# 基本情報

## アスキーコード

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Dec | Hex | Char |  | Dec | Hex | Char |  |
| 0 | 0 | NUL | Null (NUL) | 16 | 10 | DLE | Data link escape (DLE) |
| 1 | 1 | SOH | Start of heading (SOH) | 17 | 11 | DC1 | Device control 1 (DC1) |
| 2 | 2 | STX | Start of text (STX) | 18 | 12 | DC2 | Device control 2 (DC2) |
| 3 | 3 | ETX | End of text (ETX) | 19 | 13 | DC3 | Device control 3 (DC3) |
| 4 | 4 | EOT | End of transmission (EOT) | 20 | 14 | DC4 | Device control 4 (DC4) |
| 5 | 5 | ENQ | Enquiry (ENQ) | 21 | 15 | NAK | Negative acknowledge (NAK) |
| 6 | 6 | ACK | Acknowledge (ACK) | 22 | 16 | SYN | Synchronous idle (SYN) |
| 7 | 7 | BEL | Bell (BEL) | 23 | 17 | ETB | End of transmission block (ETB) |
| 8 | 8 | BS | Backspace (BS) | 24 | 18 | CAN | Cancel (CAN) |
| 9 | 9 | HT | Horizontal tab (HT) | 25 | 19 | EM | End of medium (EM) |
| 10 | 0A | LF\* | Line feed (LF) | 26 | 1A | SUB | Substitute (SUB) |
| 11 | 0B | VT | Vertical tab (VT) | 27 | 1B | ESC | Escape (ESC) |
| 12 | 0C | FF\* | New page/form feed (FF) | 28 | 1C | FS | File separator (FS) |
| 13 | 0D | CR | Carriage return (CR) | 29 | 1D | GS | Group separator (GS) |
| 14 | 0E | SO | Shift out (SO) | 30 | 1E | RS | Record separator (RS) |
| 15 | 0F | SI | Shift in (SI) | 31 | 1F | US | Unit separator (US) |

表（shift-JIS）



情報元：<http://www.isc.meiji.ac.jp/~re00108/ch11/asciicode.html>

## ANSIとShift-JISの違い

Shift-JIS(コードページ932)

## 情報系の団体

### IPA

独立行政法人情報処理推進機構

### mitre att&ck

マイターアタック

米国連邦政府が資金提供している非営利組織

対象分野はAI、直感的なデータサイエンス、量子情報科学、医療情報学、宇宙安全保障、政策と経済、信頼できる自律性、サイバー脅威の共有、サイバーレジリエンスなど

# ハブ

## 概要

スイッチングハブでは全二重、ハブでは半二重通信を行う。

なお、半二重通信ではCSMA/CD方式が用いられる。

## 用語（ハブ）

MDI

Auto MDI

MDI-X

Store-and-Forward

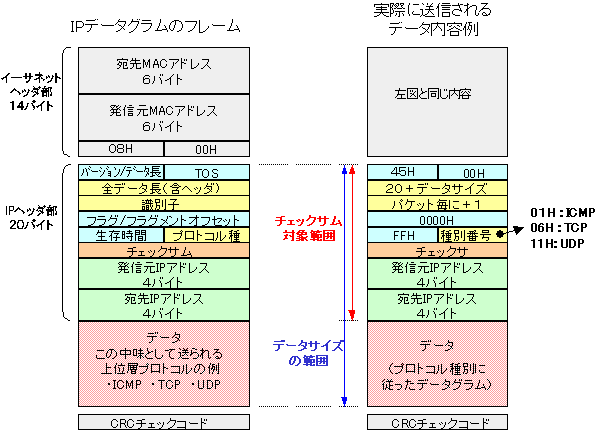
SW LSI

# イーサネット

## 注意点

IP固定の時に、IPv4が２つ割り振られたりする場合がある。その場合は、指定したIPアドレスを持つ機器が既に同ネットワークに存在している可能性が高い。別のIPを割り振りなおす。

## パケット



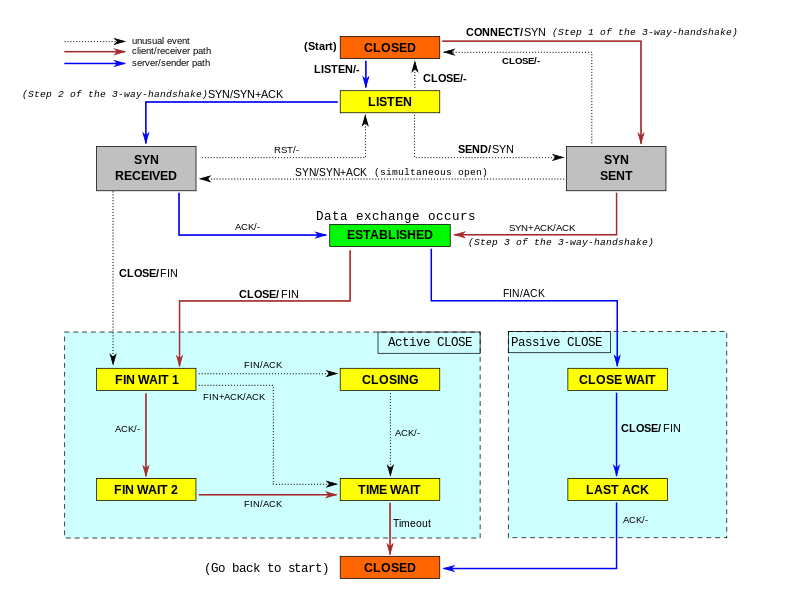
※この前にプリアンブル(7byte)、SFD；Start Frame Delimiter (1byte)　というのが先立つらしい。

