卒業論文　　2009年度 (平成21年度)

量子鍵配送を利用したIPsecのための IKE拡張

慶應義塾大学 総合政策学部

氏名：永山 翔太

卒業論文要旨 - 2009 年度 (平成 21 年度)

量子鍵配送を利用した IPsec のための IKE 拡張

本研究では，数学的に安全性が証明された暗号化通信の実現のために，Internet Key Exchange（IKE）の量子鍵配送を拡張した新しいプロトコルを設計し実装した．

現在の IKE は，因数分解問題を安全性の根拠とする Diffie-Hellman 鍵共有を利用している．しかし，この鍵共有は因数分解を効率的におこなう量子コンピュータが実現すればその有効性が失われてしまう．一方，量子鍵配送は安全性が物理的かつ数学的に証明された鍵共有方法である．量子鍵配送プロトコルの一つである BB84 は Bennett と Brassard によって提案され，NEC や NTT などによって装置が開発されている．

本研究では，量子鍵配送で生成した鍵を利用して IPsec を実現するための要件を整理し，IKE の量子鍵配送拡張プロトコルを設計した．そして IKE の WIDE プロジェクトによる実装である racoon2 を拡張し，raQoon2 としてこの新プロトコルを実装し，実証実験を実施した．本プロトコルは IETF での標準化を目指し，Internet Draft として提案した．

本研究により，量子鍵配送を利用した IKE と IPsec が実現し，量子鍵配送がインターネットで利用可能であることを確認した．また，量子鍵配送を利用した IPsec の安全性を Diffie-Hellman 鍵共有を利用した IPsec との比較によって定性的に確認した．そして量子鍵配送を利用するための，他の暗号化通信のプロトコルにも応用可能な IKE プロトコルであることを示した．

キーワード
1．量子鍵配送, 2．Internet Key Exchange, 3．IPsec

慶應義塾大学 総合政策学部

永山 翔太

Abstract of Bachelor's Thesis

Academic Year 2009

Modifying IKE for IPsec with Quantum Key Distribution

I extended the Internet Key Exchange Protocol to use keys created via Quantum Key Distribution (QKD), a key exchange algorithm which uses quantum effects to statistically prove the absence of eavesdroppers.

The Diffie-Hellman key exchange (D-H) in common use today, which depends on the difficulty of factoring large numbers, has never been proven to be secure, and may be broken in the future when quantum computers large enough to effectively factor large numbers are developed. QKD, in contrast, does not depend on the computational difficulty of any mathematical problem.

I researched requirements for crypto systems using Quantum Key Distribution and designed extenstions to the IKE protocol to use Quantum Key Distribution-generated keys for IPsec. I implemented and tested the new protocol extensions, extending the open source racoon2 IKE implementation, naming it raQoon2. Additionally, I submitted an Internet Draft to standarize the protocol through the Internet Engineering Task Force (IETF).

This thesis shows that Quantum Key Distribution is practical for IP-based networks, and the security level of the new system is compared to standard IPsec with Diffie-Hellman key exchange.

Keywords :
1. Quantum Key Distribution, 2. Internet Key Exchange, 3. IPsec

Keio University, Faculty of Policy Management

Shota Nagayama

目 次

第 1 章　はじめに **1**

1.1 序論 1
1.2 情報秘匿性の有効期限 2
1.3 本研究の目的 2
1.4 本研究の成果 3
1.5 本論文の構成 3

第 2 章　要素技術 **4**

2.1 Diffie-Hellman 鍵共有 4
2.2 IPsec 5
2.2.1 IPsec の仕組み 5
2.2.2 Security Association 5
2.3 Internet Key Exchange 6
2.3.1 IKE の管理する Security Association 6
2.3.2 IKE トランザクション 7
2.3.3 エラー処理 10
2.4 量子コンピューティング 10
2.4.1 重ね合わせ 11
2.4.2 量子ビット 11
2.4.3 エンタングルメント 11
2.4.4 Bell 測定 13
2.4.5 量子テレポーテーション 13
2.4.6 エンタングルメントスワッピング 13
2.4.7 エンタングルメント純化 15
2.4.8 量子リピータ 16
2.4.9 量子ネットワーク 16
2.5 量子非クローン定理 16

2.6 量子鍵配送 . . . 17
2.7 量子鍵配送と IPsec . . . 19
2.7.1 Quantum cryptography in practice . . . 19
2.7.2 Quantum Key Distribution (QKD) and Commodity Security Protocols: Introduction and Integration . . . 20
2.7.3 Quantum cryptography based on Bell's theorem . . . 20

第 3 章 IPsec with QKD 21
3.1 問題定義 . . . 21
3.2 解決手法 . . . 21
3.3 プロトコル設計 . . . 23
3.3.1 トランザクション . . . 23
3.3.2 ペイロード . . . 27
3.4 システム設計 . . . 29
3.4.1 本研究の要件 . . . 29
3.4.2 フォールバック設計 . . . 31

第 4 章 実装 33
4.1 量子鍵配送装置 QUICS . . . 33
4.2 実装環境 . . . 33
4.3 設計要件 . . . 34
4.4 raQoon2 実装 . . . 35
4.4.1 モジュール図 . . . 35
4.4.2 機能の達成 . . . 36
4.4.3 racoon2 改変部 . . . 37
4.4.4 鍵管理インタフェースの実装 . . . 40
4.4.5 raQoon2 のコンフィグレーション . . . 46

第 5 章 評価 49
5.1 展示/実験 . . . 49
5.1.1 量子鍵配送実用展示 . . . 49
5.1.2 ユーザの存在する環境での実用実験 . . . 50
5.2 評価 . . . 51
5.2.1 動作検証実験 . . . 52

5.2.2 定性評価 54
5.2.3 フォールバック方法評価 56
5.2.4 既存のシステムとのセキュリティ比較 57
5.2.5 既存のシステムとの情報秘匿性の有効期限比較 58
5.3 まとめ 59

第 6 章 結論 **60**
6.1 まとめ 60
6.2 今後の展望 61
6.2.1 完全な安全性を持つ暗号化通信について 61
6.2.2 Security Consideration 63
6.3 今後の課題 63

謝辞 **64**

付 録 A setkey コマンドによる SAD のダンプ **A**

付 録 B インターネットドラフト **B**

図 目 次

2.1 IKE 開始トランザクション 7
2.2 CREATE_CHILD_SA 交換トランザクション 10
2.3 エンタングルメントスワッピングの概念 14

3.1 IPsec with QKD ネットワーク 22
3.2 量子鍵配送対応 IKE 開始トランザクション 24
3.3 通常運転時の量子鍵配送対応 IKE_SA 更新トランザクション 25
3.4 フォールバック運転時の量子鍵配送対応 IKE_SA 更新トランザクション . . 26
3.5 鍵識別子交換ペイロード 27
3.6 フォールバック方法交換ペイロード 28

4.1 raQoon2 全体モジュール図（黒塗りは本研究で作成した部分，灰色は本研究で追加実装した部分） 35
4.2 qkd_choosekey_QUICS() の実装 41
4.3 qkd_fetch_key_QUICS() の実装 42
4.4 qkd_rm_keyfile_QUICS() の実装 43
4.5 鍵管理インターフェイス初期化機構の実装 44
4.6 図 4.5 を基にした，新しい鍵管理インタフェースの追加実装 45
4.7 lex ファイル改変場所 46
4.8 yacc ファイル改変場所 47
4.9 raQoon2 のコンフィグ例 48

5.1 ORF2008 展示ネットワーク 49
5.2 ORF2008 にておこなった展示の様子 50
5.3 WIDE 合宿実験ネットワーク 51
5.4 raQoon2 検証ネットワーク 52
5.5 情報秘匿性の有効期限比較 59

6.1 情報秘匿性の有効期限比較 62

A.1 複数の SA を単一の IPsec Gateway との間で張った様子その 1 . . . . . A i
A.2 複数の SA を単一の IPsec Gateway との間で張った様子その 2 . . . . . A ii
A.3 単一の SA を複数の IPsec Gateway との間で張った様子その 1 . . . . . .A iii
A.4 単一の SA を複数の IPsec Gateway との間で張った様子その 2 . . . . . .A iv
A.5 複数の SA を複数の IPsec Gateway との間で張った様子その 1 . . . . . A v
A.6 複数の SA を複数の IPsec Gateway との間で張った様子その 2 . . . . . .A vi

表 目 次

2.1 IPsec で利用されるプロトコル一覧 5
2.2 暗号化に関連する SA パラメータ 6
2.3 二種類の SA の役割 7
2.4 IKE ペイロード表 8
2.5 エンタングルしていないときの量子状態と確率 12
2.6 エンタングルしているときの量子状態と確率（例） 12
2.7 BB84 の実行例（記号は表 2.8 参照） 18
2.8 表 2.7 中の記号 18

3.1 IKE_SA_INIT 交換における交換ペイロードの比較（◯ は交換，× は交換なし） 23
3.2 IKE_AUTH 交換における交換ペイロードの比較（網掛けは暗号化，△ は選択） 25
3.3 CREATE_CHILD_SA 交換における交換ペイロードの比較（網掛けは暗号化） 27
3.4 フォールバック方法と識別番号 29
3.5 本研究の実装におけるシステム要件一覧 30

4.1 QUICS の特徴 33
4.2 実装環境 34
4.3 qkd.c に定義した量子鍵配送を利用するための関数一覧 36
4.4 racoon2 を構成するモジュール群 37
4.5 racoon2 改変ファイル一覧 38

5.1 システム動作実験における使用ホスト 52
5.2 本研究の実装におけるシステム要件と評価 54
5.3 フォールバック方法特性比較 56
5.4 本研究で構築したシステムと既存システムの安全性の比較 57

6.1 量子鍵配送と Diffie-Hellman 鍵共有並びに OTP と暗号化アルゴリズムの弱点比較 61

第1章　はじめに

1.1　序論

インターネットは登場以来コンピュータの発展と共に機能の拡張を続け，今や人々の生活に欠かせないインフラストラクチャとなっている．その用途は多岐に渡り，インターネットは民間公共問わず様々なサービスに利用されている．ニュース配信などの世界に向けて公開する情報のみならず，個人情報などの秘密に扱う必要のある情報の通信にも利用される今日のインターネットでは，暗号化通信は最も重要な技術の一つである．暗号化通信は，データの漏洩や改竄を防ぐために利用される．データが暗号化されていれば盗聴者は通信を傍受してもデータの内容がわからない．暗号化通信は暗号化するための鍵と暗号化アルゴリズムにより実現され，暗号の安全性は，鍵の安全性と暗号化アルゴリズムの安全性により決定する．このどちらかが破られれば，暗号は解読され，通信内容が漏洩する危険性がある．

現在，鍵の共有には公開鍵暗号と Diffie-Hellman 鍵共有が利用されることが多い．これらは，共に因数分解問題を利用したアルゴリズムである．因数分解問題とは，因数分解をおこなう効率的で現実的なアルゴリズムが存在しないことによる問題である．これらのアルゴリズムは，現状ではどんなコンピュータも効率的に因数分解を解けないことに安全性の根拠を置いており，効率的に因数分解を解く手法が実現されない限り今後も安全であると考えられている．

量子コンピューティングは，量子力学を利用してコンピュータを作る試みから始まったコンピューティングである．量子力学的相互作用の利用により，量子コンピューティングは現在のコンピューティングとは異なる計算能力を持つことが示されている．1980 年に始まった量子コンピューティングの研究では，量子コンピュータと量子鍵配送の開発に注力されて来た [1]．量子コンピューティングの研究は開発の困難性から一時は下火になったが，1994 年に Shor により効率的に因数分解をおこなう量子コンピュータアルゴリズムが発見されたのをきっかけに再び盛んになり，2009 年にはイェール大学によって 2 量子

ビットでの量子計算が実現された [2] [3]．量子鍵配送は，量子効果を利用した二者間での乱数共有方法である．量子力学によって数学的に安全性が保証されているため，因数分解が効率的に解かれるようになった後も使用に耐えうる鍵共有方式として期待されている．

量子コンピュータが開発された時，効率的な因数分解が可能になり，公開鍵暗号と Diffie-Hellman 鍵共有は無力化される．これは，因数分解問題によらない鍵共有による暗号化通信実用化の必要性を示唆している．

1.2 情報秘匿性の有効期限

数学的な安全性が証明されていない暗号化通信の秘匿性には，有効期限が存在する．例えば，100 時間の計算で解読できてしまう鍵共有アルゴリズムを利用した暗号化通信は，100 時間後には漏洩しても問題ない情報の通信にしか利用できない．因数分解問題に基づく鍵共有アルゴリズムも同じ問題を持っている．現在は解読不可能な暗号でも，将来的に技術の発展によって解読可能となるのであれば，暗号化通信の情報秘匿性の有効期限は，解読技術が実現したときである．公開鍵暗号や Diffie-Hellman 鍵共有は因数分解問題に基づく鍵共有アルゴリズムにあたり，量子コンピュータは因数分解問題を解く技術にあたる．将来量子コンピュータが開発されたとき，量子コンピュータが開発される前におこなわれた暗号化通信も，通信データが保存されていれば情報が漏洩する．

1.3 本研究の目的

本研究の目的は，安全性が証明された鍵共有方法を利用して，インターネットでの暗号化通信を実現することである．Diffie-Hellman 鍵共有は，因数分解問題に安全性の根拠をおいていることにより，将来因数分解が効率的に解かれるようになったとき利用できなくなることが示されている．しかしながら，安全性を担保し続けることができる他の鍵共有を利用するシステムは未だ整備されていない．また，Diffie-Hellman 鍵共有を利用した暗号化通信は，通信内容が将来的に漏洩すると予測されている．情報セキュリティではコスト対効果の考え方が重要であるため見落とされがちであるが，利用できるセキュリティが将来破られることは問題である．

1.4 本研究の成果

本研究では第 1.3 節で述べた目的を達成するため，安全性が証明された鍵共有方法として，量子効果の利用により統計数学的に盗聴者がいないことを確認可能な量子鍵配送を採用した．また，暗号化通信には様々な暗号技術を利用可能な IPsec を採用した．IPsec を利用することで量子鍵配送と様々な暗号技術を組み合わせた暗号化通信が可能となり，安全性の証明された鍵共有と様々な暗号技術を組み合わせた暗号化通信が可能となった．量子鍵配送で生成した鍵で IPsec の暗号化通信をおこなうために，IPsec の管理プロトコルである Internet Key Exchange（IKE）を拡張し，量子鍵配送に対応する新プロトコルを設計した．また，IKE の WIDE プロジェクトによる実装である racoon2 をもとにこの新しいプロトコルを実装し，raQoon2 と命名した．これにより，鍵共有アルゴリズムを攻撃することによる暗号解読が不可能な暗号化通信を実現した．また，これまで利用されてこなかった量子鍵配送の採用に伴い，量子鍵配送を利用するための新しいプロトコルの要件整備と設計をおこなった．この新しいプロトコルについては Internet Engineering Task Force（IETF）にてインターネットドラフトの提案をおこなった．

1.5 本論文の構成

本論分は，6 章から構成される．第 2 章では，本研究の背景となる要素技術を整理する．第 3 章では，本研究のアイデアについてまとめ，本研究で実装する raQoon2 のシステム設計について述べる．第 4 章では raQoon2 の実装についてまとめる．第 5 章では本研究のアイデアと実装したシステムについて評価する．第 6 章では本論文の結論と，今後の方針を述べる．

第2章　要素技術

本章では，本研究において関係するIPsec関連の技術及び量子情報技術の概要をまとめる．これらの技術の説明を通して，量子鍵配送を利用するIPsecの意義を述べる．

2.1　Diffie-Hellman鍵共有

Diffie-Hellman鍵共有は，ネットワークを介する二者間で共有秘密鍵を生成するためのアルゴリズムである[4]．Diffie-Hellman鍵共有は，以下の計算式で秘密鍵を計算する．

$$A = s^a \bmod p \tag{2.1}$$

$$B = s^b \bmod p \tag{2.2}$$

$$K_A = B^a \bmod p \tag{2.3}$$

$$K_B = A^b \bmod p \tag{2.4}$$

$$K_A = K_B = s^{ab} \bmod p \tag{2.5}$$

Diffie-Hellman鍵共有は，因数分解問題に基づいている．ある数nの因数分解をおこなうには，nを割る試行を1から順番に，最大で$\sqrt{n}$まで試してみなければならない．この際，n（二進数）の桁数が一つ増えれば，因数分解における割り算の最大試行数は$\sqrt{2}$倍に増えることになる．従って，逐次計算では，因数分解にオーダー$\mathrm{o}(2^n)$の時間がかかることになる．nを大きくしていくと，因数分解にかかる時間は天文学的な数字となり，現実的ではなくなる．即ち，巨大数の因数分解は実質不可能である．これが因数分解問題であり，Diffie-Hellman鍵共有が安全性の根拠としている．

2.2 IPsec

IPsec は，OSI7 階層中第 3 層ネットワーク層で通信を暗号化する暗号化プロトコルである [5]．第 3 層で暗号化することにより，第 4 層以上の情報はヘッダも含めて全て暗号化される．IPsec による通信は，複数の暗号化プロトコルの集合体として構成される．表 2.1 に，IPsec で利用されるプロトコルと役割を挙げる．

表 2.1: IPsec で利用されるプロトコル一覧

プロトコル名	役割
ESP（Encapsulating Security Payload）	パケットの秘匿性，完全性の確保
AH（Authentication Header）	パケットの完全性の確保
IPComp（IP Payload Compression Protocol）	パケット圧縮
IKEv2（Internet Key Exchange version 2）	IKEv2 通信

IPsec の運用形態はトンネルモードとトランスポートモードの二種類があり，トンネルモードは拠点間における暗号化通信に利用され，トランスポートモードは拠点ネットワークと単体間における暗号化通信に利用される．

2.2.1 IPsec の仕組み

カーネルは IPsec の挙動を二つのデータベースによって制御している．一つは，Security Association Databese（SAD）で，SA を管理している．もう一つは Security Policy Database（SPD）と呼ばれるデータベースで，パケットの処理ポリシーを管理している．カーネルは，送信もしくは転送するパケット一つ一つについて，SPD を参照して処理を決定する．その結果パケットが IPsec を適用する条件に適合した場合，カーネルは SAD から該当する SA を探し，SA のパラメータに従って暗号化する．パケットが SPD の持つ条件に適合しなかった場合，そのパケットには IPsec を適用されず，そのまま送信される．

2.2.2 Security Association

IPsec によって作られた暗号化通信路は Security Association（SA）と呼ばれるパラメータの集合によって認識される．SA のパラメータには，SA 検索に利用する Security Parameter Index や各種暗号化アルゴリズム，暗号化鍵，ライフタイムなどが含まれる．この際，お互いの IPsec Gateway で異なる鍵や異なる暗号化アルゴリズムを SA に設定した場合暗号

化通信は失敗する．お互いの IPsec Gateway で，SAD に同じパラメータを持つ SA を登録するには，事前に SA パラメータを折衝する必要がある．表 2.2 に，SA パラメータの例として暗号化に関係するパラメータを挙げる．

表 2.2: 暗号化に関連する SA パラメータ

パラメータ	用途
Security Parameter Index（SPI）	SA 検索に利用される SA 固有の値
AH Authentication algorithm	AH で利用される認証アルゴリズム
AH key	AH で利用する鍵
ESP Encryption algorithm	ESP で利用する暗号化アルゴリズム
ESP Encryption key	ESP における暗号化で利用する鍵
ESP Integrity algoritym	ESP で利用する認証アルゴリズム
ESP Integrity key	ESP における認証で利用する鍵
ESP combined mode algorithm	ESP で利用する認証付暗号化アルゴリズム
ESP combined mode key	ESP における認証付き暗号化で利用する鍵
SA Lifetime	SA の有効期限
IPsec Protocol mode	IPsec のモード

2.3 Internet Key Exchange

Internet Key Exchange(IKE) は，IPsec を管理するためのプロトコルである [6]．IKE の主な役割は二つある．一つは，暗号化鍵の共有である．IKE は暗号化鍵の共有に Diffie-Hellman 鍵共有を利用する．もう一つは，鍵の使い方の調整と管理である．IKE は暗号アルゴリズムの折衝や鍵の使用時間の設定など IPsec における鍵の利用方法を管理する．

2.3.1 IKE の管理する Security Association

IKE が管理する二種類の SA を表 2.3 に整理する．CHILD_SA は，データ通信を暗号化するための SA である．CHILD_SA は片方向であるため，一つの暗号化通信路作成に際して二つ作成される．IKE_SA は，IKE や各 SA の状態管理，更新，新しい SA の折衝を暗号化するための SA である．IKE_SA 自身の更新にも IKE_SA が利用される．CHILD_SA の更新，折衝は全て IKE_SA の中でおこなわれる．

2.3.2 IKE トランザクション

本小節では IKE 開始トランザクションと IKE_SA 更新トランザクションについて述べる．

IKE 開始トランザクション

IKE を開始する際のトランザクションを図 2.1 に示す．図中の各記号が表すペイロードを表 2.4 に示す．添字の i と r は，それぞれ始動者（initiator）と応答者（responder）を表す．AUTH は，認証情報を運ぶペイロードである．CERT は公開鍵証明書を使って認証する場合に証明書を運ぶ．CERTREQ は証明書要求をおこなう場合に利用される．HDR は

表 2.3: 二種類の SA の役割

SA の種類	役割
CHILD_SA	データ通信の暗号化
IKE_SA	IKE の管理，SA の更新，新しい SA の折衝

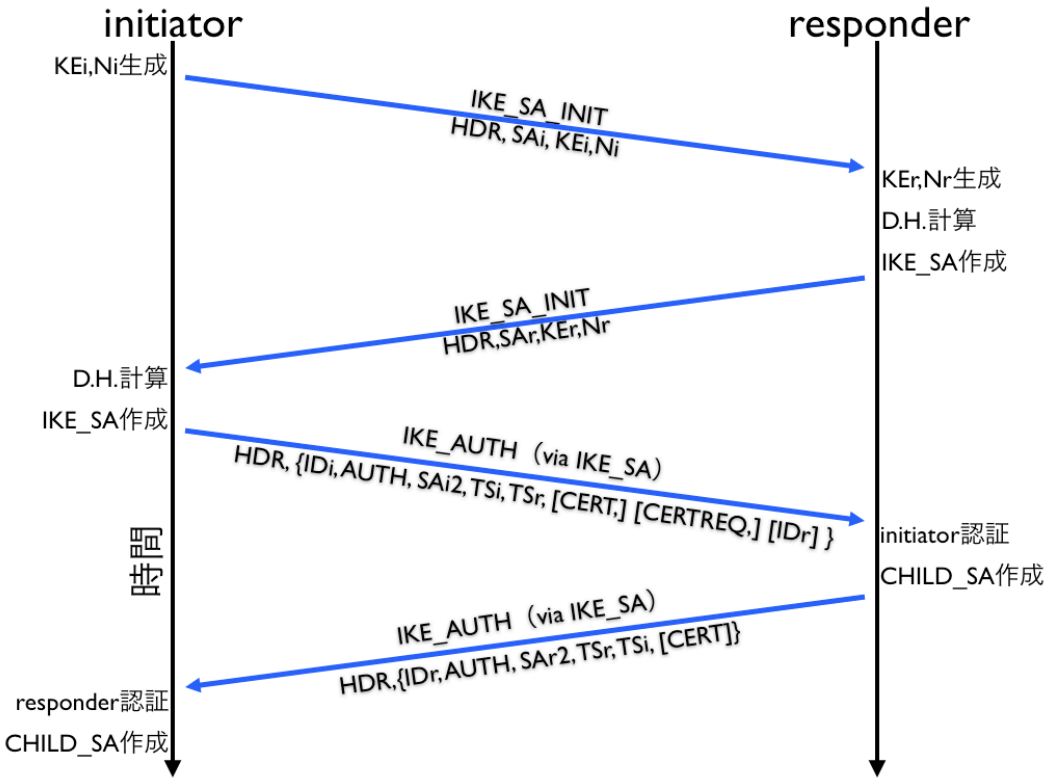


図 2.1: IKE 開始トランザクション

表 2.4: IKE ペイロード表

記号	ペイロード
AUTH	認証ペイロード
CERT	証明書
CERTREQ	証明書要求ペイロード
HDR	IKE ヘッダ
IDi, IDr	ID ペイロード
KEi, KEr	Diffie-Hellman 鍵共有ペイロード
Ni, Nr	乱数ペイロード
SAi, SAr	SA 提案ペイロード
TSi, TSr	トラフィックセレクタペイロード
N	通知ペイロード

