A High Speed Quantum Communication Testbed

Carl J. Williams, Xiao Tang, Mikko Hiekkero, Julie Rouzaud, Richang Lu, Andreas Goedecke, Alan Migdall, Alan Mink, Anastase Nakassis, Leticia Pibida, Jesse Wen [a], Edward Hagley [a], and Charles W. Clark

National Institute of Standards and Technology, Gaithersburg, MD 20899
[a] also Wen, Hagley, and Associates, Rockville, MD 20850

ABSTRACT

We describe the status of the NIST Quantum Communication Testbed (QCT) facility. QCT is a facility for exploring quantum communication in an environment similar to that projected for early commercial implementations: quantum cryptographic key exchange on a gigabit/second free-space optical (FSO) channel. Its purpose is to provide an open platform for testing and validating performance in the application, network, and physical layers of quantum communications systems. The channel uses modified commercial FSO equipment to link two buildings on the Gaithersburg, MD campus of the National Institute of Standards and Technology (NIST), separated by approximately 600 meters. At the time of writing, QCT is under construction; it will eventually be made available to the research community as a user facility. This paper presents the basic design considerations underlying QCT, and reports the status of the project.

Keywords: quantum cryptography, free-space optical communications, gigabit ethernet

1. INTRODUCTION

Quantum key distribution (QKD) has progressed rapidly since the first tabletop experiments[1]. For example, the BB84 protocol[2] has recently been implemented over channels consisting of 64 kilometers of optical fiber[3] and 10 km of open air[4]. The latter accomplishment is of particular significance because the optical depth of the path is equivalent to that of an earth-to-satellite link. Thus, quantum key distribution has matured to the level of a technical demonstration project, and commercial turnkey systems are coming to market[5]. However, there is still a relatively limited basis of experience with quantum cryptographic key exchange in a realistic modern network environment, and the commercial viability of quantum communication will depend upon its demonstrated performance in such environments. The National Institute of Standards and Technology (NIST), in collaboration with the Defense Advanced Research Project Agency (DARPA) has thus undertaken an effort to build a high-speed quantum communication test-bed facility (QCT). This facility is aimed at providing a 1GHz dedicated quantum channel, which can be used to test and validate performance of quantum communications systems in the application, network, and physical layers.

The NIST QCT has several fundamental differences from previous quantum communication systems. First, it is aimed at being a realistic high-speed quantum communications facility that is integrated into the internet. The goal of this facility is to provide a prototype system and to develop the infrastructure for producing a scalable quantum communications network that can be interfaced to modern communications systems at the physical and network layers. The QCT will include capabilities for characterizing photon sources and detectors and for testing component systems in a well-characterized environment at GHz clock rates. Other papers in this conference describe NIST's complementary efforts on single-photon sources[6] and detectors[7].

The first stage of the QCT consists of two nodes connected by a two free-space optical (FSO) channels: a classical channel, which carries the clock, provides framing information, and performs the "classical" communications used in QKD; and a quantum channel, which consists of an attenuated photon source in the first instance. Incorporation of a third node connected by optical fiber is planned for the future. The principal technical challenge presented by the QCT system is its target clock speed of 1 GHz. This requires accurate time synchronization of transmitter and receiver in order for photons in the quantum channel to be uniquely distinguished. In most previous systems, a classical pulse heralds the arrival of photons in the quantum channel. As clock speed increases, heralding becomes a major time limitation. Thus, we have designed the QCT to have a dedicated classical channel carrying timing information, which is locked to the quantum channel. We now describe the general hardware being developed and conclude with a discussion of the software interface, which will eventually be integrated in the Secure Socket Layer (SSL) of the internet.

2. HARDWARE

Our implementation of the BB84 algorithm has relegated a large portion of data processing to the hardware that stands between the control computer and the photonic components. The focus of our hardware design, also referred to as a crypto-module, is a field-programmable gate array (FPGA). The FPGA, along with other supporting chips, is placed on an application specific printed circuit board (PCB) that communicates with the control computer via a standard PCI bus interface. The PCB photonics interface meshes to the unidirectional quantum channel as well as with the bi-directional classical channel, which is separate from the internet channel used by the software to authenticate the participants' identity and to resolve quantum channel errors. In a commercial version the FPGA would be replaced by a specially designed circuit board. However, for testing and design purposes the FPGA provides improved flexibility.

