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 8 月 27 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

1 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 に示す。下表 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「クラウドサービス型」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、オンプレミス型もクラウドサービ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。

本研究では、図 1 のような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し、その中で小規模と大規模アプリケーションで分類し検討を行うとともに、そこで使用されている遠隔アクセス技術、認証技術の解説も行う。

ここでまとめることで、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

	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　テレワークの６種類のパターン

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。

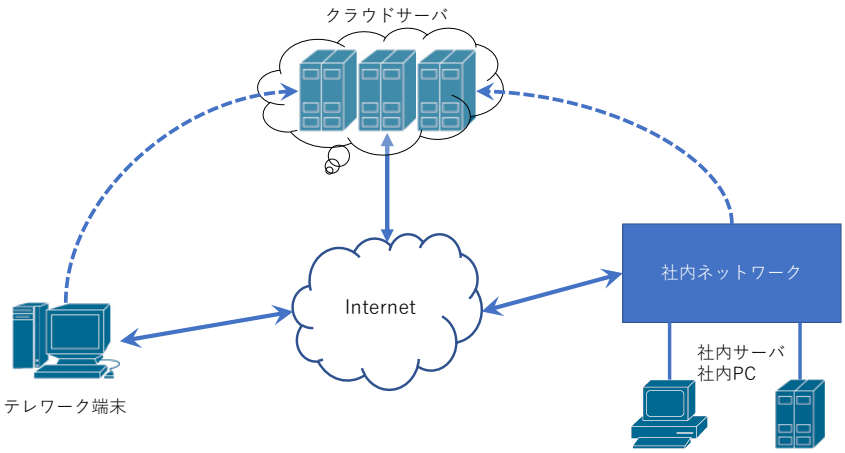

図１　クラウドサービス型

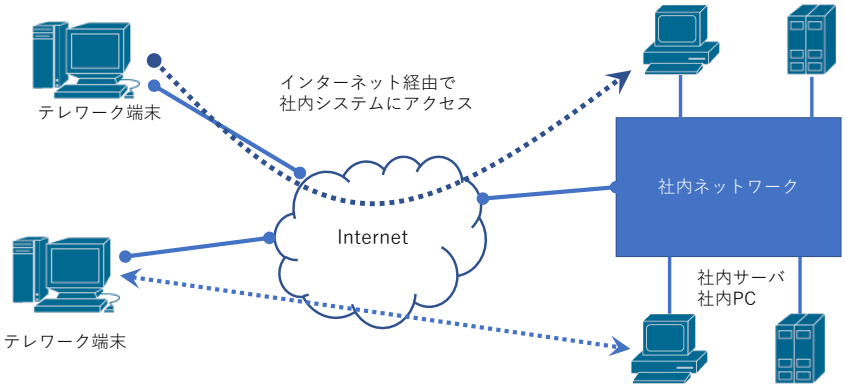

図 2　オンプレミス型

1.3　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 CentOSv8 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 3 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。Gateway Server

