2005－CSEC－29（8）
2005／5／19

ストリーム暗号の適応による OpenVPNの高速化と操作性向上について

元家 宏美† 大東 俊博† 白石 善明‡ 森井 昌克††

† 徳島大学工学部知能情報工学科〒 770-8506 徳島市南常三島町 2-1

‡ 近畿大学理工学部情報学科〒 577-8502 東大阪市小若江 3-4-1

†† 神戸大学工学部電気電子工学科〒 657-8501 神戸市灘区六甲台 1-1

E-mail: †{motoie,ohigashi}@is.tokushima-u.ac.jp, ‡zenmei@info.kindai.ac.jp, ††mmorii@kobe-u.ac.jp

あらまし 地理的に離れた拠点間で仮想的な専用ネットワークを構築するための VPN という技術がある．特に SoftEther 等のソフトウェアで実現する VPN は特別な機器を必要とせず安価に導入できるため注目されている．本稿ではソフトウェアのみで VPN を構築可能な GPL ソフトウェアである OpenVPN を対象とし，通信速度に対するオーバーヘッドを軽減し，使用者の操作負荷を低減する方法について述べる．設定をサーバに委託し，クライアント側での特別な設定を不要にすることで操作負荷の低減を図っている．そして，ブロック暗号だけをサポートしている OpenVPN に対して高速なストリーム暗号を適用することにより，通信全体のオーバーヘッドの軽減を実現している．

Speed-Up by Adapting Stream Cipher and an Improvement of Usability for OpenVPN

Hiromi MOTOIE†, Toshihiro OHIGASHI†, Yoshiaki SHIRAISHI‡, and Masakatu MORII††

†Department of Information Science and Intelligent System, The University Tokushima
2-1 Minamijyousanjima, Tokushima-shi 770-8506, Japan.

‡Department of Informatics, Kinki University
3-4-1 Kowakae, Higashi-Osaka 557-8502, Japan.

††Department of Electrical and Electronics Engineering, Kobe University
1-1 Rokkodai, Nada, Kobe 657-8501, Japan.

E-mail: †{motoie,ohigashi}@is.tokushima-u.ac.jp, ‡zenmei@info.kindai.ac.jp, ††mmorii@kobe-u.ac.jp

Abstract Virtual Private Network (VPN) is a technology to construct a private network over public networks. Software-based VPN products are popular, because they don't need any appliance. OpenVPN is one of the most popular software-based VPN products and has high flexibility. However, the usability of OpenVPN is not so high because its setting requires expert knowledge of VPN. Additionally, the throughput decreases by the overhead of encryption because the encryption time of block cipher is large. This paper presents a method to improve the usability. A client can set VPN up on a VPN setting server without expert knowledge. In addition, we demonstrate that the overhead of the encryption is reduced by adapting software-oriented stream ciphers.

1 まえがき

ブロードバンド通信環境の普及により，大容量のデータの送受信が安価に利用できるようになっている．また様々な場所からインターネットに接続することが可能となったことから，自宅や出先など遠隔地から企業内部のリソースへアクセスするリモートアクセスのニーズが高まってきている．公共のネットワークを経由して社内などの内部ネットワークに対して安全にアクセスを行うための技術としてVPN(Virtual Private Network) が注目されている．VPN は公共のネットワークを利用して仮想的に専用回線のような安全なネットワークを構築する技術である．VPN は主にトンネリングと暗号化技術の 2 つの技術から成り立っている．トンネリングはある通信プロトコルを別のプロトコルで包み込むカプセル化と呼ばれる方法を用いて，ネットワーク上の拠点間を接続するための技術である．暗号化はカプセル化したデータを第三者には分からないように変換し，データの改ざんや盗聴を防ぐための技術である．

VPN 通信によって出先から内部ネットワークにリモートアクセスする場合，特別な機器を必要とせずソフトウェアのみで接続できることが望まれる．また内部ネットワークのリソースへアクセスを考える場合，カプセル化の対象となるパケットは OSI 参照モデルの下位層であることが望ましい．なぜなら下位層を対象とすることで上位層のプロトコルで動作する全てのソフトウェアに対して VPN を適用することが可能となるからである．これらの条件を満たし，かつ幅広い接続オプションで柔軟な VPN を構築することの出来るソフトウェアとして OpenVPN[1] がある．OpenVPN は GPL(General Public Lisense)[2] に基づき配布されるオープンソースソフトウェアである．カプセル化の対象は IP パケットか ethernet フレームであり，それらを TCP もしくは UDP プロトコルで包み込んで通信を行う．OpenVPN の最大の特徴は多様な接続オプションを持ち，様々な状況での VPN 構築に対応可能なことである．また OpenVPN は OpenSSL[3] ライブラリによってサポートされたアルゴリズムを暗号化・認証に利用でき，暗号化の処理に関してはブロック暗号のみの共通鍵方式か TLS(Transport Layer Security)[4] 認証ベースの公開鍵方式が選択可能である．