FCS；Frame Check Sequence

イーサネット通信で受信したフレームに誤りがないかどうかを調べるために付加されるデータ。 CRCという値が含まれる。イーサネットフレームのデータの次、最後の4バイト。

## 状態移管図(TCP)

いわゆる「ハンドシェイク」の状態移管図



## ケーブル

### カテゴリ

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| カテゴリ | ケーブル印字 | 最大通信速度 | 伝送帯域 | 備考 |
| カテゴリ 5 | CAT.5 | 100Mbps | 100MHz |  |
| カテゴリ 5e | CAT.5e | 1Gbps | 100MHz |  |
| カテゴリ 6 | CAT.6 | 1Gbps | 250MHz |  |
| カテゴリ 6A | CAT.6A | 10Gbps | 500MHz |  |
| カテゴリ 7 | CAT.7 | 10Gbps | 600MHz |  |
| カテゴリ 7A | CAT.7A | 10Gbps | 1000MHz | 負荷の高いサーバー |
| カテゴリ 8 | カテゴリ8 | 40Gbps | 2000MHz | 業務用 |

### ピンアサイン

#### 100Base-TX

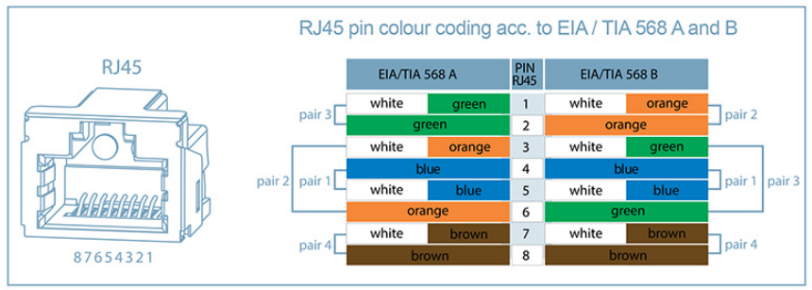
#### 1000Base-TX

#### 1000Base-T

### T568

T568-B

通称B結線。市販のストレートケーブルはほとんどB結線。

T568-A 通称A結線。

### UTPとSTP

UTP；Unshielded Twisted Pair

STP；Shielded Twisted Pair

実際にはUTPが採用される事が多い

・一般家庭ではアース接続する事は稀な為、シールドに溜まった電気を逃がせず、新たなノイズの原因になる。

・

|  |  |  |
| --- | --- | --- |
|  | UTP | STP |
| 使われる場所 | 一般家庭、オフィス | データセンター、工場 |
|  |  |  |

