副テーマ研究報告書
隔離ネットワークにおける通信制御の調査

1910225 嶂南　秀敏

副テーマ指導教員　知念　賢一　特任准教授

2020 年 9 月 15 日

概 要

現在、ユーザの遠隔サーバへの利用を容易にするアプリケーションは多く存在する。その一方で、サーバーへの通信接続をシステム管理者が容易に制御管理するためのアプリケーションは少ない。そのような通信制御のアプリケーションなしでは、システム管理者はユーザーの接続ごとに手動でコマンドを実行し管理しなければならず、サーバーを利用するユーザーが多くなるに従いシステム管理者への負担が多くなってしまう。

そこで、本研究ではサーバーへの通信制御に有効なアプリケーションを調査することを目的とし、三台のサーバーと一台の L2 スイッチを用いて隔離ネットワークを構成し、ゲートウェイサーバーに個人のソースコード投稿サイトに公開されている通信制御のアプリケーションを試用することで、通信制御の容易さ、通信の可視化度合い、セキュリティ、三点について検討を行う。

目 次

第1章　はじめに

1.1　研究背景

クラウドサーバーサービスの普及により、一般ユーザーが遠隔サーバーにアクセスし、その資源を利用する機会が増えてきた。Google のクラウドサービス (GCP)、Amazon のクラウドサービス (AWS)、Microsoft のクラウドサービス (Azure) などのようにユーザーの目的に合わせて、遠隔サーバーの利用を容易にするアプリケーションが多く存在する。

しかし、一方でサーバーへの通信接続をシステム管理者が容易に制御管理するためのアプリケーションは少ない。そのような通信制御のアプリケーションなしでは、システム管理者はユーザーの接続ごとに手動でコマンドを実行しなければならず、サーバーを利用するユーザーが多くなるに従いシステム管理者への負担が多くなってしまう。

令和２年現在新型コロナウィルス感染症の世界的大流行により在宅勤務 (テレワーク) が推奨され、自宅から社内サーバーへのアクセスを余儀なくされている。そして、同年７月下旬に政府が在宅勤務７割を企業に要請した [1] ことにより、今後ますますテレワークの人口が増大していきそれに伴い、大規模なリモートアクセスにおけるセキュリティがより重要になり、社内のシステム管理者の負担がますます増加していくことが予想できる。

1.2　研究目的

総務省テレワークセキュリティガイドライン [2] には、テレワークの利用方法が６パターンに分類されており、それとともに対策すべき観点もまとめられている。それを以下の表 1.1 に示す。下表 1.1 によると６パターンのうち 1、2、6 の３パターンが外部から社内サーバにアクセスを必要とする「オンプレミス型」のもので、3、4、5 の３パターンはインターネット上のクラウドサービスを利用している「オフプレミス型」である。

それぞれのパターンの細かな違いは、テレワーク利用端末にデータを保存できるかなどであり、オンプレミス型もオフプレミス型どちらもインターネットを介して、サーバにアクセスしている。

	パターン1	パターン2	パターン3	パターン4	パターン5	パターン6
	リモートデスクトップ方式	仮想デスクトップ方式	クラウド型アプリ方式	セキュアブラウザ方式	アプリケーションラッピング方式	会社PCの持ち帰り方式
概要	オフィスにある端末を遠隔操作	テレワーク用の仮想端末を遠隔操作	クラウド上のアプリケーションを社内外から利用	特別なブラウザを用いて端末へのデータ保存を制限	テレワーク端末内への保存を不可とする機能を提供	オフィスの端末を持ち帰りテレワーク端末として利用
テレワーク端末に電子データを保存するか？	保存しない	保存しない	どちらも可	保存しない	保存しない	保存する
オフィスの端末と同じ環境を利用するか？	同じ	テレワーク専用の環境	クラウド型アプリに関しては同じ	ブラウザ経由で利用するアプリに関しては同じ	テレワーク専用の環境	同じ
クラウドサービスを利用するか	しない	しない	する	する	する/しない どちらも可	する/しない どちらも可
高速インターネット回線の必要性	必須	必須	望ましい	望ましい	望ましい	不要

表 1.1: テレワークの 6 種類のパターン

本研究では「トンネリング」や「VPN」などのテレワークの根底にある技術の解説を行うと同時に、社内ネットワークを想定した、隔離ネットワークを構成し、そこにソースコード投稿サイトに公開されている通信制御アプリケーションを試用することで、その中で小規模と大規模アプリケーションで分類しソフトウェアの検討を行う。

本研究でまとめることで、ブラックボックス的にリモートアクセスを行うのではなく、どのような技術や仕組みを用いて実現しているか、高いセキュリティの実現のためにどのようなプロトコルを用いているかを、リモートアクセス利用者の理解を深めることを目的とする。それと同時にシステム管理者が通信接続の制御をすることを容易にするためのアプリケーションを調査する。

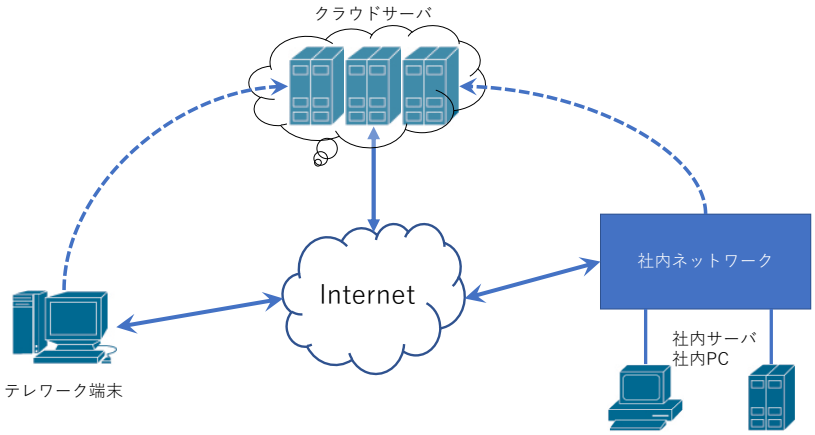

図 1.1: オフプレミス型

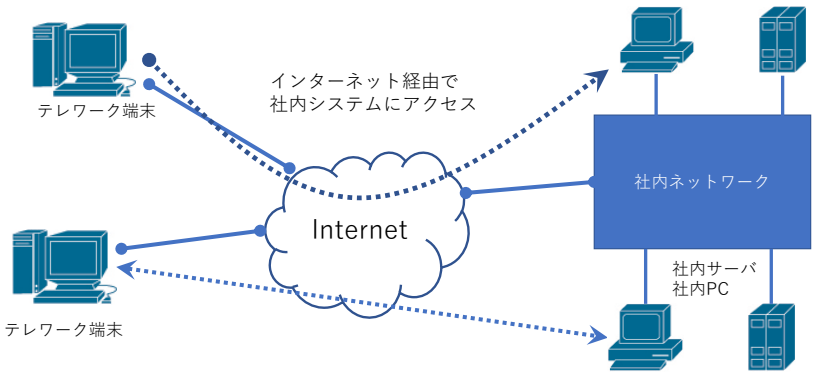

図 1.2: オンプレミス型

第2章　遠隔アクセス技術

本章では遠隔サーバーに接続し操作するための技術やソフトウェアの解説を行う。

2.1　暗号化技術

暗号化技術は、情報の保護やコンピュータセキュリティに欠かせない技術である。通信内容の保護のために、一般的に次に示す二種類の暗号化技術を使用して、認証及び暗号化通信を行っている。

- 共通鍵暗号方式
- 公開鍵暗号方式

それぞれの暗号方式は様々なアルゴリズムによって実現されるが、元となる「平文」データを「鍵」を使って「暗号文」に変換している。また、「暗号文」は「鍵」を使用して、「平文」に復号できる。

2.1.1　共通鍵暗号方式

A と B で共通の鍵を使用して、暗号化と復号化を行う。そのため、暗号化通信をする前にこの共通鍵を事前に秘密に共有する必要がある。共通鍵暗号方式の暗号化を下図 2.1 に示す。

共通鍵暗号方式は、後述の公開鍵暗号方式に比べて、演算の処理量が少ないとういう利点ある。そのため、SSH プロトコルでは、通信内容の暗号化にはこの共通鍵暗号方式を採用している。

代表的な暗号化アルゴリズムに「AES」が存在する。

2.1.2　公開鍵暗号方式

公開鍵暗号方式は、二種類の鍵である公開鍵と秘密鍵をペアで使用する。この公開鍵と秘密鍵には次に示す性質があり、公開鍵暗号方式はこれらの性質を利用して暗号化や署名を実現している。

