Multimed Tools Appl (2016) 75:10691–10707
DOI 10.1007/s11042-014-2299-1

Authentication in virtual private networks based on quantum key distribution methods

Marcin Niemiec • Petr Machnik

Received: 31 January 2014 / Revised: 23 September 2014 / Accepted: 25 September 2014 /
Published online: 14 October 2014

Abstract Quantum physics has a major influence on modern computer science and communications. New quantum-based solutions continue to be proposed by researchers. However, only a few techniques are possible to implement in practice. One of them is quantum key distribution, which ensures the confidentiality of digital data. This article introduces a new concept: quantum distribution of pre-shared keys. This approach provides end-users with very secure authentication, impossible to achieve using currently-available techniques. Secure authentication is a key requirement in virtual private networks (VPN)—popular protection in computer networks. The authors simulated quantum-based distribution of a shared secret in a typical VPN connection. Using a dedicated simulator, all individual steps of the quantum key distribution process were presented. Based on the created secret, a secure IPsec tunnel in a StrongSwan environment was established between AGH (Poland) and VSB (Czech Republic). It allows end-users to communicate at very high security levels.

Keywords Security · Authentication · Quantum key distribution · Virtual private networks · IPsec protocol

1 Introduction

Interest in quantum-based mechanisms for communications is growing rapidly. Currently, solutions such as quantum cryptography (QC) and quantum random generators are more than just concepts, and they are being put into practice [19]. Quantum key distribution (QKD) in particular is a technique whose popularity continues to grow [10].

Today, end-users of network services can buy and use devices which support symmetric encryption with QKD techniques. Devices such as Cerberis created by ID Quantique in Switzerland [13] or QPN-8505 created by MagiQ Technologies [15] in the US are available in the communications market. Before these devices can be widely used in communication

M. Niemiec (✉)
AGH University of Science and Technology, Mickiewicza 30 Ave., 30-059 Krakow, Poland
e-mail: niemiec@kt.agh.edu.pl

P. Machnik
VSB-Technical University of Ostrava, 17. listopadu 15, 708 33 Ostrava, Czech Republic
e-mail: petr.machnik@vsb.cz

networks, technical problems need to be resolved (i.e. quantum repeaters). However, this is not the only solution to implement proprietary QKD in practice. The first services based on QKD are starting to appear in the communications market, since service providers are building quantum channels and deploying their own quantum devices. As a result, QKD is currently being offered to customers as a service.

The development of quantum-based techniques may change our approach to the most popular security services and protocols in the near future, for example the methods used to build virtual private networks (VPN). Currently, the most popular authentication method in VPN is asymmetric cryptography – private keys and certificates with public keys. Unfortunately, this technique requires additional resources to work properly (i.e. deployed Public Key Infrastructure), and it will be possible to crack it easily when quantum computers are created in the future. Therefore, pre-shared keys (also known as secrets) may be the best alternative in the near future, in particular when these secrets can be distributed securely using techniques such as QKD. This article proposes a pre-shared key distributed using quantum methods for authentication, and presents an implementation of this idea in practice.

The article consists of six chapters. Chapter 2 describes selected security services, such as authentication, encryption and quantum key distribution. Chapter 3 introduces VPNs, mainly based on the IPsec protocol. The StrongSwan environment where the practical tests were conducted is also presented. Chapter 4 describes the implementation issues: the key generation process in a QC simulator, and tunnel establishment using StrongSwan. Chapter 5 presents the results of implementation: the working VPN tunnel between the AGH University of Science and Technology (Poland) and the VSB-Technical University of Ostrava (Czech Republic). Additionally, the effect of final key reduction was simulated and presented in detail. The final chapter concludes the article.

2 Security services

Security services are being applied in network environments in order to protect transmitted or stored data [20]. The services presented in this chapter are used to verify the identity of end-users and protect confidential data. Solutions such as quantum key distribution are able to establish secret strings of bits in a network environment at a higher security level.

2.1 Authentication

Authentication is usually the first security step used during communication between network entities. Usually, at the start of the communication process, the end-user must present their identity. After it is verified successfully, the end-user is authorized in the system, and they can use specific services or resources.

The most popular authentication methods available currently are based on logins and passwords. Each end-user has an individual identity in the system, known as a login. The end-user also needs to know a confidential string of bits, known as a password or secret. During the authentication process, the end-user presents their login and secret password. If the presented secret is successfully verified, the end-user gains access to the system. This method is based on the fact that only the user knows their secret password, and therefore they are able to present it during the authentication process.

There are many different types of authentication methods based on secret passwords. During one such process, known as the challenge-response method, one entity presents a question (known as a ‘challenge’) and another entity must provide a valid answer (known as a

'response'). This response is usually based on a secret password, also known as a pre-shared key. In a typical scenario, one user sends to another a randomly-generated string of bits as the challenge, whereupon the receiver must return the result of a cryptographic function. This response is computed using the challenge as the input of the cryptographic function and the secret as the key. An example of such a function is a keyed-Hash Message Authentication Code (HMAC). If both entities share the same secret, they are able to compute the response. At the same time, an intruder is not able to provide a valid authentication, because they do not know the secret and cannot calculate the correct response.

2.2 Encryption

Encryption processes are commonly used to ensure confidentiality in modern communication networks. The encryption transforms the message rendering it unreadable to anyone except certain entities (i.e., the sender and the recipient). Formally, the encryption is a function E defined by:

$$E_K(M) = C$$

where M denotes a message (plaintext), C denotes a ciphered message (ciphertext) and K is the key. In symmetric-key cryptography, the same key is used to encrypt and decrypt confidential data, therefore the key must be secret [8]. In modern ciphers, the key is a long string of bits. The distribution or agreement of these strings is crucial to data confidentiality.