# ルーター

## ホスト名の設定

Windowではコマンドでhostname もしくはipconfig /all で確認できる。

### 無線LAN

無線LANの出力は電波法により最大10mW。これは、まったく障害物のない状況であれば、直線距離100ｍくらいなら通信が可能。最大250ｍでの通信に成功。

ただし2010年に電波法は改正され、今では屋内利用で200mWまでは無免許で使用できる。

## MACアドレス制限

・Ethernet通信でMACアドレスは暗号化されない

・ルーターを超えたパケットは、すべてそのルーターのMACアドレスを使う事になる

・MACアドレスの偽装は非常に容易。

以上の３つからMACアドレス制限にはそれほどの意味はない。

参考）<https://the01.jp/p000132/>

## SSID

セキュリティ対策として変更した方がいいと言う声もあるが、MACアドレスを基にした固有IDがデフォルトの場合もある。その場合は変える必要はない。

デメリット：

・通信料が増え、通信速度が落ちる

・モバイル機器の場合、バッテリーを多く消費する可能性

ファームウェアの更新

WAN側がWeb接続していないとできない。

### VLAN

ルーターはL3スイッチを用いる

ルーターは各VLANを１つのネットワークとして扱う。同じVLAN内での通信は可能。

異なるVLANにはブロードキャストすら届かない。

# 通信全般

## 参照モデル

### OSI基本参照モデル

|  |  |  |
| --- | --- | --- |
| 層 | プロトコル |  |
| 7. アプリケーション層  （レイヤー7） | HTTP・HTTPS・FTP・SFTP・DHCP・  SMTP・POP（メールそのもの）・SSH | 実際にユーザーが目にするGUI |
| 6. プレゼンテーション層 | SMTP（送信相手）・  Telnet・TLS（下層へ）  FTP（メール） | 一貫したデータの整合性を保つ。エンコード／デコード。暗号化、文字コード  テキスト、画像、ストリーミングといったデータ形式を区別 |
| 5. セッション層 | SSL/TLS・NetBIOS | 論理的な通信路を確立。例えばhtmlはページ情報を細かい**セッション**に分割してデータを処理している。 |
| 4. トランスポート層  （レイヤー4） | TCP・UDP | **IP+Port**。ACK, SYN, FIN  通信品質の確保。エラー訂正。データ単位はパケット。 |
| 3. ネットワーク層 | **IP**・ARP・RARP・ICMP | ネットワーク間の通信を管理。論理アドレス(**IP**) を割り振る。  カプセル化等。該当デバイスは**ルーター**（L3スイッチ）。 |
| 2. データリンク層 | PPP・Ethernet | ノード間の**MAC**アドレス管理。レイヤ１を密接に関係。  データの正当性や電気信号の誤り訂正。単位は**フレーム**。  該当デバイスはスイッチングハブ（L2スイッチ） |
| 1. 物理層 | 各種**ケーブル**・スイッチ | 電気信号の電圧・クロック、コネクタ形状など。  該当デバイスはハブ。 |

### TCP/IP 参照モデル

|  |  |  |
| --- | --- | --- |
| 層 | プロトコル |  |
| アプリケーション層  （OSI の5, 6, 7） | HTTP・FTP・DHCP  SSH・IRC・DNS  SMTP・IMAP | 通信アプリケーションで管理する。  Webサーバーソフト、ブラウザ、FTP通信ソフト。 |
| トランスポート層 | TCP・UDP | 通信品質の確保。エラー訂正。ACK, SYN, FIN  データ単位はパケット。データにヘッダをつける |
| インターネット層 | **IP**・IPsec  ICMP・IGMP | ネットワーク間の通信を管理。論理アドレス(IP) を割り振る。  カプセル化等。ルーターで使用。 |
| ネットワークアクセス層  （OSI の1と2） | PPP・Ethernet  ARP・RARP  Wifi（IEEE802.11） | 各種ケーブル、スイッチなど。プロトコルはPPP, Ethernet  デバイスドライバが対応。  ネットワークインターフェース層とも言われる |

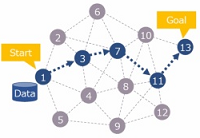
### Cisco３階層モデル

ネットワーク機器を構成する際に，各セグメント

|  |  |
| --- | --- |
| コア層 | 配下のスイッチ類を束ね，高速スイッチングを行いコアサーバーなどに接続。フルメッシュ型。 |
| ディストリビューション層 | 各フロアのスイッチを束ね，ルーティング処理を行う。  VLAN間のルーティングを行う。パーシャルメッシュ型。 |
| アクセス層 | クライアントPCやIP-Phoneなどのユーザのデバイスが接続する層。スター型。 |

### 通信トポロジー

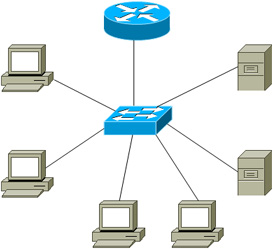
別名：ネットワークトポロジー

マルチホップ

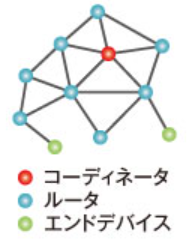
各デバイスに中継器機能を持たせ、バケツリレー式に繋いでいく

電力会社のスマートメーターが良い例。

ツリー型

スター型

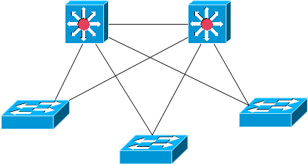
Wifiのアクセスポイントが良い例



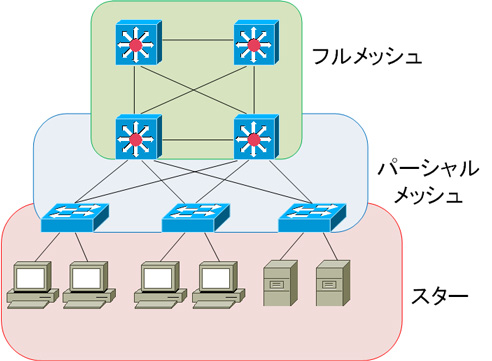
フルメッシュ型

中心ノードの周りに中継機をメッシュ状に配置した構成

必ずどこかの機器を経由して通信できる為，耐障害性が高い。

パーシャルメッシュ型

必要な部分だけメッシュ化し，コストと耐障害性のバランスを取る

ハイブリッド型

端末の役割によって左図のような複合型になる事が多い。

## データ形式

### 概要

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | データ形式 | セパレーター | 用途 | 備考 |
| JSON | Key-Value(入れ子) |  |  | Webで良くみられる |
| YAML | Key-Value(入れ子) | インデント |  |  |
| XML | Key-Value(入れ子) | タグ |  | マイクロソフト系に多い |
| TOML | Key-Value(入れ子) | インデント | 設定ファイル | 入れ子ができるiniファイル |
| CSV | テーブル | コンマ |  | 古いシステムに多い |
| TSV | テーブル | タブ |  | あまり使われてないような？ |
| Parquet | バイナリ | - |  | Apacheが開発  python:pyarrow |
| Avro | バイナリ | - | ビッグデータ | JSONに似ている |
| ORC | バイナリ | （列指向） |  |  |
| protobuf | バイナリ |  |  | Googleが開発 |
| Msgpack | バイナリ |  |  |  |