This interface between the processor and the photonics covers a broad area of functionality. At the lowest level we are dealing with the integrity and timing of GHz signals, while at the highest level we are dealing with data handling and communication bus protocols. Figure 1 shows the block diagram of our two PCB designs, one for the source (Alice) and the other for the destination (Bob). Figure 2 shows the block diagram of our two FPGA designs, one for the source (Alice) and the other for the destination (Bob).

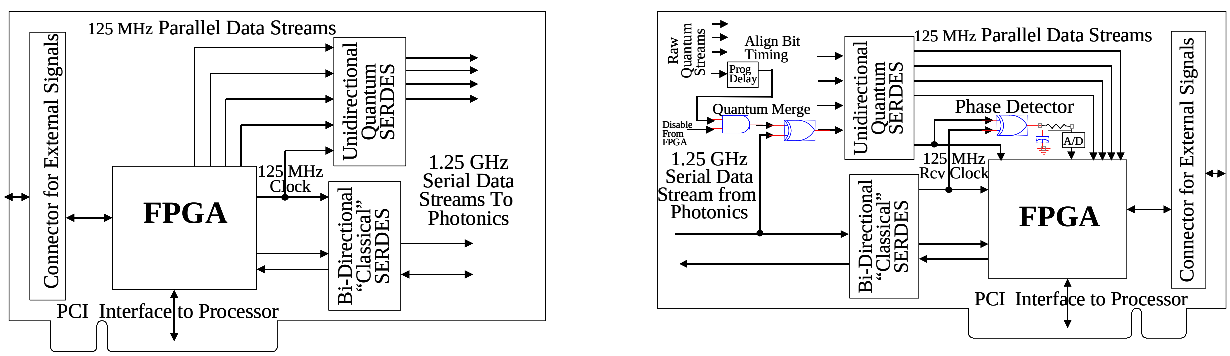

Fig. 1: Block diagram of PCB designs for Alice (right) and Bob (left).

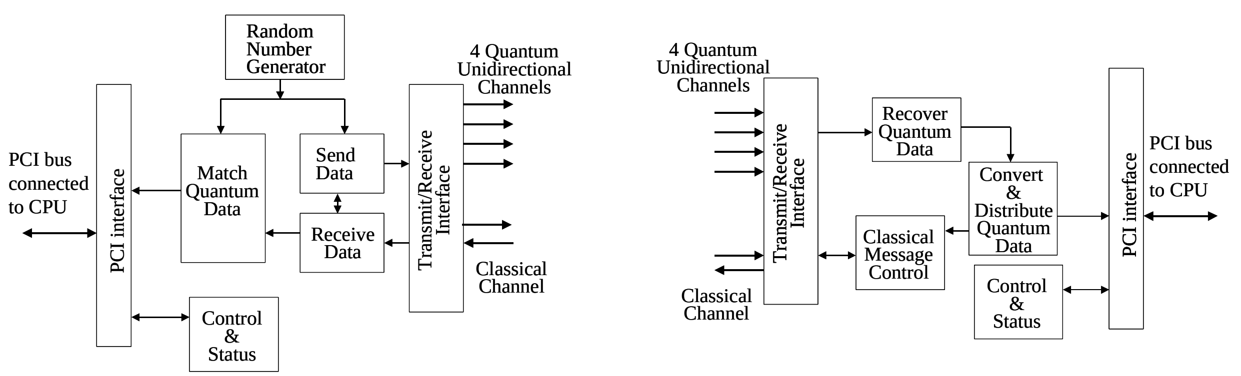

Fig. 2: Block diagram of FPGA designs for Alice (right) and Bob (left).

The source transmits a unidirectional quantum channel that transmits four possible polarization states, and one bi-directional classical channel. The destination receives quantum and classical channel, and also transmits a return classical channel. The destination PCB deals with the detection and time alignment of the recovered quantum data relative to the classical data. Since the high-speed receiver circuitry is looking for continuous differential signals, it would never recognize the non-continuous, sparse signal from the quantum detectors. Our solution is to merge (XOR) the non-continuous GHz quantum data stream with the continuous GHz classical data stream and then at the lower parallel data rate, within the FPGA, remove that known classical data to recover the quantum data. This also allows us to use the classical data stream synchronization characters to achieve timing alignment on the quantum stream. Further alignment of the classical and quantum streams is done within the FPGA. The combined quantum-classical system initiates a process to de-skew or phase lock the two channels by transmitting classical pulses across the quantum channel.