Currently used algorithms (e.g., Diffie–Hellman key agreement protocol [21]) are able to establish a shared secret key over an insecure communications channel, although they are vulnerable to some types of attacks. The best currently-available methods of establishing secret keys in a secure way are quantum key distribution methods.

2.3 Quantum key distribution

Secure distribution or agreement of encryption keys are crucial to data confidentiality. Currently, when we use modern ciphers with popular key distribution methods, we are not sure if an intruder is eavesdropping on the communication. In this way, a hidden intruder can scan the network and obtain sensitive data. Quantum key distribution ensures a very high level of security, because it is not possible to eavesdrop on the communication in a passive way [1]. If an eavesdropper reads the distributed key, this will change the quantum states of the photons and will thus be revealed. This is possible because measurement influences the quantum state [6], and it is not possible to clone an unknown quantum state [24].

Popular quantum key distribution protocols, such as BB84 [3], are based on the polarization of single photons, which carry information coded in quantum states (i.e. different polarizations: vertical, horizontal, diagonal). In this way, the recipient and potential eavesdropper do not know which detector should be used to measure the polarization precisely. It is not a problem for the intended recipient – when they announce the configuration of detector which was used during the measurement of a received photon, the sender confirms that the obtained result is correct or asks for this bit to be deleted from the final key because they obtained result is not certain.

What about eavesdropping on a quantum distributed key? If the eavesdropper chooses an inappropriate detector to perfectly measure the polarization, the polarization of the photon is changed. The sender and recipient uncover the eavesdropper if they compare parts of the obtained key, thus rendering passive eavesdropping impossible. If someone wants to

eavesdrop on photons and read confidential information, they would need to change the quantum states of the photons.

Quantum key distribution ensures a very high level of security [11]. However, it is only part of the complete key establishment process [17]. For example, the sender and recipient must estimate errors in the distributed key by computing the Quantum Bit Error Rate (QBER). The QBER is defined as the ratio of the number of wrong bits to the total number of bits. It is worth emphasizing that not only eavesdropper is responsible for introducing errors; they may occur because of disturbance in the quantum channel, optical misalignment, noise in detectors, and so on.

After the bit error estimation, two users, referred to as Alice and Bob, use key distillation protocols. These protocols usually involve two steps:

- key reconciliation—in this step the sender and recipient must find and correct or delete errors [2, 12],
- privacy amplification—the sender and recipient should improve their privacy and construct the final key by deleting some of the distributed bits [4, 5].
- Bit error estimation and key distillation processes are able to effective manage the security and efficiency in quantum cryptography [16].

3 Virtual private networks

A virtual private network (VPN) is a means to securely and privately transmit data over an unsecured and shared network infrastructure. VPNs secure the transmitted data by encapsulating the data, encrypting the data, or both encapsulating and then encrypting the data. Encapsulation is often referred to as tunneling because data is transmitted from one network to another transparently across a public network infrastructure. Typically, a VPN is a protected connection between two entities (specific devices or particular networks) that are not necessarily directly connected.

A good VPN solution should address all of the following issues:

- Protecting data from eavesdropping by using encryption.
- Protecting packets from tampering by using hash functions to ensure packet integrity.
- Protecting against man-in-the-middle attacks by using identity authentication mechanisms.
- Protecting against replay attacks by using sequence numbers when transmitting protected data.
- Defining the mechanics of how data is encapsulated and protected, and how protected traffic is transmitted between devices.
- Defining what traffic actually needs to be protected.

The most important types of VPN technologies are currently IPsec, SSL, GRE, MPLS VPN, PPTP, and L2TP. We employed the widely-used IPsec technology within the experiment described in this paper.

3.1 IPsec protocol

IPsec (Internet Protocol Security) VPNs encrypt data at the network layer of the OSI reference model, offering secure VPN solution by providing authentication, data confidentiality, anti-

replay protection and data integrity protection. IPsec is one of the most popular VPN technologies used by the commercial sector, service providers, and government networks. IPsec is a standards-based protocol, and therefore it supports interoperability across products from multiple vendors. The IPsec framework is defined in RFC 2401, although the implementation of IPsec is defined in several different RFC recommendations.

IPsec provides the following services [9]:

- Data confidentiality: This is achieved via encryption to protect data from eavesdropping attacks. Supported encryption algorithms include DES, 3DES, and AES.
- Data integrity: This is accomplished via HMAC functions to verify that packets have not been tampered with and are being received from a valid peer, preventing man-in-the-middle attacks or session hijacking attacks. Supported HMAC functions include MD5 and SHA.
- Anti-replay protection: This is achieved by including encrypted sequence numbers in data packets to ensure that a replay attack does not occur from a man-in-the-middle device.
- Peer authentication: This is done to ensure that before data is transmitted between peers, the peers' identities are verified. Device authentication is supported with symmetric pre-shared keys, asymmetric pre-shared keys, and digital certificates. Remote access connections also support user authentication using XAUTH (Extended Authentication).

The two main groups of standards that IPsec uses are [9]:

- ISAKMP (Internet Security Association and Key Management Protocol)/IKE (Internet Key Exchange): These standards are used to set up secure management connections, determine keying information for encryption, and using signatures for authentication of the management connection. This connection is used so the two IPsec peers can share IPsec messages with each other.
- AH (Authentication Header) and ESP (Encapsulating Security Payload): These standards are used to provide protection for user data. They can provide confidentiality (only ESP), data integrity, data origin authentication, and anti-replay services.

IPsec does not support encapsulation of multicast and broadcast IP packets and all other non-IP packets. In this case, the GRE (Generic Routing Encapsulation) protocol is used to encapsulate these packets and to create unicast packets that can be encapsulated by IPsec afterwards.