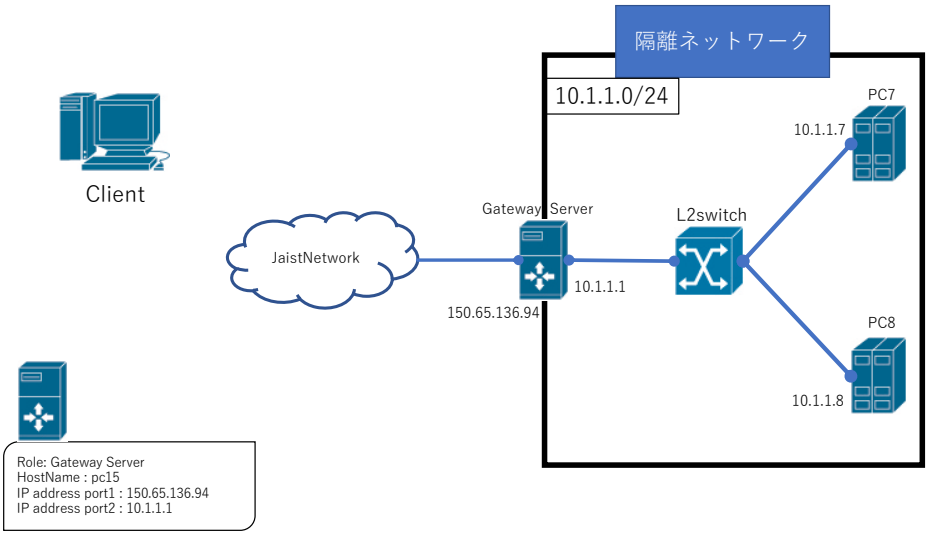

図 3　ネットワーク構成図

それと同時に、その中で使われている遠隔操作、アクセスの技術と認証技術についての解説も行う。

本研究では、システム管理者のために

そして、今回導入の規模に合わせて、小規模と大規模に分類し表に表す。

2　遠隔アクセス技術

本節では遠隔サーバーにアクセスし操作するための技術、ソフトウェアの解説を行う。

2.1　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。

2.1.1　Telnet の仕組み、使用法

Telnet の接続までの流れは以下の図 7 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。

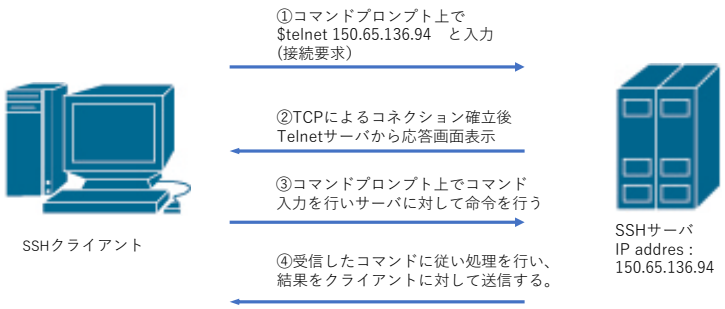

図 4　Telnet 接続フロー

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。

2.2　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。

2.2.1　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 5 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。

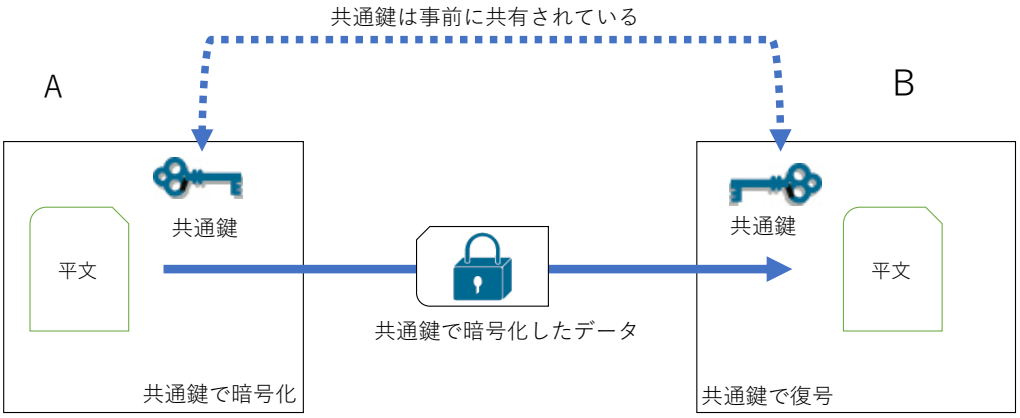

図 5　共通鍵暗号方式での暗号化

代表的な暗号化アルゴリズムに「AES」が存在する。

2.2.2　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。

- 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 6 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。