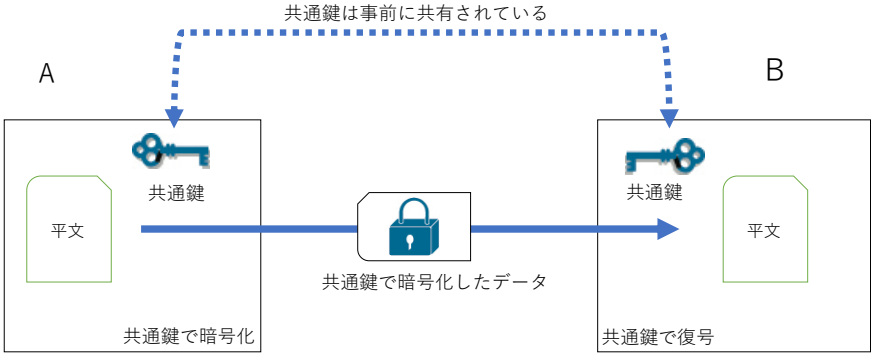

図 2.1: 共通鍵暗号方式での暗号化

- 公開鍵で暗号化した平文は、秘密鍵で復号できる
- 公開鍵で暗号化した平文は、公開鍵では復号できない
- 秘密鍵で暗号化したデータは、公開鍵で復号できる
- 公開鍵から秘密鍵を生成できない。

公開鍵暗号方式での暗号化について下図 2.2 に示す。この図では、鍵ペアを作成した B が、公開鍵を A に公開してる。A は、B の公開鍵を使用して平文を暗号化して、B へ送付する。送付された暗号文は、B 自身の秘密鍵でのみ復号できる。

代表的な暗号化アルゴリズムに「RSA 暗号」「Diffi-Hellman 鍵共有」等がある。

2.2 トンネリング (Tunneling)

ここでは、トンネリングの概要と 2.5 節に関係する VPN トンネリングプロトコルの説明を行う。

トンネリングとは

トンネリングとは、「パケットやフレームを他のパケットやフレームの中にカプセル化すること」である。これにより、プライベートアドレスの隠蔽や、カプセル化後に暗号化をすることでデータの保護を行うことがで

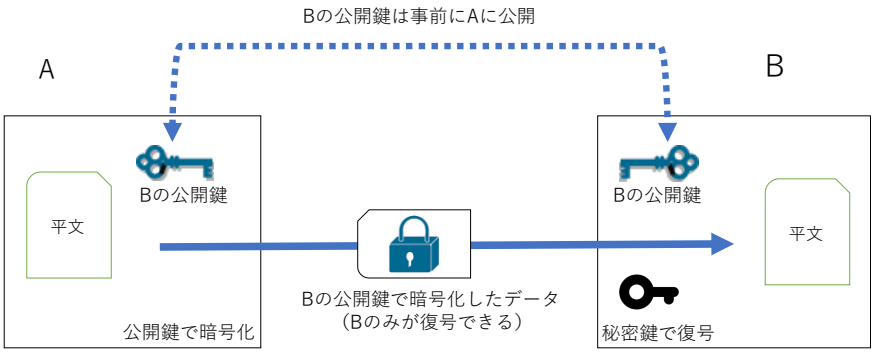

図 2.2: 公開鍵暗号方式での暗号化

きる。後に説明する VPN においてもトンネリング技術は必要不可欠なものとなっている。ただし、「トンネリング=VPN」であるわけではないというのは重要な点である。

トンネリングの機能を以下に示す。

- データ転送の容易化
  トンネリングには、パケットまたはフレーム全体を直接特定の場所に転送する、「宛先を切り替える」機能を持つ。これにより、それらが宛先ネットワークに到着するとそのネットワークを管理している組織のセキュリティやネットワークポリシーによる制御を受けることができる。

- 組み込みセキュリティ機能
  トンネリングプロトコルの中では、追加セキュリティ（暗号化、認証など）がプロトコルに組み込まれているものがある。そのようなプロトコル（プロトコル群）の代表例として「IPSec」、「L2TP」や「PPTP」がある。

2.2.1 IPSec

IPSec（IP Security Architecture）は、IP パケットのデータの完全性 (Integrity) と機密性 (Confidentiality) を保証する機能を持っているプロトコル（プロトコルの集合）であり、ネットワーク層で動作し IP プロトコルでしか使用できない。これらの機能を提供するための方法として、IPSec

は以下の３つの要素を持ち、「AH」、「ESP」、「IKE」などのプロトコルから構成されている。

- 認証 (ユーザレベルでなく、パケットレベルの)
  データの送信者が間違いなく本人であること、及び送信されたデータが受信されたデータと同一であることを検証する動作である。

  実現のためのプロトコル: AH、ESP

- 暗号化
  適切な鍵を持たない人にはデータを読み取ることができないようにする動作である。

  実現のためのプロトコル: ESP

- 鍵管理
  送信者と受信者の間でセキュリティ「鍵」の値を一致させたり、取り決めさせたりする動作である。

  実現のためのプロトコル: IKE

IPSec を構成するプロトコル

- AH（Authentication Header）

  パケットが改ざんされていないかどうか認証を行うが、パケットの暗号化はできない。
  RFC2402[3] で定義されている

- ESP(Encapsulated Security Payload)
  パケットが改ざんされていないかどうか認証を行い、パケットのペイロード部の暗号化 (DES or 3DES or AES) を行う。
  RFC2406[4] で定義されている。

- IKE(Inernet Key Exchange)
  秘密鍵情報の交換を安全に行う。(Difiie-Hellman 鍵共有などを用いる)
  RFC2409[5] で定義されている。

IPSec では、事前に通信を行うホスト同士が認証・暗号化のアルゴリズムや暗号鍵を共有するコネクションを形成する必要がある。この関係をSA（Security Association）という。

IPSec に対応したホストやファイアウォールではデータベース内に複数のSA を保有することができ、あるIPSec のパケットがどのSA に対応して

いるかは、IPSec のヘッダ中に含まれる SPI(Security Parameters Index) というポインタに指定される。

通信モード

IPSec は通信モードが２つあり、「トンネルモード」と「トランスポートモード」である。ここでは２つのモードを簡単に説明する。

まず、「トンネルモード」では、それぞれ別のネットワークとして構成されているネットワークで発生する通信パケットを、ゲートウェイでカプセル化して転送する。つまり、元の IP パケットをゲートウェイで「外側」の IP パケットの内部にカプセル化し、元のパケットをペイロード (正味のデータ部分) としてまったく新しいパケットを作成し転送している。その後ゲートウェイの IP に従って経路制御され、通信相手のゲートウェイでカプセル化されたパケットが取り出され、宛先となっているネットワークに配送される。

次に「トランスポートモード」は、端末間での IPSec 通信を実現するために、転送する前に元のパケットのペイロードだけを暗号化する。このとき、IP ヘッダは暗号化されず、そのまま経路制御のために利用される。

2.2.2 PPTP

PPTP（Point-to-Point Tunneling Protocol）[6] はデータリンク層で動作するトンネリングプロトコルの一つである。リモートクライアントからプライベートな企業のサーバーにデータを安全に伝送することをようにするための手段として設計された。サーバ/クライアントモデルで、データリンク層でカプセル化を行う。最も古い VPN プロトコルの一つであるが設定が最も容易で計算速度が速いため、古く遅いデバイスで使用される。深刻なセキュリティ上の脆弱性が存在するため、2020 年現在使用することは勧められていない。

PPTP の通信手順

以下に PPTP の通信手順を簡単に説明する。

1. 制御用コネクションの確立 (PPTP トンネルのための準備)
   トンネルの構築、維持、切断処理のための制御情報をやり取りするために、トンネル以外に制御用情報の交換のためのコネクションを確立している。制御コネクションには TCP を利用している。

2. PPP セッションを利用して PPTP トンネルを構築
   ここでは、PPP（Point-to-Point Protocol）が本来持っている仕組みを利用して、PPTP トンネルを構築するための認証及び暗号化の方法を選んで、実際に認証と鍵交換を行う。

2.2.3 L2TP

L2TP（Layer 2 Tunneling Protocol）[7] は L2F(Layer 2 Forwarding Protocol) と PPTP のアップグレードとして設計された。VPN に利用することができるが、L2TP そのものは暗号化の仕組みを持っていないため、IPSec の暗号化機能と組み合わせて VPN の暗号化通信を実現している。L2TP は IPSec とは異なり、実装次第でトンネルするネットワークが必ず IP ネットワークである必要がない。そして、重要な点として L2TP ではカプセル化されたデータは UDP のデータとして（L2TP のメッセージとして）IP ネットワーク上を配送される。

L2TP の通信手順

L2TP では以下のような手順で通信経路の確立を行う。

1. 初めに制御コネクションを確立してトンネルを構築する。
2. その後同じトンネルを使ってユーザセッションを確立する。
3. 最後に PPP によるリンクを確立する。

2.3 Telnet

Telnet は、ネットワークに接続された機器を遠隔操作するために使用するアプリケーション層の技術である。サーバ・クライアント方式で提供され、Telnet サーバが操作される側、クライアントが操作する側で動作する。Telnet を使うことでオフィスのデスクにいながら、マシンルームにあるサーバ、ルータ等の機器をパソコン上で操作できる。PC には telnet クライアント、ルータなどの機器には telnet サーバーのサービスが有効であることが前提である。基本的にはポート番号 23 を使用する。

