smerovače neexistujú, to znamená že celé smerovanie je vykonávané na daných uzloch.

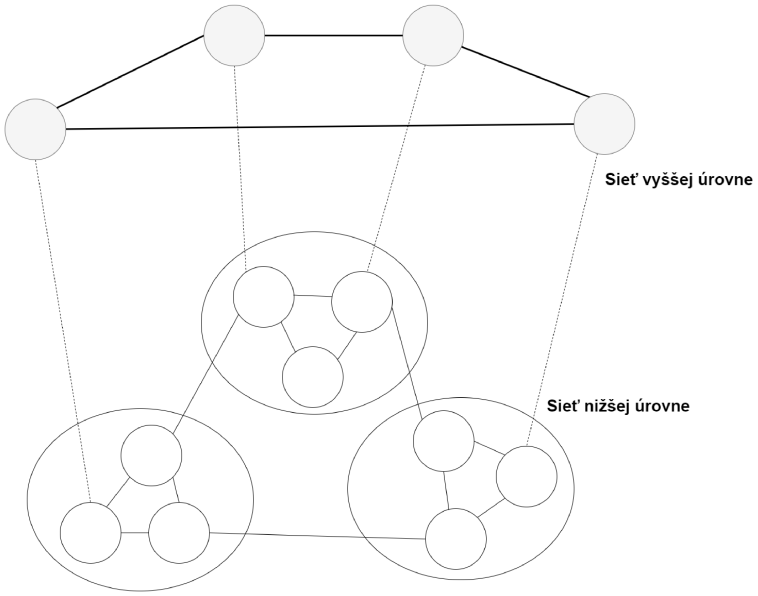


Obr. 2.11: Trusted node QKD sieť

2.7.6 QKD prekrývaná sieť

Zatiaľ čo predchádzajúce druhy QKD sietí sa odkazovali na organizáciu kvantových kanálov tak QKD prekrývaná sieť sa odkazuje na realizáciu verejného kanálu. Hlavným cieľom takejto siete je dosiahnutie lepšej QoS a lepšie využitie prostriedkov siete nižšej úrovne. Takáto sieť sa snaží byť nezávislá od siete nižšej úrovne, napríklad tým že nevyužíva smerovanie takejto siete ale vytvára si vlastnú smerovaciu politiku. Hľadanie alternatívnych ciest medzi dvoma uzlami a prípadne presmerovanie alebo využívanie rozličných ciest súčasne predstavuje kľúčové vlastnosti takéhoto prístupu. Využitie viaccestnej komunikácie je obvykle navrhované riešenie pre využitie čo najväčšieho poten-

ciálu priepustnosti siete, ponúka ochranu pred nedostupnosťou cesty pri možnom zlyhaním jednej liniek a tak ďalej [48, 49, 10].



Obr. 2.12: QKD prekrývaná sieť

2.7.7 QKD sieťové atribúty

QKD predstavuje novú generáciu bezpečnosti v rámci konvenčne používaných sietí, QKD sa nespolieha na matematické problémy ale na zákony kvantovej fyziky. Aj tak je potrebné aby QKD sieť spĺňala určité atribúty a kritéria pre implementáciu do dnešných sieti.

- **Rýchlosť generovania kľúčov**
  Rýchlosť generovania kľúčov je kľúčový atribút pre QKD sieť. Vzhľadom na to že operácie ako šifrovanie a dešifrovanie dát nemôžu byť vykonané bez dostatočného množstva kľúčov v úložisku kľúčov. Teda pomer medzi tým ako rýchlo sú tieto úložiská napĺňanie kľúčmi a ako rýchlo sú tieto kľúče spotrebovávané pre šifrovanie a dešifrovanie, zohráva dôležitú úlohu pre výkonnosť celej siete.

- **Dĺžka spoju**
  Tento atribút bol vysvetlený v kapitole 2.7.1. QKD spoj je rozdelený na verejný a kvantový kanál. Kvantový kanál ma obmedzenú dĺžku čo sa vzťahuje na dĺžku QKD spoju.

- **Ochrana kľúčov**
  Hlavny význam QKD je výmena kľúčov pre následne šifrovanie, ale taktiež je dôležité aby boli tieto kľúče bezpečne uložené v úložisku a musí byť znemožnený prístup do takéhoto úložiska pre tretie strany. Bezpečnosť kľúčov teda nezávisí len pri ich výmene ako bolo popísane

pri protokole BB84 ale aj pri ich skladovaní, manažmente a následnom použití. Je dôležité zabezpečiť bezpečnosť na všetkých leveloch QKD sieťovej architektúry.

- **Využitie kľúčov**
  Vzhľadom na obmedzený zdroj kľúčov ktorý QKD spoj ponúka. Je potrebné aby komunikácia v sieti bola znížená na minimum nakoľko každý jeden vyslaný paket znamená spotrebu kľúčov. Preto je potrebné aby smerovanie v takej sieti bolo čo najkratšie, nakoľko komunikácia je vykonávaná vo forme hop-by-hop a všetky uzly na tejto ceste musia byť dôveryhodné. Dlhšie cesty znamenajú vyššiu spotrebu kľúčov, teda zredukovanie počtu hopov znamená zníženie spotreby kľúčov [50].

- **Robustnosť**
  Vzhľadom na náklady a spôsob implementácie sa očakáva, že sieť QKD bude postupne implementovaná do dnes používaných sieti. Preto je dôležité zabezpečiť robustnosť siete QKD, ktorá sa prejavuje postupným a bezproblémovým pridávaním nových uzlov a zriadenie nových prepojení v sieti QKD. Sieť QKD musí poskytnúť adekvátne náhradné cesty, ktoré zabránia uzlom, ktoré sú chybné alebo sú vystavené vážnym útokom.

Kapitola 3

NS-3 a jeho využitie pri sieťových simuláciách

NS-3 je sieťový simulátor, ktorý dokáže metódy využívané v reálnych sieťových systémoch, rozdeliť na radu logicky separovaných procesov ktoré sú simulované v čase. Tento simulátor je určený pre výskum a edukačné účely. NS-3 je licencovaný pod *GNU GPLv2* licenciou a voľne dostupný. NS-3 podporuje simulácie ako pre IP založené siete tak aj pre nie IP založené siete. Užívatelia NS-3 môžu taktiež simulovať Wi-Fi, LTE a 5G siete. V tomto simulátore môže nájsť rôzne smerovacie protokoly sieťovej vrstvy ISO/OSI [51] modelu ako OSPF, DSDV, AODV, taktiež protokoly transportnej vrstvy ako TCP a UDP. Celý simulátor je vytvorený pomocou programovacieho jazyka C++, s voliteľnými väzbami, ktoré sú avšak napísané v programovacom jazyku Python. Taktiež je možné do simulácii vložiť aj fyzické zariadenia, na ktoré môžme posielať dáta zo simulácie, prípadne NS-3 môže poslúžiť ako framework ktorý môžme implementovať napríklad medzi dve virtuálne zariadenia [52].

3.1 QKD Network Simulation Module

QKD Network Simulation Module (QKDNetSim), je simulátor pre simulovanie QKD sieti, základ tohto simulátora tvorí NS-3 simulátor. Jadro tohto simulátora tvorí NS-3. QKDNetSim tvorí rozširovací modul. Tento simulátor sa snaží o čo najväčšie zjednodušenie QKD spoju z pohľadu simulácie a zameranie sa na meranie výkonu siete, smerovacích protokolov, správu komunikácie, generáciu kľúčov ale aj ich spotrebu.

3.1.1 Návrh

Kvôli vlastnostiam QKD spoju QKD siete sú obmedzené na metropolitné územia. V takýchto územiach môže byť sieť rozdelená na niekoľko autonómnych systémov. V takomto scenári, má každý

autonómny systém vlastnú smerovaciu politiku a vlastnú adresáciu. Musíme taktiež zobrať v úvahu, že v takáto sieť nebude použitá len pre komunikáciu v rámci QKD ale aj pre komunikáciu pre iné protokoly a služby, to môže vyústiť v nepredvídateľný pokles výkonnosti siete. Pre realizáciu zadania tejto práce sme sa rozhodli využiť prekrývaný model QKD siete, ktorý bol opísaný v predchádzajúcej kapitole. Týmto krokom budú mať QKD uzly nezávislú adresáciu od siete nižšej úrovne a smerovanie nebude závisle od autonómnych systémov. V tejto kapitole budú opísané elementárne triedy a ich funkcie spojené so simuláciou QKD sietí v tomto simulátore.

3.1.1.1 QKD Key

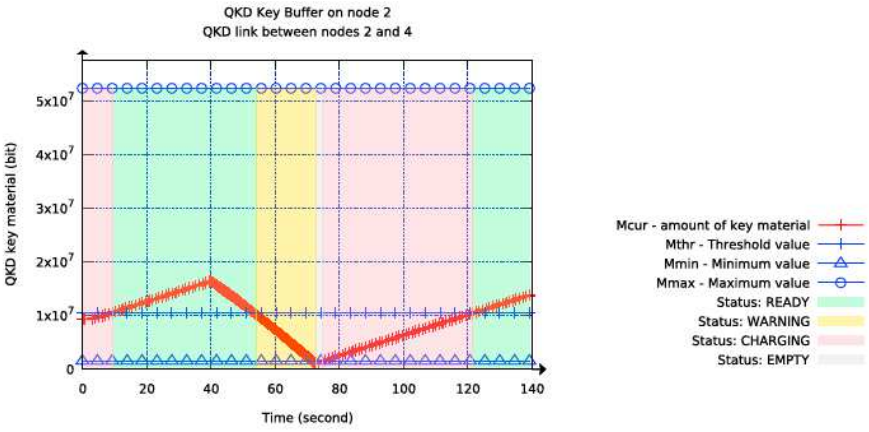
QKD Key je trieda ktorá slúži pre popis kľúča, ktorý je zostavený pri QKD procese. Kľúč je charakterizovaný podľa niekoľkých parametrov, najdôležitejšie z nich sú:

- identifikátor kľúča (ID)
- veľkosť kľúča
- hodnota kľúča ktorá môže byť vyjadrená ako *std::string* alebo *byte*
- časová značka kedy bol kľúč vytvorený

3.1.1.2 QKD Buffer

Kľúče použité pre následné symetrické šifrovanie sú uložené v medzipamäti, v prípade simulátora sa trieda pre ukladanie týchto kľúčov nazýva Buffer. Táto medzi pamäť je charakterizovaná podľa nasledujúcich parametrov:

- Hodnota M_{min} označuje minimálne množstvo kľúčov v medzipamäti, ktoré môže byť použité len na vyjednanie nových kľúčov.
- Hodnota M_{max} označuje maximálne množstvo kľúčov, ktoré dokážu byť uložené v medzipamäti.
- $M_{cur}(t)$, reprezentuje aktuálny obsah medzipamäte v čase merania t, musí platiť $M_{cur}(t) \leq M_{max}$
- M_{thr}, je prahová hodnota medzipamäte, ktorá nám indikuje stav medzipamäte, no musí platiť $M_{thr}(t) \leq M_{max}$. Tento parameter môže byť statický alebo dynamický.