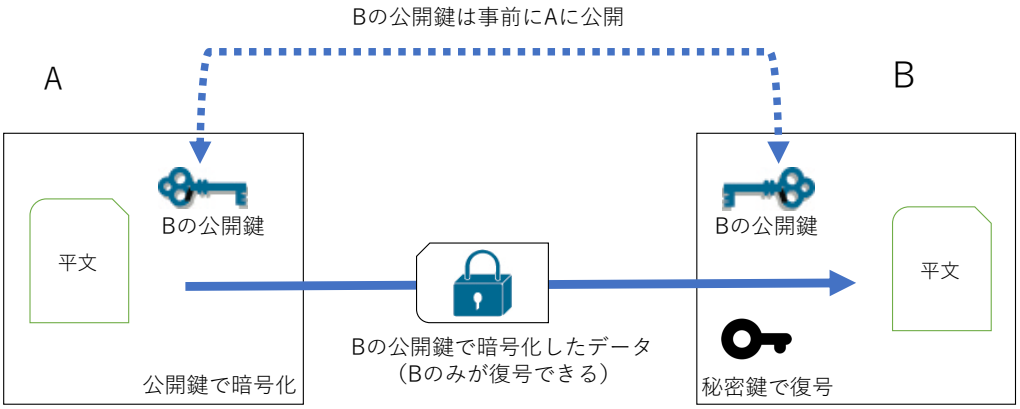

図 6　公開鍵暗号方式での暗号化

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。

2.3　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。

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

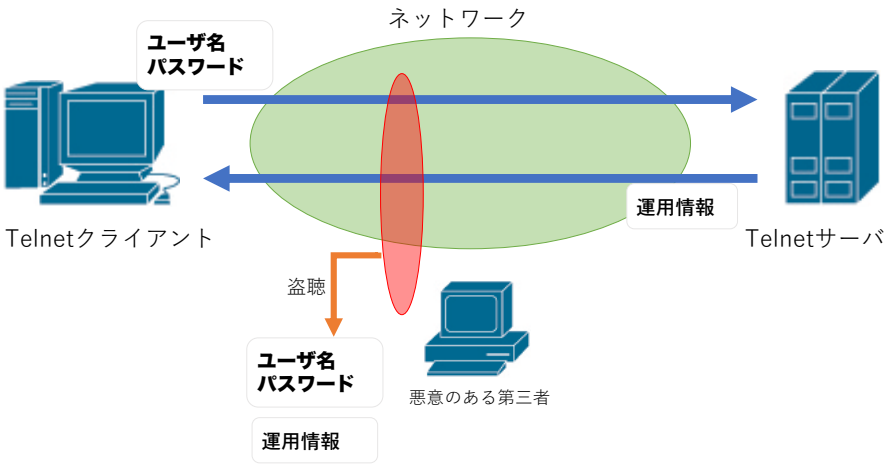

図 7　telnet 接続による脅威

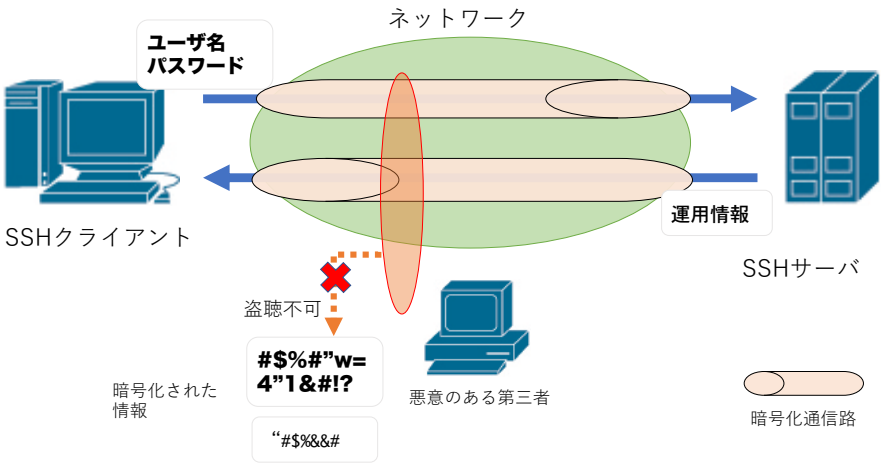

図 8　SSH 接続によるセキュアな運用管理

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。

2.3.1 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。

2.3.2 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 9 のように示す。

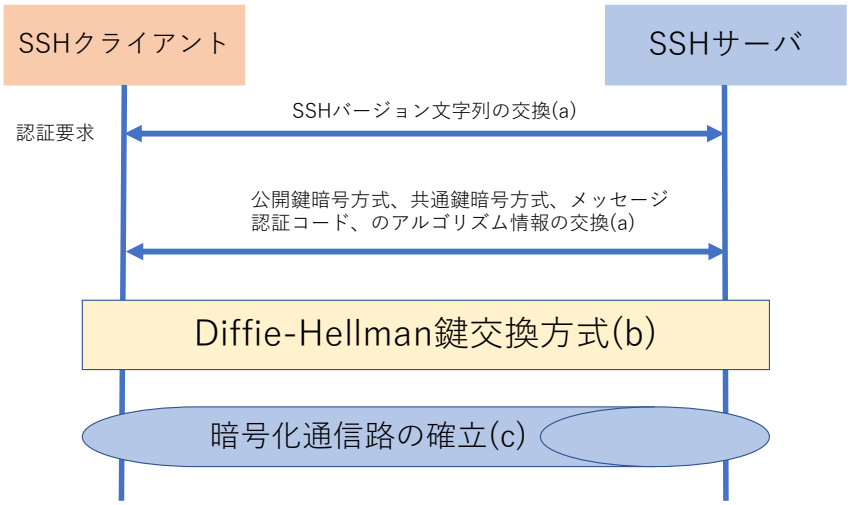

図 9　SSH 接続確立までのフロー

(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。

2.4 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 は 2 種類があり、インターネットなどの公衆通信網で使用するものと、通信事業者の閉域網で使用する VPN が存在する。本論文では前者を「公衆 VPN」、後者を「専用 VPN」を定義して説明する。

2.4.1 VPN の仕組み、使用法

3 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 ref に示す。

3.1 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のみにインストールすれば使用できる。本論文では、図 3 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 3 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 10 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。