OpenVPN は多様な設定を行えるという特徴を持っているが，設定方法が繁雑なために利便性が高いとはいえない．また，共通鍵方式においてブロック暗号のみのサポートでは VPN の通信を行う際の速度に関して十分な配慮がなされているとは考え難い．本稿では OpenVPN を用いた VPN 構築方法とその問題点について述べ，解決策としてサーバ側でクライアントの設定ファイルを生成することでクライアント側で設定を意識せずに VPN を構築する方法を提案する．また暗号化による通信のオーバーヘッドを減らすためにブロック暗号よりも高速とされているストリーム暗号を実装し，速度調査を行う．実装し評価を行う暗号アルゴリズムは RC4[5] と SSSM[6] である．RC4 は内部状態遷移型ストリーム暗号の一つであり，ストリーム暗号の中でも最も広く利用されている．SSSM は 32 ビット演算の CPU を考慮して設計された内部状態遷移型ストリーム暗号であり，最も高速なストリーム暗号の一つである．これらの提案・実装により利用者側の設定負荷の軽減，そしてオーバーヘッドによる通信速度の低下を解消する．

2 OpenVPN

OpenVPN は様々な接続オプションを持ち，柔軟性が高く，様々な状況に対応した VPN が構築可能である．本章では OpenVPN を用いた VPN の構築方法と共通鍵方式での暗号化の概要について述べ，それぞれの問題点を説明する．

2.1 VPN の構築方法

OpenVPN を用いた通信の流れを図 1 に示す．OpenVPN は仮想ネットワークデバイス TUN/TAP[7] からデータを受け取り暗号化・認証等の処理を行い，TCP/UDP プロトコルを利用して相手先に送り出す．TUN/TAP はユーザが VPN 通信の相手先に送信しようとするデータを受け取り OpenVPN に渡す役割を持っている．VPN を構築するためには必要な接続オプションを設定ファイルに記述しなければならない．接続オプションの一部を以下に示す．

dev 仮想デバイスの設定である．tun を選択した場合は IP パケット，tap を選択した場合 Ethernet フレームをカプセル化の対象とする．

ifconfig VPN で用いるアドレスの設定である．

secret 共通鍵ファイルの名前の設定である．ファ

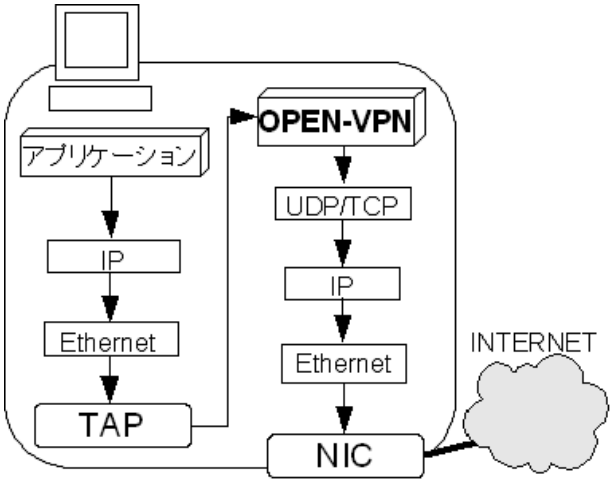

図 1: OpenVPN を用いた場合の VPN 通信の流れ

サーバ側

```
dev tun
ifconfig 10.0.0.1 10.0.0.2
secret /home/test.key
proto udp
port 5500
```

クライアント側

```
dev tun
ifconfig 10.0.0.2 10.0.0.1
secret /home/test.key
proto udp
port  5500
remote server.vpn
```

図 2: 図 3 の VPN を構築するための設定例

イル名は絶対パスで示す.

port カプセル化したパケットを通すポート番号の設定である.

proto 相手先に送るためのカプセル化のプロトコルの設定である.

remote クライアント側でサーバ側のホスト名 (IP アドレス) を指定する.