IKE ヘッダであり，常に利用される．IDi，IDr は送信者が何者であるかを伝えるのに利用される．KEi，KEr は Diffie-Hellman 鍵共有の公開鍵を運ぶ．Ni，Nr は乱数を運ぶ．この乱数はセッションの生存の確認とリプレイ攻撃への対策に利用される．SAi，SAr は，送信者が許可されている暗号アルゴリズムの情報を運ぶ．TSi，TSr は，CHILD_SA を適用するトラフィックの指定に利用される．N は通知ペイロードで，エラーの通知などに利用される．IKE_SA 初作成時には，IKE_SA_INIT 交換と IKE_AUTH 交換が実行される．まず，IKE_SA_INIT 交換により IKE_SA を作成する．次に，作成した IKE_SA を用いて暗号化して，IKE_AUTH 交換により認証と CHILD_SA の作成をおこなう．以下に IKE_SA_INIT 交換と IKE_AUTH 交換について詳解する．

・IKE_SA_INIT 交換

IKE_SA_INIT 交換は，IKE_SA を作成するのに使われる交換である．IKE_SA_INIT 交換では，始動者は IKE ヘッダ，SA 提案ペイロード，Diffie-Hellman 鍵共有ペイロード，乱数ペイロードを送信する．IKE ヘッダは，セキュリティパラメータインデックスを保持し，SA の判定に使われる．SA 提案ペイロードは，始動者の許可されている暗号化アルゴリズム群の情報を保持している．Diffie-Hellman 鍵交換ペイロードは，Diffie-Hellman 鍵交換のための公開鍵を保持している．乱数ペイロードは，Diffi-Hellman 鍵交換公開鍵と共に鍵の生成に利用される乱数を保持する．応答者は，IKE ヘッダ，SA 提案ペイロード，Diffie-Hellman 鍵共有ペイロード，乱数ペイロードを送信する．この際，SA 提案ペイロードには，始動者から送られて来た

暗号化アルゴリズム群の中から一つ情報を載せ送信する．このとき，IKE_SA_INIT 交換は，一切暗号化されない．IKE は IKE_SA_INIT 交換の Diffi-Hellman 鍵交換ペイロードと乱数ペイロードを利用して鍵を生成し，SA 提案ペイロードを利用して暗号化アルゴリズムを折衝することで，IKE_SA を作成する．

・IKE_AUTH 交換

IKE_AUTH 交換では，互いの IPsec Gateway の認証と CHILD_SA の折衝をおこなう．IKE_AUTH 交換では，始動者は常に IKE ヘッダ，始動者の ID ペイロード，認証ペイロード，CHILD_SA のための SA 提案ペイロード，トラフィックセレクタペイロードを交換し，必要に応じて証明書ペイロード，証明書要求ペイロード，応答者の ID ペイロードを送信する．応答者は，常に IKE ヘッダ，始動者の ID ペイロード，認証ペイロード，CHILD_SA のための SA 提案ペイロード，トラフィックセレクタペイロードを交換し，必要に応じて証明書ペイロードを送信する．この際，IKE ヘッダ以外のペイロードは全て IKE_SA によって暗号化される．CHILD_SA の暗号化アルゴリズムは，IKE_SA と同様に SA 提案ペイロードを用いて折衝される．CHILD_SA で利用する鍵は，IKE_SA 折衝時に，IKE_SA で必要な鍵長よりも長い鍵を折衝しておいて利用する．

IKE_SA 更新トランザクション

CHILD_SA の新折衝と更新には，CREATE_CHILD_SA 交換を用いる．CREATE_CHILD_SA 交換のトランザクションを図 2.2 に示す．IKE_SA の更新には，IKE_SA_INIT 交換を用いる方法と CREATE_CHILD_SA 交換を用いる方法がある．racoon2 では，IKE_SA_INIT 交換と CREATE_CHILD_SA 交換を交互に用いて IKE_SA の更新をおこなう．以下に CREATE_CHILD_SA 交換について詳解する．

・CREATE_CHILD_SA 交換

CREATE_CHILD_SA 交換は，新しい SA を作成するのに使われる交換である．CREATE_CHILD_SA 交換では，IKE ヘッダ，SA 提案ペイロード，乱数ペイロードを交換し，必要に応じて通知ペイロード，Diffie-Hellman 鍵交換ペイロード，トラフィックセレクタペイロードを交換する．SA 提案ペイロードによって新しく作成する SA の暗号化アルゴリズムを折衝し，乱数ペイロードと CREATE_CHILD_SA 交換をおこなう IKE_SA が持っている乱数を利用して Diffie-Hellman 鍵共有を実行し，鍵を生成する．

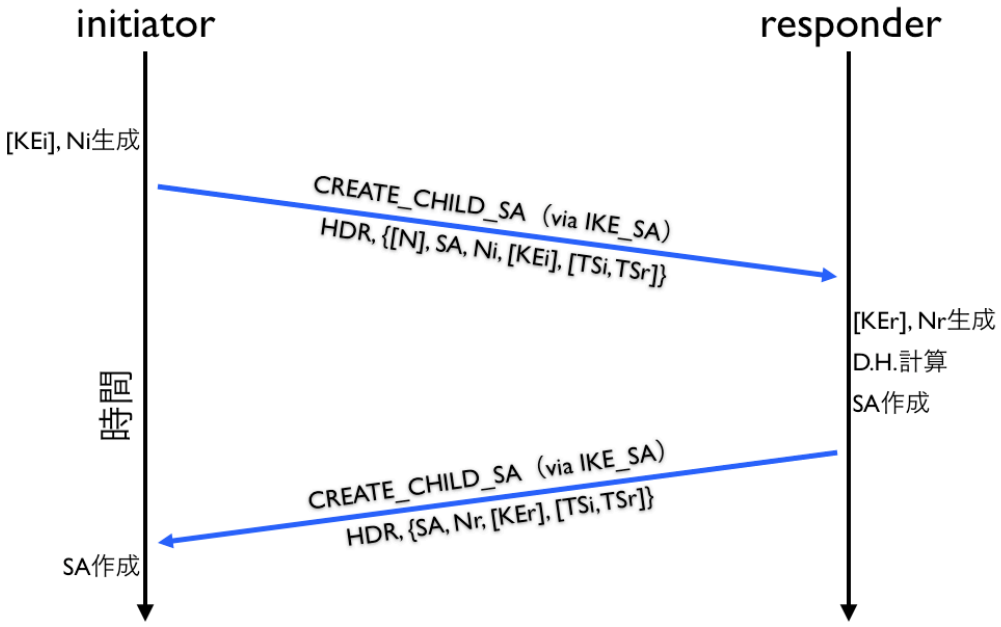


図 2.2: CREATE_CHILD_SA 交換トランザクション

2.3.3 エラー処理

IKE におけるエラー通知は，通知ペイロードと INFORMATIONAL 交換を利用しておこなう．SA の折衝中にエラーが起きた場合は，通知ペイロードを利用してエラーを通知し，暗号化通信中にエラーが起きた場合もしくは SA の有効期限切れによる消失などは，INFORMATIONAL 交換を利用して通知する．

2.4 量子コンピューティング

量子コンピューティングは，量子を素子として扱う新しいコンピューティングである．量子コンピューティングには，量子コンピュータ，量子鍵配送，量子ネットワークなどがある．

2.4.1 重ね合わせ

ある状態にある量子は定まった物理量を持っておらず，量子は物理状態が確率的に決定する性質がある．量子コンピューティングにおいて論理値は量子の物理状態によって表される．論理値を表す量子の物理状態が確率的に決定するため，量子コンピューティングの論理値は確率を用いて表現される．このときの，確率的に 0 と 1 のどちらにもなりうる論理値の状態を重ね合わせと呼ぶ．重ね合わせの様子は，量子力学に基づいて，波動関数を用いた数式で表される．量子を観測すると，波動関数が収束して量子の状態が確定し，0 か 1 を一意に表す状態になる．このとき，重ね合わせは失われる．

2.4.2 量子ビット

現在のコンピューティングにおける bit にあたるものを量子コンピューティングでは量子ビット，もしくは qubit と呼ぶ．bit は一意の論理値を持つのに対し，重ね合わせを持つ qubit は確率を用いて表現される．このとき，重ね合わせ状態にある 1qubit はベクトルを用いて式 2.6 のように表される．

$$|\Psi\rangle = \frac{1}{\sqrt{2}}(\alpha\ |0\rangle + \beta\ |1\rangle)\ (\text{ただし}, |0\rangle = \begin{pmatrix}1\\0\end{pmatrix}, |1\rangle = \begin{pmatrix}0\\1\end{pmatrix}) \tag{2.6}$$

このとき，$|0\rangle$ の係数を二乗した値がこの qubit から 0 が検出される確率となり，$|1\rangle$ の係数を二乗した値が 1 が検出される確率となる．この $|\Phi\rangle$ で表される量子の集合は密度行列で表され，密度行列 2.7 のようになる．

$$\rho = \frac{1}{\sqrt{2}}(\alpha\ |0\rangle\langle 0| + \beta\ |1\rangle\langle 1|) \tag{2.7}$$

2.4.3 エンタングルメント

通常，複数の系が存在するとき，それぞれの系の状態を決定する確率はそれぞれ独立している [7]．しかしながら，量子においては複数の量子の状態を決定する確率が独立せず，従属している場合がある．この状態をエンタングルメントと呼び，エンタングルメントの状態にあることを，量子がエンタングルしていると言う．エンタングルメントは量子コンピューティングにも影響し，二値の状態決定に関与する．式 2.8 は，エンタングルしていない 2 量子の状態を表す式である．

$$|\Psi\rangle = \frac{1}{2}(|0_1 0_2\rangle + |0_1 1_2\rangle + |1_1 0_2\rangle + |1_1 1_2\rangle) \tag{2.8}$$

表 2.5: エンタングルしていないときの量子状態と確率

二つの qubit の状態	この状態をとる確率
$\mid 00\rangle$	25%
$\mid 01\rangle$	25%
$\mid 10\rangle$	25%
$\mid 11\rangle$	25%

表 2.6: エンタングルしているときの量子状態と確率（例）

二つの qubit 状態	この状態をとる確率
$\mid 00\rangle$	50%
$\mid 01\rangle$	0%
$\mid 10\rangle$	0%
$\mid 11\rangle$	50%

この際，表 2.5 のように，qubit1 と qubit2 がそれぞれ 0 と 0 を取る確率は 25%，0 と 1 を取る確率は 25%，1 と 0 を取る確率は 25%，1 と 1 を取る確率は 25%である．この場合，qubit1 の状態に関わらず qubit2 の状態は 0 を取る確率が 50%かつ 1 を取る確率が 50%であり，逆も同様である．

一方，式 2.9 はエンタングルしている 2 量子の状態を表す式である．

$$\mid \Psi\rangle = \frac{1}{\sqrt{2}}(\alpha \mid 0_1 0_2\rangle + \beta \mid 1_1 1_2\rangle) \tag{2.9}$$

この式においては，確率的に取りうる状態が二つのみ存在する．このとき，qubit1 が 0 である場合 qubit2 は 0 を取り，qubit1 が 1 である場合 qubit2 は 1 を取る．表 2.5 と同様に表すと表 2.6 のようになる．表 2.6 を見れば明らかなように，エンタングルしている状態ではそれぞれの状態を取る確率が偏っているため，片方の qubit の二値が決定すればもう一方の qubit の二値も決定する．

エンタングルしている二つの量子を Bell ペアと呼ぶ．エンタングルメントの代表的な状態に 4 つの Bell 状態がある．各 Bell 状態は以下の式で表される．

$$\mid \Psi^+\rangle_{12} = \frac{(\mid 0\rangle_1 \mid 1\rangle_2 + \mid 1\rangle_1 \mid 0\rangle_2)}{\sqrt{2}} \tag{2.10}$$

$$\mid \Psi^-\rangle_{12} = \frac{(\mid 0\rangle_1 \mid 1\rangle_2 - \mid 1\rangle_1 \mid 0\rangle_2)}{\sqrt{2}} \tag{2.11}$$

$$|\Phi^+\rangle_{12} = \frac{(|0\rangle_1 |0\rangle_2 + |1\rangle_1 |1\rangle_2)}{\sqrt{2}} \tag{2.12}$$

$$|\Phi^-\rangle_{12} = \frac{(|0\rangle_1 |0\rangle_2 - |1\rangle_1 |1\rangle_2)}{\sqrt{2}} \tag{2.13}$$

Bell 状態は量子テレポーテーションにも利用される重要な状態である.

2.4.4 Bell 測定

Bell 測定は，エンタングルしている二つの量子が式 2.10，式 2.11，式 2.12，式 2.13 で表される四つの Bell 状態のうちどれを取っているかを観測する測定である．Bell 測定は観測された量子それぞれの状態を一意に確定はしない．そのため，量子状態は破壊されず，量子計算を続けることができる.

2.4.5 量子テレポーテーション

量子テレポーテーションは量子状態を転送する技術である [8]．量子テレポーテーションには送信者と受信者の間に Bell ペアが必要である．Bell ペアの送信者が持つ量子を量子 A とし，受信者が持つ量子を量子 B とする．このとき量子 A と送信したい量子状態を持つ量子にある特殊な操作を加えて Bell 測定をおこなうと，量子 A と量子 B がエンタングルしている事により送信したい量子状態と量子 B の量子状態の差異が求まる．この差異を量子操作によって量子 B に反映する事で量子状態の転送をおこなう.

qubit である量子を直接送信者から受信者に飛ばすと，量子状態が壊れて情報を失う恐れがある．予めエンタングルメントを作った後に量子テレポーテーションをおこなうことで，この危険性がなくなる.

2.4.6 エンタングルメントスワッピング

エンタングルメントの連結の概念図を図 2.3 に表す.

$$|\Psi^+\rangle_{\alpha\beta} = \frac{(|0\rangle_\alpha |1\rangle_\beta + |1\rangle_\alpha |0\rangle_\beta)}{\sqrt{2}} \tag{2.14}$$

$$|\Psi^+\rangle_{\beta\gamma} = \frac{(|0\rangle_\beta |1\rangle_\gamma + |1\rangle_\beta |0\rangle_\gamma)}{\sqrt{2}} \tag{2.15}$$

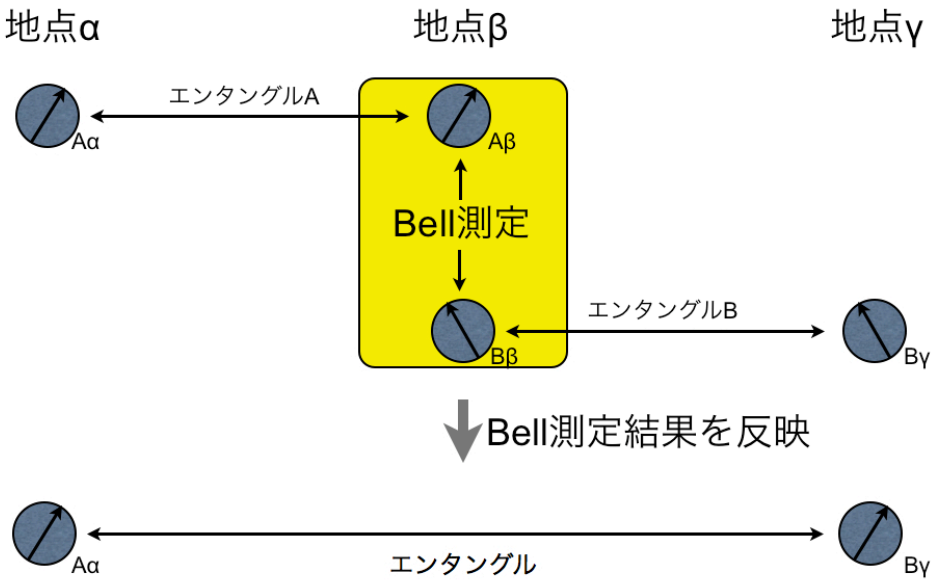


図 2.3: エンタングルメントスワッピングの概念

三地点 α，β，γ が存在して，α-β 間に式 2.14 で表される Bell ペア A が存在し，β-間 γ に式 2.15 で表されるに Bell ペア B が存在するとき，地点 β において A_β と B_β について Bell 測定をおこなう．このとき，地点 α にある A_α と地点 γ にある B_γ の二量子の状態の差異が判明する．

$$|\Psi\rangle = \frac{1}{\sqrt{2}}(|00\rangle + |11\rangle) \tag{2.16}$$

A_β と B_β 式が 2.16 で表される関係にあるとき，A_α と B_γ 式は 2.17 で表される状態にあることがわかる．

$$|\Phi^+\rangle_{\alpha\gamma} = \frac{(|0\rangle_\alpha |0\rangle_\gamma + |1\rangle_\alpha |1\rangle_\gamma)}{\sqrt{2}} \tag{2.17}$$

また，A_β と B_β 式が 2.18 で表される関係にあるとき，A_α と B_γ 式は 2.19 で表される状態にあることがわかる．

$$|\Psi\rangle = \frac{1}{\sqrt{2}}(|01\rangle + |10\rangle) \tag{2.18}$$

$$|\Psi^+\rangle_{\alpha\gamma} = \frac{(|0\rangle_\alpha |1\rangle_\gamma + |1\rangle_\alpha |0\rangle_\gamma)}{\sqrt{2}} \tag{2.19}$$

このように，A_α と B_γ はエンタングルしている状態になる．この，二つのエンタングルメントを連結して一つの長距離のエンタングルを作成する技術をエンタングルメントスワッピングと呼ぶ．

2.4.7 エンタングルメント純化

エンタングルメントには質が存在する．量子は時間経過や外部要因により状態が少しずつ変化していく．この状態変化と共に，エンタングルメントの質は劣化する．エンタングルメントの純化（精製，purification）は，Bell ペア群から質の悪いエンタングルメントをしている Bell ペアを排除して，質の良いエンタングルメントをしている Bell ペアを残すものである [9][10]．式 2.20 のような状態の量子を大量に含む，密度行列 2.21 で表される状態が最初に存在する．

$$| \Psi\rangle = g \mid 0\rangle + (1-g) \mid 1\rangle \tag{2.20}$$

$$\rho = g \mid 0\rangle\langle 0 \mid + (1-g) \mid 1\rangle\langle 1 \mid \tag{2.21}$$

このとき密度行列 2.21 中の量子を二つ選んで CNOT をおこなうと，それらの状態は密度行列 2.22 のようになる．コントロールビットに A，ターゲットビットに C の添字をつける．

$$\rho_{AC} = (g^2 \mid 0\rangle_A\langle 0 \mid + (1-g)^2 \mid 1\rangle_A\langle 1 \mid) \otimes \mid 0\rangle_C\langle 0 \mid + g(1-g)(\mid 0\rangle_A\langle 0 \mid + \mid 1\rangle_A\langle 1 \mid) \otimes \mid 1\rangle_C\langle 1 \mid \tag{2.22}$$

この状態で C を観測して 0 であった場合の A は密度行列 2.23 で表される状態となる．

$$\rho'_A = g^2 \mid 0\rangle_A\langle 0 \mid + (1-g)^2 \mid 1\rangle_A\langle 1 \mid \tag{2.23}$$

しかしながらこの状態では g^2 と $(1-g)^2$ の和が 1 とは限らないため，正規化して密度行列 2.24 の状態になる．

$$\rho''_A = g' \mid 0\rangle_A\langle 0 \mid + (1-g') \mid 1\rangle_A\langle 1 \mid (\text{ただし}, g' = \frac{g^2}{g^2 + (1-g)^2}) \tag{2.24}$$

この A のみを集めると，$g > 0.5$ であれば，$\mid 0\rangle\langle 0 \mid$ となっている量子の割合は元の量子の集合よりも増加する．これがエンタングルメント純化の原理である [11][12]．エンタングルメント純化を繰り返すと，質の低い大量のエンタングルメントからやや質の良い複数のエンタングルメントを精製でき，精製したエンタングルメントでさらにエンタングルメント純化をおこなうことで高品質のエンタングルメントを精製することが出来る．

2.4.8 量子リピータ

量子リピータは，エンタングルメントの生成，連結，純化する機能を持つ装置である．現在研究されている量子リピータは，光子を利用した装置である．光ファイバーを介して光子を転送してエンタングルメントを生成する [13][14][15]．[16]

2.4.9 量子ネットワーク

量子ネットワークは量子情報を転送するためのネットワークである．量子ネットワークは量子リピータをノードとして，光ファイバーでパスを構成することにより実現する．複数の量子機器が量子リピータと接続されているとき，量子リピータは選択的にエンタングルメントスワッピングをおこなう機器を決定することにより，任意の二機器の間でエンタングルメントを生成できる．これを連鎖的におこなって量子ネットワーク上の任意の二カ所でエンタングルメントを作成し，量子テレポーテーションをおこなうことで量子情報のルーティングが可能となる．量子ネットワークを用いると，現状途中にどんな機器も存在しない一本の光ファイバーで接続されていなければならない量子鍵配送も，量子ネットワークに接続されている任意の量子鍵配送装置と量子鍵配送をおこなう事が可能になる [17].