Obr. 3.1: QKD medzipamäť (buffer) [52]

Na obrázku 3.1 si môžme všimnúť grafický výstup medzipamäte. Medzipamäť môže nadobudnúť jeden zo štyroch stavov:

- READY - keď $M_{cur}(t) \geq M_{thr}$,
- WARNING - keď $M_{thr} > M_{cur}(t) > M_{min}$ a predchádzajúci stav bol READY,
- CHARGING - keď $M_{thr} > M_{cur}(t)$ a predchádzajúci stav bol EMPTY,
- EMPTY - keď $M_{min} \leq M_{cur}(t)$ a predchádzajúci stav bol WARNING alebo CHARGING.

3.1.1.3 QKD Crypto

Táto trieda je využívaná k zašifrovaniu, dešifrovaniu, autentifikácii a opätovným zostavením predtým fragmentovaných paketov. QKD crypto používa kryptografické algoritmy a schémy z *Crypto++* čo je *C++* kryptografická knižnica. Medzi podporovanými šiframi môžme nájsť AES-128 a OTP.

3.1.1.4 QKD Virtual Network Device

V prekrývanej sieti, QKD Virtual Network Device si môžme predstaviť ako virtuálnu sieťovú kartu, ktorá spravuje operácie smerovacieho protokolu, šifrovanie, a autentifikáciu prichádzajúcich rámcov. Využitie tejto triedy je exaktne popísané v podkapitole 3.1.1.8.

3.1.1.5 QKD Manager

QKD manažér je inštalovaný na každý QKD uzol a predstavuje základný prvok pre fungovanie celého QKDNetSim simulátora. Obsahuje znalosť o všetkých IP adresách daného uzla či už pre sieť vyššej alebo nižšej vrstvy. Prepája triedu QKD buffer a QKD crypto, nakoľko pre vykonávanie šifrovania QKD crypto trieda vyžaduje kľúčový materiál z medzipamäte kľúčového materiálu pre

daný spoj. Taktiež slúži ako prostredník pri komunikácii medzi sieťou vyššej a nižšej úrovne v prekrývanej sieti.

3.1.1.6 QKD Post-processing Application

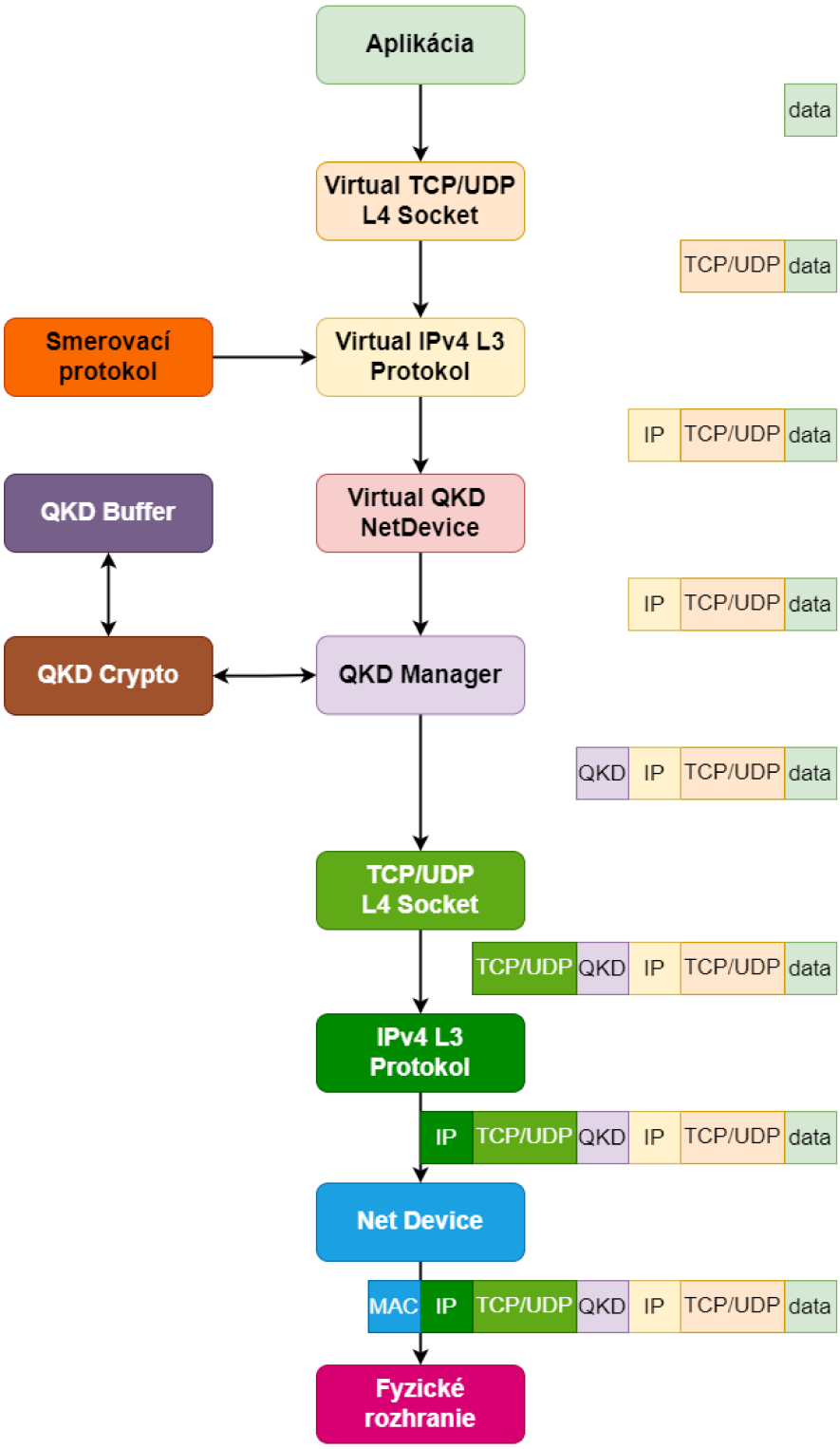
Vyjednanie kľúčového materiálu je neodmysliteľná súčasť QKD sieti. Ako už bolo spomenuté v predchádzajúcich kapitolách, tak výmena samotných kľúčov je vykonaná skrz kvantový kanál, avšak operácie ako overenie vymeneného kľúčového materiálu alebo autentifikácia sú vykonané skrz verejný kanál. Operácie vykonávané skrz verejný kanál sú nazývané tiež ako *post-processing* operácie. V simulátore sú tieto operácie vykonávané špecifickými aplikáciami ako *QKDAppChargingHelper* spojenie je nadviazané medzi dvoma susednými uzlami pomocou protokolu TCP. Táto aplikácia sa stará aj o naplnenie medzipamäte novým kľúčovým materiálom.

3.1.1.7 QKD Helper

Táto trieda je pomocná trieda pre vytvorenie QKD spojenia. V konštruktore tejto triedy môžme nájsť hodnoty pre QKD buffer ako minimálna hodnota, maximálna hodnota a prahová hodnota. Umožní inštaláciu objektu triedy *QKD Manager* a aj inštaláciu samotného spojenia a nastavenie jednotlivých parametrov spoju.

3.1.1.8 Prekrývaný TCP/IP model

QKDNetSim povoľuje realizáciu prekrývanej siete pričom toto riešenie môže byť použité pre rôzne účely aj mimo QKD sieti. Každý QKD uzol obsahuje takýto prekrývaný TCP/IP model, ktorý umožňuje nezávisle smerovanie na oboch vrstvách. Na Obr. 3.2, je zobrazený priebeh zapuzdrenia dát. Aplikácia odosiela dáta na *Virtual TCP/UDP L4 Socket*, ktorý k dátam pridá TCP/UDP hlavičku spojenia na úrovni siete vyššej vrstvy. Dáta v kombinácii s TCP/UDP hlavičkou sú poslané na *Virtual IPv4 L3 Protokol*, ktorý komunikuje so smerovacím protokolom v tomto kroku sa pridá IP hlavička. Dáta putujú na *Virtual QKD NetDevice*, kde by mala byť pridaná MAC hlavička avšak tá pridaná nie je a dáta putujú na *QKD Manager*. V tomto kroku sa pripojí QKD hlavička, ktorá obsahuje autentifikačný tag a informácie o použitej šifre. Ďalšie kroky zapuzdrenia dát vykonávajú objekty siete nižšej vrstvy. Kroky sú obdobné, samozrejme trieda *QKD Manager* je vynechaná, a trieda *NetDevice* pridá MAC hlavičku k zvyšným hlavičkám. Takýto rámec je následne odoslaný po sieti príjemcovi. Príjemca takéhoto rámcu spätnými krokmi tak, ako odosielateľ odstráni jednotlivé hlavičky aby mohol spracovať prijaté dáta.



Obr. 3.2: Zapuzdrenie dát v QKD prekrývanej sieti

Kapitola 4

Návrh viaccestnej komunikácie a jej implementácia v QKDNetSim

Táto kapitola sa bude venovať úprave vybraného smerovacieho protokolu pre podporu viaccestnej komunikácie, jeho implementácii v simulátore QKDNetSim, úprave správania smerovacieho protokolu pre zohľadnenie vlastností QKD spojov pri zostavení cesty zo zdroja do cieľa.

4.1 QKD a MANET siete

Mobile Ad Hoc Network (MANET) [53], predstavuje sieť ktorú tvoria mobilné uzly, ktoré sú medzi sebou bezdrôtovo prepojené pričom tieto uzly nie sú centralizovane ovládane a zvyčajne tvoria len dočasnú sieť. QKD siete a MANET siete majú niekoľko spoločných vlastností ktoré si následne vymenujeme:

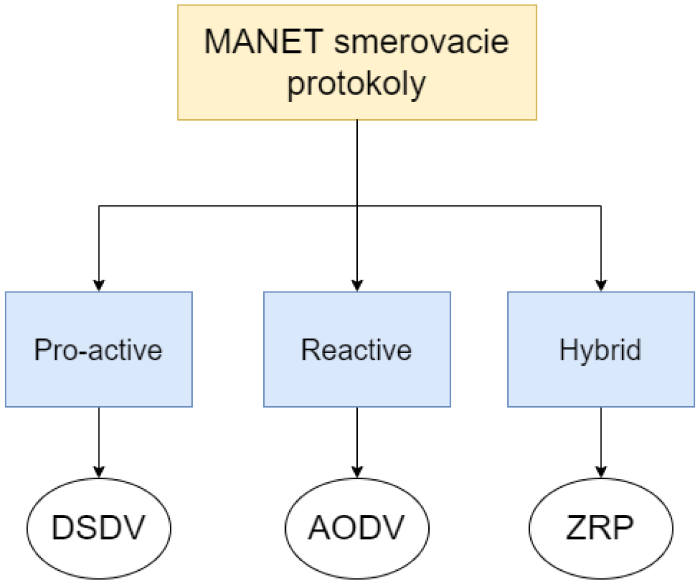
- QKD kvantový kanál, ako už bolo spomenuté v prechádzajúcich kapitolách má obmedzenú dĺžku, toto môžme prirovnať k bezdrôtovému prepojeniu medzi MANET uzlami, teda pokiaľ by boli takéto uzly prepojené Wi-Fi technológiou, prepojenie by sa prerušilo po presiahnutí maximálnej vzdialenosti.