### json

JavaScript Object Notation

### YAML

### XML

### Parquet

### ORC

### SGML；Standard Generalized Markup Language

htmlやxmlの元になった汎用マークアップ言語

## キャッシュ

### 概要

### ユースケース

・データベースへの問い合わせ結果をキャッシュ

・複数アプリ／クライアント間でのメモリ共有

### 比較

|  |  |  |  |
| --- | --- | --- | --- |
|  | Redis | Memcached |  |
| プロセス | シングルスレッド | マルチスレッド |  |
|  |  |  |  |
| 特徴 | 複数のデータ型操作に対応 | ・文字列を文字列キーでインデックス  ・ディスクI/Oは一切ない |  |
| 永続化  （ディスクへの保存） | スナップショット  AOFログ | 不可 |  |

共通しているもの

OSS

低レイテンシー

多くのオープンソースクライアントがある。

Java、Python、PHP、C、C++、C#、JavaScript、Node.js、Ruby、Goなど。

複数ノードにデータを分散可能。

古いデータを捨てるタイミングを指定できる。

### Redis

#### 概要

インメモリDB

インストール

sudo apt install redis-server

ライセンス：BSD

設定ファイル：/etc/redis/redis.conf

Github：<https://github.com/redis/redis>

マニュアル　非公式？日本語訳：<http://redis.shibu.jp/index.html>

特徴

・Key-Value型のNoSQL

・サーバー／クライアント型

・FIFO(First In First Out)も可能

・スナップショットをディスクに保存する事ができる

・1 プロセス、1スレッドで動作

SHUTDOWN SAVEのようにリブートの時のみデータベースを永続化する事が可能？

情報元：<https://yakst.com/ja/posts/3243>

補足

・python標準のqueue.Queue()クラスを使っても良い

#### Redis Cluster

#### 設定ファイル

|  |  |  |  |
| --- | --- | --- | --- |
|  | データ型 | 概要 | 初期値 |
| bind | str |  | 127.0.0.1 |
| port | int |  | 6379 |
| daemonize | yes/no |  | yes |
| pidfile |  | daemonize=yesの時に必要 | /var/run/redis/redis-server.pid |
| databases | int | Redis のデータベースの個数 |  |
|  |  |  |  |
| loglevel | str | debug, verbose, notice, warning |  |
| maxclients | int |  | 10000 |
| slaveof |  | レプリケーションの設定（slave側に記述）  slaveof <masterip> <masterport> |  |
| maxmemory | str |  | 500MB |
| save |  |  |  |
| rdbcompression | yes/no | 圧縮保存 | yes |
| dbfilename | str | 保存ファイル名 | dump.rdb |
| dir | str | 保存ディレクトリ名 | /var/lib/redis |

参考：<https://qiita.com/wind-up-bird/items/f2d41d08e86789322c71>

公式：<https://redis.io/docs/management/config-file/>

#### コマンド

set 値を代入

set test "hoge"

get 値を取得

expire 特定のkeyにttlの制限を設ける

expire test 10

ttl

### Memcached

インメモリDB

### 用語

|  |  |  |
| --- | --- | --- |
| 項目 | 俗称 | 説明 |
| FIFO(First In First Out) | キュー | 先に入ったもを先に出す。 |
| LIFO(Last In First Out) | スタック | 後に入ったものを先に出す。 |

# 映像系

H.264

Iフレーム

Pフレーム 前のフレームの差分のみを扱う

# 通信プロトコル

## IP（internet protocol）

### IPv4

#### 基本

##### 概要

IPアドレスはネットワーク部とホスト部で構成される。

ホスト部は更にサブネット化をする事ができる。

##### グローバルアドレス

WAN(Wide Area Network)に直接接続するルーターなどに割り当てるアドレスの事。IPv4のアドレス不足に伴い考え出された。ICANN（Internet Corporation for Assigned Names and Numbers）と呼ばれる非営利組織が，個々のIPアドレスが国や組織によって重複しないように管理している。

##### プライベートアドレス

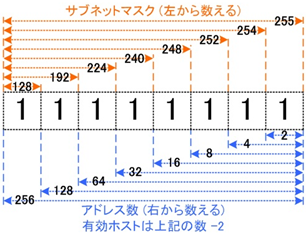
別名：ローカルアドレス

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| アドレス範囲 | プライベート | ネット数 | ホスト数 | CIDR | サブネットマスク | クラス |
| 0.0.0.0～  127.255.255.255 | 10.0.0.0～  10.255.255.255 | 27-2  =126 | 224-2  ≒16百万 | /8 | 255.0.0.0 | クラスA |
| 128.0.0.0～  191.255.255.255 | 172.16.0.0～  172.31.255.255 | 214-2  =16,382 | 216-2  =65,534 | /16 | 255.255.0.0 | クラスA |
| 192.0.0.0～  223.255.255.255 | 192.168.0.0～  192.168.255.255 | 221-2  ≒2百万 | 28-2  =254 | /24 | 255.255.255.0 | クラスA |