3.2 StrongSwan software

The StrongSwan software package [23] provides an open-source IPsec VPN solution. StrongSwan is intended primarily for Linux devices. It is fully compatible with other standard IPsec VPN implementations, and thus can be used in networks with mixed equipment.

The main benefits of StrongSwan IPsec VPNs are as follows:

- StrongSwan supports various popular platforms—computers with the Linux, Mac OS X or FreeBSD operating systems, and smartphones with the Android operating system.
- StrongSwan implements both IKEv1 and IKEv2 (Internet Key Exchange) protocols, and it fully supports IPv6.
- StrongSwan enables dynamic IP address and interface updates with the IKEv2 Mobility and Multihoming Protocol, and IKEv2 Multiple Authentication Exchanges.

- It enables automatic insertion and deletion of IPsec policy-based firewall rules.
- StrongSwan supports NAT-Traversal via UDP encapsulation and port floating.
- The XAUTH functionality is based on IKEv1 Main Mode authentication.
- The device authentication is based on X.509 certificates or pre-shared keys.
- StrongSwan enables secure IKEv2 EAP (Extensible Authentication Protocol) user authentication.
- RSA private keys and certificates can be stored on smart cards or USB tokens supporting the PKCS #11 interface.

4 Implementation

In this chapter, the practical implementation of a VPN tunnel between AGH (Poland) and VSB (Ostrava) is described. The first step is the key generation process to ensure the secure pre-shared key at both ends of the tunnel. The key – generated in a QKD simulator – is used during the authentication process. This step differs from such implementations as in SECOQC project [19] or commercial deployments [10] where physical devices were used in the network scenarios. Following successful authentication, an IPsec tunnel is established, and all data transmitted between users is transmitted securely.

4.1 Key generation

To generate the secure pre-shared key, the authors used the QKD Protocol Simulator application written in C++ [18]. The simulator implements all steps of the BB84 protocol, as well as error estimation, key reconciliation, and privacy amplification. The application provides detailed logs to report all performed steps, which makes it possible to analyze the QKD protocol.

At the start of the key generation process, the QKD protocol (BB84—the most popular protocol in practice) and the length of the key (1024 bits) were selected. The key is as follows:

```
————————STARTING KEYS————————
Original key - 1024 bits:
01000001110111001110001011011110100011010001001000
11110001111111110001000000010000111000111010000000100011
00010000100110001110100001110001110101010001011101
10011110001010111001011101
11100000001111000111000001101010010101001011011011110
10010001110000010010010011101000111100000111100100001
10100101100000001000000000000100100111111101001
10100001001101001111111011001001000101110010111010
10110101110011111110011011000000101100110111000
01001010111111111110001101001110100001101000001
01111011110000010101101011100000110110000000000111
00000111101000110100111101101000011010010010111000
```

```
0 1 1 1 1 0 0 1 1 0 0 0 1 0 1 1 0 0 0 0 0 0 1 0 1 1 1 1 0 0 0 1 0 0
1 0 1 0 1 0 0 1 1 0 0 1 0 0 1 1 1 0 1 1 0 1 0 0 0 0 1 1 0 1 1 0 1 1 0 0 0 0 0 0 1 0
1 0 0 1 0 0 1 1 1 1 1 1 1 0 1 1 0 0 1 0 0 1 0 1 0 0 1 1 1 1 1 1 0 0 0
1 1 0 0 1 1 0 1 1 1 1 1 1 1 1 1 0 1 0 0 1 1 1 0 1 0 0 0 0 1 1 0 1 0 0
00110101101011000011010010101011000010011001010011001100010001
010010001100111000010110110100010101000000110010111110101010
1 1 0 0 1 0 1 1 0 0 1 1 0 0 1 0 1 0 1 0 1 0 0 0 1 1 0 1 0 0 0 1 0 1 1 1 0 1 0
0010010111011110100101011101101010001
```

Next, a quantum channel with 1 % noise intensity was modeled, providing some errors in the starting key. The following mask was used in the process (ones indicate the locations where individual bits of the starting key were changed):

```
——————CHANNEL MODEL - NOISE ——————
Mask for introducing noise:
M:
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 1 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0000000010000000000000
```