2.5 量子非クローン定理

任意の量子状態の完全なクローンを作る事はできない．仮に量子状態のクローンが可能であるとしたとき，式 2.25 中のように，二つの量子状態 $|\Psi\rangle$ と $|s\rangle$ が存在するとき，ある変換 U によって $|t\rangle$ は $|\Psi\rangle$ に変換される．

$$U(|\Psi\rangle\otimes|t\rangle) = |\Psi\rangle\otimes|\Psi\rangle \tag{2.25}$$

同様に，$\Phi\rangle$ についても式 2.26 の変換が施せる．

$$U(|\Phi\rangle\otimes|t\rangle) = |\Phi\rangle\otimes|\Phi\rangle \tag{2.26}$$

式 2.25 と式 2.26 の両辺の内積をそれぞれ取ると，左辺は式 2.27 のようになる．

$$\langle t|\otimes\langle\Phi|U^{-1}U|\Psi\otimes|t\rangle = \langle\Phi|\Psi\rangle \tag{2.27}$$

また，右辺は式 2.28 のようになる．

$$\langle\Phi|\otimes\langle\Phi|\Psi\rangle\otimes|\Psi\rangle = \langle\Phi|\Psi\rangle^2 \tag{2.28}$$

よって，式 2.27 と式 2.28 より式 2.29 と式 2.30 が導かれる．

$$\langle \Phi \mid \Psi \rangle = \langle \Phi \mid \Psi \rangle^2 \tag{2.29}$$

$$\langle \Phi \mid \Psi \rangle = 1, 0 \tag{2.30}$$

よって，$\mid \Phi \rangle$ と $\mid \Psi \rangle$ が直交しているもしくは等しいときにのみ量子状態のクローンは可能となり，一般的な任意の量子状態のクローンは作成することは出来ない [18]．量子非クローン定理は第 2.6 節で述べる量子鍵配送の安全性の根拠となる．

2.6 量子鍵配送

量子鍵配送は，量子情報によって実現される新しい秘密乱数共有アルゴリズムである．量子鍵配送の特徴は，第三者に因る盗聴を確実に検知できる点である．量子は，観測するまで状態が確定しないかつ観測すると必ず状態が一意に確定する性質を持つ．これにより盗聴者が存在する場合，送受信した光子の量子状態が送信者のもとにあるときと受信者のもとにあるときで一致しなくなる．この不一致を確認することで，量子鍵配送は盗聴者の存在を確認する．

量子鍵配送は既に実現されており，今現在実装されている量子鍵配送装置は量子状態伝達素子に光子を用い，量子情報を載せた光子を光ファイバーの中を通して通信するものである．光子の量子状態を変更しないために，この光ファイバーにはアンプを入れる事はできない．また，既存のネットワーク機器を入れる事も出来ず，量子鍵配送のルーティングには専用の量子ネットワークを構成する必要がある．

光ファイバーを用いた量子鍵配送の実現方法は，量子コンピューティングの通信方法に基づき二種類考案されている．一つは上記の通り量子情報を載せた光子を直接送受信するもので，既に実現されている．もう一つはエンタングルメントと量子テレポーテーションを用いるもので，こちらは実現されていない．エンタングルメントと量子テレポーテーションを用いた量子鍵配送は，エンタングルメントスワッピングによるルーティングが可能であるため将来の量子鍵配送として期待されている．

量子鍵配送の代表的なアルゴリズムに BB84 がある．BB84 は，Bennett と Brassard によって最初に提案されたアルゴリズムである．BB84 のアルゴリズムを以下に説明する．BB84 では，２種類の偏光からなる基底セット２組を利用して鍵の共有をおこなう．基底セットは，0 度と 90 度の縦横セットと，45 度と 135 度の斜方セットを利用する．本小節では，縦横セットの 0 度偏光を 0，90 度偏光を 1 として扱い，斜方セットの 45 度偏光を

0，135 度偏光を 1 として扱う．このとき，縦横セットの基底と斜方セットの基底は直交しておらず 45 度の角度差で交わっているため，斜方セットの偏光を持つ光子を縦横セットの基底で観測すると $\frac{1}{2}$ の確率で 0 が検出され，$\frac{1}{2}$ の確率で 1 が検出される．同様に，縦横セットの偏光を持つ光子を斜方セットの基底を用いて観測すると $\frac{1}{2}$ の確率で 0 が検出され，$\frac{1}{2}$ の確率で 1 が検出される．これらを前提として，BB84 は 1 つ 1 つの光子を以下の手順で処理する．表 2.7 に，五つの qubit を用いて，以下で示す手順に対応する例を挙げる．

1. 送信者側でランダムに 0 もしくは 1 を選択する
2. 送信者側でランダムに基底を選択する
3. 送信者側で，選択した基底にのっとり光子に偏光をかける
4. 送信者から受信者に光子を送信する
5. 受信者側でランダムに基底を選択する
6. 受信者側で，選択した基底にのっとり光子を観測する

表 2.7: BB84 の実行例（記号は表 2.8 参照）

手順番号	表示物	qubit#1	qubit#2	qubit#3	qubit#4	qubit#5
1	選択した数字	0	1	1	0	0
2	送信者の基底セット	＋	＋	×	×	×
3	偏光方向	−	|	＼	／	／
4	送信	送信	送信	送信	送信	送信
5	受信者の基底セット	×	＋	×	＋	×
6	受信者が観測した数字	1	1	1	0	0
7	基底の異同	異	同	同	異	同
8	qubit の処理	破棄	1	1	破棄	0

表 2.8: 表 2.7 中の記号

＋	基底：縦横セット	×	基底：斜方セット
−	0 度偏光：縦横セットの 0	／	45 度変更：斜方セットの 0
|	90 度変更：縦横セットの 1	＼	135 度変更：斜方セットの 1

7. 古典コネクションを用いて，送信者側で選択した基底と受信者側で選択した基底を確認し合う

8. 異なる基底を利用していた場合，情報を破棄する

これらの手順を繰り返したとき，送信者と受信者の間には同じ鍵列が生成されているはずである．この後，生成された鍵列の一部を照らし合わせることにより盗聴者の有無を確認する．盗聴者は，受信者と同様に送信者の選択した基底を知った上で光子の観測をおこなえない．そのため，送信者が利用しなかった基底セットを用いて観測し，光子が実際に持っている量子状態とは異なる量子状態を得る可能性がある．盗聴者の観測によって通信中の光子の量子状態は破壊されるため盗聴者は受信者に送信する光子を用意する必要があるが，送信者が偏光した量子状態とは異なる量子状態を盗聴者が観測していた場合，盗聴者が生成する光子の偏光は送信者による偏光とは異なるものになる．量子状態の非クローン定理により，盗聴者は送信者が送った量子状態と同じ量子状態を持つ光子を受信者に送信できないからである．この偏光が異なる光子を受信者が受信して観測した場合，使用した基底セットが送信者と受信者で一致していても，送信者と受信者は $\frac{1}{2}$ の確率で異なる値を得る．盗聴者が送信者と同じ基底セットを利用する可能性も考慮に入れると，盗聴者がいた場合に送信者と受信者が異なる値を得る可能性は $\frac{1}{4}$ である．これにより，鍵列のうち n ビットを照らし合わせることにより，$1-(\frac{3}{4})^n$ の確率で盗聴者の有無を検出できる．

BB84 は，2009 年 2 月現在唯一実現している量子鍵配送アルゴリズムである．量子鍵配送のスループットは，現在のところ NTT Communications による 20km 間 10Mb/s が最速である．

2.7 量子鍵配送と IPsec

2.7.1 Quantum cryptography in practice

量子鍵配送と IPsec を組み合わせた最初の論文である．この研究では，量子鍵配送の実用実験のために IKE（IKEv1）を改変して IPsec をおこなった．この研究は以下の二点を欠いている．

- プロトコルの考察と標準化
- IPv6 対応

この研究ではプロトコルの標準化はおこなわれず，更なるセキュリティについての考察と量子鍵配送の性能向上が必要であると結論づけられている．また，この研究の実装は IPv6

に対応していない．本研究では，この研究でおこなわれていない量子鍵配送利用プロトコルの標準化を目標とし，IPv6 に対応する実装をおこなっている [19].

2.7.2 Quantum Key Distribution (QKD) and Commodity Security Protocols: Introduction and Integration

この論文では量子鍵配送と IPsec，TLS の概要がまとめられ，IPsec と TLS での量子鍵配送の利用可能性について述べられている．この論文では二点の結論が導かれている．一点目は，量子鍵配送は鍵の安全性向上に利用可能である点である．二点目は，現在利用されている認証や通信の完全性を確保するためのセキュリティプロトコルは量子鍵配送のアルゴリズム実施にも利用可能である点である．この論文では量子鍵配送の利用方針が述べられるのみで，プロトコル設計や実装はおこなわれていない [20].

2.7.3 Quantum cryptography based on Bell's theorem

この論文ではエンタングルメントを利用した量子鍵配送アルゴリズムが提案された．量子ネットワークはネットワーク上の任意の二地点間にエンタングルメントを生成する．そのため，量子ネットワークを介した量子鍵配送にはこのアルゴリズムの利用が適している [17].

第3章　IPsec with QKD

3.1　問題定義

第 2.1 節で述べたとおり，Diffie-Hellman 鍵共有は，効率的な因数分解手法が存在しないことを安全性の根拠としている．これは，効率的な因数分解が Diffie-Hellman 鍵共有の弱点であることを意味している．Diffie-Hellman 鍵共有を使用し続けることには二つの問題がある．一つは，効率的な因数分解が可能となった際に利用できなくなる点である．もう一つは，パケットが保存されていた場合効率的な因数分解が可能となる以前におこなわれた通信の鍵も解読され，情報が漏洩する点である．これは暗号化通信がおこなわれた年代によらず，因数分解が可能となった時点で過去におこなった全ての暗号化通信の安全性が損なわれることになる．

効率的な因数分解を可能とする確実な脅威が量子コンピュータである．量子コンピュータは現在開発途上であるが，将来開発されたとき効率的な因数分解が可能となり Diffie-Hellman 鍵共有は無力化する．

以上のように，安全性が数学的に保証されていない鍵共有の利用には問題がある．しかしながら，事前準備の要らない数学的に安全性が証明された鍵共有はこれまで実用化されていなかった．そのため，安全性が証明された鍵共有のための暗号化通信システムは未だ存在しない．本研究では安全性が証明された鍵共有を利用する暗号化通信がインターネット上で確立されていない点を問題とし，これに対処する．本研究で扱う IKE のバージョンは IKEv2 とする．

3.2　解決手法

量子鍵配送は数学により恒久的に安全性が証明された鍵共有である．本研究ではこの量子鍵配送を採用する．また，この量子鍵配送を用いる暗号化通信として幅広い暗号化プロトコルを利用できる IPsec を採用する．量子鍵配送を利用する IPsec の全体ネットワーク

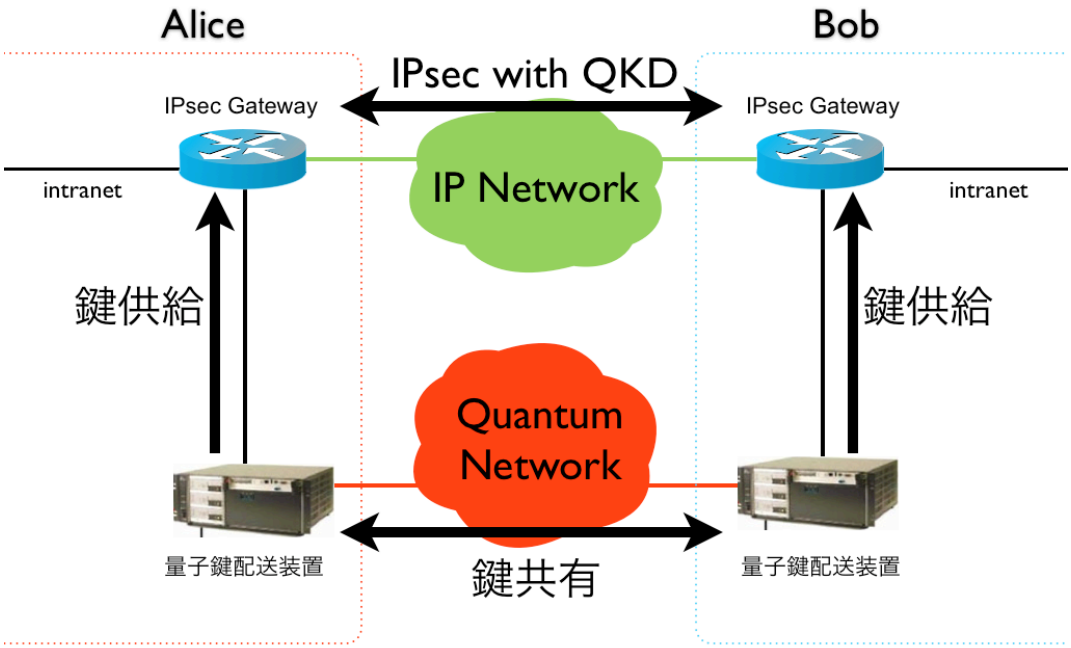


図 3.1: IPsec with QKD ネットワーク

図は図 3.1 のようになる．IPsec を行う二つの閉域網をそれぞれ Alice と Bob と呼ぶ．まず，Alice と Bob にそれぞれ一つずつ IPsec Gateway と量子鍵配送装置が存在し，ローカル接続されている．Alice の IPsec Gateway と Bob の IPsec Gateway は共に IP ネットワークに繋がっており，IP ネットワークを介して通信可能である．Alice の量子鍵配送装置と Bob の量子鍵配送装置は量子ネットワークに繋がっており，量子ネットワークを介して通信可能である．

本研究では量子鍵配送を利用する IPsec を実現するために IKE を拡張する．上記のネットワークのもとでは，IKE の量子鍵配送拡張は以下に挙げる二点を達成することで実現する．

鍵の利用部分の折衝

量子鍵配送で連続的に生成され続ける乱数を一定サイズ毎に区切って識別子を付与し，識別子を折衝することで使用する鍵を決定する．

フォールバック方法の折衝とフォールバック運転開始時の通信

IPsec Gateway と非同期に乱数を生成する量子鍵配送が，IPsec Gateway からの要

表 3.1: IKE_SA_INIT 交換における交換ペイロードの比較（◯ は交換，× は交換なし）

ペイロード	通常の IKE	IKE for IPsec with QKD
IKE ヘッダ	◯	◯
SA 提案ペイロード	◯	◯
Diffie-Hellman 鍵交換ペイロード	◯	×
乱数ペイロード	◯	×
鍵識別子交換ペイロード	×	◯

求に対して鍵を供給できないときのためのフォールバック方法を構築する．また，フォールバック運転を開始する際の通信について規定する．

3.3 プロトコル設計

第 3.1 節で述べた二件の要件のために，鍵の識別子とフォールバック方法を折衝するためのプロトコルを定義する．IKE の拡張として運用するため，第 2.3 節で整理したプロトコルをもとに設計した．

3.3.1 トランザクション

本小節では，第 2.3 節で整理された IKE トランザクションを量子鍵配送に対応するトランザクションに改変する．

IKE 開始トランザクション

量子鍵配送に対応した IKE 開始トランザクションを図 3.2 に示す．各交換の改変部分を以下に解説する．

・IKE_SA_INIT 交換

IKE_SA_INIT 交換では，新たに鍵識別子交換ペイロードを交換する．量子鍵配送利用時には Diffie-Hellman 鍵交換ペイロードと乱数ペイロードは必要ないため，これらを交換しない．よって，IKE ヘッダ，SA 提案ペイロード，鍵識別子交換ペイロードが交換される．表 3.1 に交換するペイロードの比較を示す．IKE_SA_INIT 交換においては各ペイロードは暗号化されないため，DIFFIE-HELLMAN フォールバック

を利用できない．IKE_SA 初作成の場合には CONTINUE フォールバックもおこなえないため，WAIT_QKD フォールバックのみが可能となる．

- **IKE_AUTH 交換**

  IKE_AUTH 交換では，暗号化してフォールバック方法交換ペイロードを交換する．IKE_SA_INIT 交換と異なり，IKE_AUTH 交換では通常の IKE で交換されるペイロードも全て交換される．よって，常に IKE ヘッダと送信者の ID ペイロード，認証ペイロード，CHILD_SA のための SA 提案ペイロード，トラフィックセレクタペイロード，フォールバック方法交換ペイロードを交換し，必要に応じて証明書ペイロードと証明書要求ペイロード，受信者の ID ペイロードを交換する．表 3.2 に交換するペイロードの比較を示す．

IKE_SA 更新トランザクション

量子鍵配送用の CREATE_CHILD_SA 交換には，通常運転時とフォールバック運転時の二つの動作がある．通常運転時の IKE_SA 更新トランザクションを図 3.3 に示す．フォー

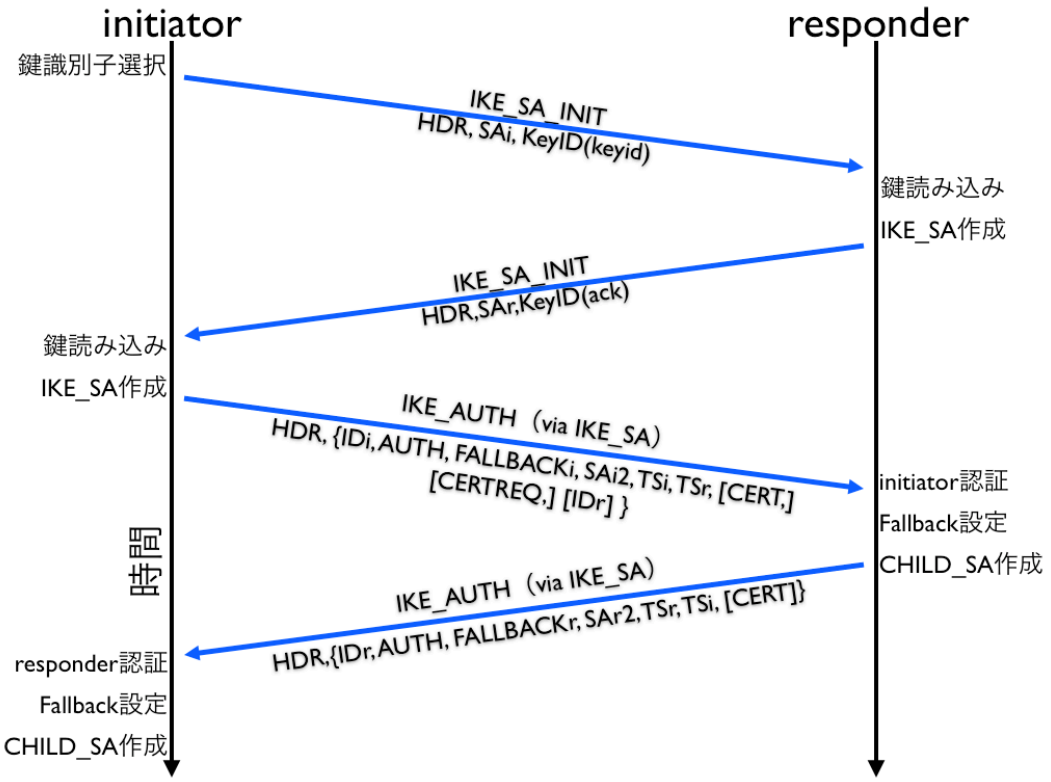


図 3.2: 量子鍵配送対応 IKE 開始トランザクション

表 3.2: IKE_AUTH 交換における交換ペイロードの比較（網掛けは暗号化，△ は選択）

ペイロード	通常の IKE	IKE for IPsec with QKD
IKE ヘッダ	○	○
送信者の ID ペイロード	○	○
認証ペイロード	○	○
CHILD_SA のための SA ペイロード	○	○
トラフィックセレクタペイロード	○	○
証明書ペイロード	△	△
証明書要求ペイロード	△	△
受信者の ID ペイロード	△	△
フォールバック方法交換ペイロード	○	○

ルバック運転時の CREATE_CHILD_SA 交換を図 3.4 に示す．

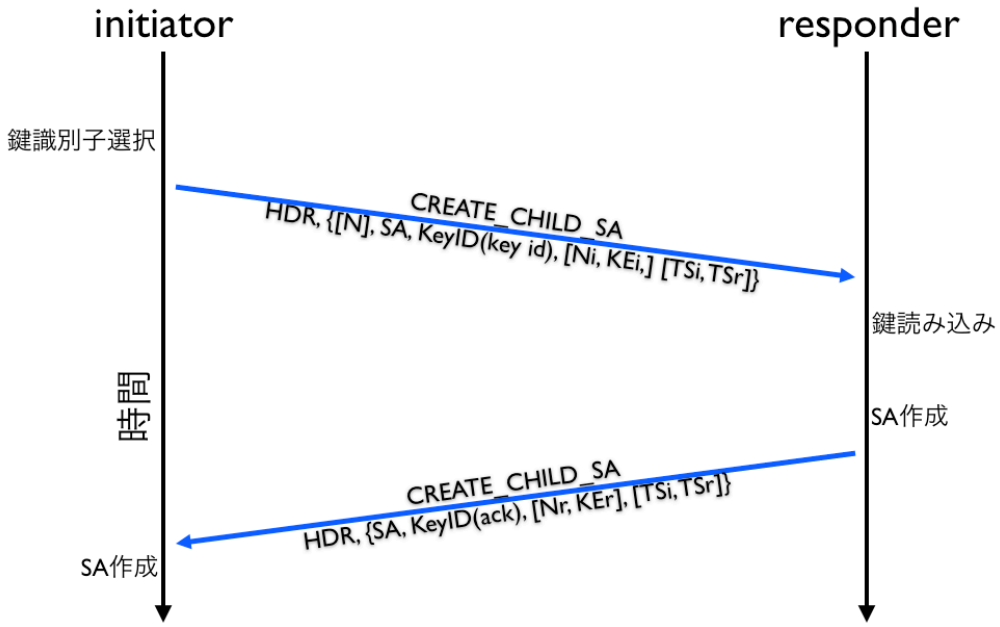


図 3.3: 通常運転時の量子鍵配送対応 IKE_SA 更新トランザクション

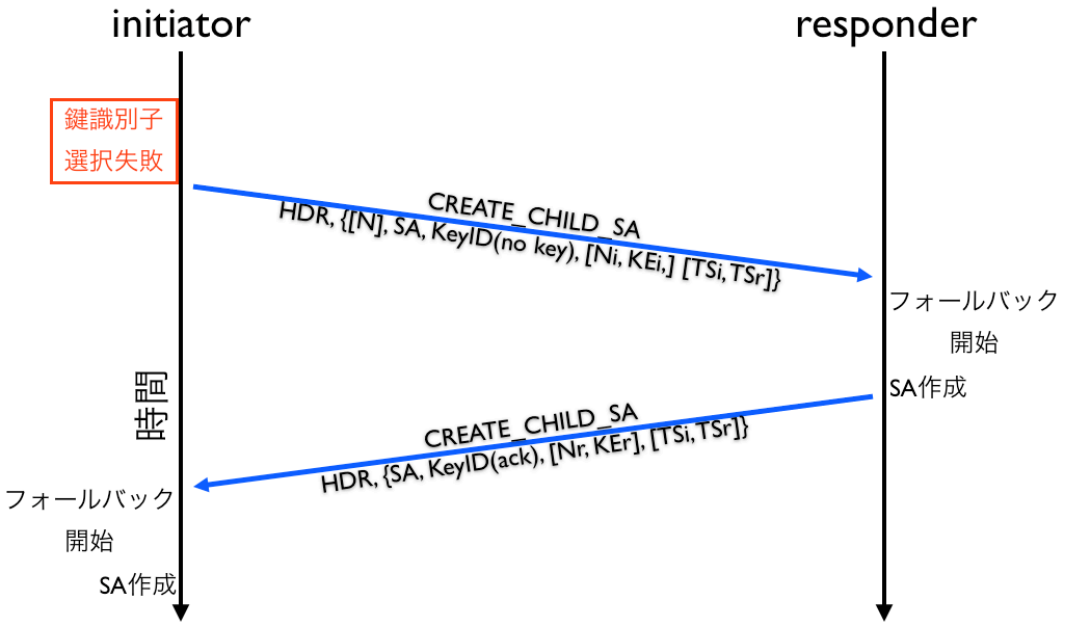


図 3.4: フォールバック運転時の量子鍵配送対応 IKE_SA 更新トランザクション

- **CREATE_CHILD_SA 交換**

  CREATE_CHILD_SA 交換では鍵識別子交換ペイロードを交換する．通常運転時は，IKE_SA_INIT 交換と同様に Diffie-Hellman 鍵共有ペイロードと乱数ペイロードを交換しない．よって，IKE ヘッダ，SA 提案ペイロード，鍵識別子交換ペイロードが交換される．フォールバック運転時は，選択したフォールバック方法によって交換ペイロードが変わる．WAIT_QKD 運転もしくは CONTINUE 運転の場合には通常運転時と同様に IKE ヘッダ，SA 提案ペイロード，鍵識別子交換ペイロードを交換する．一方，Diffie-Hellman 運転の場合には乱数ペイロードも交換し，必要に応じて Diffie-Hellman 鍵共有ペイロードも交換する．表 3.3 に交換するペイロードの比較を示す．WAIT_QKD，DIFFIE-HELLMAN，CONTINUE は三種類のフォールバック方法である．フォールバック方法については第 3.4.2 小節にて述べる．CREATE_CHILD_SA 交換では，SA 提案ペイロードと鍵識別子交換ペイロードは暗号化される．

表 3.3: CREATE_CHILD_SA 交換における交換ペイロードの比較（網掛けは暗号化）

	通常の IKE	IKE for IPsec with QKD	
		通常運転 WAIT_QKD CONTINUE	DIFFIE-HELLMAN
IKE ヘッダ	○	○	○
SA 交換ペイロード	○	○	○
Diffie-Hellman 鍵共有ペイロード	△	×	△
乱数ペイロード	○	×	○
鍵識別子交換 ペイロード	×	○	○

3.3.2 ペイロード

鍵識別子交換ペイロード

図 3.5 に鍵の識別子を折衝するためのペイロードを示す．このペイロードの一つ前に位置するペイロードの Next Payload フィールドは，鍵識別子交換ペイロードのペイロード番号を指さなければならない．このペイロードの最初の 32bit は，通常の IKEv2 共通フィールドである．図 3.5 中で”C” で表されている Critical bit は，該当ペイロードを何らかのエラーによって読み取れなかった場合に受信者が取る挙動を決定する．0 である場合にはエラーを無視し，1 である場合には処理を中断して送信者にエラーを報告する．鍵識別子の折衝失敗は鍵の共有失敗と同義の致命的エラーである．また，中間者攻撃によって安全性が損なわれる事を避ける必要がある．そのため，鍵識別子交換ペイロード中の Critical