##### サブネットマスク

どこからどこまでがサブネット部でどこからどこまでがホスト部であるかを示す。

##### ネットマスク表

例：24～32bitの時の変化

|  |  |  |  |
| --- | --- | --- | --- |
|  |  | ネットマスク | ワイルドカードマスク |
| /23 | -1 | 255.255.254.0 | 0.0.1.256 |
| /24 | 基準 | 255.255.255.0 | 0.0.0.256 |
| /25 | +1 | 255.255.255.128 | 0.0.0.128 |
| /26 | +2 | 255.255.255.192 | 0.0.0.64 |
| /27 | +3 | 255.255.255.224 | 0.0.0.32 |
| /28 | +4 | 255.255.255.240 | 0.0.0.16 |
| /29 | +5 | 255.255.255.248 | 0.0.0.8 |
| /30 | +6 | 255.255.255.252 | 0.0.0.4 |
| /31 | +7 | 255.255.255.254 | 0.0.0.2 |
| /32 | +8 | 255.255.255.255 | 0.0.0.1 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| bitが | ネット数 | ホスト数 |  |  |
| 増えると | 倍になる | （約）半分になる |  |  |
| 減ると | 半分になる | （約）倍になる |  |  |
|  |  |  |  |  |

補足：ホスト数はブロードキャストアドレスとネットワークアドレスが有るので単純に倍にはならない

アドレス数（ネットワーク，ホスト共に同じ）

2n-2 nはビット数 例n=8　28=254個

補足：サブネットマスクは28-2 (8-n) といったような考え方になる

補足：実際にはCIDRは8~32(bit)の範囲しかとらない

#### ネットワーク設定

##### デフォルトゲートウェイ

　宛先がルーティングテーブルに書かれていないネットワークだった場合の第一転送先。

他のネットワークへのルートを知っているノード。通常ルーター。

##### DNSサーバー

デフォルトゲートウェイと同じ(IP)にする設定が多い 19.05.16

#### 用語(IP4v)

CIDR表記

255.255.255.0のサブネットマスクを/24と記述する方式。

/24は上から24bit分がサブネットである事を示す。

クラスレスアドレス

「サブネットマスク」を用いたサブネットの管理方法。

現在ではクラスを用いる事の方が少なく，こちらの方が普通

（対）クラスフルアドレス

クラスフルアドレス

インターネットの初期に用いられたアドレスの管理方法。

現在でもアドレス管理の考え方の基本として残っている。

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| クラス | 先頭ビット | アドレス範囲 | 備考 | CIDR | サブネットマスク |
| A | 0 | 0.0.0.0～127.255.255.255 | 第１クオテットがネットワーク部 | /8 | 255.0.0.0 |
| B | 10 | 128.0.0.0～191.255.255.255 | 第２クオテットまでが　〃 | /16 | 255.255.0.0 |
| C | 110 | 192.0.0.0～223.255.255.255 | 第３クオテットまでが | /24 | 255.255.0.0 |
| D | 1110 | 224.0.0.0～239.255.255.255 | IPマルチキャスト用 |  |  |
| E | 1111 | 240.0.0.0～255.255.255.255 | 実験用に予約されている  実際に使用する事はない。 |  |  |

#### 特殊アドレス

##### ブロードキャストアドレス

同一セグメントの全てのノードに一斉送信する為のアドレス。

ホスト部が全て1のアドレス。

比較

|  |  |  |  |
| --- | --- | --- | --- |
|  | 相手 | IPアドレス | 備考 |
| ユニキャスト | 1対1 | 相手のIP |  |
| マルチキャスト | 特定のグループ宛 | 224.0.0 ～  239.255.255.255 | 同一動画のキャストなどに適す。 |
| ブロードキャスト | 同一ネットのすべてのデバイス | 全てのビットを1に |  |

リミテッドブロードキャスト

255.255.255.255 の事。ルーターを超えて別ネットワークへは送信されない。

ディレクテッドブロードキャスト

192.168.1.255 のようなアドレスの事。許可されている場合であれば、ルーターを超えて送信される。

##### ループバックアドレス

アドレスは127.n.n.nとなり，生存確認に用いられる。

特に「ローカルループバックアドレス」は自分自身を指し，慣例的にlocalhostと言うホスト名があてられる。

IPv4　127.0.0.1

IPv6　0:0:0:0:0:0:0:1（::1）

##### リンクローカルアドレス

DHCPサーバーなどに接続できず、自分自身（APIPA）で発行するアドレス。

ルーターにより他ネットワークへの接続は遮断される。

IPv4　169.254.0.0 /16

IPv6　 fe80::