- Smerovanie či už v MANET sieti alebo QKD sieti, je realizované pomocou samotných uzlov, nie je prítomný žiadny smerovač.

- Pokiaľ je medzipamäť prázdna pre určitý spoj tak spoj sa stane nedostupným, čo môžme prirovnať k uzlu v MANET sieti, ktorý je napájaný z batérie, uzol sa stane nedostupným vrátane priľahlých liniek v momente keď sa batéria vybije.

4.2 Smerovacie protokoly v MANET sieťach

V následujúcej podkapitole si popíšeme smerovacie protokoly v MANET sieťach a ich charakteristické rozdelenie.



Obr. 4.1: Rozdelenie smerovacích protokolov pre MANET siete.

Na obrázku 4.1 si môžme všimnúť rozdelenie smerovacích protokolov pre MANET siete.

- Pro-active: Pro-active alebo proaktívne smerovacie protokoly. Tieto smerovacie protokoly pracujú na báze priebežnej aktualizácie smerovacích tabuliek. Teda v prípade že máme implementovaný tento protokol do našej siete, jednotlivé uzly aktualizujú svoje smerovacie tabuľky v pravidelných intervaloch a každý uzol pozná cestu do ľubovoľného bodu v sieti. Typickým zastáncom takejto rodiny smerovacích protokolov je Destination-Sequenced Distance-Vector (DSDV) protokol [54]. DSDV je založený na Bellman-Ford algoritme. Každý uzol pri tomto smerovacom protokole periodicky vysiela svoju smerovaciu tabuľku svojim susedom, takýto paket sa nazýva update paket. Takýto update paket sa vysiela každých 15 sekúnd. Akonáhle susedný uzol príjme takýto paket, aktualizuje svoju smerovaciu tabuľku, zvýši hodnotu skokov o jeden v update pakete a prepošle paket ďalej po sieti. Toto pokračuje dovtedy kým každý uzol v sieti neobdrží tento update paket. Okrem periodických aktualizácii tabuliek, DSDV podporuje aj vysielanie takýchto update paketov v prípade zmeny v sieti, napríklad nedostupnosť nejakej linky [53].

- Reactive: Reactive alebo reaktívne smerovacie protokoly nepracujú ako proaktívne, teda na pravidelnej aktualizácii smerovacích tabuliek. Cesty k cieľu sú vyhľadávané len v momente kedy sú potrebné, a to keď zdrojový uzol chce komunikovať s cieľovým uzlom. Typickým

zástupcom takéhoto protokolu je Ad hoc on-demand distance vector (AODV) protokol [55]. Tento protokol bude podrobne popísaný v následujúcich podkapitolách.

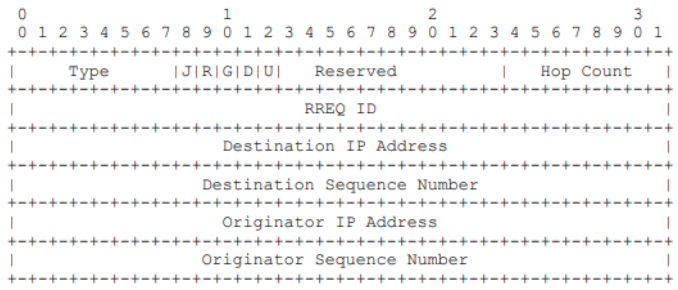
- Hybrid: Kombinuje správanie reaktívnych a proaktívnych smerovacích protokolov. Proaktívne správanie využíva pri udržiavaní záznamov o svojom lokálnom susedstve (zóna). Reaktívne správanie je využité pri komunikácii medzi susedstvami (zónami). ZRP protokol je typickým zástancom takejto hybridnej schémy [56]. ZRP využíva takzvané zóny. Každý uzol v sieti má svoju zónu a veľkosť zóny je vyjadrená ako počet hopov od uzla na okraj zóny. Čiže pokiaľ by bola veľkosť zóny jedna, tak zóna siaha len po susedné uzly [57].

4.3 AODV

Ad hoc on-demand distance vector (AODV) protokol [55], ako už bolo spomenuté je Reactive smerovací protokol. Umožňuje uzlom rýchlo získavať cesty pre destinácie a nevyžaduje aby uzly udržiavali cesty do cieľov ktoré nie sú v aktívnej komunikácii. Uzly vo svojich smerovacích tabuľkách udržiavajú len cesty ktoré sú v aktívnej komunikácii. Na základe tejto vlastnosti je AODV protokol vhodnou voľbou ako smerovací protokol pre QKD siete. V prípade implementácie napríklad DSDV protokolu do QKD siete, by sa podstatná časť kľúčov pre jednotlivé spoje spotrebovala pri periodickej aktualizácii smerovacích tabuliek, ktorá prebieha pri tomto protokole.

4.3.1 Route Request (RREQ) správa

Formát RREQ správy je znázornený na obrázku 4.2. Uzol začne šíriť RREQ správu, keď potrebuje cestu k cieľu avšak žiadna cesta k cieľu nie je dostupná v jeho smerovacej tabuľke alebo je cesta k cieľu označená ako *INVALID*. Pole *Destination Sequence Number*, je posledné známe sekvenčné číslo pre cieľ. Toto číslo pokiaľ je známe je skopírované zo smerovacej tabuľky, pokiaľ nie je známe tak vlajka *U* je nastavená. *Originator Sequence Number* pole tvorí sekvenčné číslo uzlu, ktorý chce komunikovať s cieľovým uzlom, toto číslo je zvýšené o jeden pred vložením do tohto poľa. *Hop Count* je nastavené na hodnotu 0. Cieľová adresa sa doplní do poľa *Destination IP Address* a adresa uzlu ktorý vytvára RREQ správu sa doplní do poľa *Originator IP Address*. Hodnota poľa *RREQ ID* sa zvýši o jeden od posledného RREQ ID používaného aktuálnym uzlom. Každý uzol udržiava iba jednu RREQ ID. Zdrojový uzol začne rozosielať RREQ správu a bude čakať na RREP správu.



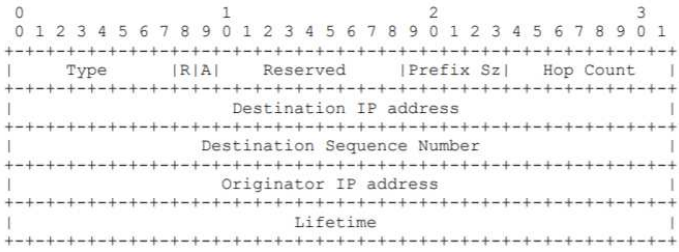
Obr. 4.2: Formát RREQ správy [55].

4.3.2 Route Reply (RREP) správa

Uzol šíri správu RREP v prípade:

- Pokiaľ pole obsahuje *Destination IP Address* RREQ správy, adresa uzla.

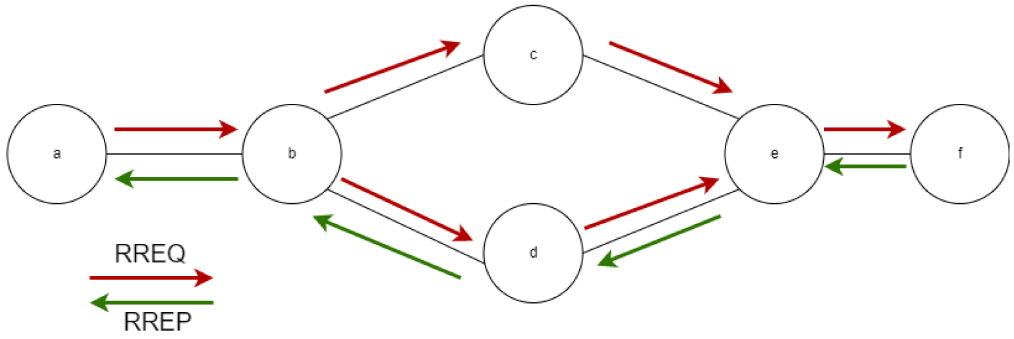
- Pokiaľ má uzol vo svojej smerovacej tabuľke aktívnu cestu do cieľa. Sekvenčné číslo cieľa v smerovacej tabuľke je väčšie alebo rovné číslu v poli *Destination Sequence Number* RREQ správy a vlajka *D* nie je nastavená (táto vlajka znamená že len cieľový uzol môže odpovedať RREP správou na RREQ správu) .



Obr. 4.3: Formát RREQ správy [55].

Pri generovaní správy RREP uzol skopíruje *Destination IP Address* a *Originator Sequence Number* do zodpovedajúcich polí v RREP správe (Obr. 4.3). Následne je RREP správa poslaná na najbližší uzol k tvorcovi RREQ správy na základe smerovacej tabuľky.

4.3.3 Príklad komunikácie pri AODV protokole



Obr. 4.4: Komunikácia AODV protokol

Na Obr. 4.4, si môžme všimnúť príklad komunikácie pre AODV protokol, kde uzol *a* chce komunikovať s uzlom *f*. Smerovacie tabuľky uzlov neobsahujú žiadne záznamy. Uzol *a* vygeneruje RREQ správu, ktorú rozpošle cez všetky svoje aktívne rozhrania. Uzol *b* príjme RREQ správu, vytvorí nový záznam do svojej smerovacej tabuľky, tento záznam bude tvoriť cesta k uzlu *a*. Teda uzol *b* bude mať v tejto chvíli len jeden záznam vo svojej smerovacej tabuľke a to záznam o ceste k uzlu *a*. Uzol *b* navýši pole *Hop Count* o jedna a pošle RREQ správu na uzly *c* a *d*, správa je poslaná na cez všetky aktívne rozhrania okrem toho, ktorý prijal RREQ správu. Obidva tieto uzly obdržia RREQ správu, vykonajú totožné kroky ako uzol *b*, a vo svojich smerovacích tabuľkách budú mať záznam o ceste k uzlu *a* Ďalší skok pre obe cesty v týchto uzloch bude uzol *b* s počtom skokov dva. Uzol *e* príjme RREQ správu od uzlu *d*, vytvorí nový záznam pre cestu k uzlu *a*. Tento uzol však tiež príjme RREQ správu od uzla *c*, avšak táto správa je odstránená nakoľko uzol *e* prijal už RREQ správu s rovnakou hodnotou *Originator IP address* a *RREQ ID*. Inými slovami, uzol *e* už má vo svojej pamäti záznam o tom že prijal RREQ správu od uzlu s rovnakým *RREQ ID* a *Originator IP address*. Tieto dve hodnoty zostávajú nemenné počas celej doby šírenia RREQ správy. Uzol *f* obdrží RREQ správu, zistí že je cieľovým uzlom pre komunikáciu. Vytvorí nový záznam do svojej smerovacej tabuľky pre cestu k uzlu *a*. Následne pošle RREP správu k uzlu *a*. Na Obr. 4.4 si môžme všimnúť putovanie RREP správy skrz sieť. Akonáhle ľubovoľný uzol na trase obdrží RREP správu, vytvorí nový záznam do smerovacej tabuľky pre uzol *f*. Akonáhle RREP dorazí k pôvodcovi RREQ správy, tak sa dátová komunikácia môže začať nakoľko všetky uzly po ceste budú poznať cestu k uzlu *a* aj *f*. Výsledná cesta bude vyzerať takto *a-b-d-e-f*.