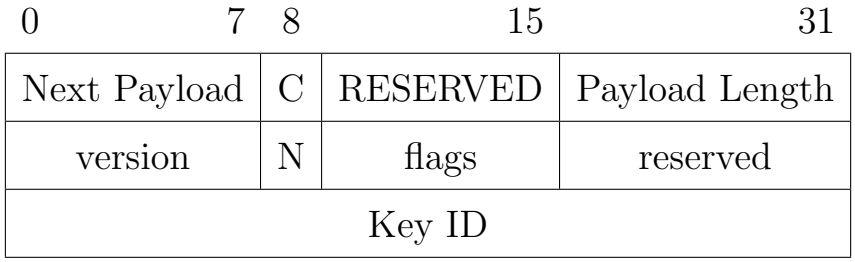


図 3.5: 鍵識別子交換ペイロード

bit は常に 1 でなければならない．このペイロード中で新たに定義された各フィールドの役割は以下の通りである．

- **version**
  鍵識別子交換ペイロードのバージョンを指定する．2010 年 2 月現在のバージョンは 1 である．

- **N**
  No key bit．利用できる鍵がなかった場合に 1 を入れて通信する．その場合，フォールバック運転を始める．

- **flags**
  フラッグ．現在フラッグは定義されていないため必ず 0 となる．

- **Key ID**
  該当 SA で利用する鍵の識別子を折衝する．

フォールバック方法交換ペイロード

図 3.6 にフォールバック方法を折衝するためのペイロードを示す．このペイロードの一つ前に位置するペイロードの Next Payload フィールドは，フォールバック方法交換ペイロードの番号を指さなければならない．このペイロードの最初の 32bit は，IKEv2 共通フォーマットである．このペイロード中で新たに定義された各フィールドの役割は以下の通りである．

- **version**
  フォールバック方法交換ペイロードのバージョンを指定する．現在のバージョンは 1 である．

- **flags**
  フラッグ．現在フラッグは定義されていないため必ず 0 となる．

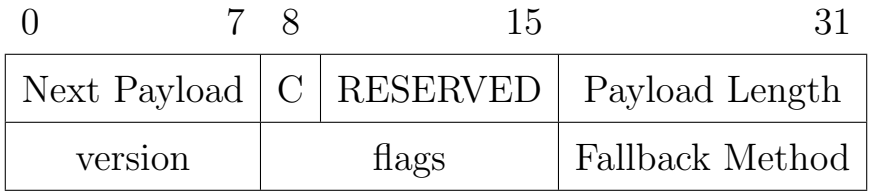


図 3.6: フォールバック方法交換ペイロード

表 3.4: フォールバック方法と識別番号

フォールバック方法	識別番号
WAIT_QKD	1
DIFFIE-HELLMAN	2
CONTINUE	3

・**Fallback Method**

フォールバック方法の折衝に利用する．

第 3.4.2 節で述べるフォールバック方法を表 3.4 のように整理する．識別番号は折衝に利用される．それぞれの方法の具体的な内容は 3.4.2 節で述べる．Fallback Method フィールド 16bit には基本的に 0 が設定される．そのうち，許可されたフォールバック方法の識別番号と同数桁目の bit を 1 にして利用する．フォールバック方法の漏洩を防ぐために，フォールバック方法交換ペイロードは暗号化されなければならない．

3.4 システム設計

本小節では本研究の要件をまとめ，要件達成に必要なフォールバック方法を設計する．

3.4.1 本研究の要件

本研究では，量子鍵配送で生成した鍵を IKE を介して IPsec で利用することを目的とする．システムの設計においては既存の IKE のシステムを出来る限り改変しないことが望ましい．またこのシステムは標準化を目指す初めての量子鍵配送利用システムである．他の暗号化通信システムを量子鍵配送に対応させる際に応用可能なシステムにすることが望ましい．本研究で構築するシステムの要件を表 3.5 にまとめる．量子鍵配送を利用する IPsec をおこなうために最低限必要な要件は以下の三つである．

要件 1:量子鍵配送装置から SAD までの機密性を確保した鍵供給

量子鍵配送の鍵共有自体の安全性は数学的に証明されている．しかし，安全な暗号化通信をおこなうには量子鍵配送を利用するのみならず量子鍵配送装置から IPsec Gateway 内の SAD までを安全に接続し，鍵の供給を安全におこなう必要がある．

表 3.5: 本研究の実装におけるシステム要件一覧

	要件	目的
必須要件	量子鍵配送装置から SAD までの機密性を確保した鍵供給	セキュリティ
	可用性の確保	セキュリティ
	外部で作成された鍵の本システムによる管理	資源管理
追加要件	量子鍵配送装置の実装によらない利用可能性の確保	実運用性
	複数の IPsec Gateway との IKE 管理	実運用性
	Diffie-Hellman 鍵共有を利用する IKE と量子鍵配送を利用する IKE の同時管理	実運用性
	IPv6 対応	将来性

要件 2:可用性の確保

可用性は，システムを必要なときに必要なように利用するための概念である．可用性が存在しないシステムは必要なときに利用できない恐れがあり，システムとして不完全である．本研究ではフォールバック運転の採用によりこの要件に対処する．

要件 3:外部で作成された鍵の本システムによる管理

本システムには，IKE 外部で生成された鍵を保持管理し，利用する機能が必要である．量子鍵配送装置の仕様によっては，量子鍵配送装置内で保持している鍵についても本システムで管理しなければならない可能性がある．IPsec Gateway のみならず，場合によっては量子鍵配送装置の資源管理も可能なシステムを構築する必要がある．

以上の三件を達成すれば，量子鍵配送を利用して IPsec をおこなうことが可能となる．要件 1 と要件 3 を達成することで，量子鍵配送で生成された鍵を暗号化へ利用可能になり，要件 2 を達成することで，利用可能な鍵がなくなっても Security Association は維持される．このように，上記の三件を達成すれば量子鍵配送を利用する IPsec の開始と開始後の維持は保証される．しかしながら，以上の三件のみでは実際のインターネットでの利用には堪えない．上記の要件のみでは，特定の量子鍵配送装置を持っている特定の相手とのみ量子鍵配送を利用した IPsec が可能となる．だが，多数の利用者が存在するインターネットでの利用を考える場合，不特定の相手と暗号化通信をいつでもおこなえる必要がある．よって，下記の三件を要件に加える．

要件 4:量子鍵配送装置の実装によらない利用可能性の確保

本研究では NEC 社製の QUICS を利用している．しかしながら量子鍵配送装置は複数のベンダーで開発されており，異なる鍵供給方式を持つ量子鍵配送装置の存在が予見される．様々な量子鍵配送装置と連動させて使用するため，本システムはインターオペラビリティを持つ設計をおこなう必要がある．

要件 5:複数の IPsec Gateway との IKE を管理

暗号化通信をおこなう必要のある相手が常に単一であるとは限らない．そのため，本システムは複数の相手との IKE を管理できる必要がある．

要件 6:Diffie-Hellman 鍵共有を利用する IKE と量子鍵配送を利用する IKE の同時管理

量子鍵配送装置を持つ相手と量子鍵配送を利用した IPsec をおこなうと同時に量子鍵配送装置を持たない相手との IPsec をおこなう場合，量子鍵配送を利用する IPsec のための IKE と Diffie-Hellman 鍵共有を利用する IKE を同時におこなう必要がある．よって，本システムは Diffie-Hellman 鍵共有を利用する IKE と量子鍵配送を利用する IKE の同時管理をおこなえる必要がある．

また，量子鍵配送を利用する IPsec の，IPv6 に対応したシステムは今だ開発されていない．そのため以下の一件も要件に加える．

要件 7:IPv6 対応

本システムは，量子鍵配送を利用する IPsec の実用化を目的としている．近い将来インターネットで主流になると考えられる IPv6 への対応は，今後のインターネットでの実用を考える上で重要である．

3.4.2 フォールバック設計

第 3.4.1 節で述べた二つ目の要件のため，有効なフォールバック方法を整理し構築する．本研究では，安全性を優先するフォールバック方法と無停止を優先するフォールバック方法を考案した．構築したフォールバック方法は以下の通りである．

- **WAIT_QKD**

  量子鍵配送が新しい鍵を生成するのを待つオプションである．これが選択されている場合，当該の SA の有効期限が切れた後 IPsec Gateway はパケットの暗号化と転送を中断する．

・**DIFFIE-HELLMAN**

Diffie-Hellman 鍵共有を利用して，通常の IKE と同様に鍵を折衝するオプションである．この際，折衝は量子鍵配送により安全性が担保されている IKE_SA によって暗号化されている必要がある．

・**CONTINUE**

新しい鍵を利用できなかった場合，当該の SA で利用中の鍵を引き続き鍵に利用する新しい SA を折衝するオプションである．

暗号化通信の安全性は WAIT_QKD，DIFFIE-HELLMAN，CONTINUE の順に高いと考えられる．しかし，WAIT_QKD は通信の安定性が保証できない．WAIT_QKD では，量子鍵配送装置が新しい鍵を供給するまで新しい SA が作成されない．この時，SPD に IPsec を適用するように登録されているパケットは，SAD を参照できないため処理を進めることができない．その結果，一時的に通信が不能となる．DIFFIE-HELLMAN を機能させるため，IKE の持つ IKE_SA_INIT を利用して IKE_SA を更新する機能は停止させなければならない．IKE_SA_INIT は暗号化されないため，Diffie-Hellman 鍵共有をおこなった場合鍵が漏洩する危険性がある．

第4章　実装

4.1　量子鍵配送装置 QUICS

本研究を実施するにあたり，NEC 社から量子鍵配送装置 QUICS をお借りした．QUICS の特徴を表 4.1 にまとめる．QUICS はコンピュータユニットと量子鍵配送ユニットから成

表 4.1: QUICS の特徴

ユニット数	2（コンピュータユニット，量子鍵配送ユニット）
量子鍵配送アルゴリズム	BB84
鍵保持方法	Unix File System にファイルで保存

る．コンピュータユニットは Unix システムを搭載し，量子鍵配送の実行を管理する．また，生成した乱数を一定サイズ毎に区切り，ファイルとして Unix File System 内に順次保存していく．本研究と raQoon2 の実装は，QUICS の利用を想定しておこなう [21]．量子鍵配送を用いるには，量子鍵配送装置は通信相手の量子鍵配送装置の認証を行わなければならない．QUICS は認証をおこなわないが，本研究においては量子鍵配送装置は自律的に認証を行うものとする．

4.2　実装環境

本研究では，量子鍵配送装置で共有される秘密乱数は全て IPsec Gateway で利用されることとする．また，量子鍵配送装置と IPsec Gateway の接続に NFS を用いる．これにより，IPsec Gateway で動作する raQoon2 は量子鍵配送装置の保持する鍵を直接操作できる．

本研究の実装環境を表 4.2 に一覧する．QUICS を借用できる期間は限られていたため，本研究では QUICS を模したエミュレータを作成し利用した．このエミュレータは鍵を模した疑似乱数を生成し，鍵プールディレクトリを模した二つのディレクトリにそれぞれ

表 4.2: 実装環境

	種別
OS	FreeBSD 7.2-RELEASE amd64
gcc	version 4.2.1
racoon2	racoon2-20090327c
量子鍵配送装置	QUICS エミュレータ

ファイルとして順次保存していくものである．二つの IPsec Gateway は，この二つのディレクトリをそれぞれの量子鍵配送装置の鍵プールとして利用する．

4.3 設計要件

本実装に必要な機能以下にまとめる．まず，量子鍵配送で生成した鍵を IPsec で利用するには，最低限以下の機能が必要である．

- 機能 1: 鍵プールから鍵を取得し，SA パラメータに登録する機能
- 機能 2:使用する鍵を折衝する機能

以上の機能を達成すれば，始動者が鍵の識別子を鍵プールから取得して応答者と折衝し，その後に鍵を取得して SA パラメータに適用することが可能になる．フォールバック機能を実装するためには以下の機能が必要である．

- 機能 3:フォールバック方法を折衝する機能
- 機能 4:フォールバック方法によって選択的にフォールバック運転をおこなう機能

また，第 3.4.1 小節で挙げたシステム要件より，要件 5 と要件 6 を達成するため，以下の三つの機能が必要である．

- 機能 5:通信相手別に量子鍵配送の鍵を管理する機能
- 機能 6:複数の相手と IKE を実行する機能
- 機能 7:量子鍵配送を用いた IKE による IPsec と通常の IKE を用いた IPsec を同時におこなう機能

また，同システム要件中の要件 4 を達成するため以下の機能が必要である．

- 機能 8:実装の異なる量子鍵配送を利用するためのインターオペラビリティの確保

4.4 raQoon2 実装

4.4.1 モジュール図

raQoon2 は racoon2 を元に設計されるため，基本的に racoon2 のモジュール群とその役割を引き継ぐ．第 4.3 節で挙げた要件を踏まえて，全体モジュールは図 4.1 のようになる．量子鍵配送装置には，量子鍵配送をおこなうユニットと生成した鍵を保存する鍵プールがある．raQoon2 には量子鍵配送装置と接続して鍵を管理するためのインタフェースモジュールが存在する．外部鍵コントロールモジュールが鍵管理インタフェースモジュールに要求を出し，鍵管理インタフェースモジュールが量子鍵配送装置と通信することで raQoon2 は鍵の取得や削除などの鍵管理をおこなう．本研究では IPsec Gateway と量子鍵配送装置の接続に NFS を利用し，量子鍵配送装置内の鍵プールはディレクトリである．

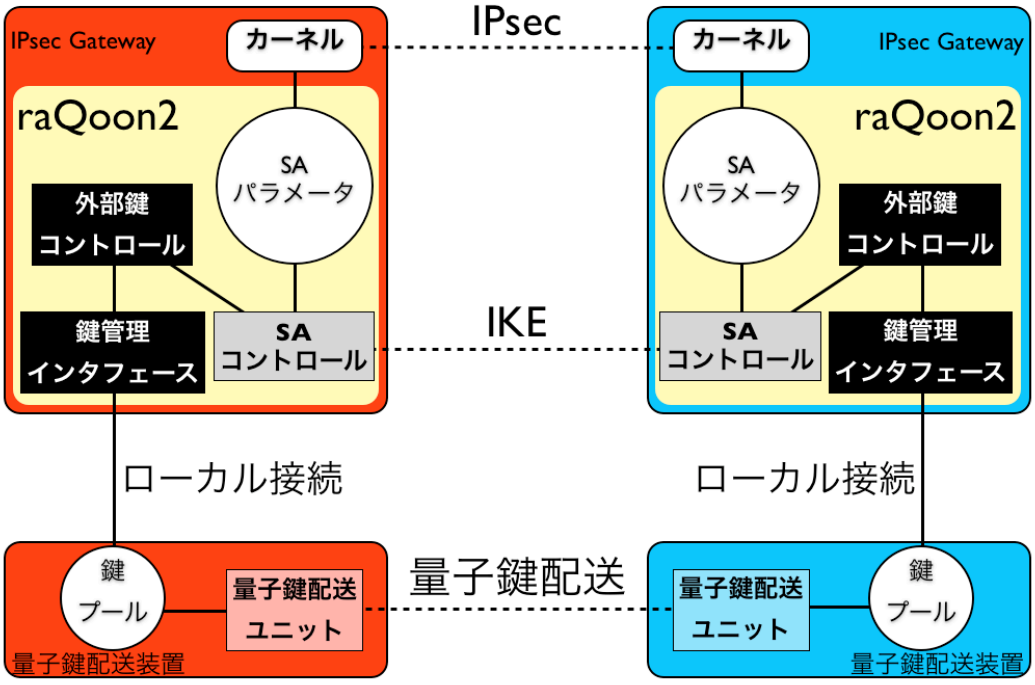


図 4.1: raQoon2 全体モジュール図（黒塗りは本研究で作成した部分，灰色は本研究で追加実装した部分）

4.4.2 機能の達成

raQoon2を実装するにあたり，第4.3小節で挙げた機能を達成するために新たに定義した関数をqkd.cにまとめた．qkd.cで定義した関数を表4.3に一覧する．機能1を達成するために鍵プールから鍵を読み込む関数qkd_fetch_key()，SAパラメータに鍵を必要な長さに合わせて登録する関数qkd_compute_keys()，鍵プールから鍵を削除する関数qkd_rm_keyfile()を定義した．機能2を達成するために鍵プールから鍵識別子を取得する関数qkd_choosekey()，鍵識別子交換ペイロードを作成する関数qkd_makepaylkeyid()，鍵識別子をSAパラメータに登録する関数qkd_set_keyid()を定義した．機能3を達成するために，フォールバッ

表 4.3: qkd.cに定義した量子鍵配送を利用するための関数一覧

モジュール	関数名	役割	達成機能
外部鍵コントロール	qkd_compute_keys()	SAパラメータへの鍵の設定	機能1
	qkd_fallback_execute_continue()	CONTINUEフォールバックの実行	機能4
鍵管理インタフェース	qkd_rm_keyfile()	使用した鍵の，ストレージからの削除	機能1
	qkd_fetch_key()	鍵の読み込み	機能1
	qkd_choosekey()	使用する鍵識別子の決定	機能2
SAコントロール	qkd_set_fallback()	フォールバック方法のSAパラメータへの設定	機能3
	qkd_merge_fallback_proposal()	フォールバック方法の折衝	機能3
	qkd_set_keyid()	鍵識別子のSAパラメータへの設定	機能2
その他	qkddev_init()	量子鍵配送装置毎の鍵管理インタフェースの差し替え	機能8
	qkd_makepaylkeyid()	鍵識別子交換ペイロードの作成	機能2
	qkd_makepaylfallback()	フォールバック方法交換ペイロードの作成	機能3

ク方法交換ペイロードを作成する関数 qkd_makepaylfallback()，お互いのフォールバック方法を比べてフォールバック方法を決定する関数 qkd_merge_fallback_proposal()，フォールバック方法を SA パラメータに登録する関数 qkd_set_fallback() を定義した．機能 4 の達成には，まず CONTINUE フォールバックのために古い SA から新しい SA に鍵をコピーする関数 qkd_fallback_execute_continue() を定義した．また，ikev2_rekey.c 内の関数を改変して，WAIT_QKD フォールバックでは SA の折衝を強制終了し，DIFFIE-HELLMAN フォールバックでは通常の IKE 交換をおこなうようにした．機能 5，機能 6，機能 7 は，SA 毎に鍵プールを管理することと，SA パラメータに量子鍵配送利用フラッグを設けることで達成した．機能 8 の達成には，qkddev_init() 関数を定義して，コンフィグで指定した量子鍵配送装置に対応する鍵管理インタフェースを利用してシステムを稼働するようにした．

4.4.3 racoon2 改変部

racoon2 は表 4.4 の通り spmd，iked，kinkd の 3 つのモジュールから成り，専用ライブラリとして libracoon を持っている．spmd は Security Policy Management Daemon で，セキュリティポリシーを管理してカーネルに設定をおこなう．iked は Internet Key Exchange Daemon であり，IKE を実行し SA を管理する．kinkd は Kerberized Internet Negotiation of Keys Daemon であり，Kerberos を利用して鍵の折衝をおこない，また SA を管理するが，本研究には関係しない．libracoon は各デーモンにカーネルと通信する際のインタフェースとコンフィグファイルを読み込む際のインタフェースを提供する．racoon2 で IKE による IPsec をおこなうには spmd，iked，libracoon を利用する．raQoon2 を実装するにあたり，ディレクトリ iked 及び libracoon を構成するディレクトリ lib 以下のファイルを書き換

表 4.4: racoon2 を構成するモジュール群

モジュール	役割	本研究への関連
spmd	セキュリティポリシーの管理	利用
iked	IKE の実行と SA の管理	改変
kinkd	KINK の実行と SA の管理	なし
libracoon	通信インタフェースとコンフィグパースインタフェースのライブラリ	改変

表 4.5: racoon2 改変ファイル一覧

モジュール	ファイル名	本研究に関わる役割
iked	ikev2.h	ペイロード定義
	ikev2_impl.h	SA パラメータ定義
	isakmp_impl.h	折衝した暗号アルゴリズムの格納
	ikev2.c	iked マネジメント，IKE_SA 初折衝
	ikev2_rekey.c	IKE_SA 更新
	ike_conf.c	コンフィグ管理
	ikev2_auth.c	認証
	ike_sa.c	IKE_SA 管理
	ikev2_child.c	CHILD_SA 管理
libracoon	cfparse.y	yacc ファイル
	cfsetup.c	コンフィグ操作
	cfsetup.h	コンフィグディレクティブ定義
	cftoken.l	lex ファイル
	rc_type.h	コンフィグパラメータ定義

えた．改変したファイルを表 4.5 に一覧する．ikev2.h には IKE パケットのペイロードが定義されている．ikev2_impl.h には IKE_SA や CHILD_SA のパラメータが定義されており，isakmp_impl.h にはコンフィグパラメータを管理する構造体が定義されている．ikev2.c では，受信した IKE パケットの仕分けや，IKE_SA 初折衝の際に利用する関数が定義されている．ikev2_rekey.c には，IKE_SA を更新する際の関数が始動者，応答者毎に定義され，ike_conf.c と ike_auth.c にはそれぞれコンフィグ管理と認証をおこなう関数が定義されている．ike_sa.c と ikev2_child.c にはそれぞれ IKE_SA の管理をする関数と CHILD_SA の管理をする関数が定義されている．

ikev2.h

ikev2.h では，IKE で定義されているペイロードがされている．raQoon2 では，ikev2.h で新たに鍵識別子交換ペイロードとフォールバック方法交換ペイロードの定義をおこなう．

ikev2_impl.h

ikev2_impl.h では，SA パラメータを格納する構造体が定義されている．raQoon2 では，この構造体にフォールバック方法とフォールバック運転実行のフラッグを新た

に格納する．

isakmp_impl.h

isakmp_impl.h では，折衝された暗号化アルゴリズムを納める構造体が定義されている．raQoon2 では，この構造体に鍵識別子を新たに格納する．

ikev2.c

ikev2.c では，IKE_SA 初生成時に IKE_SA_INIT 交換と IKE_AUTH 交換をおこなう関数が定義されている．raQoon2 では，IKE_SA_INIT 交換で鍵識別子交換ペイロードを交換し，IKE_AUTH 交換でフォールバック方法交換ペイロードを交換する．

ikev2_rekey.c

ikev2_rekey.c には，IKE_SA を更新する CREATE_CHILD_SA 交換をおこなう関数が定義されている．raQoon2 では，IKE_SA を更新する CREATE_CHILD_SA 交換で鍵識別子交換ペイロードを交換する．また，フォールバック運転の際には鍵の折衝方法が変わる．フォールバック運転は始動者が鍵識別子取得に失敗した場合に開始する．応答者は受信した鍵識別子交換ペイロードを確認し，No Key bit が 1 となっていた場合フォールバック運転を開始する．WAIT_QKD フォールバックを用いる際には，CREATE_CHILD_SA 交換の後，新しい SA の設定はしない．現行の IKE_SA の有効期限が切れた後は，IKE_SA 初折衝処理により，IKE_SA の再折衝をおこなう．CONTINUE フォールバックを用いる場合には，古い IKE_SA のパラメータ構造体に格納されている暗号化鍵を新しい IKE_SA のパラメータ構造体にも暗号化鍵として登録する．DH フォールバックでは，通常の IKE がおこなう通りに Diffie-Hellman 鍵共有ペイロードと乱数ペイロードの処理を実行する．このため，DH フォールバックをおこなっている場合に限り通常運転時には交換しない Diffie-Hellman 鍵共有ペイロードと乱数ペイロードも交換する．

ike_conf.c

ike_conf.c には，コンフィグ処理の関数が定義されている．raQoon2 では，ike_conf.c にはコンフィグにフォールバック方法が設定されているか確認し，量子鍵配送装置利用設定の初期化をおこなう．

ikev2_auth.c

ikev2_auth.c では，認証関連の関数が定義されている．raQoon2 では，乱数ペイロードが存在しない事による問題が起こらないように変更を加えた．

ike_sa.c

racoon2 は IKE_SA の更新にあたり，IKE_SA_INIT 交換と IKE_AUTH 交換を用いて IKE_SA 初作成時と同様に新しい IKE_SA を作成する手法と CREATE_CHILD_SA 交換を用いて新しい IKE_SA を作成する手法を交互に利用する．DH フォールバックをおこなうためには IKE_SA を更新する通信は IKE_SA によって暗号化されていなければならないため，暗号化されない IKE_SA_INIT 交換は利用不可能である．raQoon2 では，常に CREATE_CHILD_SA 交換を利用して IKE_SA の更新をおこなうように ike_sa.c に変更を加えた．

ikev2_child.c

ikev2_child.c には，CHILD_SA の管理や折衝に関する関数が定義されており，IKE_SA を折衝するために交換された乱数を CHILD_SA のための Diffie-Hellman 鍵共有に転用する．IKE_SA_INIT 交換で乱数ペイロードを交換しないことによる影響が出ないようにするため，IKE_SA_INIT 交換で交換するはずだった乱数ペイロードで運ばれる乱数を参照する箇所に，認証用暗号化鍵を参照する改変を加えた．raQoon2 では，IKE_AUTH 交換は IKE_SA を初生成したときにのみおこなわれるため，認証用暗号化鍵は IKE_SA の初生成時にしか利用されない．

cfparse.y cfsetup.c cfsetup.h cftoken.l rc_type.h

lib 以下のファイルである cfparse.y cfsetup.c cfsetup.h cftoken.l rc_type.h では，量子鍵配送の利用に関連するコンフィグを設定可能にするための改変をおこなった．詳しくは第 4.4.5 小節内でコンフィグ方法と併せて述べる．