### IPv6

#### 基本

表記法

基本：

16bit×8セクテット=128bit。16進数で表現=1文字2byte。

ABCD:EF01:2345:6789:ABCD:EF01:2345:6789

省略

各セクテットの先頭の0の並びは省略可能　例 0012⇒12

各セクテットがすべて0の場合は0に省略可能

一か所のみ0連続ところを::で省略できる

例：FF01:0:0:0:0:0:0:101 (マルチキャストアドレス) ⇒ FF01::101

サブネット

2001:0DB8:0:CD30:123:4567:89AB:CDEF/60

ノードアドレス 2001:0DB8:0:CD30:123:4567:89AB:CDEF

サブネット番号 2001:0DB8:0:CD30::/64 64=

（参考）<https://www.infraexpert.com/study/ipv6z2.html>

#### 特殊アドレス

##### ループバックアドレス

「::1」（0:0:0:0:0:0:0:1）

### 用語（IPv6）

#### ユニキャストアドレス

単一インタフェースの識別番号

#### エニーキャストアドレス

インタフェースの集合の識別番号で、 普通は複数の別機器に付与されます。

そのアドレスを持つ「一番近くにある」機器のインタフェースに配送されます。

エニーキャストアドレスはユニキャストアドレス空間から割り当てられ、 表記上、 エニーキャストアドレスとユニキャストアドレスの区別はつきません。

#### マルチキャストアドレス

インタフェースの集合の識別番号でアドレスを持つすべてのインタフェースに配送される。

IPv4のブロードキャストに近い

#### Egress-Onlyインターネットゲートウェイ

NATゲートウェイとインターネットゲートウェイの特徴を併せ持つIPv6専用の機能

### 用語（IP共通）

#### RIP；Routing Information Protocol

経由する可能性のあるルータをホップ数という値で数値化し、

目標値までの最短経路を動的に決定する。

#### 仮想IP

別名：Vip、Virtual IP、リロケータブルIP

複数の機器で同じIPアドレスを使用する事。システムの冗長化などに使われる。

#### その他

OUI；Organizationally Unique Identifier

MACアドレスの前半部で、メーカーごとに割り当てられる番号

ISP；Internet Service Provider

インターネット接続業者のこと。通常グローバルIPアドレスはここで割り振られる。

LLMNR

IGMPv3

Segment（セグメント）

192.168.1.0/24 などで分けられる、１つのネットワークの事。

DNS；Domain Name System

ドメイン名とIPの対応付け、 メールの宛先ホストを指示するためのシステム

DDNS (Dynamic DNS)

固定のドメイン名で変動IP に接続する。

※プロバイダを用いた通信（PPPoE）などでは接続毎にIPが変わる。

NAT

プライベートIPアドレスとグローバルIPアドレスを変換して通信する技術。

問題点として、プライベートネット内でIPアドレスが増えすぎると対応できなくなる。

SNAT；Source NAT プライベートIP→グローバルIP

DNAT；Destination NAT グローバルIP→プライベートIP

NAT Traversal

別名：NAT越え

NATを使う時の各種問題を解決するために特殊なプロトコルを使用して通信するアルゴリズム。

中継するゲートウェイがNAT Traversalに対応している必要がある。

具体的にはSTUN、TURN、ICEなどが使われる。

NAPT(Network Address Port Translation)

アドレス変換に加えてポート番号も変換する技術。

NetBIOS

クライアントのIDを示す文字列。NetBIOS自体は現在では使われていないが、仕様は今も使われている。

NetBIOS名は15byte + 0x00 の固定。未使用バイトは0x20（スペース文字）で埋める

Node

データ端末。再配布ポイント（データ回線終端装置など）かエンドポイント（データ端末装置など）のいずれか

PPPoE

PPP over Ethernet。パスワードとIDが必要な通信。通常プロバイダとの接続を指す。

## IP系プロトコル

### TCP

### ARP

Address Resolution Protocol

調べたいIPアドレスを同一ネットワークにブロードキャストで投げて（arp リクエスト；who has…? というやつ）

対象のIPを持つ端末がmacアドレスを返す。（arpリプライ）ちなみにリプライはunicast。

あまり知られていない（らしい）が、ARP 応答を受け取ったときだけでなく、ARP 要求を受け取ったときにも ARP テーブルの更新が実施される。つまり、同セグメントにいる全ての端末が、ある意味ではARPリクエストの影響を受ける。

通常、期限が切れる一定時間まえにArp更新がUnicast で行われるが、あるルーターでは１分間隔にまとめてArp Request したりする。WindowsOS は2分毎らしい。

### IPsec；Security Architecture for Internet Protocol

暗号化によってパケットの秘匿や改ざん検知を実現するプロトコル

### ICMP；Internet Control Message Protocol

データが送信元から送信先に届くまでの処理で発生したエラーや状態を送信元に報告するためのプロトコル。

ちなみに、pingはICMPのうちEchoのみを使用する。

ICMPリダイレクト L3スイッチが特定の宛先に対応するゲートウェイを通知する機能

### SOCKS

TCP/UDP通信を代理（プロキシ）する為のプロトコル

現在ではネットワーク機器が内外の通信を中継するために使われているが、もともとはファイアウォールの一部として、他のセキュリティ製品とともに管理を容易にするために作られたセキュリティプロトコルだった。

ポート番号には1080がよく使われている。

### SSDP

UPnP（Universal Plug and Play）の通信プロトコルの内の１つ。HTTPに近い形のテキストをマルチキャストで送信し、対象機器は発信元に対しリプライを送信する。