4.4 Prahová hodnota QKD uzlov

Doteraz spomenuté smerovacie protokoly pre MANET siete pracovali s metrikou vo forme počtu skokov k cieľovej destinácii. Tento prístup ale nie je dostačujúci pre QKD siete. Pretože nezohľadňuje stavy QKD bufferov pre jednotlivé spoje v QKD sieti. To môže viesť k využívaniu spojov, ktoré majú

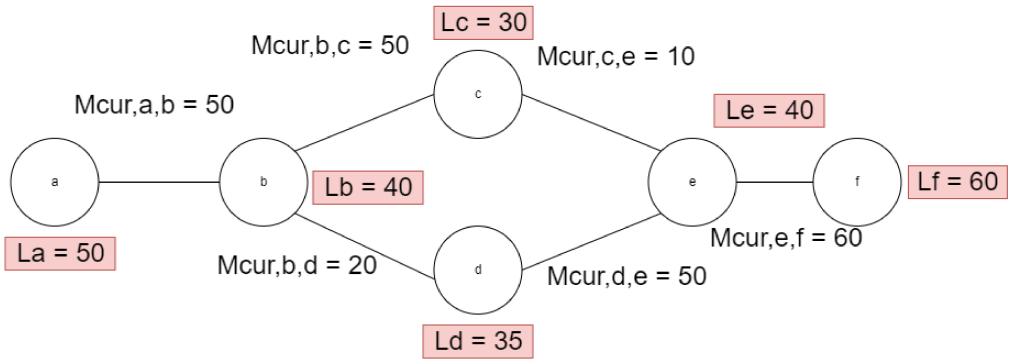
menej kľúčového materiálu než tých ktoré ho majú viac. QKD spoj ktorého buffer je prázdny sa stáva nedostupným, pokým nie je naplnený znova. V článku [11], autori predstavili prístup, ktorý zohľadňuje tento faktor. Tento prístup bol využitý pri zohľadnení kvality cesty v tejto práci pri návrhu viaccestnej komunikácie v QKD sieťach. Popísaný je rovnicou:

$$L_i = \frac{\sum_{k=0}^{N_i} M_{cur,i,k}}{N_i} \tag{4.1}$$

Rovnica 4.2, vyjadruje hodnotu L_i čo je prahová hodnota pre uzol i. Pravá strana rovnice vyjadruje podiel medzi sumou aktuálnej hodnoty množstva kľúčového materiálu všetkých QKD spojov uzla i a počtom spojov QKD (N_i). Majme QKD sieť zobrazenú na Obr.4.5, hodnota L_i je aplikovaná pre všetky uzly v sieti. Napríklad pre uzol b by rovnica 4.2 vyzerala nasledovne:

$$L_b = \frac{\sum_{k=0}^{3} M_{cur,b,k}}{3} = \frac{50 + 50 + 20}{3} = 40 \tag{4.2}$$

Tento prístup bol implementovaný do triedy *QKDManager*. Počas simulácie v QKDNetSim, každý uzol volá objekt triedy *QKDManager* a pomocou funkcie *FetchThresholhValue* tak získavu túto prahovú hodnotu. Spätná hodnota je 32 bitové kladné celé číslo.



Obr. 4.5: Príklad prahových hodnôt pre QKD sieť

4.5 Rozšírenie AODV protokolu pre viaccestnú komunikáciu v QKDNetSim

Pre zavedenie prahovej hodnoty QKD uzlov do AODV protokolu, sme museli upraviť´ samotný formát RREQ a RREP správy. Rozšírenie AODV protokolu pre podporu viaccestnej komunikácie vyžadoval úpravu správania protokolu a spôsob akým je naložené so signalizačnými správami tohto protokolu. Tieto úpravy budú opísané v následujúcich kapitolách.

4.5.1 Rozšírená RREQ správa

Správa RREQ bola rozšírená o dve nové polia. Toto sme dosiahli úpravou triedy *RREQHeader*. Nová RREQ správa v QKDNetSim simulátore:

- *uint8_t flags*
- *uint8_t reserved*
- *uint8_t hopCount*
- *uint32_t requestID*
- *Ipv4Address dst*
- *uint32_t dstSeqNo*
- *Ipv4Address origin*
- *uint32_t originSeqNo*
- *uint32_t L*
- *uint32_t L_summar*

Do RREQ hlavičky boli pridané *uint32_t L* a *uint32_t L_summar*, tieto premenné sú tridsaťdva bitové kladné celé čísla. Funkcie začínajúce so *Set* prepisujú hodnoty *uint32_t L* alebo *uint32_t L_summary* v RREQ správe, funkcie začínajúce so *Get* vracajú hodnoty *uint32_t L* alebo *uint32_t L_summary* v RREQ správe. Pomocné funkcie pre prácu s týmito premennými:

- *void Set_L(uint32_t L)*
- *uint32_t Get_L const()*
- *void Set_L_summary(uint32_t L_summary)*
- *uint32_t Get_L_summary const()*

```
RreqHeader::RreqHeader (uint8_t flags, uint8_t reserved, uint8_t hopCount,
                        uint32_t requestID, Ipv4Address dst, int32_t dstSeqNo,
                        Ipv4Address origin, uint32_t originSeqNo, uint32_t L,
                        uint32_t L_summary)
  : m_flags (flags),
    m_reserved (reserved),
    m_hopCount (hopCount),
    m_requestID (requestID),
```

```
    m_dst (dst),
    m_dstSeqNo (dstSeqNo),
    m_origin (origin),
    m_originSeqNo (originSeqNo),
    m_L (L),
    m_L_summary (L_summary)
```

Listing 4.1: Konštruktor pre triedu RreqHeader

4.5.2 Rozšírená RREP správa

V totožnom smere bola rozšírená aj správa RREP. Nová RREP správa v QKDNetSim simulátore:

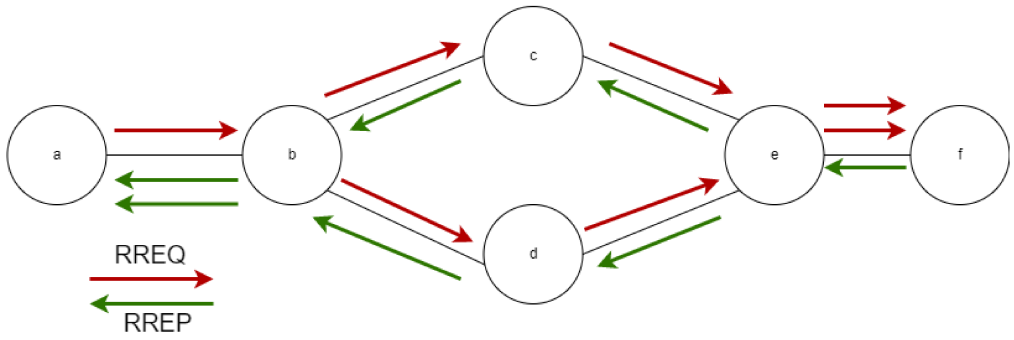
- *uint8_t prefixSize*
- *uint8_t hopCount*
- *Ipv4Address dst*
- *uint32_t dstSeqNo*
- *Ipv4Address origin*
- *Time lifeTime*
- *uint32_t L*
- *uint32_t L_summary*

Pomocné funkcie zostali rovnaké ako aj pri správe RREQ, ktoré sú popísané v podkapitole 4.5.1.

```
RrepHeader::RrepHeader (uint8_t prefixSize, uint8_t hopCount, Ipv4Address dst,
                        uint32_t dstSeqNo, Ipv4Address origin, Time lifeTime,
                        uint32_t L, uint32_t L_summary)
  : m_flags (0),
    m_prefixSize (prefixSize),
    m_hopCount (hopCount),
    m_dst (dst),
    m_dstSeqNo (dstSeqNo),
    m_origin (origin),
    m_L (L),
    m_L_summary (L_summary)
```

Listing 4.2: Konštruktor pre triedu RrepHeader

4.5.3 Príklad komunikácie pri rozšírenom AODV protokole



Obr. 4.6: Komunikácia rozšíreného AODV protokol

Na Obr. 4.6, je zobrazený priebeh komunikácie pri rozšírenom AODV protokole. Priebeh je podobný ako priebeh popísaný v podkapitole 4.4. Nájdeme tu ale niekoľko markantných rozdielov a jeden z nich je že pri generovaní RREQ správy uzol a, vyplní aj polia L a $L\_summary$ pričom pole L je aktuálna prahová hodnota uzlu. Pole $L\_summary$ je pre uzol a taktiež aktuálna prahová hodnota, avšak ako sa RREQ správa šíri sieťou, tak každý uzol zvýši hodnotu poľa $L\_summary$ o svoju aktuálnu prahovú hodnotu. Uzol b obdrží RREQ správu od uzlu a, aktualizuje hodnotu L, hodnotu $L\_summary$ a hodnotu *Hop Count* v správe RREQ. Vytvorí nový záznam do svojej smerovacej tabuľky o ceste k uzlu a, kde môžme nájsť novú hodnotu L_route, ktorá je tvorená aktuálnou prahovou hodnotu uzlu a prahovou hodnotou $L\_summary$ z RREQ správ.

$$L\_route = L + L\_summary \tag{4.3}$$

L_route z rovnice 4.3, tvorí hodnotu $L\_summary$ aktualizovanej správy RREQ, ktorá sa rozpošle cez zvyšné rozhrania k ostatným uzlom. Tento princíp platí pre všetky uzly či už pri RREQ alebo RREP správe. V momente keď uzol e obdrží dve RREQ správy, tak nezahodí ani jednu z nich ale obe ich prepošle na uzol f. Predtým než ich odošle si vytvorí dva nové záznamy do smerovacej tabuľky. Jeden z týchto záznamov bude tvoriť cesta k uzlu a cez uzol e a druhý záznam bude tvoriť cestu cez uzol d. Oba záznamy budú obsahovať aj nové L položky. Uzol f obdrží dve RREQ správy od rovnakého pôvodcu. Avšak narozdiel od uzlu e, vo svojej smerovacej tabuľke bude mať len jeden záznam pre uzol a, nakoľko obe RREQ obdržal na rovnakom rozhraní. Uzol f ako cieľ pre budúcu komunikáciu odpovie RREP správou, nové polia ohľadom prahových hodnôt vyplní obdobne ako uzol a pri RREQ správe. Uzol e nakoľko pozná dve cesty k uzlu a, aktualizuje prahové hodnoty v RREP správe a prepošle RREP správu po oboch cestách. Princíp poli L a $L\_summary$ je totožný ako pri RREQ správe. Uzol b obdrží obe RREP správy, a vytvorí dva nové záznamy do smerovacej tabuľky o cestách k uzlu f obdobne, ako uzol e. RREP správy sú preposlané uzlu a,

ktorý ale vytvorí len jeden záznam do svojej smerovacej tabuľky, pretože obe RREP správy obdržal na rovnakom rozhraní.