Once the errors were inserted into the starting key, the BB84 protocol commenced. Alice and Bob performed all the steps of the BB84 protocol to obtain 512-bit raw keys:

```
—————— QUANTUM PROTOCOL ——————
Raw keys 512 bits:
Alice:
1 1 1 1 1 1 1 0 1 1 1 0 1 0 1 0 1 0 0 1 0 1 0 1 0 1 1 0 0 1 0 0 1 1 1 0 0 1 1 0 0 0 1 0 0 0 0 1 0 0 0 1 0 0
1 1 0 0 0 0 0 1 1 1 1 0 1 1 1 1 1 0 1 1 1 0 1 0 0 0 1 0 0 1 0 1 1 0 0 0 0 0 0 1 0
0 1 1 0 1 0 1 1 1 0 1 0 0 0 1 1 1 1 0 0 0 1 1 0 0 0 0 1 0 0 0 0 0 1 1 1 0 0 0 0 0 0 1 0 1 0 1 1 0
```

```
0 1 1 1 0 0 0 0 1 1 1 0 0 1 0 1 0 0 0 0 0 0 1 1 1 1 0 1 0 1 0 0 1 1 1 1 1 1 0 1 0 0 1 0 0 1 0 0 1 0
0 0 1 1 0 1 1 1 0 1 1 1 1 1 1 0 0 1 0 1 0 1 1 1 0 0 0 1 1 0 0 1 0 1 1 0 0 0 0 1 0 0 0 1 1 1 1 0 0
1 0 1 0 1 0 0 0 0 0 0 1 1 1 1 0 1 1 1 0 0 1 0 1 0 0 1 1 0 0 0 0 1 0 1 0 0 0 0 0 1 1 0 0 0 1 1 0 0 1 1
1 1 1 1 0 1 0 1 0 0 0 0 1 0 0 1 1 0 1 0 0 0 0 1 0 0 1 1 1 1 0 1 0 0 1 0 0 1 0 0 0 0 1 1 1 1 1 1 0 0 0 1
0 0 0 0 0 0 1 1 0 0 0 1 1 0 1 0 0 1 0 1 0 0 1 0 1 0 1 1 1 1 0 1 0 1 1 1 0 0 1 1 1 0 0 0 0 0 1 1 1 1 0 1 1
1 1 0 0 0 1 0 1 0 0 1 1 1 1 1 1 1 0 0 0 1 0 1 1 1 0 0 0 1 1 0 0 1 0 1 1 0 0 0 0 1 0 0 0 1 1 0 0 0 0 0 1
0 1 1 1 1 1 1 0 0 1 0 1 0 1 1 1 0 0 0 1 1 0 0 1 0 1 1 0 0 0 0 1 0 0 0 1 1 1 1 0 1 1 1 0 0 1 0 1 1 0 1 1
1 0 1 0 0 0 0 0 1 1 0 0 0 1 1 1 1 1 1 1 0 1 0 1 0 0 0 0 0 1 0 0 1 1 0 1 1 0 0 0 0 0 0 1 1 1 1 0 0 0 0 0
0 0 0 0 0 0 1 0 0 1 1 1 1 0 0 0 0 1 1 1 1 0 1 1 0 0 0 0 1 0 1 1 0 0 1 0 1 1 0 1 0 1 0 1 0 1 0 1 1 0 0 1
0 1 0 1 0 1 1 0 0 1 0 0 0 0 0 1 0 0 0 0 1 1 1 1 1 1 0 0 0 0 1 0 1 1 1 0 0 0 1 0 1 1 1 0 1 0 1 1 1 0 1
0 1 1 0 0 1 1 1 1 1 1 0 1 1 0 1 0 0 1 1 0 1 0 0 0 0 0 0 0 0 1 0 0 1 0 0 1 0 0 0 0 0 0 1 1 0 0 1 0 1 0
0 0 0 0 0 1 1 0 0 1 1 0 1 0 0 0 1 1 0 1 0 0 1 0 1 0 0 1 0 1 0 1 1 0 1 0 1 0 1 1 0 0 1 0 1 0 1 0 1 0 1
1100111100010000111001101011101011011010 01
```

Bob:

```
1111111011101010100101010110010011100110001000011001001100
0 0 0 1 1 1 0 1 1 1 1 0 1 1 1 0 1 0 0 0 1 0 0 1 0 1 1 0 0 0 0 0 0 1 0 0 1 1
0 1 0 1 1 1 0 1 0 0 0 1 1 1 1 0 0 0 1 1 0 0 0 0 1 0 0 0 0 0
1 1 1 0 0 0 0 0 1 0 1 0 1 1 0 0 1 1 1 0 0 0 0 1 1 1 0 0 1 0 1 0 0 0 0 0 0 1
1 1 1 0 1 0 1 0 0 1 1 1 1 1 1 0 1 0 0 1 0 0 1 0 0 1 0 0 0 1 1 0 1 1 1
0 1 1 1 1 1 1 0 0 1 0 1 0 1 1 1 0 0 0 1 1 0 0 1 0 1 1 0 0 0 0 1 0 0 0 1 1
1 1 0 0 1 0 1 0 1 0 0 0 0 0 0 1 1 1 1 0 1 1 1 0 0 1 0 1 0 0 1 1 0 0 0 0
1 0 1 0 0 0 0 0 1 1 0 0 0 1 1 1 1 1 1 0 1 0 1 0 0 0 0 0 1 0 0 1 1 0 1 1
0 0 0 0 0 0 1 0 0 1 1 1 1 0 0 0 0 1 1 1 1 0 1 1 0 0 0 0 1 0 1 1 0 0 1 0 1 1 0 1
0 1 0 1 0 1 1 0 0 1 0 0 0 0 0 1 0 0 0 0 1 1 1 1 1 1 0 0 0 0 1 0 1 1 1 0 0 0 1 0 1 1 1 0 1
0 1 1 0 0 1 1 1 1 1 1 0 1 1 0 1 0 0 1 1 0 1 0 0 0 0 0 0 0 0 1 0 0 1 0 0
0 0 0 0 0 1 1 0 0 1 1 0 1 0 0 0 1 1 0 1 0 0 1 0 1 0 0 1 0 1 0 1 1 0 1 0 1 0 1
1100111100010000111001101011101011011010 01
```