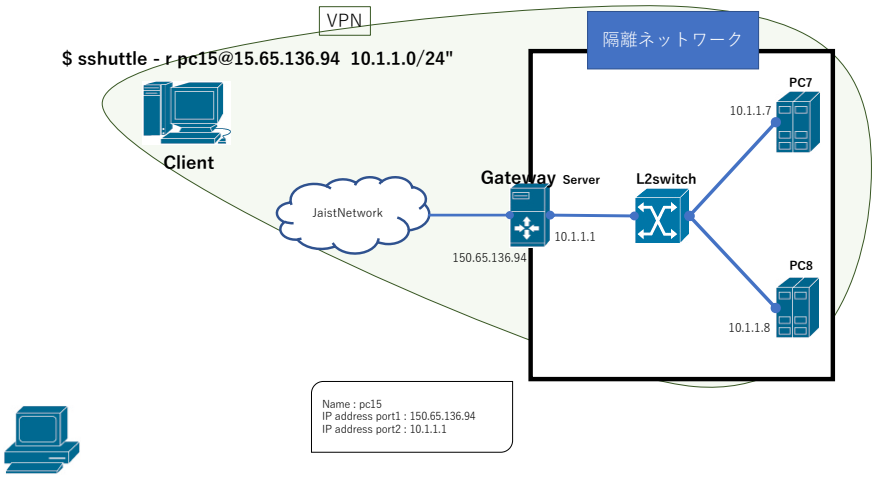

図 10　sshuttle

実際の terminal の画面出力を以下図に示す。
————————————-実際の画面————————————-

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 3 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。

3.2 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る、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 222 -l admin key setup default)”
- これによって

ユーザーを招待する。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 データベースにアカウントを作成する。

- config>user invite bob@
-

ユーザーの場合

評価

sshportal はインストールと管理が簡単にできるように作成されており、新規ユーザーの追加を管理者が動的に行うことができる。また、インストール先は Gateway Server のみで良いため、容易に使用することができる。

デメリットとしては、ユーザーの管理を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。

3.3 sshpiper

4 認証技術

大規模のアプリケーションを実現するために、使われている認証技術の解説を行う。

4.1 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 (Relatinal 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 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。
   ——————ここで図をパスワードを LDAP にまとめたような図を挿入
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。