4.5.4 Load Balancing

V príklade ktorý sme si opísali v podkapitole 4.5.3, sme vysvetlili rozšírenie AODV protokolu o podporu viaccestnej komunikácie. Zoberme v úvahu uzol b, ktorý má pre cieľový uzol f, vo svojej smerovacej tabuľke dva záznamy. Smerovacia tabuľka uzla b je interpretovaná v Tabuľke 4.1.

Tabuľka 4.1: Smerovacia tabuľka uzla b

Uzol b			
Cieľ	Ďalši skok	L_route	Počet skokov
f	c	1200	3
f	d	1500	3

Zavedenie viaccestnej komunikácie do QKD sietí, zvýši ich samotnú bezpečnosť. Pri používaní iba jednej cesty môže nastať scenár, kedy na jednom alebo viacerých z QKD spojov na danej ceste vznikne nedostatok kľúčov. Nedostatkom šifrovacích kľúčov sa stane QKD spoj nedostupný a tým aj cesta k cieľu. Predstavujeme formu load balancingu, kde základ tvorí generátor náhodných čísel ktorý má rovnomerné rozdelenie. Hustota pravdepodobnosti pre rovnomerné rozdelenie je definovaná rovnicou 4.4.

$$f(x) = \begin{cases} \frac{1}{b-a}, & x \in (a;b) \\ 0, & x \notin (a;b) \end{cases} \tag{4.4}$$

Rovnomerné rozdelenie na intervale $(a;b)$ je také, pri ktorom platí že hustota pravdepodobnosti je konštantná na danom intervale a všade inde je nulová. Inými slovami, je rovnaká pravdepodobnosť výskytu ľubovoľného čísla na intervale $(a;b)$. Pre zohľadnenie počtu skokov k cieľu a prahovej hodnoty ciest sme zaviedli vzťah:

$$Celková\ prahová\ hodnota\ k\ cieľu = \frac{L\_route}{Počet\ skokov} \tag{4.5}$$

Celková prahová hodnota pre uzol b k cieľu f je:

$$\frac{1200}{3} + \frac{1500}{3} = 900 \tag{4.6}$$

Cesta cez uzol c tvorí 44% z celkovej prahovej hodnoty. Cesta cez uzol d tvorí 56% z celkovej prahovej hodnoty. Generátor náhodných čísel je nastavený na generovanie celých čísel na intervale $(1;100)$. Ak generátor vygeneruje číslo na intervale $(1;44)$ tak paket ktorý má byť smerovaný na uzol f bude presmerovaný na uzol c. Ak generátor vygeneruje číslo na intervale $(45;100)$ tak paket, ktorý má byť smerovaný na uzol f bude presmerovaný na uzol d.

Na Listingu 4.3, si môžeme všimnúť pseudokód ktorý vykonáva rozhodovaciu úlohu ktorou cestou sa paket bude posielať po sieti. Smerovacie záznamy aj s percentuálnou hodnotou sú uložené v C++ mape, c++ mapa je triedený asociatívny kontajner, ktorý obsahuje páry, pár tvorí kľúč a unikátna hodnota tohto kľúču. Kľúč musí byť jedinečný v celej mape. Pre nás je kľúč percentuálne rozdelenie z celkovej prahovej hodnoty a hodnota kľúča je *RoutingTableEntry*, čo je objekt triedy pre smerovací záznam.

```
if(mnozstvo ciest k cielu == 1){
      continue;
  }
  else{
      for (i = ukazovatel na prvu moznu cestu k cielu; i != ukazovatel na prvok za
           poslednu moznu cestu k cielu;)
        {
          if(hranicny limit <= nahodne cislo && nahodne cislo <= percentualna
             prahova hodnota pre danu cestu)
            {
              vybrana cesta = ukazovatel na cestu;
              break;
            }
            else {
              hranicny limit = percentualna prahova hodnota pre danu cestu;
              ++i;

            }
          }
  }
```

Listing 4.3: Pseudo kód pre časť C++ kódu vrámci load balancingu

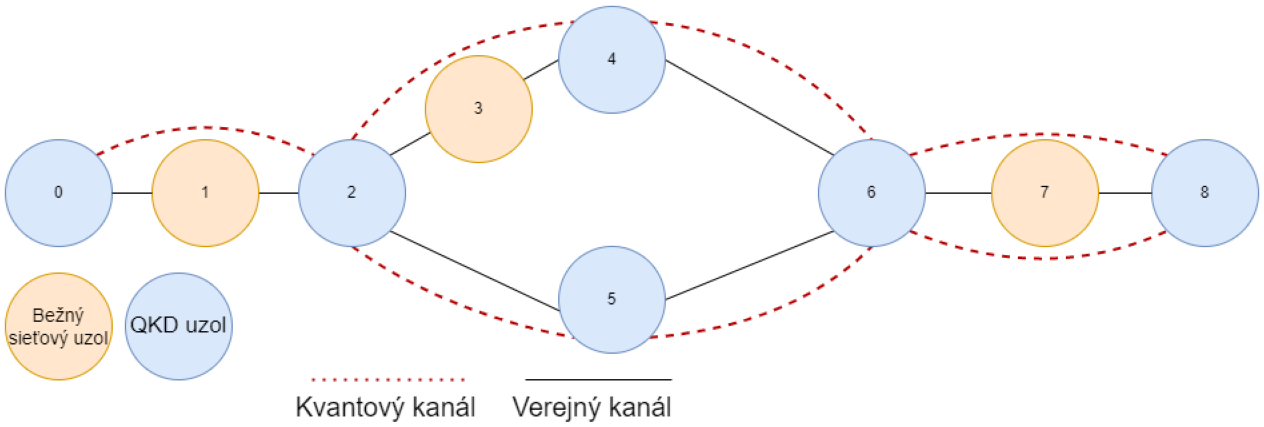
Kapitola 5

Experimentálne overenie viaccestnej komunikácie v simuláciách

V kapitole 4 sme popísali jednotlivé úkony pre úpravu AODV protokolu. Tieto kroky boli implementované do simulátoru QKDNetSim. Simulátor bol nainštalovaný na virtuálne zariadenie, ktoré obsahovalo operačný systém Ubuntu 20.04.03 LTS. Procesor tvoril Intel(R) Xeon(R) Silver 4116 CPU @ 2.10GHz a pridelená operačná pamäť bola 32GB. Výstup simulácii tvoria grafy QKD bufferov pomocou týchto grafov dokážeme určiť spotrebu kľúčového materiálu a logovací výstup zo simulátora, kde nájdeme jednotlivé kroky, ktoré uzly vykonali (naplnenie bufferov, správanie upraveného smerovacieho protokolu či prijaté pakety). V týchto simuláciách bola využitá symetrická šifra OTP kde bol šifrovaný celý paket o veľkosti 1024 bitov, zašifrovanému paketu bola pridaná QKD hlavička a *Internal Next Hop Tag*. Tento tag obsahoval adresu ďalšieho skoku, teda adresu najbližšieho uzlu na danej ceste k cieľu. Každý paket ktorý bol odoslaný ľubovoľným uzlom bol šifrovaný.

5.1 Topológia a parametre simulácie

Nakoľko QKDNetSim nemá žiadne GUI, tak výsledky simulácií sú znázornené v logovacom súbore a takýto výstup je úplne dostatočný pre pozorovanie priebehu simulácie v čase. Správanie QKD bufferov je znázornené v grafoch, kde v čase vidíme spotrebu kľúčového materiálu, y-ová os tvorí zostávajúci kľúčový materiál v bitoch, x-ovú os tvorí čas v sekundách.



Obr. 5.1: Topológia pre simulácie

Na Obr. 5.1, je znázornená topológia použitá vrámci simulácii. Pre simulácie bola použitá prekrývaná sieť. Uzly mali pridelené IP adresy pre sieť nižšej úrovne z rozsahu 10.0.0.0/16. Adresáciu pre sieť vyššej úrovne mali len QKD uzly a to z rozsahu 11.0.0.0/16. Vylepšený AODV protokol pracuje len na úrovni vyššej vrstvy to znamená že uzly siete nižšej vrstvy sú pre neho neznáme.

Tabuľka 5.1: Parametre simulácie

Parametre simulácie	
Čas simulácie [s]	110
Zdroj šírenia dát	QKD uzol 0
Cieľový uzol	QKD uzol 8
Rýchlosť prenosu dát [kb/s]	25
Maximálna hodnota QKD bufferu pre spoj medzi uzlami 2 a 4	585760
Maximálna hodnota QKD bufferu pre všetky ostatné spoje	1085760
Priepustnosť jednotlivých liniek [Gb/s]	1

Simulácie boli spúštané v príkazovom riadku pomocou príkazu: **./waf –run scratch/qkd_overlay_channel_test** .Výsledné grafy boli vykreslené príkazom: **gnuplot *.plt**. Šírenie RREQ a RREP správy prebiehalo rovnako ako bolo popísané v predchádzajúcej kapitole kde sme popisovali rozšírenie AODV protokolu. Čo je ale z nášho pohľadu najviac zaujímavé je to akým spôsobom uzol *2*, ktorý vo svojej smerovacej tabuľke má dva záznamy o ceste k uzlu *8*, narába s paketmi, ktoré majú byť smerované k tomuto uzlu. Smerovacia tabuľka je zobrazená na Obr. 5.2.

```
AODVQ Routing table
Destination Gateway     Interface   Flag    Expire      Hops    L_route
11.0.1.1    11.0.1.1    11.0.1.2    UP  3.00            1       998912
11.0.2.2    11.0.2.2    11.0.2.1    UP  3.00            1       717178
11.0.3.2    11.0.3.2    11.0.3.1    UP  3.00            1       100
11.0.6.2    11.0.3.2    11.0.3.1    UP  4.06            3       1284239
11.0.6.2    11.0.2.2    11.0.2.1    UP  4.07            3       1197848
127.0.0.1   127.0.0.1   127.0.0.1   UP  9223371974.07   1       100
```

Obr. 5.2: Smerovacia tabuľka uzlu *2*

IP adresa 11.0.6.2 predstavuje cieľový uzol, obe tieto cesty majú rozdielny ďalší skok k cieľu aj inú hodnotu L_route. V smerovacej tabuľke si môžme všimnúť aj záznamy o cestách k susedom. Tieto záznamy sú tam kvôli *QKDAppChargingHelper*, táto aplikácia slúži pre správu QKD bufferov, teda vymieňanie kľúčového materiálu a kontrolu stavu. Čas spustenia tejto aplikácie bol v desiatej sekunde, UDP aplikácia pre prenos dát bola spustená v päťdesiatej piatej sekunde. Každý spoj má pridelený svoj QKD buffer, napríklad uzol *2* obsahuje tri takéto *QKDAppChargingHelper* aplikácie pre správu týchto bufferov. V momente keď uzol *2* obdržal paket ktorý mal byť smerovaný na uzol *8*, pomocou load balancingu vybral jednu z týchto dvoch ciest.