2.3.1 Telnet の仕組み、使用法

Telnet の接続までの流れは以下の図 2.4 のようになっている。PC からの Telnet はコマンドプロンプトや Terminal から、「telnet 150.65.136.94」というように telnet コマンドと接続したいサーバの IP アドレスを入力するか、Windows では Tera Term 等で IP アドレスと入力して Telnet 接続を行う。TCP によるコネクション確立後 PC のコマンドプロンプトで Telnet サーバから応答画面が表示される。Telnet で遠隔操作を行うためには対象の機器にログインする必要があるため、最初の応答画面ではパスワード要求がされる。

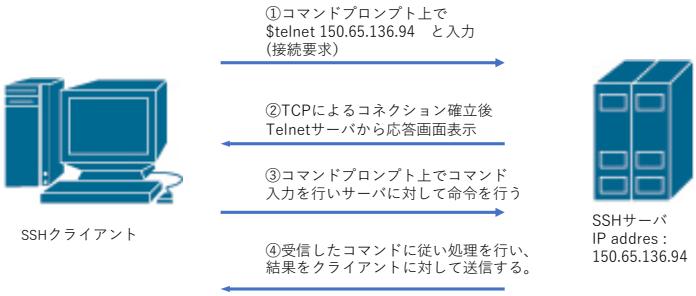

図 2.3: Telnet 接続フロー

問題点として、認証も含めすべての通信を暗号化せずに平文のまま送信するため、パスワードを盗むのは比較的容易である。同様の機能を持ち、情報を暗号して送信することができる SSH が存在し、セキュリティの観点から Telnet よりも SSH が推奨されている。

2.4 SSH

上記の Telnet は、遠隔操作するサーバーの認証情報を含め、通信を暗号化せずに平文のまま通信を行う。その結果、通信内容を盗聴されると認証情報や通信内容が簡単に盗まれてしまう危険性がある。この問題を解決してくれるのが、Telnet と同様な機能を持ちかつ通信内容を暗号化してくれる「SSH」(Secure SHell) である。SSH には「SSHv1」と「SSHv2」

の二種類のバージョンが存在し、本論文では安全性の高い「SSHv2」のみの解説を行う。

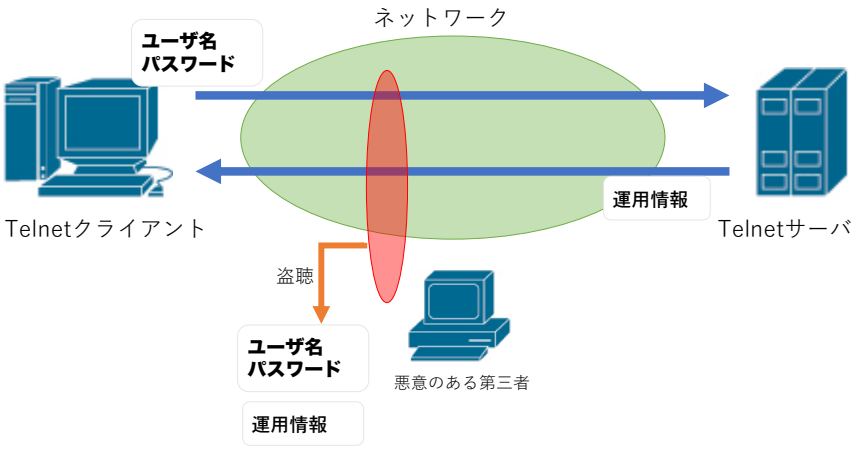

図 2.4: telnet 接続による脅威

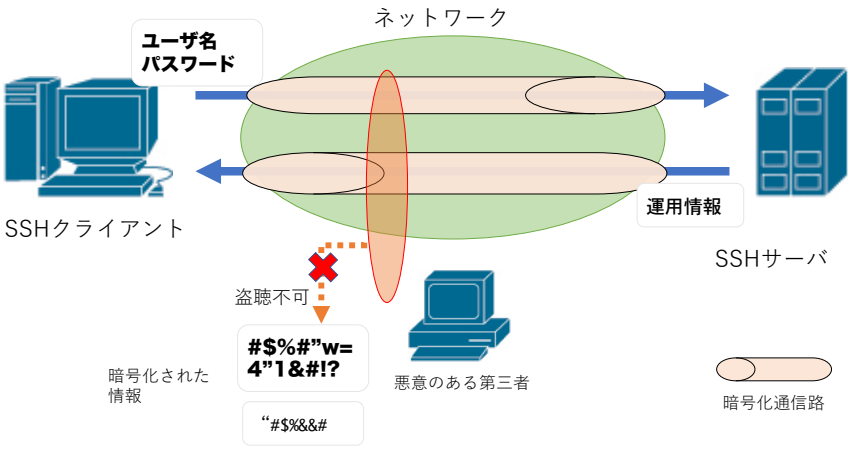

図 2.5: SSH 接続によるセキュアな運用管理

SSH の技術を用いるために、「OpenSSH」というソフトウェアが一般的に使用される。認証方式として大きく 2 つあり、「パスワード認証」と「公開鍵認証」が使われる。「パスワード認証」は、ログイン時に利用するアカウント情報をそのまま異利用し、ID とパスワードが一致すれば認証を行う。「公開鍵認証」は、クライアントが公開鍵と秘密鍵を生成し、クライアント側が持つ秘密鍵が、サーバー側が持つ公開鍵に対応するものであ

るかどうかで認証する。クライアント側が持つ秘密鍵はネットワーク上に送信されることはなく、サーバ側が持つ公開鍵から秘密鍵を推測されないため、パスワード認証よりも安全な認証を行える。SSH の公開鍵とユーザ ID が”˜/.ssh/known_host” ファイルに登録される。

2.4.1 SSH の機能

1. セキュアリモートログイン
   通常, Secure Shell（SSH）と呼ばれる機能のこと。セキュアリモートログインを使用すると、インターネット経由でも安全に、運用端末から SSH サーバーへログインできる。また、通信内容を他者に見られないため、安全な運用管理を実現できる。

2. セキュアコピー (scp)
   セキュアコピーを使用すると、SSH サーバからファイル転送を受け取ることできる。また、通信内容を他者に見られたり、改ざんされたりすることがないため、安全な運用管理を実現できる。

3. セキュア FTP(sftp)
   セキュア FTP を使用すると、SSH サーバーにファイルを転送することができる。セキュアコピーと同様に、通信内容の盗聴や、改ざんを防ぐことができる。

2.4.2 SSH 接続までの流れ

まず初めに暗号化通信路の確立までの流れを以下図 2.6 のように示す。

(a) バージョンと各種暗号方式の交換
   最初、サーバとクライアントの間で SSH バージョン文字列を交換し、SSHv1 で接続するか、SSHv2 で接続するかを決定する。サーバとクライアントの間で、使用できる鍵交換方式、希望する公開鍵暗号方式、共通鍵暗号方式、メッセージ認証コード、のアルゴリズムの各リストを交換する。

(b) ホスト認証と暗号化通信路の確立
   各 SSH サーバーは、それぞれ異なるホスト鍵ペア（ホスト公開鍵とホスト秘密鍵）を保持している。ホスト鍵ペアはインストール時に生成される。クライアントは、サーバの正当性を確認するために、これらの鍵を使用する。サーバ及びクライアントは、交換した共通鍵暗号方式やメッセージ認証コードのリストから、使用するアルゴリズムを

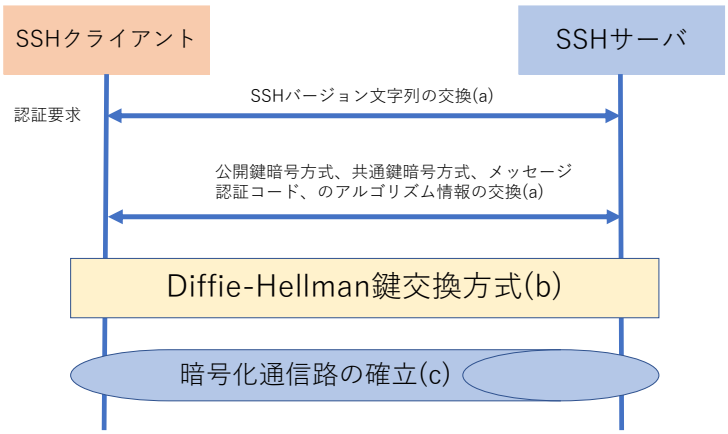

図 2.6: SSH 接続確立までのフロー

決定する。その後、Diffi-Hellman 鍵交換方式で、暗号化通信路に使用する共通鍵を交換する。共通鍵の交換中に、サーバのホスト公開書きをクライアントで保持しているホスト公開鍵のデータベースと照合して、ホスト認証も行う。ここで、Diffi-Hellman 鍵交換方式は、交換する鍵を直接送ることなく、両者で鍵を共有できるアルゴリズムである。