A good way to understand the functional description is to follow the operational flow, which we now discuss. The source FPGA generates two streams of random numbers, one for the two bases (+90, 0 and +45, -45) and one for the two values (0 and 1) characteristic of the BB84 protocol. For each bit position of these streams, the two bits are decoded into one of four states, with only one state being active. These four state streams are aligned with a message on the classical data stream, and all are synchronously sent, in parallel form, to a SERDES (serialization/de-serialization) chip for serialization at a 1.25 Gbits/s output data rate. These serial data streams are sent to photonic interfaces for free-space transmission. The received classical data stream at the destination is split, one copy is fed into a SERDES to de-serialize the data into a parallel format at a lower speed and then passed to the FPGA for processing. The other copies are individually merged (XOR) with each of the quantum streams, after the quantum stream is synchronized to a bit period by a Programmable Delay chip. The merged quantum streams are then de-serialized by another SERDES, and that parallel data is also passed to the FPGA for processing. The AND gate between the programmable delay and the XOR gate provides the ability to block the quantum data and pass only the classical data stream. This allows us to use the SERDES alignment capabilities to word align that stream based on the synchronization characters in classical data stream. A phase detection circuit between the parallel classical data clock and each of the parallel quantum channel data clocks, from the SERDES, is used to determine when the quantum data streams require re-synchronization.

When a classical channel synchronization message is received at the destination FPGA, it signals the quantum channel receivers to begin assembling a packet of quantum data. Each of the quantum receivers then applies a previously programmed bit delay for that channel when assembling a packet. As the quantum packets are being assembled and aligned, the recover crypto-module processes them for the presence and value of any quantum data received. Once recovered, the quantum data is then converted from a bit-stream format to an encoded format of bit position within the packet and its basis and bit value. This encoding does increase the volume of the data, but due to the expected loss the overall data volume will be reduced. The encoded quantum data list is then distributed to both the PCI interface, to be sent to Bob's processor, and to the classical channel – but with the quantum bit value stripped off, to be sent back to the source FPGA in the source crypto-module. Bob will treat this data as the starting values for error correction, reconciliation, and privacy amplification.

When the source FPGA receives the quantum data in the encoded format on the classical channel, it passes it to the match module to further cull this list. The match module has a copy of the original quantum data transmitted. It compares the basis value of each quantum item in the list and discards those items whose basis value is incorrect, thus further reducing the list size. For the remaining items, the match module adds the bit value, which was stripped off by Bob. This reduced list is then passed to the PCI interface to be sent to Alice's processor. Alice will treat this data as the starting values for the software error resolution algorithm. Notice that Alice and Bob's lists are not the same, since Alice's list has eliminated items that have incorrect basis information. The software error resolution algorithm, conducted over the open internet channel, will resolve the differences between these two lists. The volume of data that has to be handled by the software error resolution algorithm is greatly reduced by sending Bob's list back to Alice over the classical channel and starting with that culled list.

In the initial setup, the quantum channel will use attenuated vertical cavity surface emitting lasers (VCSEL's) operating at 850nm and Si avalanche photodiodes for the quantum channel detectors. These may eventually be replaced by internally developed sources and detectors or provided by the user community.

3. SOFTWARE

The software we are developing aims at implementing a prototype for the BB84 QKD protocol, and to develop applications that demonstrate the use of the shared secrets. In order to verify the actual security of a given QKD system, the intelligent design of the key negotiation protocol requires some knowledge of the interfaces of the protocol that uses the keys (the security protocol). Moreover, any design must make choices informed by environmental assumptions. As the application and computer security environment in which BB84 protocols will be deployed is not known, we need to develop modular designs based on the functionality of the protocols in question rather than the particulars of the algorithms in vogue. In general, security protocol interfaces need wide acceptance. As such, the prototype we are developing needs to make minimal assumptions and rely on minimal interface design.