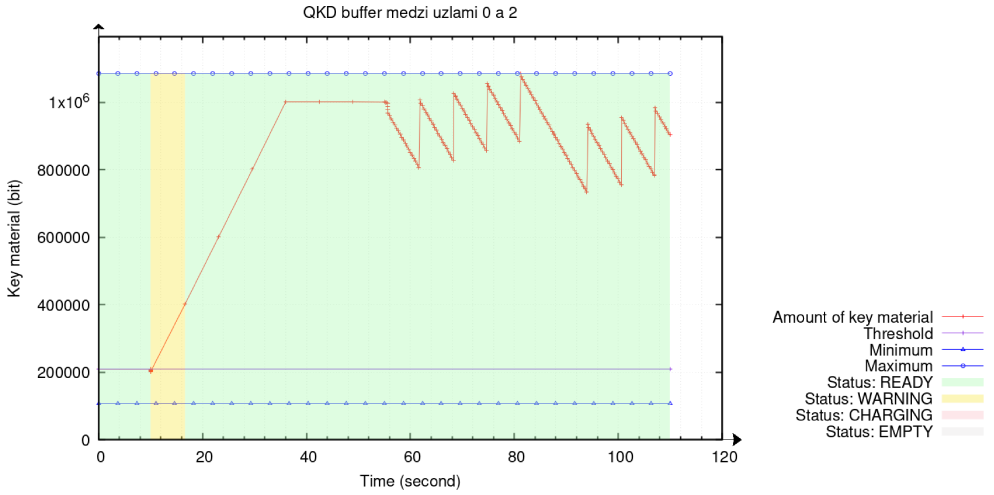
```
[node 2] AodvqRoutingProtocol:Forwarding(): [DEBUG] Random Number: 89
[node 2] AodvqRoutingProtocol:Forwarding(): [DEBUG] Percentage path after round 48
[node 2] AodvqRoutingProtocol:Forwarding(): [DEBUG] Percentage path after round 100
[node 2] AodvqRoutingProtocol:Forwarding(): [LOGIC] 11.0.3.1 forwarding to 11.0.6.2 from 11.0.1.1 packet 736997
```

Obr. 5.3: Load Balancing s náhodným číslom 89 v experimente

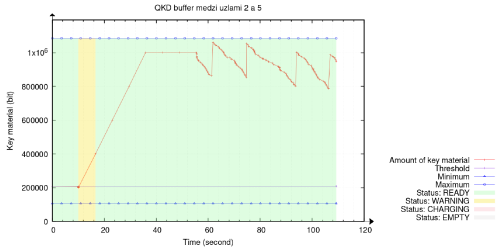
```
[node 2] AodvqRoutingProtocol:Forwarding(): [DEBUG] Random Number: 4
[node 2] AodvqRoutingProtocol:Forwarding(): [DEBUG] Percentage path after round 48
[node 2] AodvqRoutingProtocol:Forwarding(): [DEBUG] Percentage path after round 100
[node 2] AodvqRoutingProtocol:Forwarding(): [LOGIC] 11.0.2.1 forwarding to 11.0.6.2 from 11.0.1.1 packet 742161
```

Obr. 5.4: Load Balancing s náhodným číslom 4 v experimente

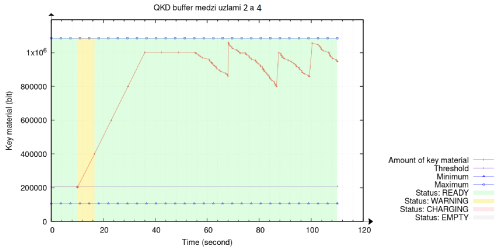
Rozhodovací proces je znázornený na Obr. 5.3 a Obr. 5.4. Interval tvorenia náhodných čísel $1; 100$ je rozdelený na dva intervaly, $1; 48$ pre cestu ktorej predstavuje výstupné rozhranie 11.0.2.1 a $49; 100$ pre cestu ktorej predstavuje výstupné rozhranie 11.0.3.1.



Obr. 5.5: QKD buffer medzi uzlami 0 a 2

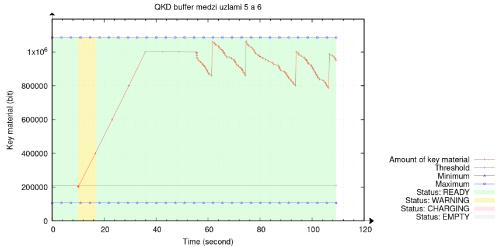


(a) QKD buffer medzi uzlami 2 a 5

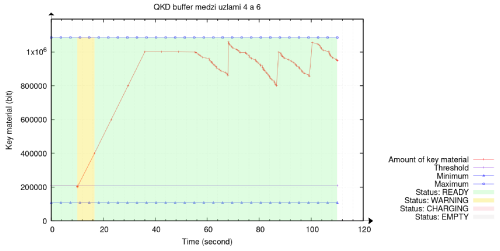


(b) QKD buffer medzi uzlami 2 a 4

Obr. 5.6: QKD buffery medzi uzlami 2 a 5, 2 a 4

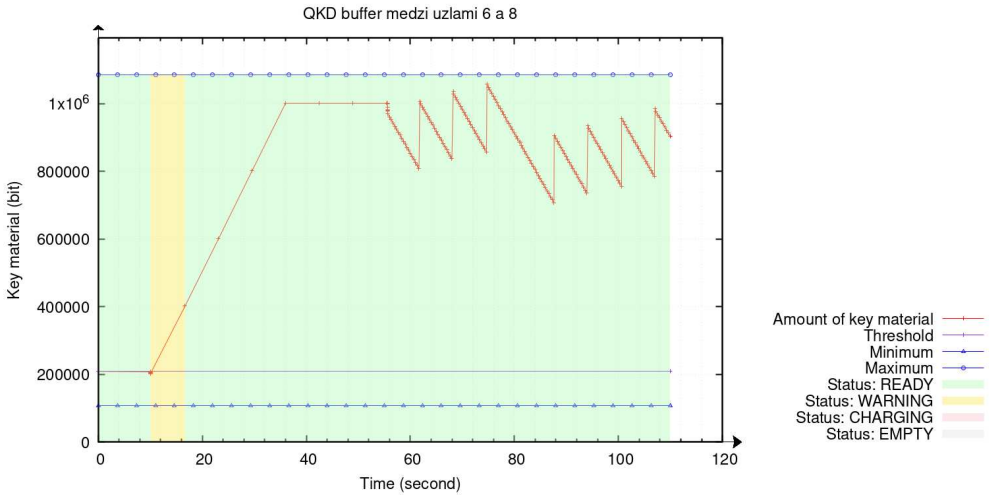


(a) QKD buffer medzi uzlami 5 a 6



(b) QKD buffer medzi uzlami 4 a 6

Obr. 5.7: QKD buffery medzi uzlami 5 a 6, 4 a 6



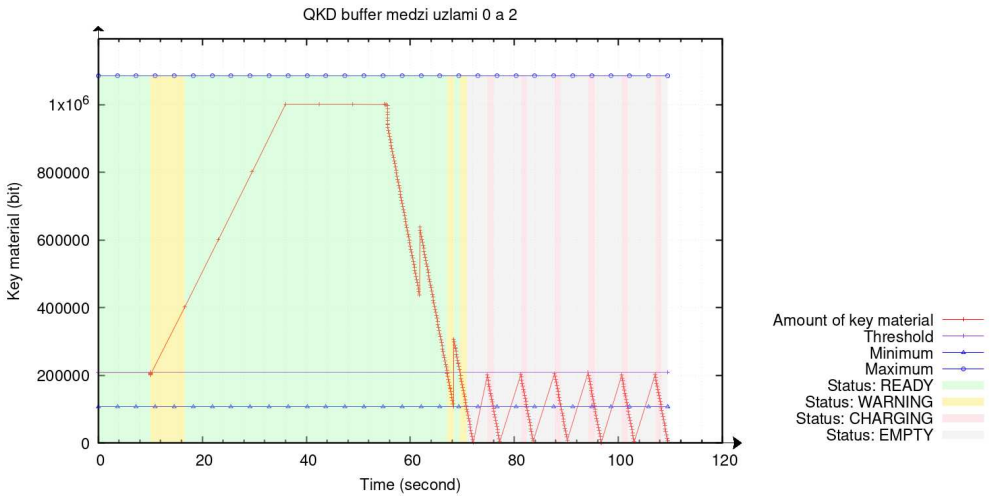
Obr. 5.8: QKD buffer medzi uzlami 6 a 8

Na Obr. 5.5 až Obr. 6.4, sú zobrazené QKD buffery a ich využitie počas simulácie. Napĺňanie bufferov kľúčovým materiálom začalo v desiatej sekunde a prvá spotreba kľúčového materiálu nastala približné v päťdesiatej šiestej sekunde. Pílovitý priebeh nám napovedá že QKD buffery sa počas času kedy danými spojmi neprebiehala žiadna komunikácia stihli naplniť o určitú hodnotu kľúčového materiálu. Napĺňanie bufferov prebiehalo dovtedy kým trieda *QKDCrypto* nevyžadovala materiál pre šifrovanie dát. Pomer odoslaných a prijatých paketov v tejto ukážkovej simulácii bol jedna to znamená všetky odoslané pakety boli aj prijaté v cieľovom uzly.

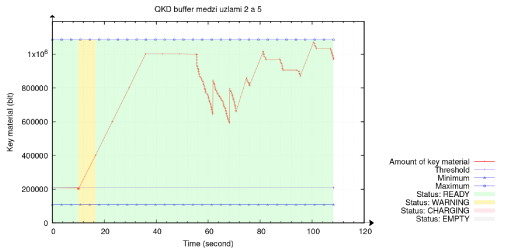
Kapitola 6

Zhrnutie dosiahnutých výsledkov

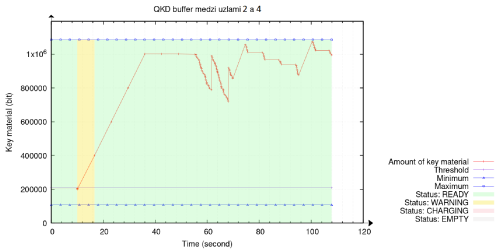
V predchádzajúcej kapitole sme popísali priebeh jednej zo simulácii. Takýchto simulácii prebehlo niekoľko s rôznymi parametrami spojov a použitými šifrovacími algoritmami. V simuláciách keď QKD spoje mali dostatok kľúčového materiálu pre šifrovanie, prenos dat prebehlo úspešne a množstvo odoslaných dat sa rovnalo množstvu prijatých dat. Avšak v simuláciách kde nastal nedostatok kľúčového materiálu pre QKD spoj, vznikol problém pri dosiahnutí nového kľúčového materiálu. Problém pretrváva na kvantovej vrstve QKD siete kde aplikácia *QKDAppChargingHelper*, ktorá síce naplňovala QKD buffery novým kľúčovým materiálom avšak pri dosiahnutí M_{thr} hodnoty sa začal kľúčový materiál znova spotrebovávať. Čo viedlo k prirýchlemu spotrebovaniu nadobudnutého materiálu. Pri vytváraní tejto aplikácie nie je možné zmeniť parametre pre ovplyvnenie danej vlastnosti.