(c) ユーザ認証
ホスト認証後、暗号化通信路が確立されると、公開鍵暗号方式またはローカルパスワードによるユーザ認証を行う。

(1) 公開鍵暗号方式によるユーザ認証
遠隔サーバにはあらかじめユーザの公開鍵を登録しておく。クライアントでは、登録されているユーザ公開鍵に対応した、ユーザが所持している秘密鍵を使用して認証する。SSHv2 では、「電子署名」という方法を使用する。
まず、クライアントでは、ユーザ名、ユーザの公開鍵、ユーザの公開鍵アルゴリズムを記述した認証要求メッセージを作成する。そして、作成した認証要求メッセージに対して、ユーザの秘密鍵を使用して電子署名を作成する。最後にサーバーに対して認証要求メッセージに最後に、サーバに対して、認証要求メッセージに電子署名をつけたものを送付する。
サーバでは、送付された認証要求メッセージから、ユーザ名とユーザ公開鍵を取り出し、登録済みのユーザとユーザの公開鍵で

あることを確認する。また、登録されているユーザの公開鍵を試用して、送付された電子署名を審査し、正しいユーザの電子署名であることを確認できると、ユーザの認証成功となる。

(2) ローカルパスワードによるユーザ認証
telnet と同様に、サーバでローカルに設定されたパスワードを使用してユーザ認証を行う。しかしパスワードは暗号化された通信路を経由するため、第三者には見えない

(d) ログイン後
ユーザ認証に成功すると、セッションが確立し、ユーザはログインする。ここで通常はターミナルのセッションが開始される。

2.5 VPN

VPN とは

VPN とは、「Virtual Private Network」の略であり、「仮想専用線」や「仮想閉域網」と訳され、通信事業者のネットワークやインターネットなどの公衆ネットワーク上で作られる、仮想的な専用ネットワークの総称である。VPN を利用することで、公衆回線（インターネット）での脅威を防ぎ、安全なリモートアクセスを実現することができる。VPN の実現のための主な方法として、ここでは「IPSec-VPN」と「SSL-VPN」を挙げる。まず初めに下表 2.1 に 2 つの違いを簡単にまとめ、その後各々の説明を行う。

	IPSec-VPN	SSL-VPN
リモートアクセス端末への専用ソフトインストール/環境設定	**必要** 環境設定も複雑 （専用ソフトは、IPSec-VPN装置と同一メーカ製品が原則）	**不要** 専用ソフトが必要な場合は自動インストール、自動環境設定
リモートアクセス端末機器	△ 専用ソフトが対応している装置 （パソコンが中心）	○ パソコン、 携帯電話（WEBブラウザ使用）
コンテンツやサーバーに対するアクセス制御	△ 難しい	○ 容易
初期導入コスト	○ 低い	△ 高い
運用管理コスト	△ 高い	○ 低い
既存ネットワークへの適用性	△ NAT（アドレス変換）、ファイアーウォール越えなどの考慮が必要	○ シームレスに導入可能
性能（処理速度・アクセス速度）	○ SSL-VPNより高速	△ IPSec-VPNより低速

表 2.1: IPSec-VPN と SSL-VPN の違い

2.5.1 IPSec-VPN

IP-VPN とは、2.2 節で説明した IPSec の技術を利用して VPN を構成する方法である。IPSec-VPN では、VPN ゲートウェイ装置との間に VPN トンネルを作るため、リモートアクセス端末に専用のソフトをインストールする必要がある。また、暗号化や認証のための設定などの環境設定を事前に行う必要があり、IPSec-VPN は、複数のオフィス間で VPN を張る等、固定的なエンドポイント同士での VPN 接続に向いている。

モバイルデバイスと社内システムを VPN で接続するなどには、より手軽に利用できる SSL-VPN を使用するのが一般的である。

2.5.2 SSL-VPN

SSL-VPN とは、暗号化に SSL 技術を使用した VPN である。IPSec-VPN はクライアント PC に必ず VPN Client のソフトウェアをインストールする必要があるのに対して、SSL-VPN の場合は Web ブラウザさえあれば通信可能である。

SSL-VPN には、「リバースプロキシ」、「ポートフォワーディング」、「L2 フォワーディング」の 3 つの方式がある。まずはじめに 3 つの方式の比較を簡単に下表 2.2 に示した後、それぞれを簡潔に説明する。

	リバースプロキシ方式	ポートフォワーディング方式	L2フォワーディング方式
リモートアクセス端末側構成要素	WEBブラウザ	WEBブラウザ＋モジュール（WEBからダウンロード、自動インストール）	WEBブラウザ＋モジュール（WEBからダウンロード、自動インストール）
使用可能アプリケーション	△ WEBアプリケーション	○ 通信中ポート番号が変わるものは使用できない場合あり	◎ ほとんどのアプリケーションで使用可能
リモートアクセス端末機器	○ WEBブラウザが動く端末	△ モジュールの仕様によって制限 利用時に管理者権限が必要	△ モジュールの仕様によって制限
用途	出張先の端末などから簡単に使いたい。 仕様アプリケーションはWEBメールやWEB型グループウェアなどWEBページ中心	クライアント端末のOSが様々である。 ある程度の種類のアプリケーションを使いたい	アプリケーションを制限なく使いたい。 使用されるクライアント端末の種類は限られている。

表 2.2: SSL-VPN の実装方式の比較

SSL-VPN(リバースプロキシ方式)

「リバースプロキシ」は、通常のクライアント側（利用者側）の「プロキシ」ではなく、サーバー側（アクセスされる側）に位置するものである。簡単な構成を下図 2.7 に示す。

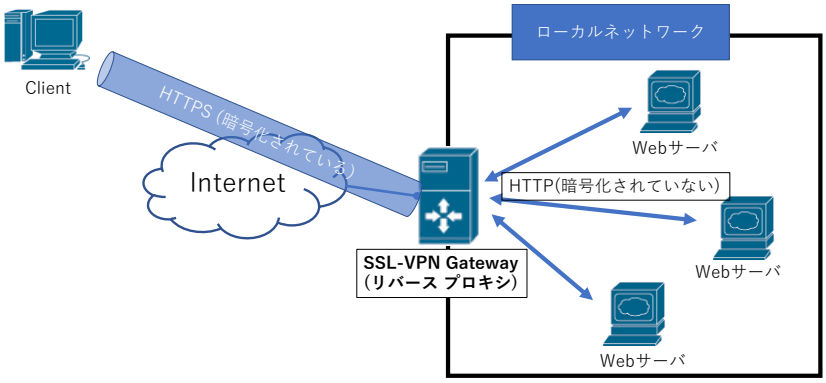

図 2.7: リバースプロキシの図

クライアントからは Web サーバにアクセスしているように見えるが、実際は SSL-VPN Gateway（リバースプロキシ）が仲介者として Client の Request に沿って各ローカルの Web サーバにアクセスしている。

これにより、Web サーバに外部からの直接的なアクセスができなくなり、改ざんや不正侵入などのリスクを減らすことができる上に SSL-VPN Gateway にファイアウォールや認証機能を追加することでセキュリティの強化を行うことができる。また、SSL-VPN Gateway だけに SSL 証明書を適用すればよくなる。

リバースプロキシ方式は、Web ブラウザで動作しないアプリケーションは使用できないという問題があった。そこでリモートアクセスしてくるクライアントに Java や ActiveX で作成されたモジュールを追加させた上で SSL-VPN 通信を行う方法が考えられた。それが、「ポートフォワーディング」と「L2 フォワーディング」である。

SSL-VPN(ポートフォワーディング方式)

一般的に上記のリバースプロキシ方式では、SSL-VPN Gateway にファイアウォールを設置し、外部から利用できるアプリケーションを制御（Web

サーバは閲覧できるが、Telnet はできないなど）している。その制御には、通信されるデータに含まれる、ポート番号と呼ばれるアプリケーションの種類を示す情報を使用している。

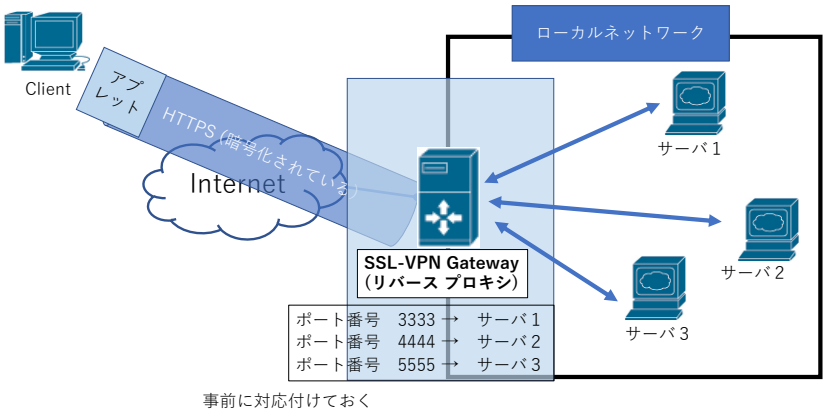

図 2.8: ポートフォワーディングの図

ポートフォワーディングとは、ファイアーウォールを通過できないアプリケーションのデータのポート番号を、通過できるアプリケーションのポート番号に変換することにより、ローカルネットワークとグローバルネットワーク（インターネット）との通信を可能にする機能である。ポートフォワーディング方式では、この機能を用いて任意のアプリケーションの通信を上図 2.8 のように HTTPS のポート番号に変換し、ファイアーウォールを通過させることで SSL-VPN を実現する。