We have designed our protocol with the following basic assumption. The BB84 secret key creation mechanism is slow with respect to the overall throughput of a busy host, and will probably remain so in the future. Therefore, cryptographic key resources must be created in anticipation of future needs. This contrasts to a scheme where the required key is generated on the fly. A one time-pad encryption will be available on a per-need basis (e.g., applications that are the equivalent of a secure telephone unit). Therefore, our prototype will support and demonstrate security on demand. Further, because we place no restrictions on how a key will be used, user applications will be free to use a length of key in an unrestricted manner even though this may increase the risk of the resulting transmission. We now describe the software of the BB84 protocol for a generic application.

Classical cryptography relies on a design in which key-management protocols negotiate security associations that can be seen as distributed databases. Networking protocols decide which data streams require security services and provide them. The interactions between the security association and the networking protocols are local and require minimal, if any, synchronization. Security based on one-time pads requires synchronization to ensure that “Alice” and “Bob” do not attempt to simultaneously use the same one-time pad and thereby create a potential security risk. This can be avoided if we operate with two pads per host pair: one used to encrypt, with a corresponding decrypting pad in the remote host, and one used to decrypt with a corresponding remote encrypting pad. Even then we must allow for the fact that we work over an overall unreliable network service that can reorder packets and that we need to address the issue of when and why bits in the decryption pads are discarded. We assume therefore that we will create two keypads per BB84 connection, and we shall implement a mechanism for discarding bits in the decryption pad. Finally, we assume that the BB84 implementation pulls sifted bits as needed from the low layer protocol.

The basic software design we adopted is based on a three tier architecture of which the main one is the one in the middle. The top layer consists of a very simple application to demonstrate how security is provided. When the application is interacting with a remote peer, the two ends open a socket to the mid layer and obtain an agent. Subsequent communications are labeled clear or secret and are handled by the agent. The application and testing software to mimic the mid layer have already been coded. Diagrammatically the top layer application and the middle layer are shown in Fig. 3; the functions of the middle layer are expanded in Fig. 4.