Now, Alice and Bob must check whether the raw key was eavesdropped. They estimate the QBER using the 204 bits (known as the 'sample key'). It should be noted that the real QBER is approx. 0.96 %, but the calculated QBER is larger (approx. 1.96 %). However such a value of QBER is normal in practical quantum channels, therefore Alice and Bob decide that the raw key is secure.

```
———————— QBER ESTIMATION ————————————
Sample keys - 204 bits

Raw keys (without sample) 308 bits:

Alice:
1111111110001100101000100000101100001111101001000100000
1001010100110010100011000101100111011101000000110111001
0010110111111010111000101011000001100101000001011010000
0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 0 1 0 0 0 0 0 0 1 0 1 1 0 0 0 1 0 0 1 0 0 0 1 0 1 0 0 0 1 0 0 1 0
1 1 0 1 0 1 1 0 0 0 0 0 0 0 0 0 1 1 1 1 0 1 1 1 0 0 0 0 1 0 1 1 1 0 0 1 1 1
1110101010000010000011000101010101000111011100100101101110 1
Bob:
1111111110001100101000100000101100001111101001000100000
0100001001010100110010100011000101100111011101000000110111001
0 0 0 1 1 0 1 1 1 0 0 1 0 0 1 0 1 1 0 1 1 1 1 1 0 1 0 1 1 1 0 0 0 1 0 1 0 1 1 0 0 0 0 0 1
1 0 0 1 0 1 0 0 0 0 0 1 0 1 1 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 0 1
0000001011000100100010100110010110101100000000111101
1 1 0 0 0 0 1 0 1 1 1 0 0 1 1 1 1 1 1 0 1 0 1 0 1 0 0 0 0 0 1 0 0 0 0 0 1 1
0001010101010001110111001001101110 1
```

```
General QBER: 0,9765625 %
Sample QBER: 1,96078431372549 %
```

Once the QBER was estimated, the raw key was shortened to 308 bits. Because of errors provided in the channel, the keys were not the same. Therefore, the key reconciliation process was started:

```
———————— RECONCILIATION ————————————
Input QBER: 0,00324675324675325
Round 1
Block length: 63 bits
Round 2
Block length: 126 bits
Round 3
Block length: 252 bits
QBER after round 3: 0 %
KEYS ARE THE SAME (Alice and Bob have the same key).
```

Using the CASCADE algorithm [7], where the parity of each block is checked (in three rounds with different length of blocks), all errors were corrected. After this step, Alice had the same raw key as Bob. Unfortunately, during the previous steps some information about the raw key could be obtained by eavesdropper. Therefore, Alice and Bob perform the final step – the privacy amplification process:

```
————————— PRIVACY AMPLIFICATION ————————————
Security Parameter:9
Amplified keys 292 bits:

 0011100011010101110110101011110110010100011100
1100100111011111000000011100000000101010100100011
0010010111100100000100010110000100000011010010000
1011000111100011010011010010100010000100001011001100101111010
0011001011110000110010100111010111100110110010101110100001001
1001010000100001111101001
```

The final key has a length of 292 bits – significantly shorter than the staring key at the beginning of the process. This string of bits will be used as a secure pre-shared key during the authentication process. Since in the StrongSwan environment the keys are stored in the HEX format, the final key was transformed into the following form:

```
38D5DABD9473277C0E02A91C979045881A4163C69A5108599749978653AF36574265043E9
```

4.2 Establishing the IPsec tunnel

The StrongSwan software was employed to create an IPsec tunnel between two computers at VSB and AGH. The topology of the test scenario is shown in Fig. 1.

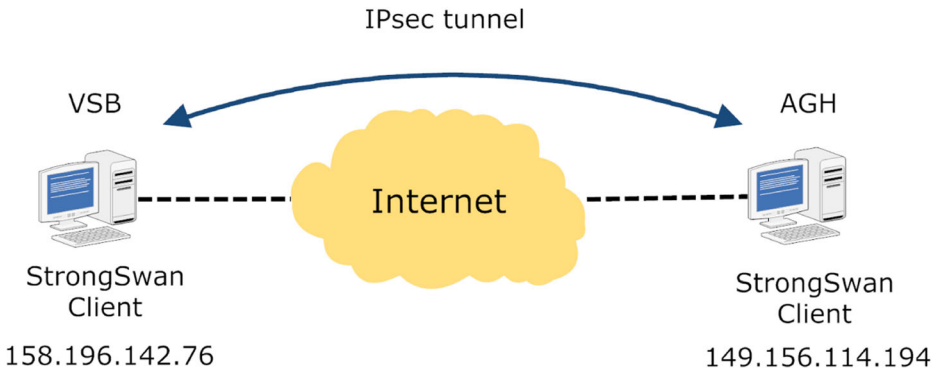

Fig. 1 Test topology

The key generated by the QKD simulator was used as a pre-shared key for the authentication of both IPsec tunnel endpoints. The configuration of the StrongSwan software which was used for establishing the tunnel is as follows:

ipsec.conf file

```
config setup

conn %default
  ikelifetime=60 m
  keylife=20 m
  rekeymargin=3 m
  keyingtries=1
  ike=aes256-sha1-modp1536
  esp=aes256-sha1
  authby=secret
conn VSB-AGH
  left=158.196.142.76
  right=149.156.114.194
  auto=start
```

ipsec.secrets file

```
158.196.142.76 149.156.114.194 : PSK 0x38D5DABD9473277C0 E02A
91C979045881A4163C69A5108599749978653AF36574265043E9
```

The first file *ipsec.conf* contains the configuration of the tunnel, while the second file *ipsec.secrets* contains the pre-shared key used for authentication. The configuration at both ends of the tunnel differs only in the local (left) and remote (right) IP address.

Once StrongSwan is installed on both computers and the configuration files are ready, the environment starts. If the connection process is completed successfully, the VPN tunnel will be ready to transmit secure data.

5 Results

This chapter presents a summary of the obtained results. It covers basic measurements and the status of the VPN tunnel created. Simulation results of the final key during the key establishment process are also presented. The authors take into account the effect of key reduction dependent on the noise intensity in the quantum channel.