4.4.4 鍵管理インタフェースの実装

鍵管理インタフェースの三つの関数を，QUICS 用の実装と併せて以下に説明する．

- **qkd_choosekey()**

  本研究では，QUICS 用の qkd_choosekey() を qkd_choosekey_QUICS() として実装した．qkd_choosekey_QUICS() は NFS でマウントしている量子鍵配送装置の鍵プールディレクトリを走査して，使用可能な鍵が存在すればファイル名を識別子に変換して返り値として返し，存在しなければ SA パラメータのフォールバック運転フラッグに 1 を代入して返り値として 0 を返す関数である．図 4.2 に qkd_choosekey_QUICS() のソースコードを載せる．構造体”ike_sa” の中にある”path” には，コンフィグファイルから読み込んだ鍵プールディレクトリのパスが保持されている．/* A */で示

```
int qkd_choosekey_QUICS(struct ikev2_sa *ike_sa){
  ike_sa->qkd_fallback_state = 0;
  int keyid = 0;
  DIR *dir;
  struct dirent *dirp;
  char path[PATH_LENGTH];
  memcpy(path,ike_sa->qkd_key_dir->v,ike_sa->qkd_key_dir->l);
  path[ike_sa->qkd_key_dir->l] = '\0';
  if ((dir = opendir (path)) == NULL){
/* D */
    syslog(LOG_NOTICE,"qkd_choosekey_QUICS:failed opendir()\n");
    ike_sa->qkd_fallback_state = 1;
    return 0;
  }
/* A */
  while(dirp = readdir(dir)){
    if(strncmp(dirp->d_name,".",1) == 0 ||
           strncmp(dirp->d_name,"..",2) == 0)
      continue;
/* B */
    keyid = strtol(dirp->d_name,NULL,16);
    ike_sa->qkd_fallback_state = 0;
    closedir(dir);
    return keyid;
  }
/* C */
  ike_sa->qkd_fallback_state = 1;
  return 0;
}
```

図 4.2: qkd_choosekey_QUICS() の実装

される部分で利用可能な鍵を探し，/* B */の部分で鍵ファイルのファイル名を識別子として返している．/* C */の部分は鍵ファイルが存在せず，フォールバック運転を開始する場合にのみ処理される．qkd_choosekey() を入れ替えることで，異なる実装の鍵プールに対応することが出来る．qkd_choosekey() が構造体”ike_sa” の中にある”qkd_fallback_state” に 1 を代入し，返り値として 0 を返した場合 raQoon2 は

フォールバック運転を開始する．/* D */のように，鍵を保存するディレクトリへのアクセスに失敗した場合にも，利用可能な鍵が存在しなかったものとして 0 を返り値として返しフォールバック運転を開始する．

- **qkd_fetch_key()**

  本研究では，QUICS 用の qkd_fetch_key() を qkd_fetch_key_QUICS() として実装した．qkd_fetch_key_QUICS() は，鍵プールディレクトリから指定された鍵識別子を持つ鍵ファイルを開いて鍵を読み込む関数である．読み込んだ鍵列を切り分けて各種暗号化鍵として SA パラメータに格納する作業は qkd_fetch_key() の読み出し元関数でおこなう．図 4.3 に qkd_fetch_key() のソースコードを載せる．/* E */で示される部分で鍵プール内の鍵ファイルを開き，/* F */で示される部分で鍵を必要なサイ

```
rc_vchar_t *qkd_fetch_key_QUICS
(char *keyname, struct ikev2_sa *ike_sa, int required_len){
  FILE *fp = 0;
  char path[PATH_LENGTH];
  memcpy(path, ike_sa->qkd_key_dir->v, ike_sa->qkd_key_dir->l);
  path[ike_sa->qkd_key_dir->l] = '\0';
/* E */
  if((fp = fopen(strncat(path,keyname,PATH_LENGTH-strlen(path)), "r"))
      == NULL ){
    syslog(LOG_NOTICE, "qkd_fetch_key_QUICS: failed fopen\n");
    return NULL;
  }
  rc_vchar_t *keys = NULL;
  keys = rc_vmalloc(required_len);
/* F */
  if(fread(keys->v, keys->l, 1, fp) != 1){
    syslog(LOG_NOTICE, "qkd_fetch_key_QUICS: failed fread\n");
    return NULL;
  }
  fclose(fp);
  return keys;
}
```

図 4.3: qkd_fetch_key_QUICS() の実装

```
int qkd_rm_keyfile_QUICS(char *keyname, struct ikev2_sa *ike_sa){
  char path[PATH_LENGTH];
  memcpy(path, ike_sa->qkd_key_dir->v, ike_sa->qkd_key_dir->l);
  path[ike_sa->qkd_key_dir->l] = '\0';
  strncat(path,keyname,PATH_LENGTH-strlen(path));
/* G */
  if(unlink(path) != 0){
    syslog(LOG_NOTICE,
              "qkd_rm_keyfile_QUICS: failed remove %s\n",path);
    return -1;
  }
  return 0;
}
```

図 4.4: qkd_rm_keyfile_QUICS() の実装

ズ分だけ読み込む．返り値は鍵を保持するデータ構造体”rc_vchar_t” へのポインタであるが，鍵を読み込むのに失敗した場合には NULL を返す．

- **qkd_rm_keyfile()**

  本研究では，QUICS 用の qkd_rm_keyfile() を qkd_rm_keyfile_QUICS() として実装した．qkd_rm_keyfile_QUICS() は，鍵プールディレクトリから指定された識別子を持つ鍵ファイルを削除する関数である．図 4.4 に qkd_rm_keyfile_QUICS() のソースコードを載せる．/* G */で示される部分で，鍵ファイルの削除をおこなっている．

インターオペラビリティを持つ鍵管理インタフェースモジュールの実装

raQoon2 は鍵管理インタフェースの初期化において図 4.5 に示す処理をおこない，インターオペラビリティを確保している．qkd_choosekey() と qkd_fetch_key()，qkd_rm_keyfile() は，/* H */で関数ポインタ構造体 qkd_keyctrl_module に格納されている．qkddev_init() が呼び出されたとき，/* I */で示す switch 文で，qkd_key_module にコンフィグファイルで指定された量子鍵配送装置の種類に対応する関数ポインタ構造体が代入される．QUICS を指定していた場合，/* J */で示す QUICS 用の鍵管理インタフェース関数構造体へのポインタが鍵管理インタフェース関数として構造体 ike_sa に設定される．

```
/* H */
struct qkd_keyctrl_module{
  int (*qkd_choosekey)();
  rc_vchar_t *(*qkd_fetch_key)();
  int (*qkd_rm_keyfile)();
};
/* J */
struct qkd_keyctrl_module qkd_keyctrl_module_QUICS={
  qkd_choosekey_QUICS,
  qkd_fetch_key_QUICS,
  qkd_rm_keyfile_QUICS
};
int qkddev_init(struct ikev2_sa *ike_sa){
  ike_sa->qkd_dev_type = ike_sa->rmconf->ikev2->qkd_dev_type;
/* I */
  switch(ike_sa->qkd_dev_type){
  case RCT_QUICS:
    ike_sa->my_qkd_keyctrl_module = &qkd_keyctrl_module_QUICS;
    break;
  default:
    return -1;
  }
  /* check key pool */
  return 0;
}
```

図 4.5: 鍵管理インターフェイス初期化機構の実装

新しい鍵管理インタフェースの追加方法

2010 年 2 月現在の raQoon2 は，量子鍵配送装置の中でも QUICS にのみ対応している．新しく他の量子鍵配送装置に対応させるには，その量子鍵配送装置に対応する鍵管理インタフェースモジュールを実装する必要がある．図 4.6 に，鍵管理インタフェースを追加する際の例を示す．“NEWQKD” という名前の量子鍵配送装置を追加する場合，NEWQKD の実装に応じて，qkd_choosekey() の実体である qkd_choosekey_NEWQKD() と qkd_fetch_key() の実体である qkd_fetch_key_NEWQKD()，qkd_rm_keyfile() の実体である qkd_rm_keyfile_NEWQKD() を実装し，qkd_keyctrl_module_NEWQKD 構造体として

```
int qkd_choosekey_NEWQKD(struct ikev2_sa *ike_sa){
  /*  NEWQKD の実装に対応する choose_key() 関数  */
}
rc_vchar_t *
qkd_fetch_key_NEWQKD(char *keyname, struct ikev2_sa *ike_sa){
  /*  NEWQKD の実装に対応する qkd_fetch_key() 関数  */
}
int qkd_rm_keyfile_NEWQKD(char *keyname, struct ikev2_sa *ike_sa){
  /*  NEWQKD の実装に対応する qkd_rm_keyfile() 関数  */
}
struct qkd_keyctrl_module qkd_keyctrl_module_NEWQKD={
  qkd_choosekey_NEWQKD,
  qkd_fetch_key_NEWQKD,
  qkd_rm_keyfile_NEWQKD
};
int qkddev_init(struct ikev2_sa *ike_sa){
  ike_sa->qkd_dev_type = ike_sa->rmconf->ikev2->qkd_dev_type;
  switch(ike_sa->qkd_dev_type){
  case RCT_QUICS:
    ike_sa->my_qkd_keyctrl_module = &qkd_keyctrl_module_QUICS;
    break;
  case RCT_NEWQKD:
    ike_sa->my_qkd_keyctrl_module = &qkd_keyctrl_module_NEWQKD;
    break;
  default:
    return -1;
  }
  /* check key pool */
  return 0;
}
```

図 4.6: 図 4.5 を基にした，新しい鍵管理インタフェースの追加実装

まとめる必要がある．また，量子鍵配送を利用する際の初期化関数である qkddev_init() 中の switch 文で NEWQKD がコンフィグで選択された場合に対応し，構造体 ike_sa 中の my_qkd_keyctrl_module に qkd_keyctrl_module_NEWQKD のポインタを代入するよう改変する．

4.4.5 raQoon2のコンフィグレーション

図4.7にlexに追加したtokenを一覧する．ikev2_qkdは，量子鍵配送を利用したIKEをおこなう際に指定する鍵交換プロトコルである．qkd_fallback_typeはフォールバック方法を指定するディレクティブであり，qkd_key_dirは量子鍵配送で生成した鍵の保存ディレクトリを設定するディレクティブ，qkd_dev_typeは利用する量子鍵配送の実装を設定するディレクティブである．waitqkd，continue，dhはフォールバック方法を指定する識別子である．QUICSは，量子鍵配送装置QUICSを利用する際に指定する，量子鍵配送装置の種類の識別子である．図4.8にyaccファイルに追加した文法を示す．ikev2_qkd_specは，ikev2_qkdが指定されたときに利用される．ikev2_qkd_specは通常のikev2を指定した際に読み込むikev2_specを拡張したもので，量子鍵配送に対応してQKD_FALLBACK_TYPE，QKD_KEY_DIR，QKD_DEV_TYPEの読み込みをおこなう．ikev2_qkd以外を指定してこれらのディレクティブの設定をおこなうとエラーが出るようになっている． raQoon2のremoteディレクティブのコンフィグ例を図4.9に示す．図4.9では，トランスポートモードを例にしてある．remoteディレクティブ以外のディレクティブのコンフィグはracoon2に準拠する．4行目のikev2_qkdで，量子鍵配送を用いるIKEの動作について規定し，3行目のacceptable_kmpにikev2_qkdを指定することで量子鍵配送を用いたIKEの利用を指定する．10行目のkmp_qkd_fallback_typeに，使用するフォールバック方法を指定する．raQoon2におけるフォールバック方法の優先順位はコンフィグの記述順ではなく，各フォー

```
ikev2_qkd               { DP; return(IKEV2_QKD); }

qkd_fallback_type       { DP; return(QKD_FALLBACK_TYPE); }
qkd_key_dir             { DP; return(QKD_KEY_DIR); }
qkd_dev_type            { DP; return(QKD_DEV_TYPE); }

waitqkd                 { DP; return(QKD_FALLBACK_WAITQKD); }
continue                { DP; return(QKD_FALLBACK_CONTINUE); }
dh                      { DP; return(QKD_FALLBACK_DH); }
QUICS                   { DP; return(QUICS); }
```

図 4.7: lexファイル改変場所

```
ikev2_qkd_spec
        :       kmp_common_spec
                {
                        $$ = $1;
                }
        |       QKD_FALLBACK_TYPE algorithm_list_spec
                {
                  MKRCFDIR($$, CFD_QKD_FALLBACK_TYPE);
                  $$->nextp = $2;
                }
        |       QKD_KEY_DIR string
                {
                  MKRCFDIR($$, CFD_QKD_KEY_DIR);
                  $$->nextp = $2;
                }
        |       QKD_DEV_TYPE algorithm_list_spec
                {
                  MKRCFDIR($$, CFD_QKD_DEV_TYPE);
                  $$->nextp = $2;
                }
        .
        .
        .
      (続く)
```

図 4.8: yacc ファイル改変場所

ルバック方法の性質を考慮して WAIT_QKD，DH，CONTINUE の順になっているため，注意が必要である．図 4.9 では，WAIT_QKD と CONTINUE を指定している．11 行目の qkd_key_dir で，量子鍵配送で作成された鍵を保持するディレクトリを指定する．本実装では，NFS でマウントしたパーティション中に存在するディレクトリとなる．図 4.9 では，/mnt/qkdkeys/ディレクトリが指定されている．12 行目の qkd_dev_type で，量子鍵

```
remote ike_trans_remote {
        acceptable_kmp { ikev2_qkd; };
        ikev2_qkd {
                my_id fqdn "${MY_FQDN}";
                peers_id fqdn "${PEERS_FQDN}";
                peers_ipaddr "${PEERS_IPADDRESS}" port 500;
                ## Use Preshared Key
                kmp_auth_method { psk; };
                pre_shared_key "${PSKDIR}/${PRESHRD_KEY}";
                qkd_fallback_type { wait_qkd; continue; };
                qkd_key_dir "/mnt/qkdkeys/";
                qkd_dev_type QUICS;
        };
        selector_index ike_trans_sel_tcp_in;
        selector_index ike_trans_sel_tcp_out;
        selector_index ike_trans_sel_udp_in;
        selector_index ike_trans_sel_udp_out;
};
```

図 4.9: raQoon2 のコンフィグ例

配送装置の種類を指定する．図 4.9 では，QUICS が指定されている．

第5章　評価

本章では本研究を進める過程でおこなった展示と実験について述べ，構築したシステムについての評価をまとめる．

5.1　展示/実験

本研究では，システム構築過程で経過発表を兼ねて一回の展示と一回の実験をおこなった．

5.1.1　量子鍵配送実用展示

SFC Open Research Forum 2008 にて，量子鍵配送を用いた IPsec の展示として raQoon2 のプロトタイプの実動展示をおこなった．図 5.1 に実験ネットワークを示す．Alice と Bob

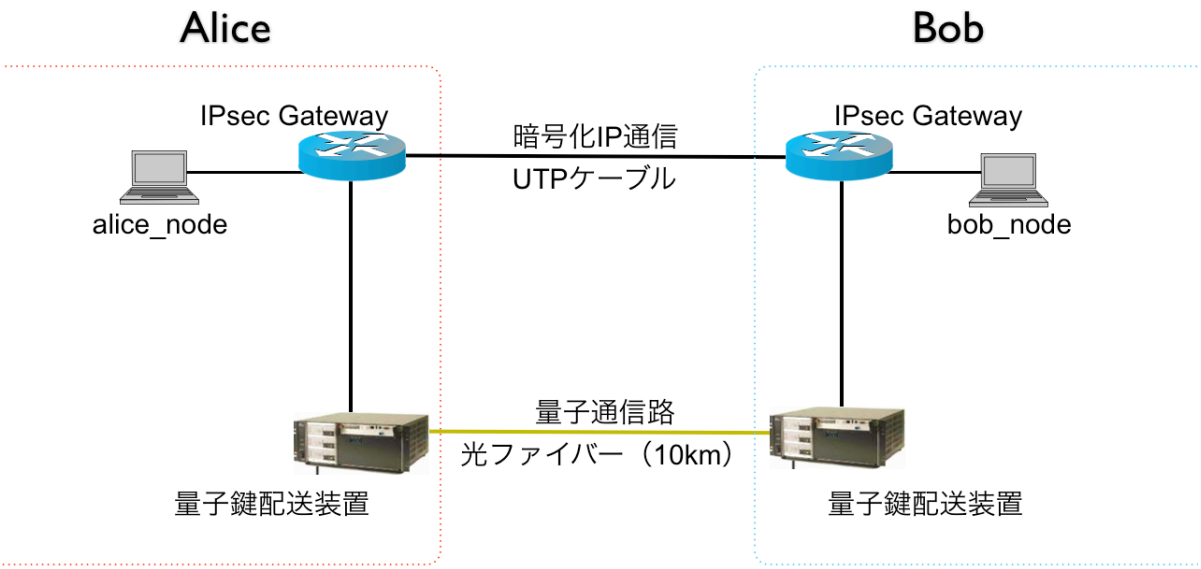


図 5.1: ORF2008 展示ネットワーク

図 5.2: ORF2008 にておこなった展示の様子

に，それぞれ IPsec Gateway と量子鍵配送装置が一台ずつ存在し，ローカル接続されている．IPsec Gateway 同士は UTP ケーブルによって接続されており，量子鍵配送装置同士は長さ 10km の光ファイバーによって接続されている．この光ファイバーの中に光子を通して量子鍵配送をおこなった．また，Alice の IPsec Gateway には Alice_node が接続されており，Bob の IPsec Gateway には Bob_node が接続されている．本展示では Alice_node から Bob_node へ ICMP echo request を送信し，Bob_node から Alice_node へ ICMP echo reply を返信した．その際トンネルモードの IPsec を利用し，IPsec Gateway 間の通信を量子鍵配送を利用した IPsec によって暗号化した．図 5.2 におこなった展示の様子を掲載する．この展示は本研究最初の発表の場となった．

5.1.2 ユーザの存在する環境での実用実験

2009 年 3 月におこなわれた WIDE プロジェクトの合宿にて，第 5.1.1 小節で利用した raQoon2 のプロトタイプを用いて実用実験をおこなった．図 5.3 に実験ネットワークを示す．IPsec Gateway と量子鍵配送装置の接続は，第 5.1.1 小節で述べた展示と同様である．

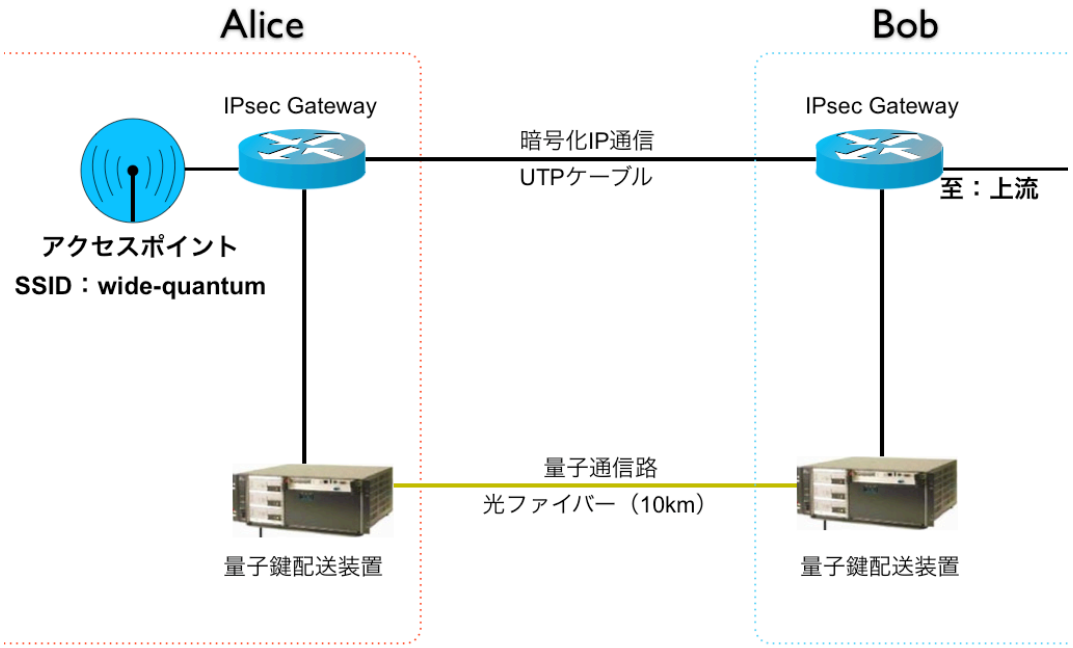


図 5.3: WIDE 合宿実験ネットワーク

前出の展示と異なる点は，Alice の IPsec Gateway に無線アクセスポイントを接続し，Bob の IPsec Gateway をインターネット上流への経路と接続した点である．これにより，この無線アクセスポイントに接続したユーザは，インターネットと通信する際に必ず，量子鍵配送を利用した IPsec を利用した．この実験は，量子鍵配送において初めての，ユーザを持つ運用実験となった．

5.2 評価

本節ではまず，本研究で構築したシステムの動作検証実験について述べる．続いてシステム評価，フォールバック方法の安全性評価について述べた後，既存のシステムとの比較を通して，本研究で構築したシステムの特徴をまとめる．

5.2.1 動作検証実験

本実験では，フォールバック運転の動作検証，単一 IPsec Gateway と複数の SA を張る IPsec の検証，複数の IPsec Gateway と単一の SA を張る IPsec の検証，複数の IPsec Gateway と複数の SA を張る IPsec の検証をおこなった．本実験をおこなったネットワークを図 5.4 に示す．また，本実験に使用したホストを表 5.1 にまとめる．本実験では QUICS を利用できなかったため，QUICS エミュレータを用いて実験をおこなった． QUICS エミュレータ 1 が生成する乱数を利用して，sheryl と valkyrie の間で量子鍵配送を利用する IPsec をおこなった．また，QUICS エミュレータ 2 が生成する乱数を利用して，sheryl と

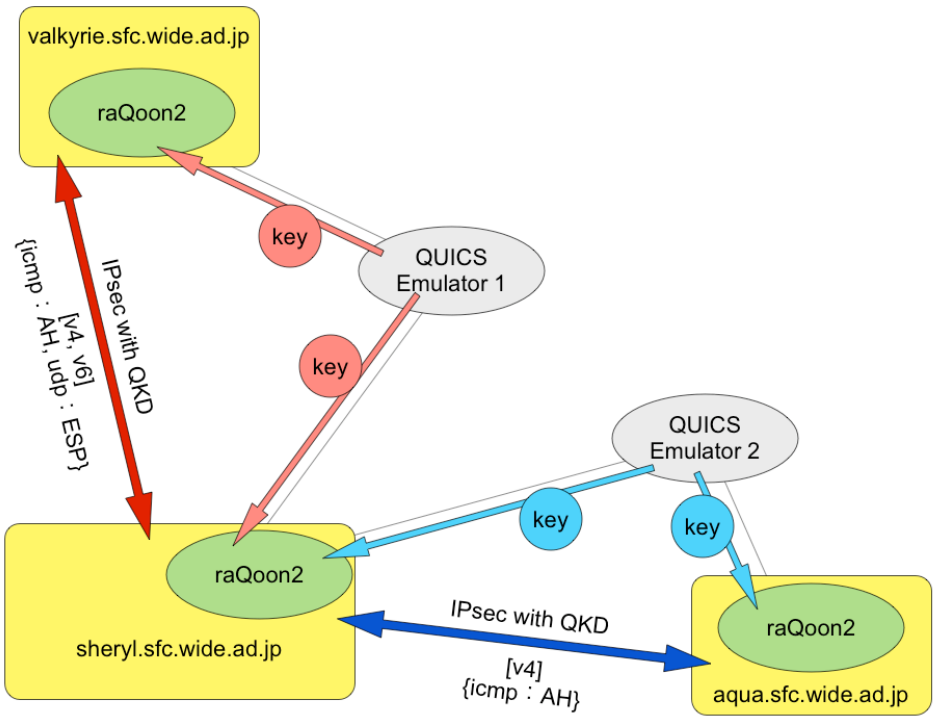


図 5.4: raQoon2 検証ネットワーク

表 5.1: システム動作実験における使用ホスト

ホスト名	IPv4 アドレス	IPv6 アドレス
sheryl.sfc.wide.ad.jp	203.178.143.26	2001:200:0:8801:203:178:143:26
valkyrie.sfc.wide.ad.jp	203.178.128.57	2001:200:0:8802:203:178:128:57
aqua.sfc.wide.ad.jp	203.178.143.71	2001:200:0:8801:203:178:143:71

aqua の間で量子鍵配送を利用する IPsec をおこなった．QUICS エミュレータ 1 と QUICS エミュレータ 2 は同一ホスト内で動かしたが，プロセスを分けているため論理的には図 5.4 のようになる．各検証実験について以下に述べる．