図 2 は共通鍵方式を用いて図 3 のような VPN を構築するための設定方法を示したものである. 設定ファイルの記述は 1 行に 1 つのオプションとそれに関する引数を記述する.

2.2 暗号化概要

共通鍵方式で暗号化処理を行う場合, 事前に両端末で共通鍵を共有しておく必要がある. デフォルトの暗号アルゴリズムは Blowfish が選択されている. OpenVPN の共通鍵方式では OpenSSL ライブラリに含まれているブロック暗号であれば選択可能である. ブロック暗号のモードは初期設定では CBC モードとなっている. また利用する共通鍵はセッションが変わっても同じものを使い続ける.

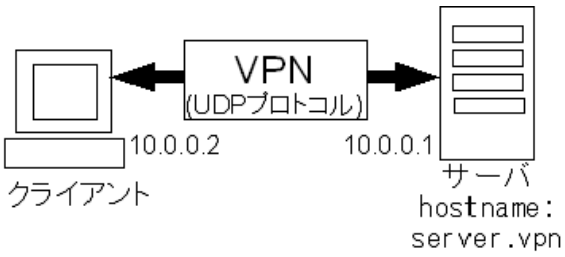

図 3: VPN の接続状態の例

2.3 問題点

OpenVPN を用いて通信を行う際の問題点を以下に述べる.

問題点 1 設定が煩雑である.

OpenVPN の特徴は設定項目が多い事であるが, 項目が多岐に渡るため利用者が必要とする項目を見つけ出し理解をすることは容易ではない. 一部のオプションは暗号化の方式やオプションの組合せによって設定方法が異なったり利用出来ない場合があるため, 利用者は OpenVPN の設定方法に対してある程度の知識を要求される. またファイルに記述を行う設定方法は CUI や記述方法を理解していない利用者には敷居が高い.

問題点 2 クライアント側でサーバの設定を把握する必要がある.

OpenVPN ではサーバ側だけでなくクライアント側でも接続するための何らかの設定オプションの記述を行わなければならない. また一部オプションに関してはサーバ側の設定に依存した設定を行う必要がある. 具体例として proto オプションが挙げられる. サーバ側で UDP プロトコルを指定した場合クライアント側でも UDP を指定しなければならない.

問題点 3 共通鍵が更新されない.

OpenVPN では共通鍵は一度共有すれば明示的に変更をしない限り更新されない. そのため同じ秘密情報を長期に渡って使用することとなり, 時間とともに危険性が高まる.

問題点 4 ブロック暗号のみのサポート．

VPN 通信を行う場合，通信を行うための処理がオーバーヘッドとなる．オーバーヘッドを考慮する場合，暗号化処理では計算量が少なく，かつ安全である暗号を選択する必要がある．OpenVPN はブロック暗号のみ選択可能であり，高速性に優れているストリーム暗号を選択することは出来ないためオーバーヘッドの軽減には不向きである．

3 OpenVPNの問題点の解決方法

前章で述べた OpenVPN の問題点に対して，設定方法と安全性，通信速度の 3 つの項目に分けて解決方法を考察する．本考察を踏まえ，次章以降で OpenVPN の問題点を解決するための具体的な方法を提案する．

- 設定方法 **(問題点 1，2 の解決)**
  設定に関する最大の問題はクライアント側でも図 2 のような設定を行わなければならないことである．この問題を解決する方法としてサーバ側から設定項目を受け取る方法が考えられる．サーバから設定を受け取ることができればクライアントは設定を行う必要はなく，サーバの設定を把握する必要もなくなる．既存の方法として OpenVPN の公開鍵方式ではサーバの仮想アドレス等を受け取る方法がある．しかしその場合，クライアント側でサーバから設定を受け取るためのオプションや証明書などの設定を記述しなければならないという問題が生じる．そこで提案方法では設定を渡す際にクライアントが意識せずに設定を行えることを考慮する．

- 安全性 **(問題点 3 の解決)**
  共通鍵に関する問題に対しては，セッション毎に暗号化に用いる共通鍵を変更するのが望ましい．解決方法として安全にセッション鍵の生成する方法を考える．

- 通信速度 **(問題点 4 の解決)**
  暗号化によるオーバーヘッドを低減させるためには，高速なストリーム暗号を実装することが有効である．ストリーム暗号はブロック暗号よりも高速な暗号とされている．実際にストリーム暗号の中で広く利用されている RC4 は OpenVPN でデフォルトに利用されている Blowfish-CBC よりも高速に処理を行える．