5.1 VPN tunnel

After starting the configured StrongSwan environment, the process creating the VPN tunnel begins. During the experiment, the entire process took 4 s, during which a few packets were exchanged between both computers. These packets were captured by the *tcpdump* packet analyzer and are presented below.

```
root@ubuntu:~# tcpdump src 149.156.114.194
16:34:00.696851 IP dhcp194.kt.agh.edu.pl.4500 > pcn312g.vsb.cz.4500: NONESP-encap:
isakmp: child_sa inf2[ I]
16:34:00.717917 IP dhcp194.kt.agh.edu.pl > pcn312g.vsb.cz: ICMP
dhcp194.kt.agh.edu.pl udp port 4500 unreachable, length 116
16:34:01.705904 IP dhcp194.kt.agh.edu.pl > pcn312g.vsb.cz: ICMP echo reply, id
7176, seq 137, length 64
16:34:02.707946 IP dhcp194.kt.agh.edu.pl > pcn312g.vsb.cz: ICMP echo reply, id
7176, seq 138, length 64
16:34:03.708830 IP dhcp194.kt.agh.edu.pl > pcn312g.vsb.cz: ICMP echo reply, id
7176, seq 139, length 64
16:34:03.789067 IP dhcp194.kt.agh.edu.pl.isakmp > pcn312g.vsb.cz.isakmp: isakmp:
parent_sa ikev2_init[ I]
16:34:03.970510 IP dhcp194.kt.agh.edu.pl.4500 > pcn312g.vsb.cz.4500: NONESP-encap:
isakmp: child_sa ikev2_auth[ I]

root@ubuntu:~# tcpdump src 158.196.142.76
16:34:00.717854 IP pcn312g.vsb.cz.4500 > dhcp194.kt.agh.edu.pl.4500: NONESP-encap:
isakmp: child_sa inf2[ R]
16:34:01.705872 IP pcn312g.vsb.cz > dhcp194.kt.agh.edu.pl: ICMP echo request, id
7176, seq 137, length 64
16:34:02.707915 IP pcn312g.vsb.cz > dhcp194.kt.agh.edu.pl: ICMP echo request, id
7176, seq 138, length 64
16:34:03.708786 IP pcn312g.vsb.cz > dhcp194.kt.agh.edu.pl: ICMP echo request, id
7176, seq 139, length 64
16:34:03.946818 IP pcn312g.vsb.cz.isakmp > dhcp194.kt.agh.edu.pl.isakmp: isakmp:
parent_sa ikev2_init[ R]
16:34:03.998277 IP pcn312g.vsb.cz.4500 > dhcp194.kt.agh.edu.pl.4500: NONESP-encap:
isakmp: child_sa ikev2_auth[ R]
```

At the start, the ISAKMP protocol was used to provide a framework for authentication and cryptographic key exchange. Additionally, we used IKE to establish a Security Association (SA) to share security attributes between both ends of the VPN tunnel. Secure communication was supported by attributes including cryptographic algorithms, mode of encryption, key length, etc.

Finally, the VPN connection between AGH (Poland) and VSB (Czech Republic) was established. The details of the tunnel's status are shown below.

```
Connections:
  VSB-AGH: 149.156.114.194…158.196.142.76
  VSB-AGH: local: [ 149.156.114.194] uses pre-shared key authentication
  VSB-AGH: remote: [ 158.196.142.76] uses any authentication
  VSB-AGH: child: dynamic === dynamic

Security Associations:
  VSB-AGH[ 3] : ESTABLISHED 89 seconds ago,
                149.156.114.194[ 149.156.114.194] …158.196.142.76] 158.196.142.76]
  VSB-AGH[ 3] : IKE SPIs: 92c1014b8b1a478b_i 9ee4cdab07c82629_r*, pre-shared key
                reauthentication in 54 minutes
  VSB-AGH[ 3] : IKE proposal: AES_CBC_256/HMAC_SHA1_96/PRF_HMAC_SHA1/MODP_1536
  VSB-AGH{ 2} : INSTALLED, TUNNEL, ESP SPIs: cd67ddfa_i c9fcb900_o
  VSB-AGH{ 2} : AES_CBC_256/HMAC_SHA1_96, 0 bytes_i, 0 bytes_o, rekeying in 12 min
  VSB-AGH{ 2} : 149.156.114.194/32 === 158.196.142.76/32
```

Authentication of both entities was based on the pre-shared key (generated using the quantum cryptography method). Confidentiality of transmitted data was censured by the AES cipher with a 256-bit key. Data was encrypted using the CBC (Cipher Block Chaining) mode. Data integrity was protected by the SHA-1 (Secure Hash Function) algorithm.

To check whether transmitted data was protected by the IPsec protocol, the *xfrm* framework for transforming encrypted packets was used. Some details are shown below:

```
src 149.156.114.194 dst 158.196.142.76
  proto esp spi 0xcb8b7ff8 reqid 1 mode tunnel
  replay-window 32 flag af-unspec
  auth-trunc hmac(sha1) 0xfb3f33d2751e3382e8d5fe58a7370c3de42c97fa 96
  enc cbc(aes)
    0xba9c4b796edb8f2888d0f6eacce326d52d7384d2333eaeb0244c485dda228b08
src 158.196.142.76 dst 149.156.114.194
  proto esp spi 0xc6db653e reqid 1 mode tunnel
  replay-window 32 flag af-unspec
  auth-trunc hmac(sha1) 0x202135ca11f23fce1b3ce8d48bdd16d2c4bf6b2e 96
  enc cbc(aes)
    0x6f642743c9ebacc20010fa9f3d433f727c078cae929bc97c0cff6b43fe5a408a
```

The established tunnel was built by ESP – member of the IPsec protocol suite. It provides confidentiality (using the AES cipher), data integrity (using SHA-1) and data-origin authentication. The entire original IP packet was encapsulated with a new packet header (IPsec). This solution made it possible to build a secure path between AGH and VSB through an untrusted network. End-users are able to use this tunnel to deliver confidential data, share joint resources or access network services in a secure way.