Obr. 6.1: QKD buffer medzi uzlami 0 a 2 pri nedostupnosti jednej z liniek

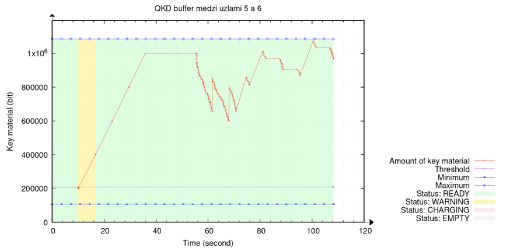


(a) QKD buffer medzi uzlami 2 a 5 pri nedostupnosti jednej z liniek

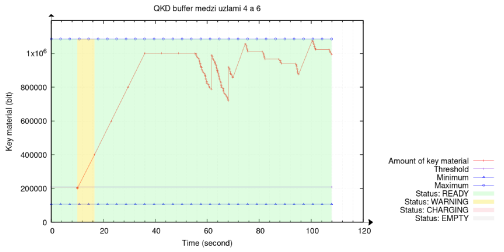


(b) QKD buffer medzi uzlami 2 a 4 pri nedostupnosti jednej z liniek

Obr. 6.2: QKD buffery medzi uzlami 2 a 5, 2 a 4 pri nedostupnosti jednej z liniek

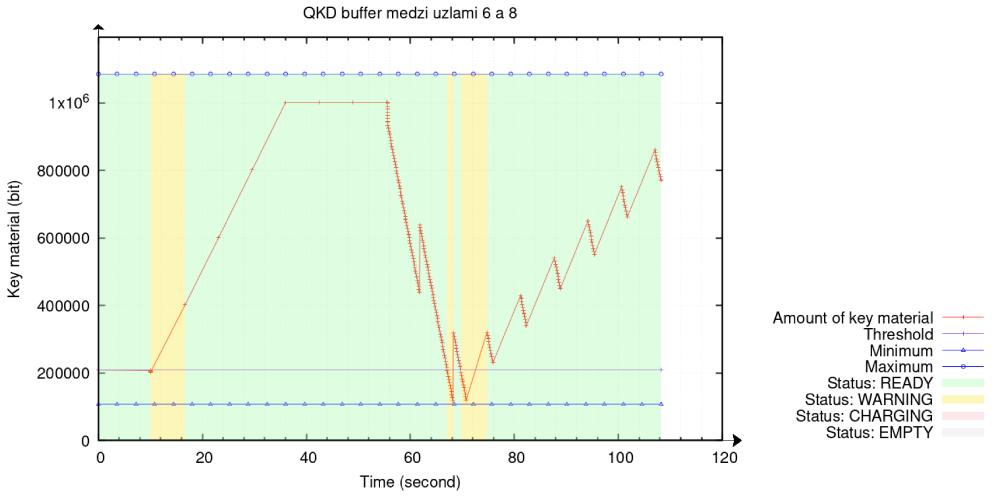


(a) QKD buffer medzi uzlami 5 a 6 pri nedostupnosti jednej z liniek



(b) QKD buffer medzi uzlami 4 a 6 pri nedostupnosti jednej z liniek

Obr. 6.3: QKD buffery medzi uzlami 5 a 6, 4 a 6 pri nedostupnosti jednej z liniek



Obr. 6.4: QKD buffer medzi uzlami 6 a 8 pri nedostupnosti jednej z liniek

Na Obr. 6.1, si môžme všimnúť správanie, kedy QKD buffer dosiahne M_{min} hodnotu. Aplikácia *QKDAppChargingHelper* zareaguje aktuálny stav tohto bufferu a pomocou zvyšného kľúčového ma-

teriálu začne znova nadobúdať nový kľúčový materiál, avšak akonáhle dosiahne QKD buffer M_{thr} materiál sa začne znova spotrebovávať. QKD buffer nedosiahne stavu Ready. Cieľom návrhu tohto smerovacieho protokolu je hlavne vyhnúť sa správaniu kedy je QKD buffer prázdny. Problém nastáva pri bodoch v sieti kedy všetky komunikačné dáta putujú cez jediný spoj. Na ostatných grafoch QKD bufferov si môžme všimnúť postupné spotrebovávanie kľúčového materiálu, v prípade spoju 6 a 8 sa stihne QKD buffer znova naplniť nakoľko prenos dát sa pozastaví na spoji medzi uzlami 0 a 2. Momentálne simulátor funguje na princípe *Trusted-relay* kde každý paket je šifrovaný novým kľúčom. Princíp *Key-relay* by bol vhodnejší, nakoľko by sa šíril len kľúč pre koncové uzly a tento kľúč by bol použitý po celú dobu komunikácie. Prípadne kľúč by nemusel byť využívaný počas celej doby komunikácie ale mal by určitú časovú platnosť, napríklad 20 sekúnd.

Kapitola 7

Záver

Táto diplomová práca sa v teoretickej časti venuje princípu fungovania kvantovej kryptografie a jej porovnaniu s konvenčne používanou kryptografiou. Kvantová kryptografia využíva princípy kvantovej fyziky narozdiel od konvenčne používanej kryptografie ktorá sa spolieha na známe matematické problémy. Práca obsahuje popis simulátoru QKDNetSim ktorý bol vyvíjaný na našej univerzite pričom simulátor je stále vo vývoji. Praktická časť sa venuje implementácii viaccestnej komunikácie v rámci QKD sietí. Na základe podobnosti MANET sietí a QKD sietí bol vybraný reaktívny protokol AODV pre rozšírenie o podporu viaccestnej komunikácie. QKDNetSim v základnej verzii obsahuje implementáciu AODV protokolu, jednotlivé triedy tohto smerovacieho protokolu napísané v C++ boli rozšírené o parametry a chovanie pre podporu viaccestnej komunikácie. Doterajšie smerovacie protokoly využívané pri simuláciách QKD sieti nezohľadňovali jej dôležité atribúty. Preto som zaviedol do smerovacieho protokolu AODV aj zohľadňovanie stavu medzipamäte kľúčového materiálu pre jednotlivé QKD spoje. Ako sa signalizačné správy šíria sieťou zbierajú užitočné informácie o QKD spojoch čo v konečnom dôsledku zvýši stabilitu samotnej QKD siete. Takto nadobudnuté informácie následne využije load balancing pre čo najefektívnejšie využitie dostupných zdrojov kľúčového materiálu. Cieľ takejto implementácie je to aby nenastal stav kedy je QKD spoj nedostupný z dôvodu nedostatku kľúčového materiálu. Táto implementácia umožňuje rovnomerné využívanie všetkých zdrojov v QKD sieti. Objavili sa určité nedostatky na kvantovej úrovni QKD siete avšak cieľom tejto práce bola práca na komunikačnej úrovni a o tomto nedostatku bude autor tohto simulátora informovaný. Všetky body zadania boli v tejto práci splnené. Kompletný zdrojový kód praktickej časti tejto diplomovej práce je dostupný na [58], nebolo možné tento kód pridať ako prílohu kvôli obmedzenej veľkosti 30 MB. Počas štúdia na magisterskom študijnom programe Informačné a komunikačné technológie som sa pridal k vedeckému tímu pod vedením prof. Ing. Miroslava Vozňáka Ph.D. Z tejto spolupráce vzišiel vedecký článok ohľadom post-quantovej kryptografie prezentovaný na konferencii Telfor [59]. V rámci spomínaného tímu sme predložili jeden článok do IEEE COMMUNICATIONS SURVEYS & TUTORIALS a ďalší na konferenciu ICAT 2022 v Sarajeve [60], obidva tieto spomínané články sú v stave under-review [61].

Literatura

1. GROVER, Lov K. A fast quantum mechanical algorithm for database search. In: *Proceedings of the twenty-eighth annual ACM symposium on Theory of computing*. 1996, s. 212–219.
2. SHOR, P.W. Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. *SIAM Review*. 1999, roč. 41, č. 2, s. 303–332. Dostupné z DOI: 10.1137/S0036144598347011.
3. MATTSSON, John Preuß; SMEETS, Ben; THORMARKER, Erik. Quantum-Resistant Cryptography. *arXiv preprint arXiv:2112.00399*. 2021.
4. KUTSCHERA, Florian; DERVISEVIC, Emir; BEHAN, Ladislav; LÓPEZ, Diego; MEHIC, Miralem; VOZNAK, Miroslav; HÜBEL, Hannes; PASTOR, Antonio; CEPEDA, Luis. Data acquisition and simulation tools for virtual QKD testbed access–examples from the OPENQD project. In: *2021 European Conference on Optical Communication (ECOC)*. 2021, s. 1–4.
5. NIEMIEC, Marcin; MEHIC, Miralem; VOZNAK, Miroslav. Risk Assessment Approach to Estimate Security of Cryptographic Keys in Quantum Cryptography. In: *International Conference on Advanced Engineering Theory and Applications*. 2018, s. 114–124.
6. NIEMIEC, Marcin; MEHIC, Miralem; VOZNAK, Miroslav. Security verification of artificial neural networks used to error correction in quantum cryptography. In: *2018 26th Telecommunications Forum (TELFOR)*. 2018, s. 1–4.
7. MEHIC, Miralem; KOMOSNY, Dan; MAUHART, Oliver; VOZNAK, Miroslav; ROZHON, Jan. Impact of packet size variation in overlay quantum key distribution network. In: *2016 XI International Symposium on Telecommunications (BIHTEL)*. 2016, s. 1–6.
8. MEHIC, Miralem; NIEMIEC, Marcin; SILJAK, Harun; VOZNAK, Miroslav. *Error Reconciliation in Quantum Key Distribution Protocols*. 2020.
9. MEHIC, Miralem; MAURHART, Oliver; RASS, Stefan; KOMOSNY, Dan; REZAC, Filip; VOZNAK, Miroslav. Analysis of the public channel of quantum key distribution link. *IEEE Journal of Quantum Electronics*. 2017, roč. 53, č. 5, s. 1–8.

10. MEHIC, Miralem; NIEMIEC, Marcin; RASS, Stefan; MA, Jiajun; PEEV, Momtchil; AGUADO, Alejandro; MARTIN, Vicente; SCHAUER, Stefan; POPPE, Andreas; PACHER, Christoph et al. Quantum key distribution: a networking perspective. *ACM Computing Surveys (CSUR)*. 2020, roč. 53, č. 5, s. 1–41.

11. MEHIC, Miralem; FAZIO, Peppino; RASS, Stefan; MAURHART, Oliver; PEEV, Momtchil; POPPE, Andreas; ROZHON, Jan; NIEMIEC, Marcin; VOZNAK, Miroslav. A novel approach to quality-of-service provisioning in trusted relay quantum key distribution networks. *IEEE-/ACM Transactions on Networking*. 2019, roč. 28, č. 1, s. 168–181.

12. MEHIC, Miralem; MARIC, Almir; VOZNAK, Miroslav. QSIP: A quantum key distribution signaling protocol. In: *International Conference on Multimedia Communications, Services and Security*. 2017, s. 136–147.

13. MEHIC, Miralem. *Using Quantum Key Distribution for Securing Real-Time Applications*. Ostrava, 2017.

14. GISIN, Nicolas; RIBORDY, Grégoire; TITTEL, Wolfgang; ZBINDEN, Hugo. Quantum cryptography. *Reviews of Modern Physics*. 2002-03, roč. 74, č. 1, s. 145–195. ISSN 1539-0756. Dostupné z DOI: 10.1103/revmodphys.74.145.