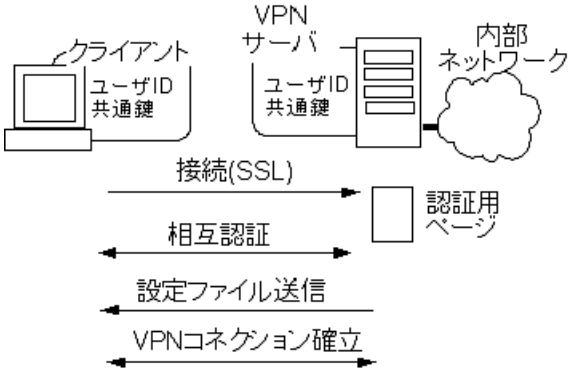

図 4: 設定ファイルの受け取り手順

設定方法の提案については 4 章で，安全性，通信速度については 5 章で述べる．

4 サーバ委託によるクライアントの設定負荷の軽減

本章ではクライアントが設定ファイルの記述や詳細を意識せずにサーバとの VPN 通信を可能とする方法を提案する．提案する方法では VPN サーバでクライアントの設定ファイルを生成することで，クライアントは余計な手間をかけず設定を行うことができる．具体的には提案手法ではクライアントが設定ファイルを受け取るために図 4 に示した以下の処理を行う．前提としてサーバ側で共通鍵を生成しユーザ ID とともに登録をしておき，クライアント側は OpenVPN のインストールを済ませ，ユーザ ID と共通鍵をサーバから受け取っておくものとする．

Step1 クライアント側から設定ファイル配布用 Web ページに SSL 通信で接続する．

Step2 設定ファイル配布用 Web ページ上でユーザ ID を入力し，送信する．

Step3 入力されたユーザ ID と両端末で所持している共通鍵を用いてチャレンジ&レスポンスなどの相互認証プロトコルを利用して認証を行う．

Step4 認証完了後設定ファイルをクライアントに送る

表 1: 実装環境

CPU	PentiumIII 1GHz
メモリ	512MBytes
OS	Vine Linux ver2.6
言語	C
コンパイラ	gcc ver2.95

表 2: 暗号化速度

アルゴリズム	速度 (Mbps)
SSSM	556
RC4	262
AES	141

Step5 受け取った設定ファイルを利用してクライアントは VPN サーバに接続する．

このようにすることでクライアントは OpenVPN をインストールし Web ページでユーザ ID を入力するだけでサーバとの VPN 接続が行えるようになる．これにより煩わしい設定やサーバ側の設定を考慮する必要がなくなる．

5 ストリーム暗号実装による暗号化処理の高速化

ストリーム暗号アルゴリズムである RC4 と SSSM を OpenVPN に実装する．RC4 と SSSM はストリーム暗号の中でも内部状態遷移型の擬似乱数生成を用いた暗号方式である．RC4 は最も普及しているストリーム暗号である．SSSM は 32 ビット演算を考慮した内部状態遷移型ストリーム暗号の中でも高速な部類に属する暗号アルゴリズムである．SSSM，RC4，ブロック暗号である AES を表 1 の環境上でソフトウェア実装した場合の速度は表 2 のようになっている．本章ではストリーム暗号の OpenVPN への実装方法とその安全性，速度測定結果について述べる．

5.1 実装方法

本節では共通鍵方式の暗号アルゴリズムを選択する際，ストリーム暗号を選択できるように実装を行う．実装した暗号化の処理手順を以下に示す．

Step1 セッション鍵を用いて内部状態の初期化を行う．(セッション開始時のみ)