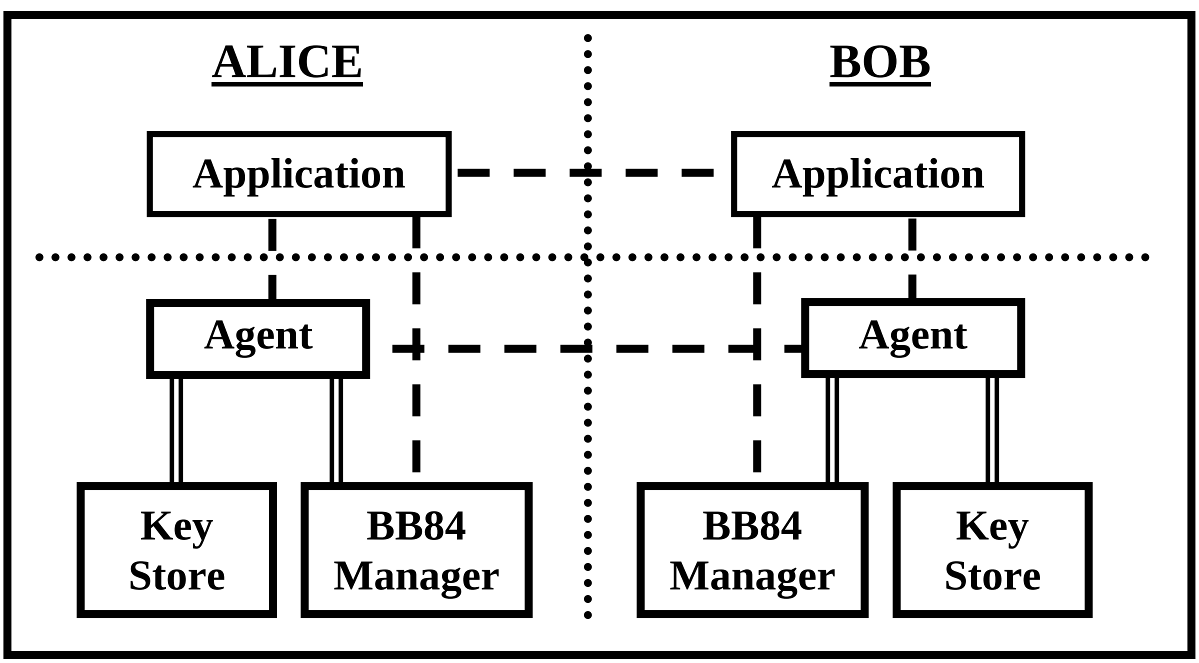

Fig. 3: Top and middle layers of the cryptographic key management system. Dashed lines denote socket based communications, double lines show Java calls. Not all communications are shown nor are the implementation details (thread structure).

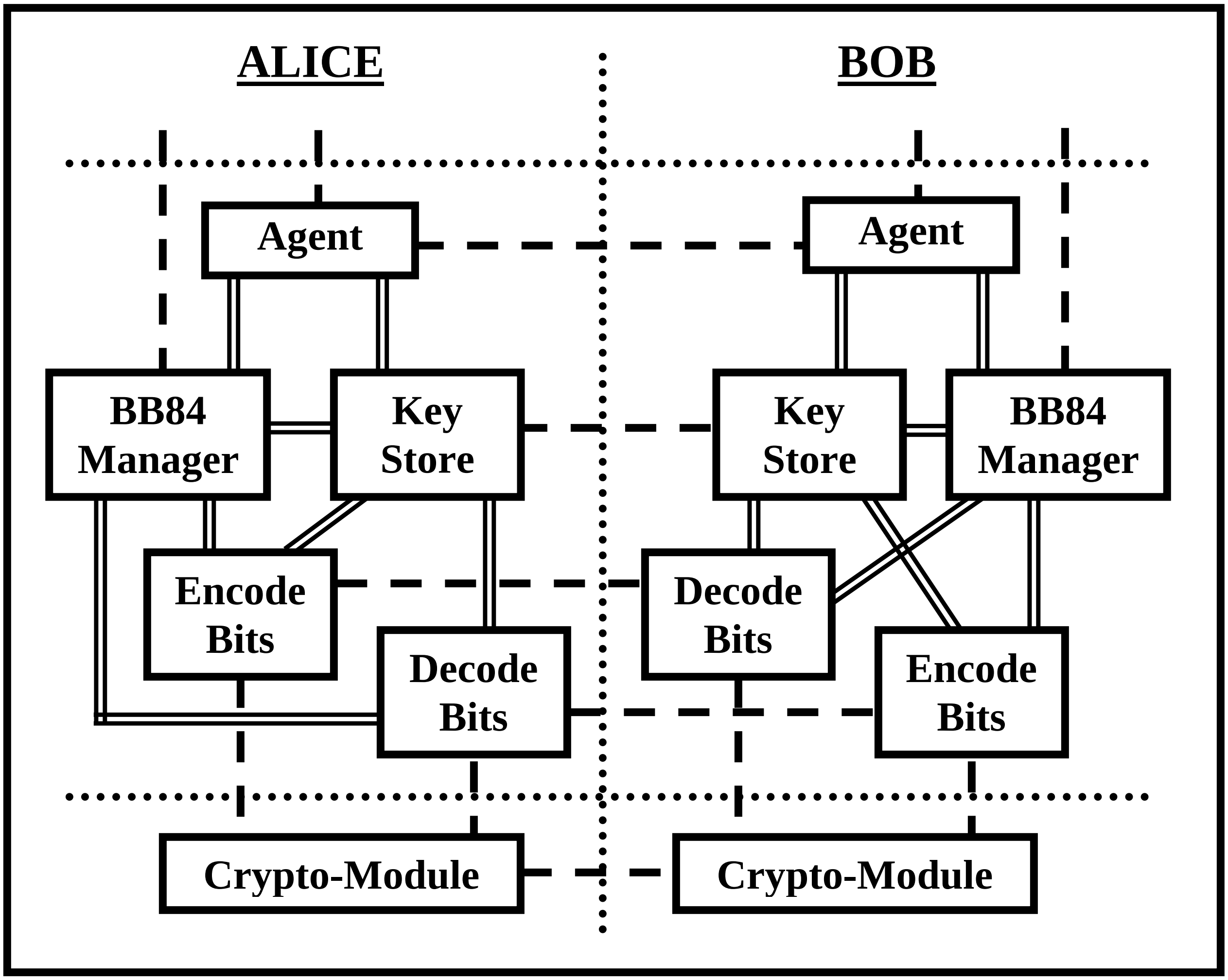

Fig. 4: Expanded view of the middle layer of the cryptographic key management system. Dashed lines denote socket based communications, double lines show Java calls. In principle, the BB84 Manager could be a manager for a different protocol, but our initial QCT design is based on BB84.