フォールバック運転の動作検証

フォールバック運転の動作について確認すべき事項は以下の三点である．

- 通常運転からフォールバック運転へ正しく移行するか
- 各フォールバック運転は正しく動作するか
- フォールバック運転から通常運転へ正しく移行するか

この三点について確認するため，WAIT_QKD，DIFFIE-HELLMAN，CONTINUE の各フォールバック方法を以下の手順で検証した．

1. 鍵プールに鍵を保持し，QUICS エミュレータを切った状態でシステムを開始する．
2. SA の更新に伴い鍵プールに保持していた鍵を使い切る．
3. 鍵プールに鍵がない状態で SA を更新し，フォールバック運転へ移行する．
4. フォールバック運転を継続する．
5. QUICS エミュレータを起動し，鍵プールに新しい鍵を生成する．
6. 新しく生成された鍵を利用して SA を更新し，通常運転へ移行する．

検証は，sheryl と valkyrie の間で IPv4 を利用しておこなった．この結果，三種類のフォールバック方法それぞれが確認すべき事項三点を達成していることを確認し，フォールバック運転が正しく動作することを確認した．

単一 IPsec Gateway と複数の SA を張る IPsec の検証

単一の IPsec Gateway と複数の SA を張る IPsec の検証は以下の要領でおこなった．sheryl と valkyrie の間で icmp の通信には AH を使用し，udp の通信には ESP を使用した．張った SA の様子を付録 A の図 A.1 と図 A.2 に示す．

複数の IPsec Gateway と単一の SA を張る IPsec の検証

複数の IPsec Gateway と単一の SA を張る IPsec の検証は以下の要領でおこなった．sheryl と valkyrie の間で icmp の通信に AH を使用し，sheryl と aqua の間で icmp の通信に ESP を利用した．張った SA の様子を付録 A の図 A.3 と図 A.4 に示す．

複数の IPsec Gateway と複数の SA を張る IPsec の検証

複数の IPsec Gateway と複数の SA を張る IPsec の検証は以下の要領でおこなった．sheryl と valkyrie の間で icmp の通信に AH を使用し，udp の通信には ESP を使用した．また，sheryl と aqua の間で icmp の通信に ESP を利用した．張った SA の様子を付録 A の A.5 と A.6 に示す．

5.2.2 定性評価

本研究で構築したシステムの評価は定性評価でおこなう．表 5.2 に表 3.5 で示したシステム要件とその評価をまとめる．

要件 1:量子鍵配送装置から SAD までの機密性を確保した鍵供給

本システムでは，量子鍵配送装置と IPsec Gateway の接続は物理セキュリティが確保された環境下で専用線を用いておこなう．これにより，量子鍵配送装置と IPsec Gateway 間で鍵を転送する際に鍵を盗まれることはなくなる．また，IPsec Gateway

表 5.2: 本研究の実装におけるシステム要件と評価

	要件	評価	備考
要件 1	量子鍵配送装置から SAD までの機密性を確保した鍵供給	○	物理セキュリティを利用
要件 2	可用性の確保	○	フォールバック運転を実装
要件 3	外部で作成された鍵の本システムによる管理	○	NFS を利用
要件 4	量子鍵配送装置の実装によらない利用可能性の確保	○	量子鍵配送装置と IPsec Gateway の，拡張性のある接続インタフェースを構築
要件 5	複数の IPsec Gateway との IKE 管理	○	鍵プールディレクトリの SA 毎の管理
要件 6	Diffie-Hellman 鍵共有を利用する IKE と量子鍵配送を利用する IKE の同時管理	○	量子鍵配送利用フラッグの新設定
要件 7	IPv6 対応	○	racoon2 の機能を活用

内では鍵をドライブ上に保存することはせず，全てメモリ上で処理をおこなう．これにより，量子鍵配送装置から SAD まで鍵を安全に供給する．

要件 2:可用性の確保

本システムでは，量子鍵配送による鍵を利用できない事態に備えてフォールバック運転を実装した．フォールバック運転の利用により，SA を更新する際には利用可能な鍵がなくとも IPsec を継続する事が可能である．

要件 3:外部で作成された鍵の本システムによる管理

本システムでは，量子鍵配送装置と IPsec Gateway の論理的接続には NFS を用いている．これにより IPsec Gateway 内の raQoon2 から量子鍵配送装置内の鍵プールを直接操作可能となり，量子鍵配送装置の資源を圧迫しない運用が可能である．

要件 4:量子鍵配送装置の実装によらない利用可能性の確保

本システムでは実際に量子鍵配送装置を操作するインタフェースを，一つのモジュールとして分けて実装した．これにより，新しいモジュールを追加実装することで様々な鍵プールの実装を利用することが可能となり，様々な量子鍵配送装置を利用する事が可能となった．

要件 5:複数の IPsec Gateway との IKE 管理

本システムでは，複数の IPsec Gateway との IKE を管理できる racoon2 をベースとして開発した．racoon2 の持つ機能を阻害せず有効利用し，なおかつ，鍵プールディレクトリのパスを SA パラメータに保持して SA 毎に鍵プールを管理する方式を取ったことで，複数の IPsec Gateway との IKE 管理が可能となった．

要件 6:Diffie-Hellman 鍵共有を利用する IKE と量子鍵配送を利用する IKE の同時管理

本システムでは，IKE_SA の設定パラメータに量子鍵配送の利用を規定するフラッグを設けた．このフラッグを用いて処理を分ける事により，Diffie-Hellman 鍵共有を利用する IKE と量子鍵配送を利用する IKE の同時管理が可能となった．

要件 7:IPv6 対応

本システムは IPv6 に対応した IKE 実装である racoon2 をベースとして開発した．racoon2 の持つ機能を阻害せず有効活用することにより，IPv6 で動く IPsec も量子鍵配送で生成した鍵を利用して実行可能である．

表 5.3: フォールバック方法特性比較

	WAIT_QKD	DIFFIE-HELLMAN	CONTINUE
フォールバック方法の利用可能性	◯	◯	◯
通常運転との安全性比較	◯	△	△
可用性	×	◯	◯

5.2.3 フォールバック方法評価

Diffie-Hellman 鍵共有を用いた IPsec では鍵が必要になったときに鍵を共有することが可能である．そのため，可用性は確保されている．しかしながら量子鍵配送を用いた IPsec は，鍵の使用と鍵の共有が非同期であり，鍵共有自体に時間がかかる，よって，鍵が利用不可能な事態がありえるため可用性は低下する．フォールバック運転は，低下した可用性を補うものである．表 5.3 に三種類のフォールバック方法と要求される安全性についての考察を示す．

フォールバック方法の利用可能性

全てのフォールバック方法において，暗号化通信の安全性は量子鍵配送によって保証されている．よって，全てのフォールバック方法の利用可能性は ◯ である．

通常運転との安全性比較

本システムの通常運転時の鍵の安全性とフォールバック運転時の鍵の安全性を比較する．同じ鍵を利用し続ける CONTINUE は，攻撃者が獲得可能な，鍵についての情報量が増えるため安全性は低下する．2010 年 1 月現在においてはこの問題は無視可能であると考えられるが，今後も無視を続けられるとは限らないため △ とした．DIFFIE-HELLMAN は，SA 更新の際に暗号化鍵は変わっているが，安全性の根拠とする鍵は更新前と同一である．よって CONTINUE と同様に，漏洩する鍵についての情報量を考えて通常運転より評価を下げ，△ とした．一方 WAIT_QKD では，鍵についての漏洩情報量は通常運転と同様であるため通常運転と同じ鍵の安全性を持っているとし ◯ とした．

可用性

各フォールバック運転時の可用性について考察する．WAIT_QKD は，量子鍵配送装置の鍵生成を待つ間に SA の有効期限が切れた場合，暗号化通信がおこなえなくなるため可用性は大幅に低下する．DIFFIE-HELLMAN では鍵の安全性は低下す

るものの，SA が存在しなくなることはないため暗号化通信の可用性は低下しない．CONTINUE は DIFFIE-HELLMAN と同様に，鍵の安全性は低下するものの SA が存在しなくなることはないため，常に暗号化通信が可能であり可用性は低下しない．

DIFFIE-HELLMAN もしくは CONTINUE を用いれば，量子鍵配送の利用に併せて起こる可用性の低下を抑えることが可能である．しかしながら，DIFFIE-HELLMAN と CONTINUE は暗号化通信の安全性の低下を起こしているため，フォールバック方法を選択する際には暗号化通信の目的と合わせてよく考慮する必要がある．

5.2.4 既存のシステムとのセキュリティ比較

表 5.4 に，本研究で構築した量子鍵配送を利用した IPsec と，Diffie-Hellman 鍵共有を利用した IPsec の安全性の比較を示す．

効率的因数分解

効率的因数分解は，現在の計算力では実現していない主題である．Diffie-Hellman 鍵共有は効率的因数分解が実現していないことを前提に安全性が保証されており，効率的因数分解が可能になれば Diffie-Hellman 鍵共有は解かれる．そのため，Diffie-Hellman 鍵共有を利用した IPsec は効率的因数分解に弱い．一方量子鍵配送は，安全性が量子数学と統計数学によって証明されており，効率的因数分解が実現しても解かれることはない．よって，本研究で構築したシステムは効率的因数分解に強いと言える．

暗号化アルゴリズムへの攻撃

IPsec は，AES や DES3，MAC などの暗号化アルゴリズムを利用している．AES，DES3 は暗号化アルゴリズムであり，MAC はメッセージ認証コードである．IPsec

表 5.4: 本研究で構築したシステムと既存システムの安全性の比較

	量子鍵配送を利用した IPsec	Diffie-Hellman 鍵共有を利用した IPsec
効率的因数分解	○	×
暗号化アルゴリズムへの攻撃	×	×
鍵共有盗聴の検知	○	×
可用性	○	○

ではこれらの他にも多くの暗号化アルゴリズムを利用できる．これらの暗号化アルゴリズムへの攻撃は 2010 年 1 月現在実現していないが，将来実現する可能性がある．暗号化アルゴリズムへの攻撃が実現したとき，攻撃される暗号化アルゴリズムを利用していれば，本研究で構築したシステムへの攻撃は成功し，Diffie-Hellman 鍵共有を利用した IPsec への攻撃も成功する．

攻撃の検知

Diffie-Hellman 鍵共有を利用した IPsec では，鍵共有トランザクションを攻撃者によって盗聴されても検知することはできない．一方，本研究で構築したシステムでは攻撃者によって鍵共有を盗聴された場合，量子鍵配送の性質により検知することが可能である．

可用性

Diffie-Hellman 鍵共有を利用した IPsec は，必要に応じて鍵を折衝するため可用性は高い．本研究で構築したシステムでは，必要に応じて鍵を折衝することはしないが，鍵についてのフォールバック方法を講じてあるため可用性は高い．

5.2.5 既存のシステムとの情報秘匿性の有効期限比較

本研究で構築したシステムの情報秘匿性の有効期限を既存のシステムの情報秘匿性の有効期限と比較する．図 5.5 に，Diffie-Hellman 鍵共有を利用した IPsec と量子鍵配送と OTP を利用した IPsec の情報秘匿性の有効期限を比較する．表 5.4 より，Diffie-Hellman 鍵共有を利用した IPsec は，因数分解問題が解決した際に安全性が損なわれ，暗号化アルゴリズムへの攻撃が成立した際にも安全性が損なわれる．仮に攻撃者が通信パケットを保存している可能性を考えた場合，この方法で暗号化した全ての情報は，いつ通信したものであっても，因数分解問題が解決した時点もしくは暗号化アルゴリズムへの攻撃が成立した時点で秘匿性を失う．一方，量子鍵配送を利用した IPsec では，暗号化アルゴリズムへの攻撃が成立した際にのみ安全性が損なわれる．量子鍵配送を利用した IPsec では，因数分解問題が解決した時点ではそれまでに暗号化通信した情報の秘匿性を失うことはなく，暗号化アルゴリズムへの攻撃が成立した時点でのみそれまでに通信した情報の秘匿性を失う．因数分解問題の解決が暗号化アルゴリズムへの攻撃の成立より早かった場合，Diffie-Hellman 鍵共有を利用した IPsec は，量子鍵配送を利用した IPsec よりも早く情報秘匿性を失う．現在の時点からすると，Diffie-Hellman 鍵共有を利用した IPsec の情報秘匿性の有効期限は，量子鍵配送を利用した IPsec の有効期限よりも短いこととなる．暗号

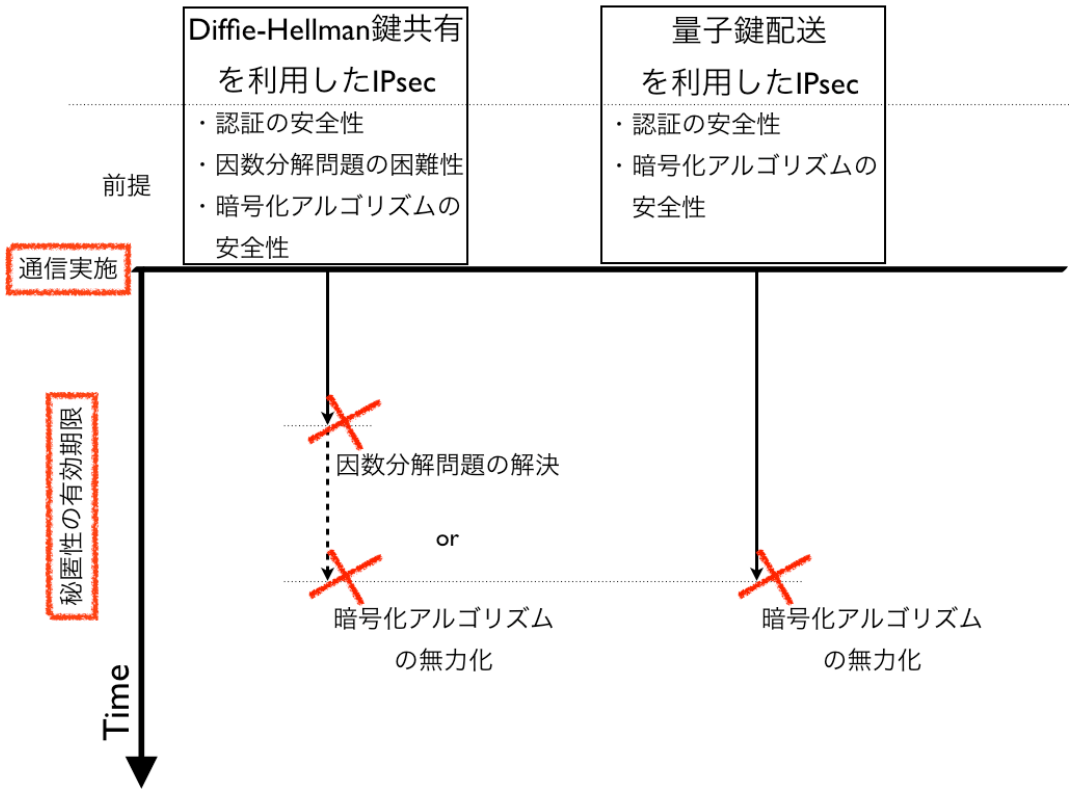


図 5.5: 情報秘匿性の有効期限比較

化アルゴリズムへの攻撃の成立が因数分解問題の解決よりも早かった場合，これら二つのシステムの情報秘匿性の有効期限は等しい．

5.3 まとめ

本章ではシステム要件と照らし合わせて，本研究で構築したシステムの定性評価をおこなった．また，Diffie-Hellman 鍵共有を用いた IPsec との比較を通して本研究で構築したシステムの安全性を確認した．その結果，本研究で構築したシステムの安全性が既存の暗号化通信システムより高いことが確認された．本システムを利用することで，従来の暗号化通信より安全で，情報秘匿性の有効期限が長い暗号化通信が実現することが確認された．

第6章　結論

6.1　まとめ

因数分解問題に基づく鍵共有方式は，将来安全性が損なわれると予測されている．本研究では暗号化通信を継続的に利用していくために，数学的に安全性が保証されている量子鍵配送に着目した．そして，IP を利用する全ての通信を暗号化できる IPsec で量子鍵配送を利用するために量子鍵配送を用いる IPsec の要件を整理し，IPsec の鍵交換プロトコルである IKE を拡張した．

量子鍵配送は IPsec Gateway の要求に応じて鍵を生成せず，独立に鍵を生成し続ける．このことは量子鍵配送を利用するシステムを考えるにあたって二点の要求を生む．一点目は，鍵の利用箇所の折衝が必要となることである．IPsec Gateway は連続して生成され続ける鍵列の中からどの部分を暗号化鍵に利用するかを折衝しなくてはならない．二点目は IPsec Gateway が鍵を必要になったとき，量子鍵配送装置と量子鍵配送装置によって作成された鍵がどういう状態にあるかを予測できないという点である．問題無く鍵が生成されている可能性もあるが，鍵の利用速度が鍵の生成速度を上回って鍵が枯渇している可能性もある．さらに，盗聴などによって量子鍵配送に失敗し，鍵を折衝できなくなることにより鍵が枯渇している可能性も考えられる．

様々な鍵の生成状況に対応するため，本研究では量子鍵配送の性質と IPsec による暗号化通信の性質を考慮した鍵管理方式を提案し，実装した．この鍵管理方式には上記の要求に対応する二つの機能を盛り込んだ．一つ目の機能は，量子鍵配送による鍵を一定サイズごとに区切って識別子を付加することである．暗号化通信を開始する際には識別子を交換して使用する暗号化鍵を決定する．二つ目の機能は，フォールバック運転の採用である．フォールバック方法には量子鍵配送によって新しい鍵が生成されるのを待つものと既存の暗号化通信中で新しい鍵を折衝するもの，既存の暗号化通信路で使用している鍵を引き続き利用するものの三種類を設計した．これらを折衝するために新しく鍵識別子交換ペイロードとフォールバック方法交換ペイロードを定義して IKE を拡張した．

この IKE 拡張プロトコルは，2010 年 1 月現在標準化の途上にある．racoon2 をもとに，raQoon2 としてこのプロトコルを実装して実動を確認した後，Internet Draft-00 を執筆

し，IETF76（Hiroshima）において発表した [22]．本研究は，IPv6 に対応した初めての量子鍵配送利用暗号化通信である．また，本研究の成果の一つである raQoon2 は，オープンソースとして公開する予定である [23]．

6.2 今後の展望

6.2.1 完全な安全性を持つ暗号化通信について

情報秘匿には，量子鍵配送による鍵と One-Time Pad（OTP）による暗号化の併用が最も望ましい．理論上では鍵を入手しない限り OTP を破る事はできない．一方，他の暗号化アルゴリズムでは解読により通信の内容が漏洩しうる．

表 6.1 に，量子鍵配送を用いた場合と Diffie-Hellman 鍵共有を用いた場合，並びに OTP を用いた場合と他の暗号化アルゴリズムを用いた場合の暗号化通信の弱点の違いを整理する．Diffie-Hellman 鍵共有を利用すると，巨大数の効率的な因数分解が弱点となる．また，AES など OTP 以外の暗号化アルゴリズムを利用した場合，暗号化アルゴリズムへの攻撃が成立すると暗号化通信は破られることになる．これらをもとに情報秘匿性の有効期限について以下に考察する．図 6.1 に，Diffie-Hellman 鍵共有と OTP 以外の暗号化アルゴリズムを利用した暗号化通信，Diffie-Hellman 鍵共有と OTP を利用した暗号化通信，量子鍵配送と OTP 以外の暗号化アルゴリズムを利用した暗号化通信，量子鍵配送と OTP を利用した暗号化通信の情報秘匿性の有効期限を比較する．表 6.1 より，Diffie-Hellman 鍵共有と OTP 以外の暗号化アルゴリズムを利用する暗号化通信は，因数分解問題が解決した時点もしくは暗号化アルゴリズムへの攻撃が成立した時点で安全性を失う．仮に攻撃者が

表 6.1: 量子鍵配送と Diffie-Hellman 鍵共有並びに OTP と暗号化アルゴリズムの弱点比較

鍵共有方法	量子鍵配送	量子鍵配送	Diffie-Hellman 鍵共有	Diffie-Hellman 鍵共有
暗号化アルゴリズム	OTP 以外	OTP	OTP 以外	OTP
効率的因数分解	○	○	×	×
暗号化アルゴリズムへの攻撃	×	○	×	○
鍵共有盗聴の検知	○	○	×	×

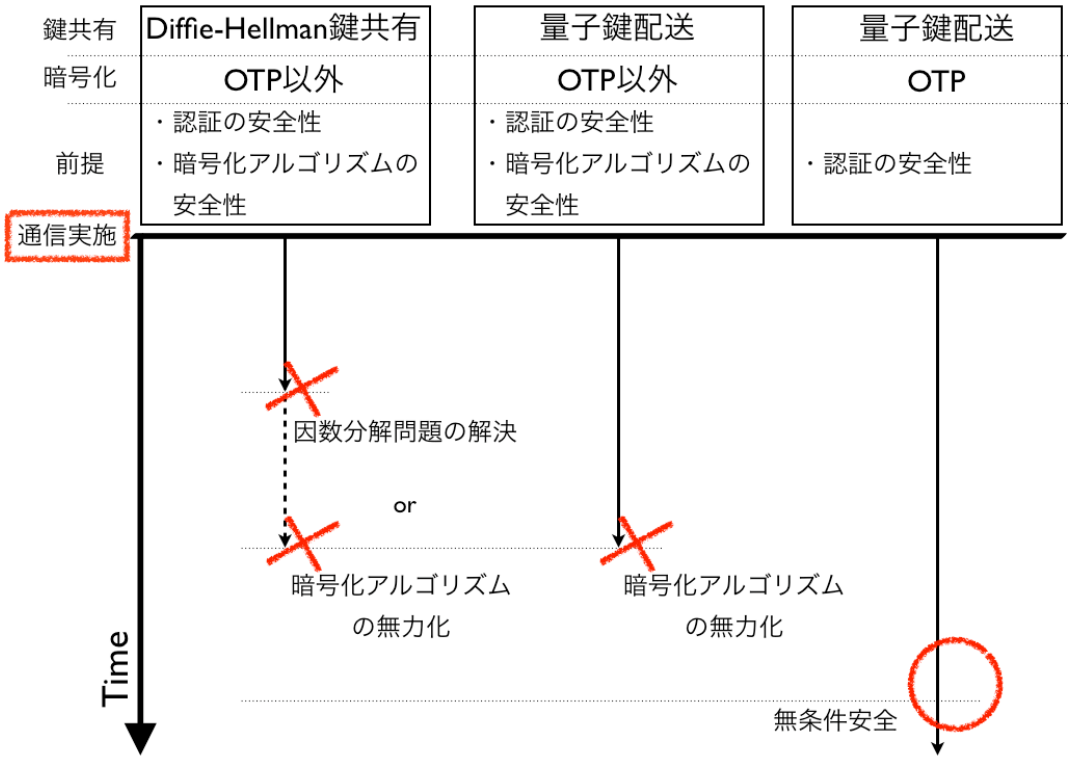


図 6.1: 情報秘匿性の有効期限比較

暗号化通信パケットを保存している可能性を考えると，過去におこなった全ての暗号化通信から情報の秘匿性が失われる．これは，Diffie-Hellman 鍵共有と OTP 以外の暗号化アルゴリズムを利用する暗号化通信の情報秘匿性の有効期限が切れることを意味する．同様に，Diffie-Hellman 鍵共有と OTP を利用した場合には，因数分解問題が解決した時点で情報の秘匿性を失い，情報秘匿性の有効期限が切れる．また，量子鍵配送と OTP 以外の暗号化アルゴリズムを利用する暗号化通信は暗号化アルゴリズムへの攻撃が成立した時点で情報秘匿性の有効期限が切れる．量子鍵配送と OTP を利用した場合，暗号化通信に弱点は存在しない．そのため，情報秘匿性の有効期限は無期限となり，将来に渡って暗号化通信した情報は漏洩しない．よって，完全に安全な暗号化通信をおこなうには量子鍵配送と OTP を併用しなければならない．しかしながら 2010 年 1 月現在量子鍵配送は 20km の距離で 10Mb/s 程度のスループットにとどまっており，OTP をおこなうには速度不足である．OTP では，鍵共有のスループットがそのまま暗号化通信のスループットになる．完全な暗号化通信実現のため，量子鍵配送のスループットの更なる向上が期待される．