※家庭用無線LANルーターでUPnP機能を有効にしておくと，Windows OSでデフォルトで有効になっているSSDPのマルチキャストパケットに反応してしまい，ルーターがSSDPの発信元MACアドレスを探る為にARP Requestのブロードキャストを行う。RequestでARPテーブルに追加されるが、あまりに量が多いとARPの期限切れよりも前にテーブルからあふれ、再度ARP Requestを送る、という悪循環が生み出される。

### NBNS；NetBIOS Name Service

ネットワーク内のコンピューターの NetBIOS名などの問い合わせと回答を行うプロトコル。

nbtstat などで使用。

### SNTP

Simple Network Time Protocol

NTPパケットを利用した、簡単な時計補正プロトコル

NTP（プロトコル）の方が精度が高い？2021-09-07

SNMP；Simple Network Management Protocol

IPネットワーク上のルータやスイッチ、サーバ、端末など様々な機器をネットワーク経由で遠隔から監視・制御するためのプロトコル。UDP/IPベースでCPU使用率、トラフィックなどの監視を行う。

LAN（イントラ）で良く使用される。

（関連）Telemetry（テレメトリー）

トラップ

SNMP監視サーバーに通知を非同期で送る為の仕組み。

デバイスの障害やエラー、トラフィックの異常、不正アクセスなどのセキュリティイベント、パフォーマンスの低下、デバイスの再起動　などのイベントを通知する事ができる。

### OSPF；Open Shortest Path First

各ルータがネットワーク全体の「リンク状態」を学習し、その内容にしたがってパケットの中継動作を行う。（リンクステート型）

各ルータは自らと直接繋がれたルータのリストをLSA（Link State Advertisement）パケットで他のルータに知らせ、ルータ間のリンク状態を集約したデータベース（LSDB：Link State Database）を構築する。

規模の大きい企業ネットワークに対応できるルーティングプロトコル

## Ethernet系

### EtherCAT

Ethernet for Control Automation Technology

Beckhoff Automationによって開発されたイーサネットに基づくフィールドバスシステム

主にPLCや工作機など繋ぐ目的で使われる。

機器の動作電源（主に24VDC）を含んだEtherCAT Pという規格も存在する。

（比較）CC-Link IE

### TLS；Transport Layer Security

通信する際のセキュリティのプロトコル

デジタル証明書を使用して、送信側（または受信側）の正当性を検証する仕組み

SSLのバージョンによってTLSのバージョンも変わってくる

TLSプロトコルに組み込まれた3つの要素：

認証 - 提供されたIDの正当性を検証できること

暗号化 - 1つのホストから別のホストに送信された情報を難読化できること

完全性 - 偽造と改ざんを検出できること

### TSN

Time-Sensitive Networking

イーサネットをベースにしながら時間の同期性が保証されており、リアルタイム性を担保できるネットワーク

## メール

POP(Post Office Protocol)

ポート：110 or 995(SSL)

※通信パケットにパスワードも含まれる為、丸見えだったりする

IMAP(Internet Message Access Protocol)

ポート：143 or 993(SSL)

POPと同じくメール受信のプロトコル。但しサーバーにメッセージが残る。

APOP

SMTP(Simple Mail Transfer Protocol)

送信用プロトコル

ポート：25(一般) or 587(投稿ポート) or 465（SMTP over SSL）

SPF；Sender Policy Framework

自ドメインのDNS情報に送受信可能リストを作成し、電子メール送信元のドメイン詐称を検知する。

平たく言うと、迷惑メール対策。

### SMTP

Simple Mail Transfer Protocol　Port: TCP25　TCP465(SSL)

送信のプロトコル（手順）

TCP587（投稿ポート）25番ポートは迷惑メールの配信元に多いので、代替として用いるポート。

### IMAP

Internet Message Access Protocol　 Port:TCP143（暗号なし） TCP993(SSL)

常にサーバー上のメールにアクセスする。サーバー側で既読・未読状態を保持する。

### POP

Post Office Protocol　 Port: TCP110（暗号なし） TCP995(SSL)