5.2 Final key length

The QKD technique is a suitable candidate for securing VPNs by establishing shared keys between end-users. This is because it provides protection against eavesdropping, and due to the nature of the shared bits. Laws of physics mean that quantum-based generators are an excellent source of randomness [14], therefore the established key is truly random. Additionally, a dedicated quantum channel makes it possible to use long strings of bits during authentication and enables frequent changes of keys.

Table 1 Relationship between the value of QBER and final key length (example simulation)

QBER [%]	Starting key length [bit]	Final key length [bit]	Keys are the same
0.00	1024	314	True
0.78	1024	310	True
1.76	1024	305	True
2.54	1024	302	True
2.73	1024	300	True
3.71	1024	295	True
4.49	1024	293	True
5.08	1024	290	True
6.05	1024	288	True
7.23	1024	284	True
8.40	1024	279	True
8.79	1024	278	True
9.57	1024	271	True
10.55	1024	267	True
11.52	1024	266	True
12.70	1024	261	True
13.48	1024	259	True
15.23	1024	252	True
15.82	1024	250	True
17.38	1024	245	True
17.77	1024	243	True
19.14	1024	238	True
19.92	1024	235	True
21.29	1024	231	True
22.07	1024	231	True
23.05	1024	228	True
24.41	1024	225	True
25.59	1024	221	True
26.37	1024	221	True
27.54	1024	0	False
28.52	1024	0	False
29.69	1024	0	False
31.25	1024	0	False
31.84	1024	0	False
33.20	1024	0	False
34.38	1024	0	False
35.35	1024	0	False
36.33	1024	0	False
37.30	1024	0	False
39.06	1024	0	False
39.65	1024	0	False
40.63	1024	0	False
41.21	1024	0	False
41.80	1024	0	False

Table 1 (continued)

QBER [%]	Starting key length [bit]	Final key length [bit]	Keys are the same
43.16	1024	0	False
43.55	1024	0	False
44.53	1024	0	False
45.31	1024	0	False
46.48	1024	0	False
47.27	1024	0	False
48.05	1024	0	False
49.02	1024	0	False
50.20	1024	0	False

Unfortunately, the QKD protocols reduces the final key significantly [22]. This reduction of key length is characteristic of all quantum key distribution protocols. For example, the original key presented in section 4.1 had a length of 1024 bits, but after all the QC steps (error estimation, key reconciliation, privacy amplification), the length of the final key was just 292 bits.

However, the final key length strongly depends on noise intensity in the quantum channel. Higher levels of noise result in a higher value of QBER and a greater reduction of the key during the key reconciliation process. A number of simulations were performed to check this relationship. The authors assumed that at the start of the QKD process, Alice sends to Bob 1024 bits using the BB84 protocol (*Starting Key Length*). All algorithms and parameters were similar to the QKD process presented in section 4.1. The only difference was the QBER, which changed from 0 to 100 %. An example of results obtained is presented in Table 1 (QBER values from 0 to 50.20 %).

The simulations confirmed that the final key length (*Final Key Length*) strongly depends on noise intensity in the quantum channel. However, when QBER exceeds the value of 30 %, it is

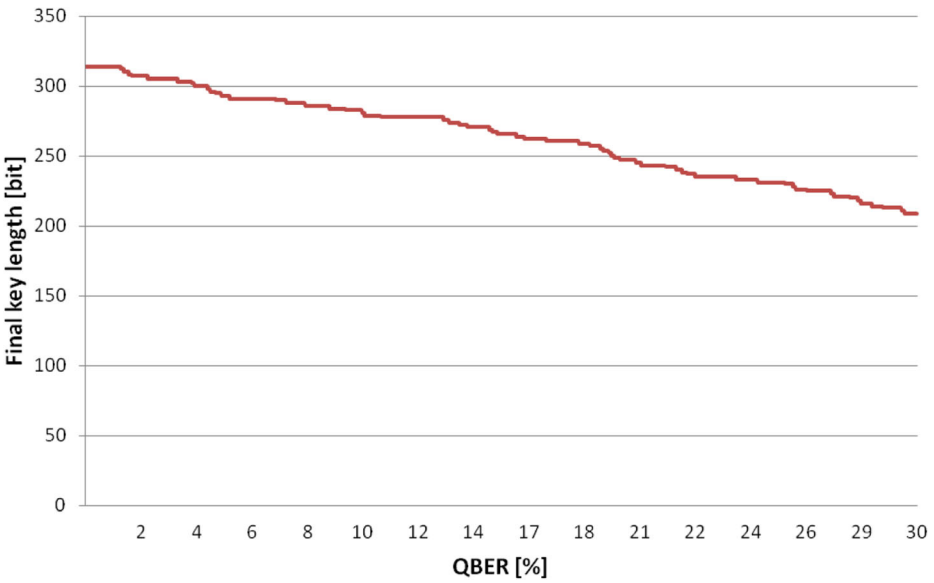

Fig. 2 Reduction of final key length

not possible to establish a final key between Alice and Bob. Therefore, almost all simulations were conducted for QBER values below 30 %. The final results are presented in Fig. 2.

The final key varies from 314 bits (QBER=0 %) to 213 bits (QBER=30 %). The reduction is significant (approx. 30 % difference between final keys). Therefore, this effect must be considered when QKD techniques are used for authentication in VPNs.

6 Conclusions

Data security is one of the most important requirements today. It also poses a major challenge, since absolute security is unreachable in practice. However, end-users of communication services strive to apply increasingly more secure methods to protect data transmitted through unsecure channels. One promising technology is quantum-based security, mainly using the quantum key distribution method. The laws of physics make it possible to uncover all eavesdroppers. Quantum cryptography is currently used to distribute encryption keys securely, although different usages may also be proposed in the future.