6.2.2 Security Consideration

量子鍵配送を利用した暗号化通信について二点追記する．量子鍵配送は盗聴者を必ず発見できる．しかしながらもし盗聴者を発見した場合には，盗聴者が操作を加えたと考えられる鍵を放棄する．そのため盗聴者の存在は鍵共有の妨げとなり，盗聴は DoS として有効な攻撃になりうる．フォールバックの主な目的は，この問題に対処し可用性を確保することである．本研究ではフォールバック方法を三つ定義した．今後はネットワークオペレータの使いやすいフォールバック方法を設計し，プロトコルや実装に取り入れていく必要がある．

量子鍵配送装置は認証を必要とする．本研究では量子鍵配送装置が独自に認証をおこなうこととした．量子鍵配送装置が認証し，なおかつ IPsec Gateway が量子鍵配送で生成した鍵を利用した暗号化通信に成功すれば，互いの IPsec Gateway が認証されている量子鍵配送装置と接続されていることの証明となるため，IPsec Gateway 同士の認証は必要ないと考えられる．

6.3 今後の課題

今後は，以下の方向性で活動を進める．

- 量子鍵配送を利用した IPsec のための IKE 拡張の標準化
- 量子鍵配送と OTP を利用した，完全安全な暗号化通信の実証実験
- NEC 社以外の量子鍵配送装置への対応

IKE の量子鍵配送拡張の標準化については，IETF76（Hiroshima）において Internet Draft-00 を発表した．付録 B に draft-nagayama-ipsecme-ipsec-with-qkd-00 を添付する．量子鍵配送と OTP を利用した暗号化通信の実証実験は，国際量子暗号会議 2010 にておこなう予定である．

量子鍵配送装置は複数のベンダーで造られている．本研究は量子鍵配送を利用する暗号化通信プロトコルの標準化と，量子鍵配送装置の特徴を活かした，全てのベンダーの量子鍵配送装置に対応する汎用的な暗号化通信システムの構築を最終目標とする．本論文ではシステム要件を整理してプロトコルを設計，実装し，NEC 社製量子鍵配送装置 QUICS に対応する暗号化通信システムを構築した．今後は他のベンダーの量子鍵配送装置にも対応するよう暗号化通信システムの改良を続け，OTP にも対応させて研究の完成を目指していく．

謝辞

本論分の執筆にあたり，多くの方々に本当にお世話になりました．まず，ご指導頂いた慶應義塾大学環境情報学部教授村井純博士，同学部教授徳田英幸博士，同学部教授中村修博士，同学部教授武田圭史博士，同学部准教授楠本博之博士，同学部准教授高汐一紀博士，同学部准教授三次仁博士，同学部准教授植原啓介博士，同学部専任講師 Rodney D. Van Meter III 博士，同学部専任講師重近範行博士，同学部専任講師中澤仁博士に感謝致します．この場をお借りして御礼を述べさせて頂きます．

Rodney D. Van Meter III 博士に重ねて感謝致します．博士には私が研究室に入室した当初から御指導御助言を頂きました．東京大学助教石原知洋氏に感謝致します．氏には，普段の研究室での生活から，研究の進め方についてまでご指導ご助力頂きました．慶應義塾大学政策メディア研究科後期博士課程水谷正慶氏と同大学卒業生金井瑛氏に感謝致します．お二方にはセキュリティの考え方，論文の書き方など多くのことをご指導頂きました．本論文を執筆するにあたり叱咤激励を頂きました慶應義塾大学政策・メディア研究科後期博士課程岡田耕司氏に感謝致します．研究室で活動を共にしました同大学卒業生中村友一氏，中里恵女史，同大学後期博士課程片岡広太郎氏，空閑洋平氏，松谷健史氏，堀場勝広氏，田崎創氏，工藤紀篤氏，久松剛氏，松園和久氏，三島和宏氏，同研究科修士課程六田佳祐氏，峯木厳氏，黒宮佑介氏，同大学環境情報学部中島明日香女史，石崎佳織女史，三部剛義氏，中村遼氏，藤原龍氏，富田千智女史と徳田・村井・楠本・中村・高汐・重近・バンミーター・植原・三次・中澤・武田合同研究プロジェクトの皆様に感謝致します．共に卒論を執筆した慶應義塾大学環境情報学部佐藤貴彦氏，波多野敏明氏，勝利友香女史と同大学総合政策学部上原雄貴氏に感謝致します．

本研究を実施するにあたり，量子鍵配送装置を貸して頂いた NEC 社に感謝致します．ご協力頂いた同社システムプラットホーム研究所田中聡寛氏，富田章久氏に感謝致します．WIDE プロジェクトの Working Group である KAME プロジェクトに感謝致します．同プロジェクトの提供する IKE の実装である racoon2 がなければ本研究は実現しませんでした．

中島明日香女史に重ねて感謝致します．女史は往復 5 時間の通学とアニメ視聴，世間の人々より長めの睡眠で非常に多忙であるにも関わらず常日頃から私を気にかけ，心配し，また支えてくれました．

黒宮佑介氏に重ねて感謝致します．氏は私に異次元での過ごし方と画質の大切さを教えて下さりました．

峯木厳氏に重ねて感謝致します．氏の持ち前の大らかさは私の卒論を救って下さりました．

マクロス Frontier のメインヒロインであるシェリル・ノーム女史に感謝致します．女史の歌への姿勢と高圧的な態度は私の胸を打ちました．思わず御名を実験ホスト名にしてしまいました．

弱肉強食の野生の世界を生きる野良猫，社長女史に感謝します．女史は本論文執筆以前からずっと私を癒し続けてくれました．もう氏と呼べないことが悔やまれてなりません．

最後に，大学入学当初から四年間に渡って私を支え続けてくれた父 隆治，母 浩子，弟 大貴，妹 ひかりに感謝致します．

以上をもって謝辞と致します．

2010 年 2 月
永山 翔太

参考文献

[1] C. H. Bennett and G. Brassard. Quantum cryptography: Public key distribution and coin tossing. In *Proc. IEEE International Conference on Computers, Systems, and Signal Processing*, pages 175–179. IEEE, December 1984.

[2] P.W. Shor. Algorithms for quantum computation: discrete logarithms and factoring. *IEEE Computer Society Press, Los Alamitos*, CA:124–134, 1994.

[3] L. DiCarlo, J. M. Chow, J. M. Gambetta, Lev S. Bishop, B. R. Johnson, D. I.Schuster, J. Majer, A. Blais, L. Frunzio, S. M. Girvin, and R. J. Schoelkopf. Demonstration of Two-Qubit Algorithms with a Superconducting Quantum Processor. *Nature*, may 2009.

[4] Whitfield Diffie and Martin E. Hellman. "new directions in cryptography". *IEEE Transactions on Information Theory*, IT-22(6):644–654, 1976.

[5] S. Kent and K. Seo. Security Architecture for the Internet Protocol. *Network Working Group RFC 4301*, December 2005.

[6] C. Kaufman. Internet Key Exchange (IKEv2) Protocol. *Network Working Group RFC 4306*, December 2005.

[7] A. Einstein, B. Podolsky, and N. Rosen. Can Quantum-Mechanical Description of Physical Reality Be Considered Complete? *Phys. Rev.*, 47(10):777–780, May 1935.

[8] C. H. Bennett, G. Brassard, C. Crépeau, R. Josza, A. Peres, and W. Wootters. Teleporting an unknown quantum state via dual classical and EPR channels. *Physical Review Letters*, 70:1895–1899, 1993.

[9] W. Dür and H.J. Briegel. Entanglement purification and quantum error correction. *Rep. Prog. Phys.*, 70:1381–1424, 2007.

[10] W. Dür and H.-J. Briegel. Entanglement purification for quantum computation. *Physical Review Letters*, 90(6):067901, 2003.

[11] Charles H. Bennett, Gilles Brassard, Sandu Popescu, Benjamin Schumacher, John A. Smolin, and William K. Wootters. Purification of Noisy Entanglement and Faithful Teleportation via Noisy Channels. *Phys. Rev. Lett.*, 76(5):722–725, Jan 1996.

[12] D. Bouwmeester, A.Ekert, and A.Zeilinger. *The Physics of Quantum Information*. 共立出版, 2007.

[13] H. J. Briegel, W. Dur, J. I. Cirac, and P. Zoller. Quantum repeaters for communication, 1998.

[14] H.-J. Briegel, W. Dür, J.I. Cirac, and P. Zoller. Quantum Repeaters: the Role of Imperfect Local Operations in Quantum Communication. *Physical Review Letters*, 81:5932–5935, 1998.

[15] W. Dür, H.-J. Briegel, J. I. Cirac, and P. Zoller. Quantum repeaters based on entanglement purification. *Physical Review A*, 59(1):169–181, Jan 1999.

[16] Rodney Van Meter, Thaddeus D. Ladd, W. J. Munro, and Kae Nemoto. System Design for a Long-Line Quantum Repeater. *IEEE/ACM Transactions on Networking*, 17(3):1002–1013, June 2009.

[17] A.K. Ekert. Quantum cryptography based on Bell's theorem. *Physical Review Letters*, 67(6):661–663, 1991.

[18] 佐川 弘幸 and 吉田宣章. **量子情報理論**. シュプリンガー・フェアラーク東京株式会社, May 2003.

[19] Chip Elliott, David Pearson, and Gregory Troxel. Quantum cryptography in practice. In *Proc. SIGCOMM 2003*. ACM, ACM, August 2003.

[20] Alan Mink, Sheila Frankel, and Ray Perlner. Quantum Key Distribution (QKD) and Commodity Security Protocols: Introduction and Integration. *International Journal of Network Security & its Applications (IJNSA)*, 1(2), July 2009.

[21] NEC. 2007.

[22] S. Nagayama and R. Van Meter. draft-nagayama-ipsecme-ipsec-with-qkd-00. 2009. expires April 22, 2010.

[23] IKE for IPsec with QKD Project raQoon2. http://aqua.sfc.wide.ad.jp/raQoon2/index.html.

付 録A　setkeyコマンドによるSADのダンプ

```
sheryl# setkey -D
203.178.143.26 203.178.128.57
ah mode=transport spi=246837585(0x0eb67151) reqid=0(0x00000000)
A: hmac-sha1  8e88a1c6 598a05f8 13af4d43 f26de6c7 9f99b79e
seq=0x00000005 replay=64 flags=0x00000000 state=mature
created: Jan 27 10:05:38 2010current: Jan 27 10:05:45 2010
diff: 7(s)hard: 20(s)soft: 17(s)
last: Jan 27 10:05:44 2010hard: 0(s)soft: 0(s)
current: 484(bytes)hard: 0(bytes)soft: 0(bytes)
allocated: 5hard: 0soft: 0
sadb_seq=3 pid=77530 refcnt=2
203.178.128.57 203.178.143.26
ah mode=transport spi=224276485(0x0d5e3005) reqid=0(0x00000000)
A: hmac-sha1  1baa9b80 9fb06242 5c40b770 20aba7b2 db34d82c
seq=0x00000005 replay=64 flags=0x00000000 state=mature
created: Jan 27 10:05:38 2010current: Jan 27 10:05:45 2010
diff: 7(s)hard: 20(s)soft: 16(s)
last: Jan 27 10:05:44 2010hard: 0(s)soft: 0(s)
current: 364(bytes)hard: 0(bytes)soft: 0(bytes)
allocated: 5hard: 0soft: 0
sadb_seq=2 pid=77530 refcnt=1
203.178.143.26 203.178.128.57
esp mode=transport spi=36200674(0x022860e2) reqid=0(0x00000000)
E: rijndael-cbc  c05348e5 b5aaf0c7 541781f7 44712376
A: hmac-sha1  eabb5a2e 088271ac 5b745790 9b71cf9c 003b208a
seq=0x00000000 replay=64 flags=0x00000000 state=mature
created: Jan 27 10:05:43 2010current: Jan 27 10:05:45 2010
diff: 2(s)hard: 20(s)soft: 17(s)
last:                       hard: 0(s)soft: 0(s)
current: 0(bytes)hard: 0(bytes)soft: 0(bytes)
allocated: 0hard: 0soft: 0
sadb_seq=1 pid=77530 refcnt=1
```

図 A.1: 複数の SA を単一の IPsec Gateway との間で張った様子その 1

```
203.178.128.57 203.178.143.26
esp mode=transport spi=115078458(0x06dbf53a) reqid=0(0x00000000)
E: rijndael-cbc  9bc16392 dc5c59c5 fe493583 b98b36d1
A: hmac-sha1  a0a0d31f 56922f90 6c6f6e11 0486db09 8a09b6bb
seq=0x00000036 replay=64 flags=0x00000000 state=mature
created: Jan 27 10:05:43 2010current: Jan 27 10:05:45 2010
diff: 2(s)hard: 20(s)soft: 17(s)
last: Jan 27 10:05:45 2010hard: 0(s)soft: 0(s)
current: 1512(bytes)hard: 0(bytes)soft: 0(bytes)
allocated: 54hard: 0soft: 0
sadb_seq=0 pid=77530 refcnt=1
```

図 A.2: 複数の SA を単一の IPsec Gateway との間で張った様子その 2

```
sheryl# setkey -D
203.178.143.26 203.178.143.71
esp mode=transport spi=72256401(0x044e8b91) reqid=0(0x00000000)
E: rijndael-cbc  75c47b81 98e4be8a 7615cb09 70b8ee39
A: hmac-sha1  586796f6 7e668462 dc6f9a48 b1f5b569 2fe62ef7
seq=0x00000000 replay=64 flags=0x00000000 state=mature
created: Jan 27 09:46:05 2010current: Jan 27 09:46:10 2010
diff: 5(s)hard: 20(s)soft: 17(s)
last:                     hard: 0(s)soft: 0(s)
current: 0(bytes)hard: 0(bytes)soft: 0(bytes)
allocated: 0hard: 0soft: 0
sadb_seq=3 pid=37245 refcnt=1
203.178.143.71 203.178.143.26
esp mode=transport spi=256914514(0x0f503452) reqid=0(0x00000000)
E: rijndael-cbc  7adad5c0 79c41697 3e1e99ba 9aa7b47f
A: hmac-sha1  71ec8fd3 5dc38a3c 7f9bc76c 0f801e24 6d11dffd
seq=0x00000000 replay=64 flags=0x00000000 state=mature
created: Jan 27 09:46:05 2010current: Jan 27 09:46:10 2010
diff: 5(s)hard: 20(s)soft: 16(s)
last:                     hard: 0(s)soft: 0(s)
current: 0(bytes)hard: 0(bytes)soft: 0(bytes)
allocated: 0hard: 0soft: 0
sadb_seq=2 pid=37245 refcnt=1
203.178.143.26 203.178.128.57
ah mode=transport spi=151824127(0x090ca6ff) reqid=0(0x00000000)
A: hmac-sha1  e7642e9b f168deb8 b25839d4 3d5815c2 2159d793
seq=0x00000002 replay=64 flags=0x00000000 state=mature
created: Jan 27 09:46:05 2010current: Jan 27 09:46:10 2010
diff: 5(s)hard: 20(s)soft: 17(s)
last: Jan 27 09:46:09 2010hard: 0(s)soft: 0(s)
current: 216(bytes)hard: 0(bytes)soft: 0(bytes)
allocated: 2hard: 0soft: 0
sadb_seq=1 pid=37245 refcnt=2
```

図 A.3: 単一の SA を複数の IPsec Gateway との間で張った様子その 1

```
203.178.128.57 203.178.143.26
ah mode=transport spi=132170663(0x07e0c3a7) reqid=0(0x00000000)
A: hmac-sha1  d4a1858c 48158e5e 782c5dba cc7ab3b1 5a5496c0
seq=0x00000002 replay=64 flags=0x00000000 state=mature
created: Jan 27 09:46:05 2010current: Jan 27 09:46:10 2010
diff: 5(s)hard: 20(s)soft: 16(s)
last: Jan 27 09:46:09 2010hard: 0(s)soft: 0(s)
current: 168(bytes)hard: 0(bytes)soft: 0(bytes)
allocated: 2hard: 0soft: 0
sadb_seq=0 pid=37245 refcnt=1
```

図 A.4: 単一の SA を複数の IPsec Gateway との間で張った様子その 2

```
sheryl# setkey -D
203.178.143.26 203.178.143.71
esp mode=transport spi=117684293(0x0703b845) reqid=0(0x00000000)
E: rijndael-cbc  66b97166 a0dd6a45 ec4eff66 340b1aa4
A: hmac-sha1  c8018411 0a3e748c d9296a17 1031f0f9 06786b93
seq=0x00000000 replay=64 flags=0x00000000 state=mature
created: Jan 27 09:58:15 2010current: Jan 27 09:58:31 2010
diff: 16(s)hard: 20(s)soft: 16(s)
last:                     hard: 0(s)soft: 0(s)
current: 0(bytes)hard: 0(bytes)soft: 0(bytes)
allocated: 0hard: 0soft: 0
sadb_seq=5 pid=3988 refcnt=1
203.178.143.71 203.178.143.26
esp mode=transport spi=112022000(0x06ad51f0) reqid=0(0x00000000)
E: rijndael-cbc  5d64a1df ea126ae0 82d36eb6 ca282f15
A: hmac-sha1  397017a3 3db8c141 bbdd2672 67e2de29 afddd05a
seq=0x00000000 replay=64 flags=0x00000000 state=mature
created: Jan 27 09:58:15 2010current: Jan 27 09:58:31 2010
diff: 16(s)hard: 20(s)soft: 16(s)
last:                     hard: 0(s)soft: 0(s)
current: 0(bytes)hard: 0(bytes)soft: 0(bytes)
allocated: 0hard: 0soft: 0
sadb_seq=4 pid=3988 refcnt=1
203.178.143.26 203.178.128.57
ah mode=transport spi=184076856(0x0af8ca38) reqid=0(0x00000000)
A: hmac-sha1  ea45e0d1 e99f5d4b becb134f 9dc937b7 095089a8
seq=0x0000000e replay=64 flags=0x00000000 state=mature
created: Jan 27 09:58:14 2010current: Jan 27 09:58:31 2010
diff: 17(s)hard: 20(s)soft: 16(s)
last: Jan 27 09:58:30 2010hard: 0(s)soft: 0(s)
current: 1456(bytes)hard: 0(bytes)soft: 0(bytes)
allocated: 14hard: 0soft: 0
sadb_seq=3 pid=3988 refcnt=2
```

図 A.5: 複数の SA を複数の IPsec Gateway との間で張った様子その 1

```
203.178.128.57 203.178.143.26
ah mode=transport spi=150108146(0x08f277f2) reqid=0(0x00000000)
A: hmac-sha1  306e0678 5096af5a 40090094 dc655551 9e071fba
seq=0x0000000e replay=64 flags=0x00000000 state=mature
created: Jan 27 09:58:14 2010current: Jan 27 09:58:31 2010
diff: 17(s)hard: 20(s)soft: 16(s)
last: Jan 27 09:58:30 2010hard: 0(s)soft: 0(s)
current: 1120(bytes)hard: 0(bytes)soft: 0(bytes)
allocated: 14hard: 0soft: 0
sadb_seq=2 pid=3988 refcnt=1
203.178.143.26 203.178.128.57
esp mode=transport spi=256082823(0x0f438387) reqid=0(0x00000000)
E: rijndael-cbc  5218abf1 7e6ef87c 722e5956 14bc6ad6
A: hmac-sha1  307c8473 1e1835a5 4066f36a 73a20368 16d8abdc
seq=0x00000000 replay=64 flags=0x00000000 state=mature
created: Jan 27 09:58:21 2010current: Jan 27 09:58:31 2010
diff: 10(s)hard: 20(s)soft: 16(s)
last:                      hard: 0(s)soft: 0(s)
current: 0(bytes)hard: 0(bytes)soft: 0(bytes)
allocated: 0hard: 0soft: 0
sadb_seq=1 pid=3988 refcnt=1
203.178.128.57 203.178.143.26
esp mode=transport spi=206088265(0x0c48a849) reqid=0(0x00000000)
E: rijndael-cbc  fecef61e 40dc1407 350d599f dc62afe3
A: hmac-sha1  9b866f7e 0f3fee66 d16b9de5 5870b2cb d0fe1b69
seq=0x000007c6 replay=64 flags=0x00000000 state=mature
created: Jan 27 09:58:21 2010current: Jan 27 09:58:31 2010
diff: 10(s)hard: 20(s)soft: 16(s)
last: Jan 27 09:58:24 2010hard: 0(s)soft: 0(s)
current: 55720(bytes)hard: 0(bytes)soft: 0(bytes)
allocated: 1990hard: 0soft: 0
sadb_seq=0 pid=3988 refcnt=1
```

図 A.6: 複数の SA を複数の IPsec Gateway との間で張った様子その 2

付 録B　インターネットドラフト

IPsec Maintenance and Extensions
Internet-Draft
Intended status: Experimental
Expires: April 22, 2010

S. Nagayama
R. Van Meter
Keio University

October 19, 2009

IKE for IPsec with QKD
draft-nagayama-ipsecme-ipsec-with-qkd-00

Status of this Memo

This Internet-Draft is submitted to IETF in full conformance with the provisions of BCP 78 and BCP 79. This document may contain material from IETF Documents or IETF Contributions published or made publicly available before November 10, 2008. The person(s) controlling the copyright in some of this material may not have granted the IETF Trust the right to allow modifications of such material outside the IETF Standards Process. Without obtaining an adequate license from the person(s) controlling the copyright in such materials, this document may not be modified outside the IETF Standards Process, and derivative works of it may not be created outside the IETF Standards Process, except to format it for publication as an RFC or to translate it into languages other than English.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF), its areas, and its working groups. Note that other groups may also distribute working documents as Internet-Drafts.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

The list of current Internet-Drafts can be accessed at http://www.ietf.org/ietf/1id-abstracts.txt.

The list of Internet-Draft Shadow Directories can be accessed at http://www.ietf.org/shadow.html.

This Internet-Draft will expire on April 22, 2010.

Copyright Notice

Copyright (c) 2009 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal

Provisions Relating to IETF Documents in effect on the date of publication of this document (http://trustee.ietf.org/license-info). Please review these documents carefully, as they describe your rights and restrictions with respect to this document.

Abstract

Quantum Key Distribution (QKD) is a mechanism for creating shared, secret, random bits. This document describes extensions to the IKEv2 protocol to use random bits created via QKD as keys for IPsec. The Diffie-Hellman key agreement mechanism is replaced with QKD. The use of QKD-generated keys with standard IPsec will extend the lifetime of privacy guarantees for IPsec-protected data: future technological advances that break Diffie-Hellman key exchange will not disclose data until such time as the encryption algorithm used for the IPsec tunnel is broken.