SSL-VPN(L2 フォワーディング方式)

L2 フォワーディング方式ではクライアント PC に SSL-VPN クライアントソフトをインストールする。

L2 フォワーディングでは、アプリケーションのデータを HTTP パケットでカプセル化して SSL 通信を行う。ポートフォワーディング方式のように、SSL-VPN Gateway で通信するサーバの IP アドレス/ポート番号を事前に定義する必要がないため、幅広いアプリケーションをサポートすることが可能である。

専用のソフトウェアをインストールすると、クライアントに SSL-VPN 用の仮想 NIC が追加され、仮想 NIC 経由の通信がすべて SSL で暗号化される。仮想 NIC の IP アドレスは、通常 SSL-VPN 接続とともに自動設定

される。そして、クライアントは仮想 NIC によって社内ネットワークに直接接続されているのと同等に扱うことができるようになる。

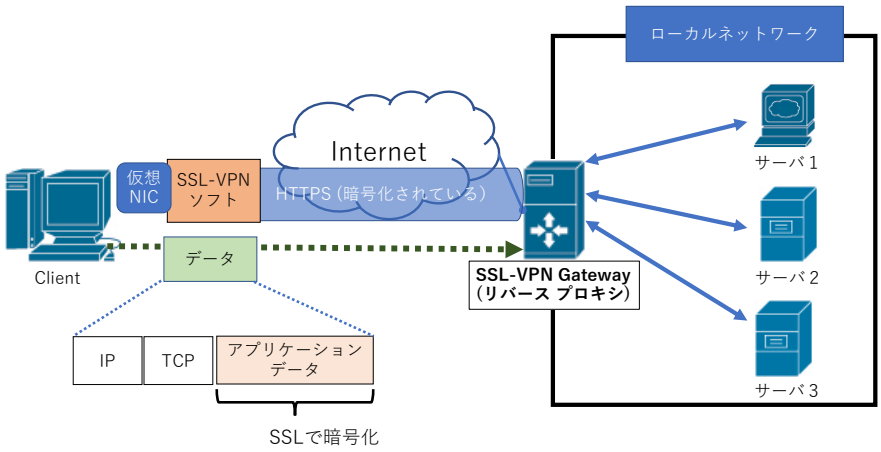

図 2.9: L2 フォワーディングの図

企業において SSL-VPN を用いてテレワークなどのために社内ネットワークに遠隔アクセスする場合、ログアウト後にそのクライアント PC（テレワーク利用端末）にデータが残らないよう自動的にデータが削除されるように実装させるのが一般的である。

参考として、Cisco の VPN ゲートウェイ装置で SSL-VPN を行う場合、「リバースプロキシ方式」、「L2 ポートフォワーディング方式」が利用できる。「ポートフォワーディング方式」は利用できない。

第3章　認証技術

認証技術の解説を行う。

3.1　LDAP

概要

LDAP(Lightweight Directory Access Protocol) は、Active Directory のようなディレクトリサービスにアクセスするためのプロトコルである。LDAP 自体はプロトコルであり、サービスやシステムを指すものではない。LDAP を実装したデータベースを LDAP サーバと呼び、代表的なものに「Open LDAP」、「Active Directory」が存在する。

ディレクトリサービス

ディレクトリサービスとは、ディレクトリと呼ばれるデータベースから、ユーザ名やマシン名などのキーを元にデータを検索、参照するためのサービスである。一般的にデータベースと呼ばれる、SQL 言語等を用いて扱う「RDB (Relational Data Base)」では、データ間の関係性を利用して、データの参照、挿入、更新、削除、といった操作を行うため、それらは少し異なるものである。グローバルでサービスを提供する場合には、分散型ディレクトリサービスが用いられ、DNS(Domain Name Service) が分散型ディレクトリサービスとして有名である。

LDAP、ディレクトリサービスの特徴

- 読み取りが高速
- 分散型の情報格納モデル
- 高度な検索機能をもつ

LDAP の特性としては、「情報の参照、検索」に特化している。このようになる理由としては、ディレクトリサービスとして利用されるものは、一般的なデータベースのように読み取りと書き込みが同じ頻度で発生することはない。そして、大規模システムでは、ユーザ情報の利用は参照検索が最も頻繁に起こるため、それらの操作に対する高い性能が必須であるため、このような特性となっている。

LDAP を用いたデータの集中管理の様子を下図 3.1 に示す。

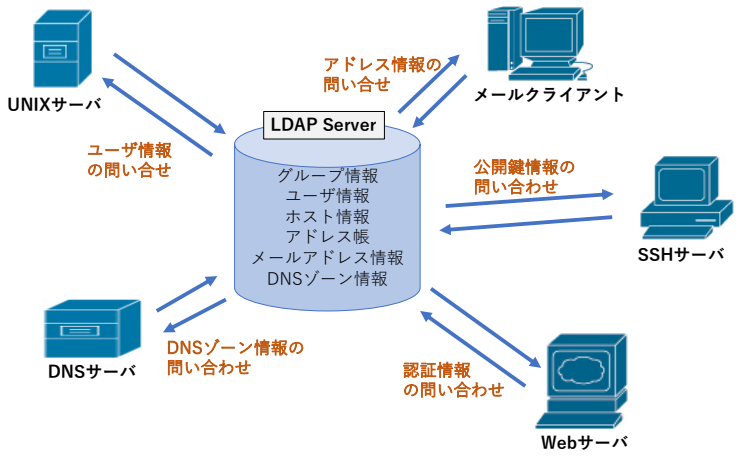

図 3.1: LDAP を用いてデータを集中管理の図

LDAP できること

1. リソースの一元管理
   多数のクライアントがある場合、１台１台に ID パスワード情報を入れることなく、LDAP サーバー１台だけ登録すれば、どのクライアントからも同じ ID パスワードでログインできるようになり、さらに環境もログイン時に取得できるようになる。

2. リソースのアクセス制御
   特定の IP アドレスからであれば、読み書き可能であるが、それ以外からは読みしかできない。

3. 各種サービスとの連携
   多くのアプリケーション (Open Source Software) と連携することができる。初期のユーザ情報作成を LDAP データベースから行い、認証を LDAP サーバに委譲することができる。これの発展形として、

一つのサーバで認証すれば、他のサーバでは認証無しでログインできる仕組みである、シングルサインオン（SSO）を実現できる。

3.2 Kerberos

ネットワーク内でのシステムの安全性と利便性は共存しにくいことがある。単純にどのサービスがネットワーク上で稼働しているか、そして、使用されているサービスの動作を管理者が追跡するだけでも膨大な時間がかかることがある。さらに、FTP プロトコルや Telnet プロトコルのようにデータを暗号化せずにネットワーク上でパスワードを送信させるというような、プロトコル自体が安全でないときネットワークサービスへのユーザ認証は危険を伴うことになる。

Kerberos とは

ネットワーク上でユーザの認証を行う方式の一つである。クライアント/サーバ間の通信を暗号化でき、比較的セキュリティが強固な認証方式となっている。ケルベロス認証では一度ログインすると、「チケット」と呼ばれるものを用いて認証を行えるようになるため、次回のログイン時に ID・パスワードを改めて入力する必要がなくなり、シングルサインオン（SSO）を実現できる。

利用例としては、Active Directory のユーザ認証の際に用いられている。名前はギリシャ神話の地獄の門を守る番犬ケルベロスに由来している。

Kerberos の構成要素

Kerberos の仕組みを解説する前に、用語「KDC、AS、TGS、プリンシパル、レルム」の説明を行う。

- KDC (Key Distribution Center)
  サーバとユーザに関する信頼関係の情報を一括管理する中央データベース。
- AS (Authentication Server)
  認証サーバで、ユーザからの認証を受け付けるサーバ。
- TGS (Ticket Granting Server)
  チケット発行サーバ。各サーバを利用するためのチケットを発行するサーバ。

- プリンシパル (principal)
  KDC 認証を行うユーザやサーバのこと。
- レルム (realm)
  同じ KDC の配下にあるシステムをグループとして定義する論理ネットワーク。