The bottom layer is a group of hardware and software components (described above) that creates sifted keys and, on demand, will produce the proper sifted bits to the requesting end and to the remote peer. Because by the time the sifted keys have been produced, the roles of qubit sender/qubit receiver are not relevant, the requesting end is always the one in need of bits for encryption purposes. The matching bits at the far end are delivered to the module in charge of producing decrypting keys. Diagrammatically this is shown in Fig. 3. We should note that the bottom layer crypto-module contains the hardware previously described in this paper along with the hardware-software interface. This interface serves to sift the presifted qubits, to provide error correction and reconciliation, and to provide privacy amplification. The privacy amplification depends not only on the error rate, but on information about the nature of the photon source. If the photon source is an attenuated laser source, then the privacy amplification will also amplify out any information that could be obtained from an attack that involves the detection and removal of individual photons from multi-photon pairs.

To be able to experiment with BB84 alternatives without using a quantum channel, we have developed and tested pure software versions of the crypto-module (based on pseudo-random number generation of sifted keys) and testing modules above so as to exercise the interface and the correctness of the crypto module's logic.

The mid layer BB84 manager is the central security agent for the overall quantum crypto system. In principle, the hardware or quantum cryptographic protocol may be modified, but the higher-level functions will remain similar. In the present paradigm, the structure of the mid layer for a host that has at least one quantum channel is the following. It contains a single manager module, shown here as the BB84 Manager, that provides an interface to the relevant neighbors and applications that need cryptographic security. The manager controls modules for generating encoding secrets, encode bits, and decoding secrets, decode bits. Finally, it is responsible for providing synchronized storage bins for the secrets created (key-store) and cryptographic services to the higher lying applications.

Our intent is to develop the modules and interfaces that have not yet been defined and to initially integrate in-house and public implementations of the cascade algorithms. Once this is done and successfully tested, we shall test alternative schemes that will seek to optimize the cost of classical communications necessary to the creation of BB84 secrets.

USE OF TRADE NAMES

Use of trade names in this article is for purposes of identification only. The National Institute of Standards and Technology has not certified the performance of any products so identified, and does not imply that such products are uniquely or preferably suitable for any given application.

ACKNOWLEDGMENTS

This work is partially supported by the Quantum Information Science and Technology (QuIST) program of the Defense Advanced Research Projects Agency under contract L492.

REFERENCES

1. C. H. Bennett, F. Bessette, G. Brassard, L. Savail, and J. Smolin, "Experimental quantum cryptography," *J. Cryptology 5*, pp. 3-28 (1992)
2. C.H. Bennett and G. Brassard, "Quantum Cryptography: Public Key Distribution and Coin Tossing", *Proc. IEEE Int. Conf. on Computer Systems and Signal Processing*, pp. 175-179 (Bangalore India, December 1984); reprint: http://www.research.ibm.com/people/b/bennetc/bennettc198469790513.pdf
3. D. Stucki, N. Gisin, O. Guinnard, G. Ribordy and H., Zbinden , "Quantum key distribution over 67 km with a plug & play system", *New Journal of Physics 4* (June 2002 in press); preprint: http://arxiv.org/abs/quant-ph/0203118
4. R. J. Hughes, J. E. Nordholt, D. Derkacs and C. G. Peterson, " Free-space Quantum Key Distribution over 10 km in Daylight and at Night" *Los Alamos Report* LA-UR-02-449 (2002)
5. For example, see http://www.idquantique.com/qkd.html
6. A. L. Migdall, D. Branning, S. Castelletto, and M. Ware, " Single Photon Source with Individualized Single Photon Certifications," these proceedings
7. Sae Woo Nam, to be published