Table of Contents

1. Quantum Key Distribution 4
2. Architecture and Assumptions 5
3. Data Formats and Information Exchange Sequences 7
  3.1. Data Formats 7
    3.1.1. QKD Key ID Payload 7
    3.1.2. QKD Fallback Payload 8
  3.2. Sequence 9
    3.2.1. Initializing IKE_SA 9
    3.2.2. Rekeying IKE_SA 11
  3.3. Considerations for Multiple SAs 13
4. Error Handling 14
5. Recommendations for use of QKD-generated keys 15
6. Security Considerations 17
7. IANA Considerations 18
  7.1. Payload Type Values 18
8. References 19
  8.1. Normative References 19
  8.2. Informative References 19
Appendix A. Implementation Considerations and Current Status . . 20
Authors' Addresses 21

1. Quantum Key Distribution

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [RFC2119].

Quantum key distribution (QKD) [BB84] creates shared, secret, random bits using quantum effects to guarantee that the probability of an undetected eavesdropper learning the secret bits is vanishingly small. Thus, the secret bits are a good source of cryptographic keying material. In the terminology proposed by the SECOQC Project[SECOQC07], a QKD network includes a "secrets plane" which delivers secret key material to other subsystems.

2. Architecture and Assumptions

This document describes modifications to IKEv2 to use keys created via QKD for the Internet Key Exchange IKE_SA[RFC4306], the key agreement protocol for IPsec[RFC4301] . With the exception of the use of the new Payloads defined below and the removal of the Diffie-Hellman key agreement information, IKEv2 system operates in standard fashion.

The system design is shown in Figure 1. Each side has an IPsec Gateway and a QKD Device. The IPsec Gateways are connected via an IP network and the QKD Devices are connected through the QKD network. The IP network and QKD network MAY share all, some, or none of the physical links comprising their networks, e.g. via wavelength multiplexing. Either end MAY initiate the QKD connection.

System Design

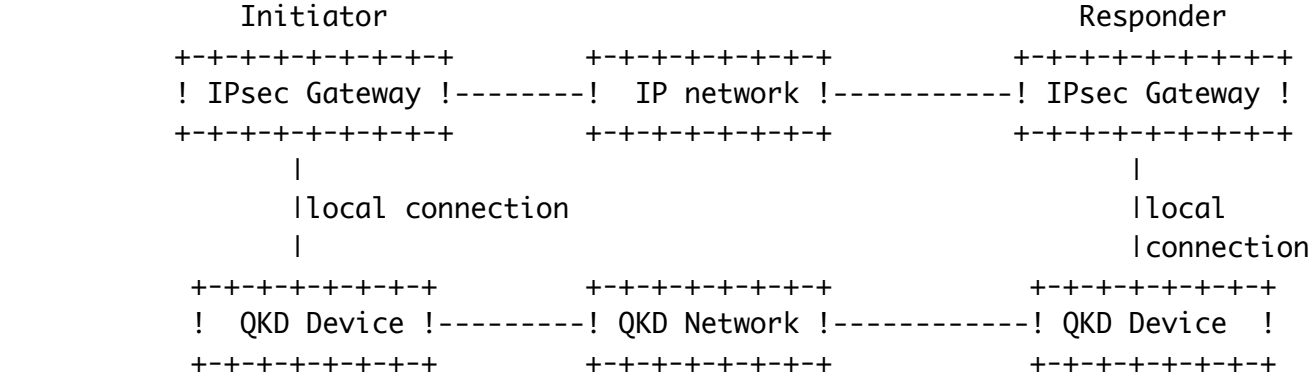


Figure 1

The connection between the IPsec Gateway and the QKD device, marked "local connection" MUST be secret, authenticated, and reliable. This MAY be achieved by incorporating both the IPsec Gateway and QKD device into a single system.

The QKD device SHALL provide secret, shared, random bits to the IPsec gateway. The bits MUST be shared with an authenticated partner only. The key material SHALL be managed in such a manner that the IPsec gateways can independently map a Key ID to matching key material. Beyond this, the interface between the IPsec Gateway and QKD device is beyond the scope of this document.

The technical details of the operation of the QKD network (including device physics, data filtering, node addressing, authentication, synchronization, etc.) are beyond the scope of this document. The QKD channel operates independently from the IP network that connects the IPsec gateways. QKD requires an authenticated classical channel

which is not part of the IPsec connection; this channel can be unencrypted. The key name (Key ID) is chosen by the QKD subsystem. It is the QKD subsystem's responsibility to ensure that key names are unambiguous, e.g. that key names are not reused within a time frame that can cause confusion.

3. Data Formats and Information Exchange Sequences

IKE must exchange two parameters to use QKD: an identifier indicating which QKD-generated key is to be used (KeyID) and the choice of fallback methods. One Key ID represents one unit of shared random bits, large enough for use as bulk data encryption key. Fallback methods are used when the QKD system key generation underruns.

3.1. Data Formats

3.1.1. QKD Key ID Payload

Figure 2 defines the payload for the QKD Key ID.

QKD Key ID Payload

Figure 2

The Next Payload field of the previous header MUST be set to the QKD Key ID payload number (see Section 7). The first 32 bits of the payload are the Generic Payload Header. To avoid a man-in-the-middle attack downgrading the negotiated security level, the Critical bit must be set to 1. The responder MUST reply with an error message when it it is incapable of using QKD (see Section 4).

- Key ID (four octets) is used to communicate which key to use for the encryption.
- Version (one octet) specifies the format and semantics of this message. The current version is 1.
- Flags holds flag bits; this field MUST be zero.
- N is the NoKey bit. 0 means that the KeyID field is valid. 1 means that the KeyID field is not valid, and the responder SHALL resort to the Fallback method, if one is specified in a Fallback Payload. This bit MUST be 0 for IKE_SA_INIT.

3.1.2. QKD Fallback Payload

The Next Payload field of the previous header MUST be set to the QKD Fallback payload number (see Section 7). The first 32 bits of the payload are the Generic Payload Header.

QKD Fallback Payload

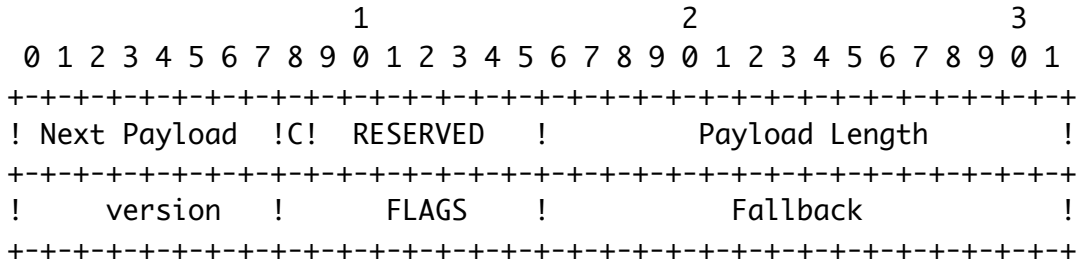


Figure 3

- Version (one octet) specifies the format and semantics of this message. The current version is 1.

- Flags holds flag bits; this field MUST be zero.

- Fallback field contains the configure of fallback methods. There are three fallback methods, listed in Table 1.

Fallback methods

Fallback method	Method Value
WAIT_QKD	1
DIFFIE-HELLMAN	2
CONTINUE	3

Table 1

The Fallback methods are as follows:

WAIT_QKD indicating that IKE MUST wait for the QKD device to deliver a new key. When the IPsec tunnel key lifetime expires, the system MUST stop encrypting packets and forwarding them across the network; the tunnel should be considered to be down.

CONTINUE indicating that IPsec MAY continue to use the most recent key until a new key becomes available.

DIFFIE-HELLMAN indicating that IKE SHALL generate a new key in the existing IKE_SA using Diffie-Hellman as defined in [RFC4718].

The Fallback payload is encrypted, relying on the security of the IKE_SA, which is guaranteed by QKD.

3.2. Sequence

To use QKD-generated keys, the Initiator and Responder must agree on a Key ID to use. This key will be used to encrypt the IKE_AUTH exchange, and does not change the IKE Sequence. Other parameters, defining the Fallback method, must be exchanged in IKE_AUTH, in the encrypted connection.

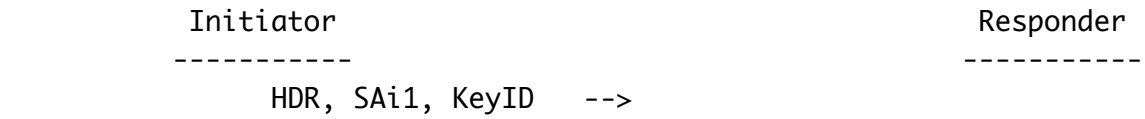
Standard IKEv2 exchanges key data for Diffie-Hellman in IKE_SA_INIT in a synchronous fashion. The principle difficulty in using QKD-generated secret bits as keys for IPsec tunnels is coordinating the activity of the QKD secrets plane with IKE, because the QKD device must operate continuously and independently to monitor its path and create secret bits, as discussed in Appendix A.

3.2.1. Initializing IKE_SA

When the initiator wishes to use QKD-generated keys, it MUST wait until the QKD device delivers one or more valid keys, shared with the responder, before sending the IKE_SA_INIT message. The initiator chooses a key and sends the Key ID in IKE_SA_INIT. The responder echos the Key ID. QKD fallback methods are exchanged in IKE_AUTH. The way to choose fallback methods follows IKE's algorithm to share configuration in [RFC4306] Section 2.7."Cryptographic Algorithm Negotiation".

The key negotation process is described below. Payload names in this document are to be interpreted as described in [RFC4306].

The IKE_SA_INIT with QKD



HDR and SAi1 are the IKE Header and a payload which states the cryptographic algorithms the initiator supports for the IKE_SA. KeyID is the QKD Key ID Payload described in Section 3.1.1. The

NoKey bit MUST be 0 in IKE_SA_INIT. The KEi and Ni payloads that are cantained in standard IKEv2 MUST be omitted because they are for Diffie-Hellman, and are not used with QKD.

```
                                  <--   HDR, SAr1, KeyID
```

Responder echos Key ID in its KeyID payload.

The IKE_AUTH with QKD exchange

```
 Initiator                                          Responder
-----------                                        -----------
 HDR, SK{IDi, [CERT,] [CERTREQ,] [IDr,]
       QKDfallback, AUTH, SAi2, TSi, TSr}  -->
```

The notation SK{...} means that payloads between {} are encrypted by the SA whose key is chosen in IKE_SA_INIT. QKDfallback payload is the QKD Fallback Payload, containing the initiator's proposed fallback method. IDi, AUTH, SAi2, TSi, TSr are payloads which state the initiator's identification, authentication and CHILD_SA's parameters and traffic selectors of initiator and responder.

```
                    <--  HDR, SK{IDr, [CERT,] QKDfallback
                                   AUTH, SAr2, TSi, TSr}
```

Responder replies with its acceptance of fallback methods in its QKD fallback payload. If the Responder does not agree with the Initiator's requested fallback method, it MUST respond with an error message and abort the IKE negotiation, as discussed in Section 4.

Normal Exchanges in initializing IKE_SA

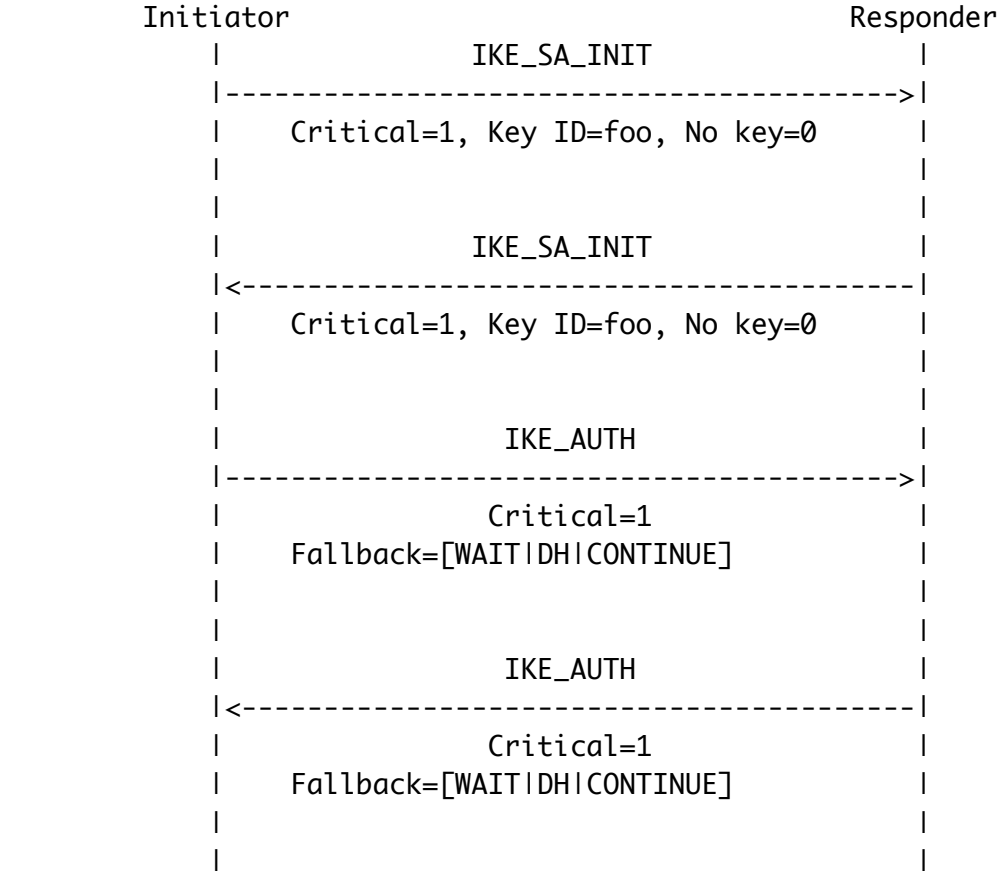


Figure 4

During initialization, IKE_SA cannot use a fallback method. The key must be generated by QKD. Thus, the Critical bit is set to 1. If the system underruns in key generation, it MUST wait for the QKD device to generate a new key.

3.2.2. Rekeying IKE_SA

In IKE two ways are defined to rekey an IKE_SA: repeating the original initiation sequence by exchanging IKE_SA_INIT and IKE_AUTH, and using the CREATE_CHILD_SA exchange. Because IKE_SA_INIT is exchanged without encryption, if the Initiator wishes to specify fallback behavior, it MUST create a child SA, rather than re-initialize.

The CREATE_CHILD_SA with QKD Exchange

```
 Initiator                                          Responder
------------                                       -----------
    HDR, SK{[N], SA, Ni, KeyID,
              [KEi,] [TSi, TSr]}  -->
```

Figure 5

The initiator sends CREATE_CHILD_SA including an IKE header, optionally a notify, a new security association, a nonce, Key ID for QKD, optionally a key exchange for Diffie-Hellman and optionally traffic selectors.

```
                          <--   HDR, SK{SA, Nr, KeyID,
                                     [KEr,] [TSi, TSr]}
```

The Responder sends CREATE_CHILD_SA which includes an IKE header, a new security association, a nonce, Key ID for QKD, optionally a key exchange for Diffie-Hellman and optionally traffic selectors.

The system SHOULD use QKD to rekey IKE_SA when possible. When the initiator rekeys using a new QKD-generated key, the KeyID payload from the initiator carries the new Key ID and the No key bit is set to 0. Responder repeats the Key ID and sets the No key bit set to 0. The Critical bits are ignored in this case. Both KEi and KEr MUST be omitted.

Normal Message Sequence in CREATE_CHILD_SA for rekeying IKE_SA

```
Initiator                                    Responder
    |              CREATE_CHILD_SA              |
    |------------------------------------------>|
    |                  Critical=1               |
    |           Key ID=foo, No key=0            |
    |                                           |
    |                                           |
    |              CREATE_CHILD_SA              |
    |<------------------------------------------|
    |                  Critical=1               |
    |           Key ID=foo, No key=0            |
    |                                           |
    |                                           |
```

Figure 6

When the SA lifetime nears expiration and it becomes necessary to rekey, if no QKD-generated key is available the Initiator SHALL rekey using the specified fallback method, if one was specified. The Initiator SHALL send the Fallback Payload with the No key bit set to 1. The initiator SHALL send the Key ID Payload with the KeyID field set to 0 and the NoKey bit set to 1. The Responder SHALL reply with the same Key ID and Fallback payloads. If Diffie-Hellman is permitted as a fallback method and the Perfect Forward Security(PFS) is configured to work, CREATE_CHILD_SA carries KEi and KEr.

Fallback Exchanges in CREATE_CHILD_SA for rekeying IKE_SA

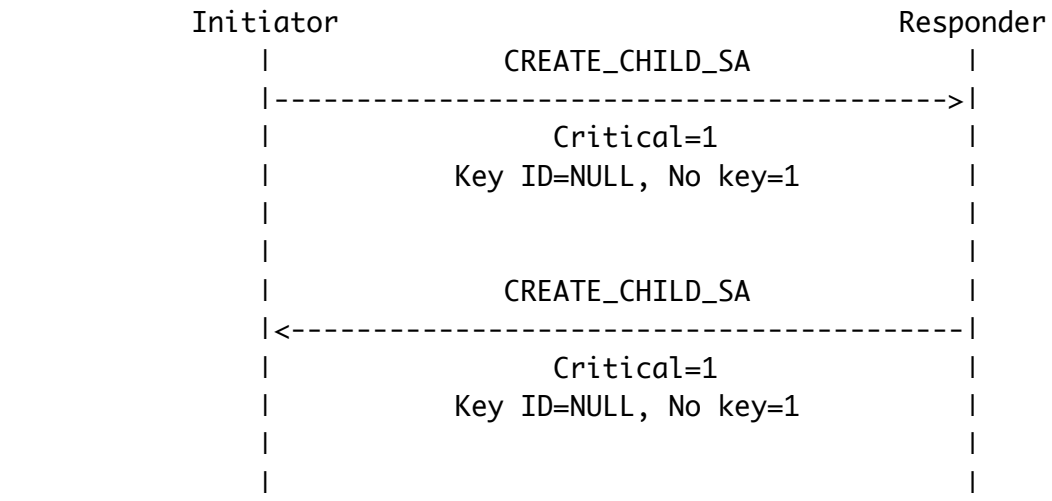


Figure 7

3.3. Considerations for Multiple SAs

IPsec can have multiple SAs between two IPsec gateways. QKD provides node-to-node keys, thus the system described in this document can also manage multiple SAs. The Initiator is free to use any QKD-generated keys for any SAs, but MUST NOT reuse any key.

4. Error Handling

Error handling beyond that already described in this document is TBD.

5. Recommendations for use of QKD-generated keys

From a data privacy point of view, the ideal use of QKD-generated keys would be as a one-time pad (OTP) to protect the data carried in the IPsec tunnel. However, as of 2009, QKD key generation rates are not adequate for high-speed OTP use; the QKD-generated keys instead will be used most commonly as key material for symmetric encryption of the IPsec tunnel. Thus, the upper bound on the secret lifetime of data remains the time until the chosen symmetric cipher can be broken. An eavesdropper who records encrypted packets today can store those packets, and decrypt them later by directly attacking the symmetric cipher, when it becomes technically feasible to do so.

However, existing IPsec/IKE implementations actually have a lower data secrecy lifetime, due to their dependence on Diffie-Hellman key agreement. The security of Diffie-Hellman depends on the difficulty of the factoring problem, which remains uncertain; factoring may prove vulnerable either to theoretical advances in algorithms, or the deployment of large-scale quantum computers. An eavesdropper who records encrypted packets today can store those packets, and decrypt them later by discovering the key, when it becomes technically feasible to do so.

QKD+IKE+IPsec offers a different point in the security space, providing secrecy under different assumptions about computational difficulty than D-H+IKE+IPsec, for all choices of IPsec tunnel encryption protocol.

In summary:

- QKD+IKE+IPsec depends on the availability of an authentication mechanism that is secure at the time of key negotiation.
- If QKD keys are used as an OTP, transferred data remains secret forever (or until disclosed through alternate means).
- If QKD keys are used for symmetric encryption, an eavesdropper may copy and store packets but cannot decrypt them until the symmetric cipher can be broken.

In contrast:

- D-H+IKE+IPsec depends on the availability of an authentication mechanism that is secure at the time of key negotiation.
- If the D-H keys are used for symmetric encryption, an eavesdropper may copy and store packets, and will be able to decrypt them when it becomes possible EITHER to factor large numbers (breaking the

D-H key agreement) OR to break the symmetric cipher.

Thus, QKD+IKE+IPsec can remove one uncertainty about the future evolution of computational security. If factoring is easier than breaking symmetric encryption, the use of QKD will extend the timeframe for maintaining the secrecy of data, even if standard, symmetric encryption is used for the bulk data encryption.

Key lifetime could be matched to QKD key generation rate; the mechanism is not specified here.

6. Security Considerations

Because QKD's principal role is to detect eavesdropping and discard possibly compromised bits, eavesdropping is a very effective denial of service (DoS) attack. One purpose of the fallback behavior negotiation is to provide network managers with a tool for alleviating this problem. Fallback methods should be used with extreme care, and SHOULD be coupled with event notification and monitoring.

One possible practice would be to define the fallback policy for an SA carrying user traffic as WAIT_QKD, and define a second, primarily dormant, SA with a more liberal fallback policy for a management station. The second SA might be used only to diagnose problems and for low-security network monitoring and management activity until the QKD connection can be restored.

This document describes a form of Internet Key Exchange protocol which is not based on the difficulty of factorization. Thus, under the circumstances described in Section 5, security may be improved.

The system consists of two logically separate channels: a classical channel between IPsec gateways and a quantum channel between QKD-devices. The QKD devices require a classical channel and authentication to prevent a man-in-the-middle attack. One keys are securely transferred to the IPsec gateway, those keys could be used as an alternative method for authenticating the IPsec gateways. Careful integration of the classical and quantum networks could eliminate authentication on one path by sharing the authentication information from the other; such a use is not specified here.

7. IANA Considerations

The following new assignments can only be made via a Standards Action as specified in [refs.IANA].

7.1. Payload Type Values

The IANA should allocate IKE Payload Type Values for the QKD Key ID Payload and the QKD Fallback Payload upon publication of the first RFC.

8. References

8.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[RFC4301] Kent, S. and K. Seo, "Security Architecture for the Internet Protocol", RFC 4301, December 2005.

[RFC4306] Kaufman, C., "Internet Key Exchange (IKEv2) Protocol", RFC 4306, December 2005.

[RFC4718] Eronen, P. and P. Hoffman, "IKEv2 Clarifications and Implementation Guidelines", RFC 4718, October 2006.

[refs.IANA] Narten, T. and H. Alvestrand, "Guidelines for Writing an IANA Considerations Section in RFCs", RFC 2434, October 1998.

8.2. Informative References

[BB84] Bennett, C. and G. Brassard, "Quantum cryptography: Public key distribution and coin tossing", 1984.

[EPT03] Elliot, C., Pearson, D., and G. Troxel, "Quantum cryptography in practice", 2003.

[SECOQC07] Alleaume, R. and et al., "SECOQC White Paper on Quantum Key Distribution and Cryptography", 2007.

[UQC09] Dodson, D. and et al., "Updating Quantum Cryptography Report ver. 1", 2009.

Appendix A. Implementation Considerations and Current Status

As of 2009, available QKD products use single photons over dedicated optical fibers and are limited in distance. Experimental demonstrations of wireless links and multi-hop networks using trusted intermediate nodes have been conducted [EPT03]. Progress is also being made toward use of satellite links and quantum entanglement-based networks of quantum repeaters that will not require trusting intermediate nodes [SECOQC07][UQC09].

In general, because QKD relies heavily on statistical evidence to determine the presence of an eavesdropper, it requires time to create a key. Thus, the IPsec implementation should be prepared for a long delay before keys become available. Moreover, the key generation rate may vary over time, typically rising over a long period from the initiation of a connection as statistical certainty improves, then settling near a sustained value around which the rate may vary as conditions change.

Authors' Addresses

Shota Nagayama
Keio University
Faculty of Policy Management
5322 Endo
Fujisawa, Kanagawa 252-8520
Japan

Email: kurosagi@sfc.wide.ad.jp

Rodney Van Meter
Keio University
Faculty of Environment and Information Studies
5322 Endo
Fujisawa-shi, Kanagawa-ken 252-8520
Japan

Email: rdv@sfc.wide.ad.jp