これらを構成すると以下の図 3.2 になる。

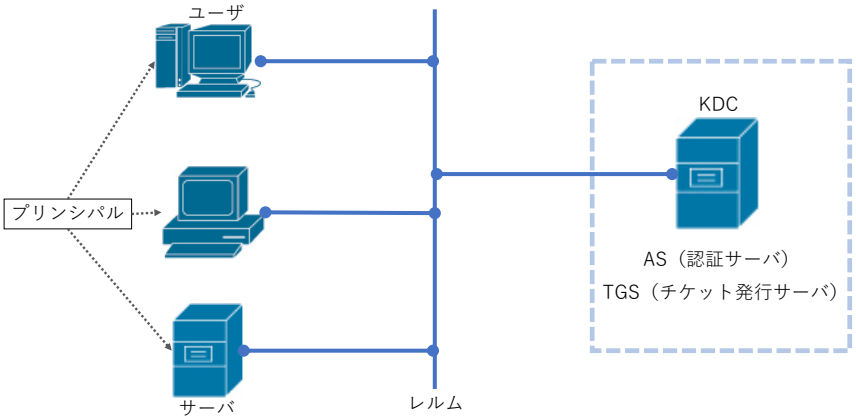

図 3.2: Kerberos 認証の構成要素

Kerberos の認証の仕組み

Kerberos 認証では、ユーザが正しいユーザ ID とパスワードを AS (Authentication Server) に送信して認証に成功すると TGS(Ticket Granting server) からチケットと呼ばれるデータを受け取れる。Kerberos 認証ではこのチケットを認証に使用する。サーバはアクセスしてくるユーザがアクセス権を持っているかどうかをユーザ ID とパスワードではなくチケット（クライアント ID、タイムスタンプ、有効期限が記されている）を使用して確認する。認証時にチケットを使用することでアカウント（ユーザ ID、パスワード）の漏洩を防いでいる。全体的な流れを以下図 3.3 に示す。
Kerberos 認証では、チケットの盗聴によるなりすましを防ぐために、時刻同期の仕組みが用意されている。チケットの中にはタイムスタンプ（送

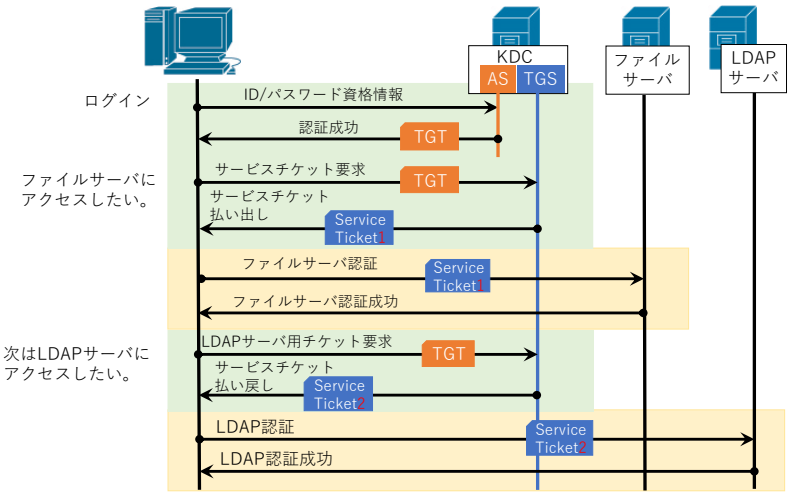

図 3.3: Kerberos 認証流れ

信時刻）が記録されており、チケットを受信したサーバがチケットのタイムスタンプとサーバの持つ時刻と 5 分以上のズレがあると認証に失敗するようにないっている。したがって、NTP（Network Time Protocol）を使用して、チケット発行側の時刻とチケット利用側の時刻が同じにする必要がある。

3.3 Radius

Radius とは

Radius(Remote Authentication Dial In User Service) は、ネットワーク上のユーザ認証プロトコルである。Radius による認証システムは、「Radius サーバ」、「Radius クライアント」、「ユーザ」の 3 つの要素で構成されている。

インターネットが普及を始めたころ、ユーザがインターネットへのアクセスするには電話世界線を使ったダイヤルアップが主流であった。Radius サーバは、ダイヤルアップサービス用の認証サーバとして開発され、ISP や企業などで利用されてきた。現在回線は光回線になっているが、ISP の認証サービスなどでは、Radius サーバが継続して使用されている。また、近年では、Wi-Fi アクセスポイントでの認証などでも、Radius サーバが使われている。

- Radius クライアント

アクセスしてくるユーザの認証要求を受け付けて Radius サーバーにその情報を転送する役割。NAS（Network Access Server）とも呼ばれる。

- Radius サーバ
  認証要求に応じて認証を実行してアクセスを許可するかどうかを決定する役割。

Radius 認証の流れ

今回、ユーザがネットワークやネットワーク機器を利用したい場合、全体の認証の流れは下図 3.4 のようになっている。

1. ユーザがネットワークに接続しようとする。
2. Radius クライアントは、ユーザに認証を要求する
3. ユーザがユーザ名、パスワードを入力し、認証を行う。
4. Radius クライアントは、受け取ったユーザ名、パスワードを使って、Radius サーバにアクセス認証の Request を出す。
5. Radius サーバは認証処理を行い、Radius クライアントに認証可否を伝える。
6. Radius クライアントは、ユーザに認証結果を伝える。

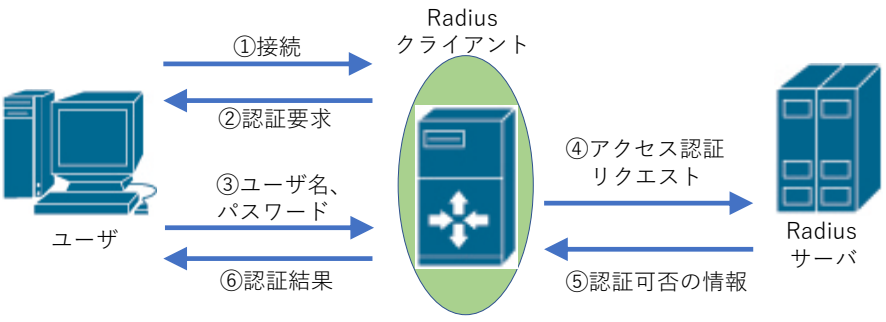

図 3.4: Radius 認証の流れ

3.4 Active Directory

Active Directory とは

マイクロソフトによって開発されたディレクトリサービスシステムで、一般的に WindowsOS で使用される。Active Directory は複数のサービスの総称である。

Active Directory とは簡単にいうと「Windows システムで認証を行う機能」である。社内システムの中にはアクセスを制限して一部の社員にしかアクセスさせたくないシステムもある。このようなシステムにアクセス可能かどうかを判断するために、Active Directory の認証機能を使用してアクセスの制限を行う。

Active Directory の機能

Active Directory は 5 つのサービスから構成されている。

- Active Directory ドメインサービス (AD DS)
  ユーザーやコンピュータの認証や、管理者が情報を安全管理したり、ユーザがファイルやディレクトリ などのリソースを簡単に検索することができる機能である。
  一般的に Active Directory というと、このドメインサービスのみを指すことが多い。

- Active Directory ライトウェイトディレクトリサービス (AD LDS)
  AD DS の簡易版という立ち位置。AD DS の構成要素のうち「データベースの仕組み」、データの検索記録が行えるようになったもの。認証を行う機能はない。

- Active Directory 証明書サービス（AD CS）
  公開鍵（PKI）を構築するための、証明書の作成と管理を行う証明機関 (Certification Authority:CA) を作成するサービス。

- Active Directory Rights Management Services (AD RMS)
  ドキュメントの権限管理やコンテンツ保護など、不正使用から情報を保護するための機能。AD RMS を利用することで、メールやドキュメントの保護が可能になり、保護されたデータは暗号化される。

- Active Directory フェデレーションサービス (AD FS)
  複数の Web アプリケーション間の認証や、異なる組織間での認証など、組織の違いを超えて認証の仕組みを連携する機能。

第4章　研究のためのネットワーク構成

本研究では、L2スイッチ一台とCentOS8.0のインストールされたサーバ三台を用いて一台をGateway Server、残りの二台をHost Serverとすることで、以下図4.1のような隔離ネットワークを構成し、そのネットワーク内でソフトウェアを試用する。ここで隔離ネットワークとは、内部のHost Server二台はGatewayServerを経由しないと外部からは接続できないようになっているネットワークと定義する。L2スイッチの設定は、PC7とGateway間、PC8とGateway間のそれぞれの間は”ping”コマンドが通るようにし、PC7とPC8間は”ping”コマンドが通らないように設定した。

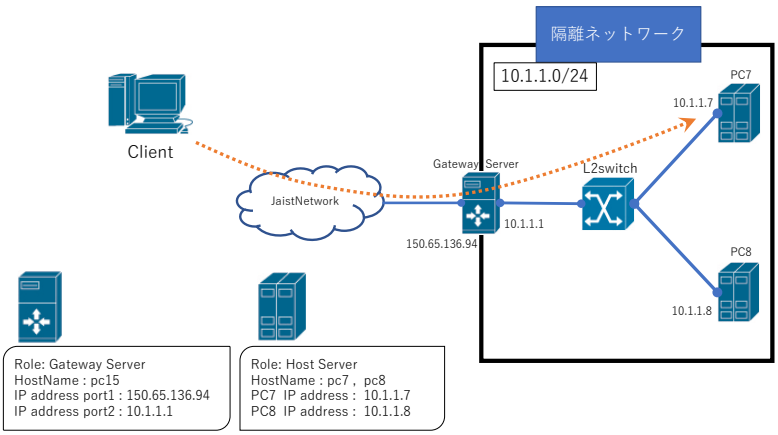

図 4.1: ネットワーク構成図

そして、本研究では、試用するソフトウェアを導入規模で小規模と大規模というように分類し、各種機能で細分化して評価をしていくが、ここで小規模アプリケーションと、大規模アプリケーションを以下のように定義する。