15. VERNAM, Gilbert S. Cipher printing telegraph systems: For secret wire and radio telegraphic communications. *Journal of the AIEE*. 1926, roč. 45, č. 2, s. 109–115.

16. SIMMONS, G.J. Symmetric and Asymmetric Encryption. *ACM Computing Surveys (CSUR)*. 1979, roč. 11, č. 4, s. 305–330. Dostupné z DOI: 10.1145/356789.356793.

17. SANN, Zarni; SOE, T; KNIN, K; WIN, Z. Performance comparison of asymmetric cryptography (case study-mail message). *APTIKOM Journal on Computer Science and Information Technologies*. 2019, roč. 4, č. 3, s. 105–111.

18. RIVEST, Ronald L; SHAMIR, Adi; ADLEMAN, Leonard M. *A method for obtaining digital signatures and public key cryptosystems*. Routledge, 2019.

19. BENNETT, Charles H.; BRASSARD, Gilles. Quantum cryptography: Public key distribution and coin tossing. *Theoretical Computer Science*. 2014-12, roč. 560, s. 7–11. ISSN 0304-3975. Dostupné z DOI: 10.1016/j.tcs.2014.05.025.

20. BENNETT, Charles H. Quantum cryptography using any two nonorthogonal states. *Physical review letters*. 1992, roč. 68, č. 21, s. 3121.

21. EKERT, Artur K. Quantum Cryptography and Bell's Theorem. In: *Quantum Measurements in Optics*. Springer, 1992, s. 413–418.

22. BECHMANN-PASQUINUCCI, Helle; GISIN, Nicolas. Incoherent and coherent eavesdropping in the six-state protocol of quantum cryptography. *Physical Review A*. 1999, roč. 59, č. 6, s. 4238.

23. SCARANI, Valerio; ACIN, Antonio; RIBORDY, Grégoire; GISIN, Nicolas. Quantum cryptography protocols robust against photon number splitting attacks for weak laser pulse implementations. *Physical review letters*. 2004, roč. 92, č. 5, s. 057901.

24. LO, Hoi-Kwong; ZHAO, Yi. *Quantum Cryptography*. 2008. Dostupné z arXiv: 0803.2507 [quant-ph].

25. DUŠEK, Miloslav; LÜTKENHAUS, Norbert; HENDRYCH, Martin. Quantum cryptography. *Progress in Optics*. 2006, roč. 49, s. 381–454.

26. NURHADI, Ali Ibnun; SYAMBAS, Nana Rachmana. Quantum Key Distribution (QKD) Protocols: A Survey. In: *2018 4th International Conference on Wireless and Telematics (ICWT)*. 2018, s. 1–5. Dostupné z DOI: 10.1109/ICWT.2018.8527822.

27. CHONG, Song-Kong; HWANG, Tzonelih. Quantum key agreement protocol based on BB84. *Optics Communications*. 2010, roč. 283, č. 6, s. 1192–1195. ISSN 0030-4018. Dostupné z DOI: https://doi.org/10.1016/j.optcom.2009.11.007.

28. HUANG, Jinxiang; WANG, Yong; WANG, Huadeng; LI, Zhaohong; HUANG, Jinxiang. Man-in-the-middle attack on BB84 protocol and its defence. In: *2009 2nd IEEE International Conference on Computer Science and Information Technology*. 2009, s. 438–439. Dostupné z DOI: 10.1109/ICCSIT.2009.5234678.

29. SHOR, Peter W.; PRESKILL, John. Simple Proof of Security of the BB84 Quantum Key Distribution Protocol. *Physical Review Letters*. 2000-07, roč. 85, č. 2, s. 441–444. ISSN 1079-7114. Dostupné z DOI: 10.1103/physrevlett.85.441.

30. AGGARWAL, Rahul; SHARMA, Heeren; GUPTA, Deepak. Analysis of various attacks over BB84 quantum key distribution protocol. *International Journal of Computer Applications*. 2011, roč. 20, č. 8, s. 28–31.

31. HAITJEMA, Mart. A survey of the prominent quantum key distribution protocols. *Access Link: https://www. cse. wustl. edu/˜ jain/cse571-07/ftp/quantum*. 2007.

32. BRAUNSTEIN, Samuel L; VAN LOOCK, Peter. Quantum information with continuous variables. *Reviews of modern physics*. 2005, roč. 77, č. 2, s. 513.

33. LI, Jian; LI, Na; ZHANG, Yu; WEN, Shuang; DU, Wei; CHEN, Wei; MA, Wenping. A survey on quantum cryptography. *Chinese Journal of Electronics*. 2018, roč. 27, č. 2, s. 223–228.

34. SILBERHORN, Ch; RALPH, Timothy C; LÜTKENHAUS, Norbert; LEUCHS, Gerd. Continuous variable quantum cryptography: Beating the 3 dB loss limit. *Physical review letters*. 2002, roč. 89, č. 16, s. 167901.

35. GROSSHANS, Frédéric; GRANGIER, Philippe. Continuous variable quantum cryptography using coherent states. *Physical review letters*. 2002, roč. 88, č. 5, s. 057902.

36. JOUGUET, Paul; KUNZ-JACQUES, Sébastien; LEVERRIER, Anthony; GRANGIER, Philippe; DIAMANTI, Eleni. Experimental demonstration of long-distance continuous-variable quantum key distribution. *Nature photonics*. 2013, roč. 7, č. 5, s. 378–381.

37. DJORDJEVIC, Ivan B. Discrete Variable (DV) QKD. In: *Physical-Layer Security and Quantum Key Distribution*. Springer, 2019, s. 267–322.

38. LAI, Jun-sen; LIN, Xiang-yu; QIAN, Yi; LIU, Lu; ZHAO, Wen-yu; ZHANG, Hai-yi. Deployment-oriented integration of DV-QKD and 100G optical transmission system. In: *Asia Communications and Photonics Conference*. 2019, T2H–1.

39. GISIN, Nicolas; RIBORDY, Grégoire; TITTEL, Wolfgang; ZBINDEN, Hugo. Quantum cryptography. *Reviews of modern physics*. 2002, roč. 74, č. 1, s. 145.

40. ALLEAUME, Romain; ROUEFF, Francois; DIAMANTI, Eleni; LÜTKENHAUS, N. Topological optimization of quantum key distribution networks. *New Journal of Physics*. 2009, roč. 11, č. 7, s. 075002.

41. PILE, David FP. Twin-field QKD. *Nature Photonics*. 2018, roč. 12, č. 7, s. 377–377.

42. FENG, Zhao; LI, Shangbin; XU, Zhengyuan. Experimental underwater quantum key distribution. *Optics Express*. 2021, roč. 29, č. 6, s. 8725–8736.

43. BUTTLER, WT; HUGHES, RJ; KWIAT, Paul G; LAMOREAUX, SK; LUTHER, GG; MORGAN, GL; NORDHOLT, JE; PETERSON, CG; SIMMONS, CM. Practical free-space quantum key distribution over 1 km. *Physical Review Letters*. 1998, roč. 81, č. 15, s. 3283.

44. NAUERTH, Sebastian; MOLL, Florian; RAU, Markus; FUCHS, Christian; HORWATH, Joachim; FRICK, Stefan; WEINFURTER, Harald. Air-to-ground quantum communication. *Nature Photonics*. 2013, roč. 7, č. 5, s. 382–386.

45. VALLONE, Giuseppe; BACCO, Davide; DEQUAL, Daniele; GAIARIN, Simone; LUCERI, Vincenza; BIANCO, Giuseppe; VILLORESI, Paolo. Experimental satellite quantum communications. *Physical Review Letters*. 2015, roč. 115, č. 4, s. 040502.

46. ETSI. Quantum Key Distribution (QKD); Protocol and data format of REST-based key delivery API (ETSI GS QKD 014). 2019-02. Dostupné tiež z: `https://www.etsi.org/deliver/etsi_gs/QKD/001_099/014/01.01.01_60/gs_qkd014v010101p.pdf`.

47. ELLIOTT, Chip. The DARPA quantum network. In: *Quantum Communications and cryptography*. CRC Press, 2018, s. 91–110.

48. MA, Zheng; SHAO, Huai-Rong; SHEN, Chia. A new multi-path selection scheme for video streaming on overlay networks. In: *2004 IEEE International Conference on Communications (IEEE Cat. No. 04CH37577)*. 2004, zv. 3, s. 1330–1334.

49. TANG, Chiping; MCKINLEY, Philip K. Improving multipath reliability in topology-aware overlay networks. In: *25th IEEE International Conference on Distributed Computing Systems Workshops*. 2005, s. 82–88.

50. KOLLMITZER, Christian; PIVK, Mario. *Applied quantum cryptography*. Springer, 2010.

51. POPESCU-ZELETIN, Radu. Implementing the ISO-OSI reference model. *ACM SIGCOMM Computer Communication Review*. 1983, roč. 13, č. 4, s. 56–66.

52. *What is ns-3*. [B.r.]. Dostupné tiež z: `https://www.nsnam.org/about/what-is-ns-3/`.

53. ALSLAIM, Mona N.; ALAQEL, Haifaa A.; ZAGHLOUL, Soha S. A comparative study of MANET routing protocols. In: *The Third International Conference on e-Technologies and Networks for Development (ICeND2014)*. 2014, s. 178–182. Dostupné z DOI: `10.1109/ICeND.2014.6991375`.

54. PERKINS, Charles E; BHAGWAT, Pravin. Highly dynamic destination-sequenced distance-vector routing (DSDV) for mobile computers. *ACM SIGCOMM computer communication review*. 1994, roč. 24, č. 4, s. 234–244.

55. QUEUE, RFC. manet-aodv (RFC 3561). [B.r.].

56. BEIJAR, Nicklas. Zone routing protocol (ZRP). *Networking Laboratory, Helsinki University of Technology, Finland*. 2002, roč. 9, č. 1, s. 12.

57. MITTAL, Shaily; KAUR, Prabhjot. Performance Comparison of AODV, DSR and ZRP Routing Protocols in MANET's. In: *2009 international conference on advances in computing, control, and telecommunication technologies*. 2009, s. 165–168.

58. BURDIAK, Patrik. *Praktická časť diplomovej práce* [`https://bitbucket.org/patrik_burdiak1/burdiak-dp-2022-qkd/src/master/`]. 2022.

59. LAUTERBACH, Filip; BURDIAK, Patrik; RICHTER, Filip; VOZNAK, Miroslav. Performance Analysis of Post-Quantum Algorithms. In: *2021 29th Telecommunications Forum (TELFOR)*. 2021, s. 1–4.

60. LAUTERBACH, Filip; BURDIAK, Patrik; ROZHON, Jan; AL., et. Quantum Channel Characteristics from the Point of View of Stability, under review. In: Sarajevo, 2022-06.

61. MEHIC, Miralem; MICHALEK, Libor; DERSEVIC, Emir; BURDIAK, Patrik; AL, et. Quantum Cryptography in 5G networks: A Comprehensive Overview, under review. *IEEE Communications Surveys & Tutorials, IF 25.249 (2020)*. 2022.