LDAP の仕組み

————ここは図を使って説明できたら良い。

4.2 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 11 になる。

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

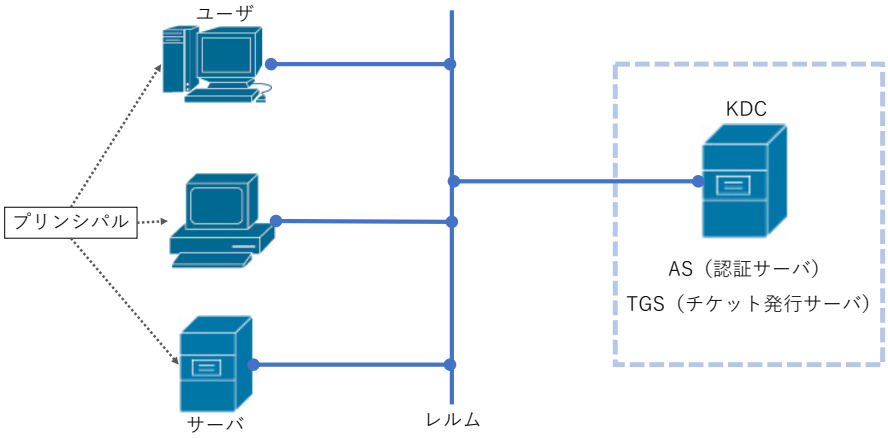

図 11　Kerberos 認証の構成要素

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 12 に示す。

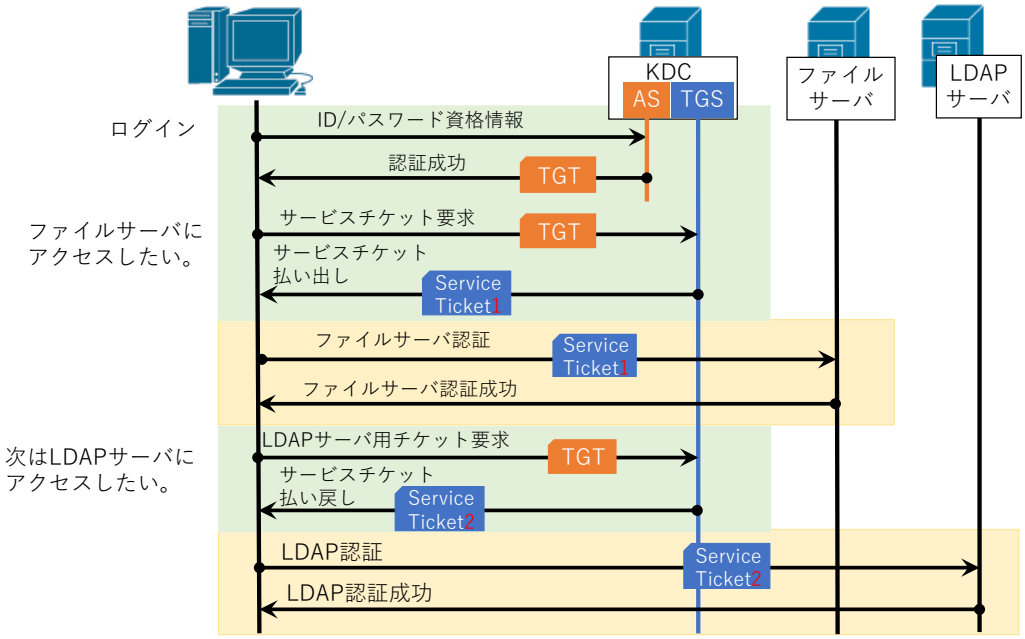

図 12　Kerberos 認証流れ

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。

4.3 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。無線 LAN や有線 LAN でのネットワーク接続時のユーザ認証のプロトコルとしても利用されている。

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。

- 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

全体の認証の流れは下図のようになっている。

————————-ここに図を挿入ーーーーーーーーーーーーーーーーーー

PC から Radius クライアントに送信されたユーザ名とパスワードは、Radius クライアントから Radius サーバへ”Access-Request”メッセージとして送信される。Radius サーバは送信されてきた情報と、Radius サーバが保持している情報とを照らし合わせて、正規ユーザかどうかを識別する。正規ユーザと判断されると Radius サーバは”Access-Request”メッセージで Radius クライアントに伝えて、その情報がユーザに伝えられる。

4.4 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。

5 大規模

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、評価表を下表に示す。

5.1 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。

————————————-ここに telport のアーキテクチャ図を————————————

————————————ここまで————————————

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

5.2 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

6　まとめ

参考文献

[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」.