- 小規模アプリケーション
  上図4.1において、「GatewayServer」だけにインストールするもの、

「Client」だけにインストールするものと定義する。

- 大規模アプリケーション
  上図 4.1 において、「GatewayServer」、「Client」、「HostServer」のそれぞれにインストールを行い、各種設定をおこなう必要があるソフトウェアと定義する。

次の章で、小規模ソフトウェアの試用を行う。

第5章　小規模ソフトウェア

インストール先がゲートウェイサーバーだけのものや、隔離ネットワークにアクセスするクライアントだけのものを小規模のアプリケーションと分類して、評価を行う。まず最初に、評価表を表 5.1 に示す。

名前	概要	ログ能力	鍵交換	webUI	適用先	適用難易度	
sshportal	動的にユーザとホストを構成するBastionサーバーツール	あり	RSA	なし	Gateway Server	少し難しい	導入や使用法などがあまり詳細に記されていない。
sshuttle	擬似的な簡易VPN	あり	※VPN	なし	クライアント	易しい	専用のサイトが用意されている上に、導入手順が詳細に記されている。
sshpiper	プロキシーのようなソフトウェア	あり	RSA	あり	クライアント	すこし難しい	導入や使用法などがあまり詳細に記されていない。

表 5.1: 小規模ソフトウェア比較表

5.1　sshuttle

概要

“sshuttle(https://github.com/sshuttle/sshuttle)” は、簡易 VPN ツールである。リモートアクセスユーザ (Client) のみにインストールすれば使用できる。本論文では、図 4.1 の Client にインストールを行なった。

インストール方法

今回、クライアント pc には MacBook を用いたため、homebrew を使ってインストールを行なった。他の OS の場合のインストール方法もいくつかここに記す。

- Ubuntu
  $ apt-get install sshuttle
- MacOS
  $ brew install sshuttle
- Centos
  $ git clone https://github.com/sshuttle/sshuttle.git
  $ cd sshuttle
  $ sudo ./setup.py install

Client のみにインストールすればよいため、とても容易に使用することができる。

使用方法

今回、図 4.1 で Client と隔離ネットワークと VPN を形成したい時を想定する。
”$ sshuttle -r pc15@15.65.136.94 10.1.1.0/24”のコマンドで、図 5.1 のように VPN を形成できた。実際の terminal の図はこちらになる。一度 VPN を形成すると、多段 ssh をする必要などなく、自由に隔離ホストにアクセスできるようになる。
例えば、一度 sshuttle で VPN を構築すると、Client の端末上で
$ ssh pc8@10.1.1.8 のコマンドで隔離サーバである PC8 に直接 ssh 接続することができる。

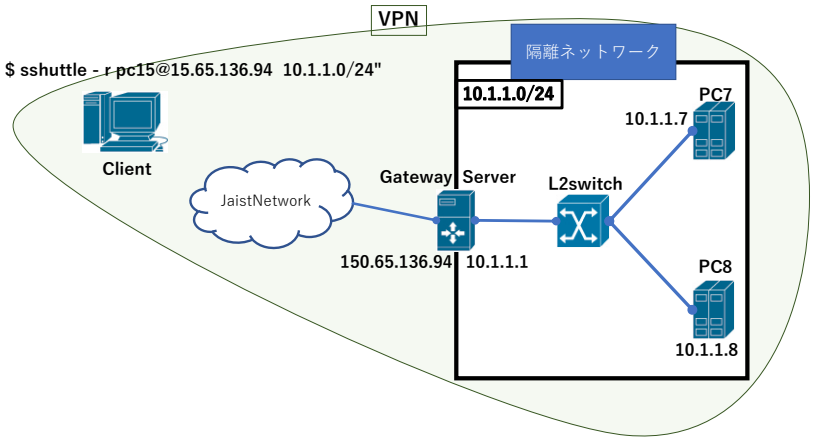

図 5.1: sshuttle の VPN 構築イメージ図

実際の terminal の画面出力を以下図 5.2 に示す。

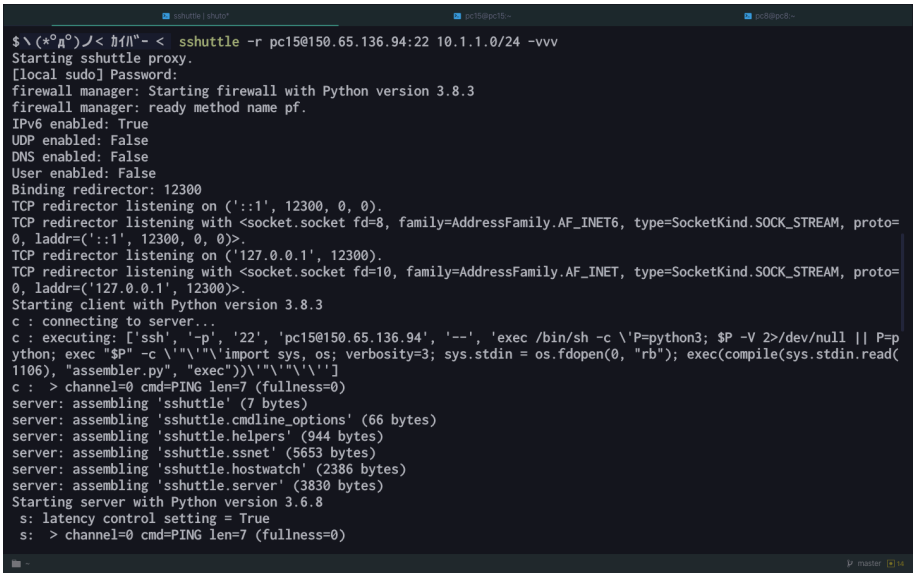

図 5.2: sshuttle の VPN 構築時のコンソール画面

図 5.3: sshuttle 後隔離ホストへの ssh 接続画面

メリット

sshuttle を使用するメリットは、「安全性」と「簡易化」である。

一般的に、図 4.1 のような隔離ネットワーク内のホストにアクセスするためには、多段 ssh や Tunneling を行う必要がある。多段 ssh というのは今回の場合、Client がまず Gateway Server に ssh を行い、その後 Gateway Server の Terminal から、隔離ホストに ssh 接続するというように複数回 ssh を行うことである。この場合、一度 ssh を切ってしまうと、再び多段 ssh する必要である上に、各プロセスごとに多段 ssh をする必要がある。しかし、sshuttle を使用することで、一度 VPN を構築すると自由に隔離ホストにアクセスすることができるため、とても容易に隔離ホストに接続できる。

また、安全性の面では一般的に多段 ssh を行う場合は、Gateway Server を経由するため、外部ネットワークと接続されている Gateway Server に Client の ssh key を保存しないといけない。しかし、sshuttle で VPN を構築することで Gateway Server を経由せずに隔離ホストと接続するため、Client-隔離ホスト間で鍵共有するだけでよいため、インターネット等と接続されている Gateway に鍵を保存するより安全である。

5.2 sshportal

概要

“sshportal(https://github.com/moul/sshportal)” とは、透過的な SSH 要塞サーバーにするソフトウェアである。Gateway Server のみにインストールすれば使用することができる。sshportal を使用することで、管理者は Gateway Server にアクセスし隔離ホストにログインでき、ユーザーを動的に管理することができる。これによって複数のユーザーを複数のホストに簡単に割り当てられる様になる。Gateway Sserver のみが両側に関する情報を知っているため、エンドユーザはホストを知る必要がなく、アクセスする必要があるすべてのものに自動的に接続される。

インストール方法

sshportal は Docker を用いることで容易にインストールができる。また、今回使用した Gateway Server の OS は CentOSv8.0 であるため Docker は使用できない。代わりに”Podman”が存在しているため、そのコマンドもここに記す。Podman の使用法はほとんど Docker と違いはない。

- Docker
  docker pull moul/sshportal
- Podman
  podman pull moul/sshportal

使用方法

ここでは「Docker」使用時のコマンドを示す。

管理者の場合

- バックグラウンドサーバーを開始する
  docker run -p 2222:2222 -d –name=sshportal -v ”$(pwd):$(pwd)” -w ”$(pwd)” moul/sshportal:v1.10.0
- ログを表示させる
  docker logs -f sshportal
- 管理者 (admin) としてログイン
  # ssh localhost -p 2222 -l admin
  その後 config >に切り替わる。ここで動的にユーザー登録を行う。

もし、サーバーにアクセスしたいユーザーがいるときの使用法

- 最初に admin ホストを作成する
  config>host create user1@10.1.1.8
- サーバーに鍵を追加する
  $ ssh user1@10.1.1.8 ”$(ssh localhost -p 2222 -l admin key setup default)”

ユーザの招待

- 例： config>user invite bob@exaple.com
  これによってユーザーを招待している。このコマンドでは、リモートサーバーにユーザーを作成するのではなく、sshportal.db というデータベースににアカウントを作成する。

ユーザーの場合

- $ ssh localhost -p 2222 -l 10.1.1.8
  これにより、user1 は Gateway から pc8 に接続することができる。