Step1-1 128bits の IV(乱数) を生成

Step1-2 OpenSSL ライブラリの HMAC 関数に共通鍵と IV を入力し 128 bits のセッション鍵を生成する．

Step1-3 セッション鍵を用いて内部状態の初期化を行う

Step2 送信するデータの長さを求め，作業用バッファの領域を確保する．

Step3 作業用バッファに平文をコピーし，作業用バッファ上で暗号化を行う．

Step4 暗号化を行った作業用バッファから平文を取り出し出力用バッファにコピーする．

Step5 セッション開始時に，IV を両端末で共有するため最初のヘッダに IV を付加する．

5.2 安全性に対する考察

2 章の問題点 3 で述べたように OpenVPN の共通鍵方式では鍵の更新が行われない．そこでセッション鍵という概念を導入した．安全にセッション鍵を利用するためには，セッション鍵の生成方法と IV 長について注意する必要がある．無線通信に利用される WEP(Wired Equivalent Privacy) の場合，IV と鍵を連結してセッション鍵が生成されていた [8]．しかしこの生成方法を用いる場合，関連鍵攻撃が可能になってしまうため攻撃者に有利な条件を与えてしまう．そこで本稿ではハッシュ関数を利用してセッション鍵を生成することにより，関連鍵攻撃が困難になるようにしている．IV 長はランダムに IV が生成されるときに重複する確率を考慮して決定する必要がある．なぜならストリーム暗号では同一のセッション鍵が利用された時，片方の平文が判明した場合にもう片方の平文が推測されてしまう問題があるからである．したがって，同じ IV を用いることで同じセッション鍵を生成してしまうことは避けねばならない．IV をランダムに生成する場合，Birthday Paradox として知られている性質より，IV 長が l bits のとき $2^{\frac{l}{2}}$ 個の IV を発生させると 50 %という高い確率で重複してしまう．OpenVPN のブロック暗号の CBC モードでの IV 長は 48bits となっているが，

表 3: 実験環境

	PC1	PC2
OS	FedoraCore 1	Turbolinux 8.0
CPU	Pentium 4 1.7GHz	Pentium Ⅱ 450MHz
メモリ	512MBytes	256MBytes
コンパイラ	gcc ver3.3.2	gcc ver2.96
OpenSSL	OpenSSL 0.9.7b	OpenSSL 0.9.7a
OpenVPN	OpenVPN 2.0-rc6	OpenVPN2.0-rc6

表 4: 通信速度測定結果 (単位:Mbps)

	UDP	TCP
SSSM	36	33
RC4	31	26
Blowfish-CBC	29	25
AES-CBC	21	20
暗号化なし	46	41

このデータ長の IV を使用した場合，2^{24} 回生成するだけで重複が生じる可能性がある．そこで IV のデータ長を 128bits として，安全性を高めている．

5.3 速度測定結果

OpenVPN へストリーム暗号を実装し，速度測定を行った．速度の比較対象とするアルゴリズムは提案手法によって実装した RC4，SSSM，そしてブロック暗号である Blowfish，AES とする．Blowfish，AES は CBC モードで暗号化処理を行う．測定を行う PC の環境を表 3 に示す．速度の調査方法は 10Mbytes データを VPN 上で FTP 転送しその転送量を測定する．PC1 からデータを暗号化し VPN を利用して PC2 へ送り，PC2 でデータを復号し受け取る．この一連の処理にかかる時間を測定し，その平均値を求める．速度測定の結果を表 4 に示す．表 4 よりデフォルトで設定されている Blowfish は平文で通信を行う時よりも速度が 4〜6 割程度低下しているのに対し，ストリーム暗号では 6〜8 割程度となっている．このことからストリーム暗号を実装することによるオーバーヘッドの軽減が行えたことがわかる．

6 むすび

VPN をソフトウェアで構築できる OpenVPN には，様々なオプションが存在する．しかし設定オプションの利用方法が煩雑であったり，共通鍵方式でブロック暗号しか対応していないなどいくつかの問題点を有していた．本稿ではそれらの問題点に関する解決策として，サーバから設定ファイルを手に入れる方法とストリーム暗号の実装を提案した．設定ファイルをサーバに委託する方法によってクライアント側での煩雑な設定を無くし，ストリーム暗号を実装することにより暗号化部分の処理を高速化し通信全体のオーバーヘッドの軽減が可能となった．

参考文献

[1] OpenVPN, *available at http://openvpn.sourceforge.net/*

[2] GNU プロジェクト, *available at http://www.opensource.jp/gpl/gpl.ja.html*

[3] OpenSSL, *available at http://www.openssl.org/*

[4] Transport Layer Security (tls) Charter, *available at http://www.ietf.org/html.charters/tls-charter.html/*

[5] B. Schneier, Applied Cryptography, Wiley, New York, 1996.

[6] 鵜川三蔵，大東俊博，元家宏美，白石善明，森井昌克，“内部状態遷移型ストリーム暗号 SSSM の提案 (第 2 報),” SCIS2005, pp.403–408, January 2005.

[7] universal TUN/TAP driver, *available at http://vtun.sourceforge.net/tun/*

[8] IEEE Computer Society, “Wireless Lan Medium Access Control (MAC) and Physical Layer (PHY) Specifications,” IEEE Std 802.11, 1999 Edition, available at *http://standards.ieee.org/reading/ieee/std/lanman/.*