This article proposes using quantum key distribution to support authentication of end-users. The pre-shared keys – secrets which confirm the identity of end-users – can be distributed using quantum protocols, such as the BB84 protocol. This way, the pre-shared keys are established using the highest secure solutions available in modern communications.

The presented idea was verified in practice: an IPsec VPN tunnel was successfully established between AGH (Poland) and VSB (Czech Republic). The authentication of both end-users was based on a pre-shared key. This key was generated by the QKD simulator, where the BB84 protocol, error estimation process, key reconciliation, and privacy amplification were implemented. Additionally, authors checked the effect of final key reduction. This reduction must be taken into account during the design process – especially, length of shared keys using for authentication and frequency of key changing. Once the tunnel was established, end-users were able to transmit confidential data in a secure way.

Today, VPN is a well known and widely used solution that protects digital data during transition across a network. In the near future, development of quantum-based techniques may well change our approach to this technique. Combining VPN and quantum key distribution may increase security to levels unachieved by any previous solutions.

Acknowledgments This work has been partially funded by the European Union, project INDECT (*Intelligent information system supporting observation, searching and detection for security of citizens in urban environment*) — grant agreement number: 218086.

References

1. Assche GV (2006) “Quantum Cryptography And Secret-Key Distillation”.Cambridge University Press
2. Bennett CH, Bessette F, Brassard G, Salvail L, Smolin J (1992) “Experimental quantum cryptography”. J Cryptol 5:3–28
3. Bennett CH, Brassard CH (1984) “Public key distribution and coin tossing”. IEEE International Conference on Computers, Systems, and Signal Processing: 175–179
4. Bennett CH, Brassard G, Crepeau C, Maurer UM (1995) Generalized privacy amplification. IEEE Trans Inf Theory 41(6):1915–1923

5. Bennett CH, Brassard G, Robert J (1988) “Privacy amplification by public discussion”. SIAM J Comput 17: 210–229
6. Bouwmeester D, Ekert A, Zeilinger A (2000) “The Physics of Quantum Information. Quantum Cryptography, Quantum Teleportation, Quantum Computation”. Springer
7. Brassard G, Salvail L (1994) “Secret-key reconciliation by public discussion”. Springer-Verlag
8. Buchmann J (2004)“Introduction to Cryptography”.Springer
9. Deal R (2006) “The complete Cisco VPN configuration guide”. Cisco Press, Indianapolis
10. Dixon AR, Yuan ZL, Dynes JF, Sharpe AW, Shields AJ (2010) “Continuous operation of high bit rate quantum key distribution”. Applied Physics Letters 96(16)
11. Ekert A, Renner R (2014) “The ultimate physical limits of privacy”. Nature 507:443–447
12. Furukawa E, Yamazaki K (2001) “Application of existing perfect code to secret key reconciliation”. International Symposium on Communication and Information Technologies: 397–400
13. ID Quantique (IDQ) company webside. http://www.idquantique.com
14. ID Quantique White paper, “Random number generation using quantum physics” (2010)
15. MagiQ company webside. http://www.magiqtech.com
16. Niemiec M, Pach AR (2012) “The measure of security in quantum cryptography”, IEEE Global Telecommunications Conference (GLOBECOM 2012)
17. Niemiec M, Pach AR (2013) “Management of security in quantum cryptography”. IEEE Commun Mag 51(8):36–41
18. Niemiec M, Romanski L, Swiety M (2011) “Quantum cryptography protocol simulator”, multimedia communications, services and security, vol 149. Springer, Berlin, pp 286–292
19. Peev M, Poppe A, Maurhart O, Lorunser T, Langer T, Pacher C (2009) “The SECOQC quantum key distribution network in Vienna”, European Conference on Optical Communication
20. Recommendation X.800: Security architecture for Open Systems Interconnection for CCITT applications (1991)
21. RFC 2631: Diffie-Hellman Key Agreement Method (1999)
22. Scarani V, Bechmann-Pasquinucci H, Cerf NJ, Dusek M, Lutkenhaus N, Peev M (2009) “The security of practical quantum key distribution”. Rev Mod Phys 81(3):1301–1350
23. StrongSwan website. http://www.strongswan.org. Accessed 11 December 2013
24. Wootters WK, Zurek WH (1982) “A single quantum cannot be cloned”. Nature 299:802–803

Marcin Niemiec obtained his M.Sc. and Ph.D. in Telecommunications at the AGH University of Science and Technology, Krakow, Poland, in 2005 and 2011, respectively. He has also studied at the Universidad Carlos III de Madrid. He is an assistant professor at the Department of Telecommunications, AGH University of Science and Technology. His research interests focus on security and data protection, in particular security services, symmetric ciphers, cryptanalysis, malware, intrusion detection, and quantum cryptography. Co-organizer of a number of international meetings, workshops, and conferences. He has actively participated in European Commission’s 6th and 7th Framework Programmes (ePhoton/ONE+, BONE, SmoothIT, INDECT), Eureka-Celtic (DESYME), and several national projects. He is the recipient of the Best Paper Award from IEEE GLOBECOM 2012. He has co-authored over 50 publications and reports (papers, deliverables, book reviews, IETF draft, and book).

Petr Machník received his M.Sc. and Ph.D. degrees in Telecommunications from VSB—Technical University of Ostrava, Ostrava, Czech Republic in 2004 and 2008, respectively. Presently, he is an assistant professor at the Department of Telecommunications, VSB—Technical University of Ostrava. His research activities are focused on computer, telecommunication and wireless networks and communication security. He has participated in several national and international research projects related to these topics, including INDECT project of the EU 7th Framework Program.