評価

sshportal はユーザの管理が簡単に行えるように設計されており、新規ユーザの追加等を管理者が動的に行うことができる。インストール先は Gateway Server のみで良いが、前述の「sshuttle」のように Github とは別に専用のサイトは用意されておらず、Github にも詳細には仕組みやインストール方法が記されていないため、導入や使用までにすこし試行錯誤が必要である。

その他のデメリットとしては、ユーザーの管理を自動ではなく、動的に管理者が行う必要があるため、ユーザーが大人数になるほど管理が大変になってしまう。

第6章　大規模ソフトウェア

インストール先がGateway Serverだけでなく、隔離ネットワーク内のホスト全てにRole別に設定を行う必要のあるものを大規模アプリケーションと分類して、評価を行う。まず最初に、評価表を下表6.1に示す。

名前	概要	ログ能力	key exchange	webUI	適用先	適用難易度	
Aker	FreeIPAを利用したBastionサーバーツール	あり	Kerberosチケット	なし	Gateway Server, HostServer, クライアント	難しい	• python2とpython3の依存関係の問題あり。 • インストールまでの説明が少ない。
teleport	リモートアクセスするためのセキュリティゲートウェイ	あり	SSL証明書	あり	Gateway Server, HostServer, クライアント	すこし難しい	• 管理者専用のサイトが用意されており、導入まで丁寧に記載されている
SoftEther VPN	レイヤ２でカプセル化やトンネリングを行う、VPN構築ソフトウェア	あり	**認証** RSA **暗号化** AES、DESなど	なし	Gateway Server, HostServer, クライアント	すこし難しい	• インストールまでのロードマップが用意されている • コンピュータネットワークへの深い知識が必要

表 6.1: 大規模ソフトウェア比較表

6.1　teleport

概要

Teleport(https://github.com/gravitational/teleport)は、リモートアクセスのためのセキュリティGatewayとなっている

SSHまたはKubernetes APIを介してlinuxサーバのクラスターへのアクセスを管理するためのGatewayである。有料版と無料版があり、今回は無料版を使用した。以下にTelportの機能の一部を示す。

- 単一のSSHアクセスGateway

- SSH 証明書ベースの認証
- 二段階認証
- SSH の役割ベースのアクセス制御
- セッションの記録を行う。
- シングルサインオン (SSO)

基本的なアーキテクチャの概要を下図に示す。

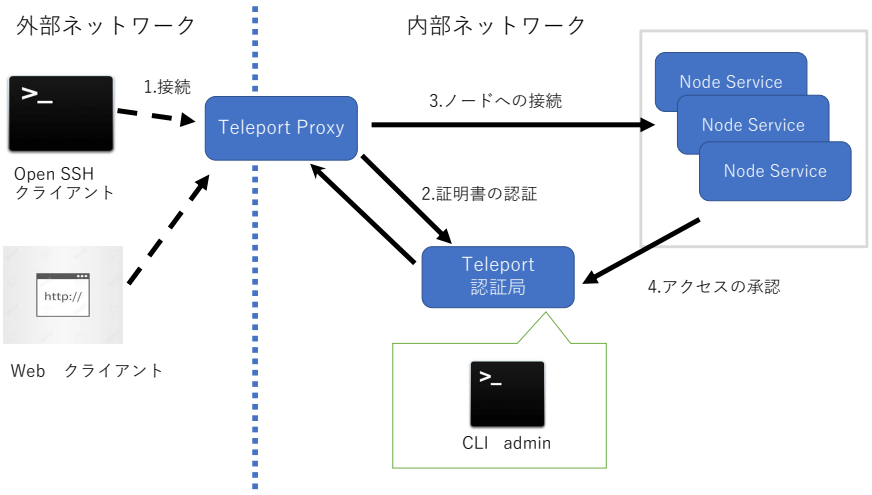

図 6.1: teleport のアーキテクチャ

6.2　teleport

Teleport の構成要素

Teleport の仕組みを解説する前に、用語の説明を行う。

- ノード
  「サーバ」または「コンピュータ」と同義語。SSH 接続できるもの。
- ユーザ
  ノード上で一連の操作を実行できる人や、マシン。
- クラスター
  連携して動作するノードのグループであり、単一のシステムとみなすことができる。

- CA（認証局）
  公開鍵/秘密鍵のペアの形式で SSL 証明書を発行する。

- テレポートノード
  テレポートサービスを実行しているノード。許可されたテレポートユーザがアクセスできる。

- テレポートユーザ
  テレポートクラスタへアクセスしたいユーザ。ユーザはユーザ名とパスワードを登録する必要がある。

- テレポートクラスタ
  １つ以上のノードで構成され、各ノードは同じ CA によって署名された証明書を持つ。

- テレポート CA
  テレポートは、認証サービスの機能として２つの内部 CA を運用する。一つはユーザ証明書の署名を行い、もう一つはノード証明書の署名に使用される。

Teleport の接続までの流れ

下図の詳細なアーキテクチャを使って説明を行う。

1: クライアント接続を開始する
クライアントは、CLI インターフェースや Web ブラウザーを使用してプロキシ（Gateway）へ SSH 接続を始める。そのとき、クライアントは証明書を提供する。
————————ここで図の挿入ーーーーーーーーー

2: クライアント証明書の認証
プロキシは、送信された証明書が以前に認証サーバ（CA）によって署名されているかどうかを確認する。もし署名がされていなかった場合（初回ログイン時）や証明書の有効期限が切れているとき、プロキシ

は接続を拒否し、パスワードと二段階認証でのログインをクライアントに求める。二段階認証は、Google Authenticator などを用いて行う。HTTPS 経由でプロキシに送信される。

3: クライアントが接続要求するテレポートノードを調べる。
ーーーーーーーーーーーーーーーーここで図を挿入しておきたいーーーーーーーーー
^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^

このステップで、プロキシーはクラスター内の要求されたノードを見つけようとする。プロキシがノードの IP アドレスを見つける検索メカニズムは 3 つある。

(1) DNS をしようして、クライアントから要求された名前解決を行う。
(2) これに登録されているノードがあるかどうか認証サーバに尋ねる。
(3) 要求された名前と一致するラベルを持つノードを見つけるように認証サーバに要求する。

その後、ノードが見つかると、プロキシはクライアントと要求されたノード間の接続を確立する。その後、宛先ノードはセッションの記録を開始し、セッション履歴を認証サーバ（CA）に送信して保存する。

4: ノード証明書の認証
ノードは接続要求を受信すると、認証サーバを使用してノードの証明書を検証しノードのクラスタメンバーシップを検証する。ノード証明書が有効な場合、ノードは、クラスター内のノードおよびユーザーに関する情報へのアクセスを提供する認証サーバー API へのアクセスを許可される。

5: ユーザノードのアクセスを許可する
ノードは認証サーバーに、接続しているクライアントの OS ユーザーのリスト（ユーザーマッピング）を提供するように要求し、クライアントが要求された OS ログインの使用を許可されていることを確認する。最後に、クライアントはノードへの SSH 接続を作成することを許可される。

使用方法

6.3 SoftEther VPN

SoftEther VPN とは

SoftEther VPN(https://ja.softether.org/) とは、筑波大学における学術目的の研究プロジェクト「SoftEther プロジェクト」により運営されている、オープンソースソフトウェアである。Windows,Linux,Mac,FreeBSD および Solaris 上で動作する。

SoftEther VPN の特徴

SoftEther VPN は、カプセル化およびトンネリングの通信をレイヤ 2、のデータリンク層で行なっている。

SoftEther VPN を使用することで、通常の LAN カード、スイッチング HUB 及びレイヤ 3 スイッチなどのネットワークデバイスをソフトウェアによって仮想的に実現し、それらの間を TCP/IP プロトコルをベースとした、「SoftEther VPN プロトコル」と呼ばれるトンネルで接続することで、柔軟性の高い VPN 構築を実現している。

柔軟なリモートアクセス VPN 及び拠点間接続 VPN を実現するために、Ehternet デバイスの仮想化する。

SoftEther VPN Server の仕様

同時接続可能な最大 VPN セッション数 4096 個

ユーザ認証：

パスワード認証

Radius 認証

Active Directory 認証

固有証明書認証

署名済み証明書認証

SoftEther VPN の評価

第7章　まとめ

関連図書

[1] 日経新聞, 2020/7/26. 「西村経財相「在宅勤務を 7 割に」　経済界に再要請へ」.

[2] 総務省. 総務省テレワークセキュリティガイドライン, 平成 30 年 4 月.

[3] S. Kent and R. Atkinson. IP Authentication Header, RFC2402. November 1998.

[4] S. Kent and R. Atkinson. IP Encapsulating Security Payload (ESP), RFC2406. November 1998.

[5] D. Harkins and D. Carrel. The Internet Key Exchange (IKE), RFC2409. November 1998.

[6] K. Hamzeh, G. Pall, W. Verthein, J. Taarud, W. Little, and G. Zorn. Point-to-Point Tunneling Protocol (PPTP), RFC2637. July 1999.

[7] W. Townsley, A. Valencia, A. Rubens, G. Pall, G. Zorn, and B. Palter. Layer Two Tunneling Protocol "L2TP", RFC2661. August 1999.