クライアントのメーラーなどでサーバー(≒Post) にアクセスしてメールをダウンロードする方式。

UIDL（メッセージに一意のID）を照合する事で受信状況を把握する。

#### POPコマンド一覧

USER ユーザー名 認証するユーザー名を指定する

PASS パスワード 認証するユーザーのパスワードを指定する

STAT メールメッセージの数とサイズを応答する

LIST [メッセージ番号] メールメッセージ番号とそれぞれのサイズを応答する。メッセージ番号が指定された場合には、該当メッセージ分のみが対象

RETR メッセージ番号 指定されたメッセージ番号のメッセージ全体を表示する

DELE メッセージ番号 指定されたメッセージ番号のメッセージを削除する

※逆に言うと、これを送信しなければ、POPであっても消えない。

NOOP 何もしない

RSET 認証確立後発生した削除処理を全て取り消す

QUIT ログアウトする

※注意！　いずれも末尾に \r\n が必要

情報元：

<http://www.atmarkit.co.jp/fnetwork/rensai/netpro07/pop3-commands.html>

### SMTP-AUTH

メール送信・転送するSMTPの拡張プロトコル。

ユーザー名、パスワードで認証を行う。

### APOP

メールサーバーのメールを取り出す時に使用するプロトコル。

通常平文で送られるユーザー名とパスワードをハッシュ化して送信する。

### POP3S

POP3 over SSL/TLS

POP通信全体をSSLで暗号化する

MX

## html関連

### WebSocket

#### 概要

サーバ・クライアント間で**双方向の非同期通信**を実現する技術。

URLスキーマとして ws:// または、暗号化された wss:// を用いる。

（参考）<https://www.tohoho-web.com/ex/websocket.html>

## 工業系

### OPC UA

マルチベンダー製品間や、異なるOSにまたがってデータ交換を可能にする高信頼の産業通信用の通信規格。

IEC62541。COM/DCOM技術がベース？SOAPベース、HTTPSベース、TCPベースのプロトコルが存在。

Native Binaryの場合、TCPでハンドシェイク→通信　　結局中継サーバーが必要らしい

python

pip install opcua

CODEYES はOPC UAサーバーにもなる。

https://misoji-engineer.com/archives/opc-ua.html

opc ua

関連ソフトウェアなど

UA Expert OPC UAのクライアントソフト

### SECS

SEMIが提唱するCIMの通信ソフトウエア規格で半導体製造装置等のプロセス設備とFA自動化のホストシステム間の通信を定義し標準化。

## 認証系

### LDAP

Lightweight Directory Access Protocol

シングルサインオンにおいて、ネットワークを介して認証する為の仕組み。

## Web 通信系

### まとめ

#### 概要

WebAPI比較

|  |  |  |  |
| --- | --- | --- | --- |
|  | REST | GraphQL | gRPC |
| 特徴 | ・5種類のhtmlメソッド  ・思想設計のみ | ・SQLに思想は近い  ・リクエストをjsonにしてhtml-PUTで投げる。 | ・Socket通信  ・複数のプログラム言語対応  ・厳格な型付け |
| データ | 一度に１つ  ⇒対応策はあるらしい | 複数のリクエストを１回で済ます | シリアライズ（ストリーム）  独自プロトコル |
| 通信 | json/xml（HTTP/1.1） | json（HTTP/1.1） | バイナリ　HTTP/2 |
| 長所 | ・高い汎用性  ・一貫した操作コマンド | ・複数データを一括で  ・型定義できる | ・より複雑なデータを一括で送受信できる  ・通信が高速 |
| 短所 | ・文字列なので型が曖昧  ・状態保持をしない  ・送受信データ量が多い | ・サーバー側で処理増  ・状態保持をしない  ・自由度ゆえに保守が難 | ・学習コスト高め  ・セキュリティ対応難  ・Webブラウザ非対応 |
| 備考 | フロントエンド向き | BFF※がほぼ必須？ | マイクロサービスに向く |

※RPC；Remote Procedure Call

※BFF：サーバーとPCの間で動くプログラム（らしい）

SOAP

### htmlメソッド

#### GET

・第一原則として、**データの取得以外何も行わない**ようにする。

・クエリ文字列を用いて良いのはGETだけ。2022-01-18なぜ？？

・反対に、リクエストボディを用いてはいけない。

参考：<https://datatracker.ietf.org/doc/html/rfc7231>

#### DELETE

・クエリ文字列もリクエストボディも用いてはいけない。

・できるだけDELETEを実装しない設計を目指す。POSTなどで対応する。

補足：最悪なのはGETに削除機能を持たせること。クローラーなどが削除してしまう場合があるらしい。

まとめ

#### メソッド比較

|  |  |
| --- | --- |
|  |  |
| GET |  |
| POST |  |
| PUT |  |
| PATCH |  |
| DELETE |  |

#### 開発言語ごとのhtmlメソッドへの対応

##### 概要

|  |  |  |
| --- | --- | --- |
| 対象 | 対応 | 備考 |
| PLC | × | Phoenix ContactのPLCは対応品があるとか？ |
| .NET | 〇 | System.Net.WebRequestを使う（らしい） |
| Python | 〇 | Requests, BeautifulSoup, Seleniumなど？ |
| C言語 | 〇 | curlが使えるらしい　　補足：Winsock2.h と ws2tcpip.h を使う？ |
| エクセル(VBA) | 〇 | MSXML2.XMLHTTPをつかう（らしい） |
| Android | 〇 | HttpUrlConnectionを使う？ |
| iOS | 〇 | NSURLSessionを使う？ |
| Arduino | △ | 自力で組むらしい？ |

##### エクセルVBA

Function RESTAPI(URL As String)

Dim objXMLHttp As Object, zipArr

Set objXMLHttp = CreateObject("MSXML2.XMLHTTP")

objXMLHttp.Open "GET", URL, False

objXMLHttp.Send

RESTAPI = objXMLHttp.responseText

End Function

ただし、標準ではjsonのパーサー（変換モジュール）が存在しないらしい。

#### 資料(htmlメソッド)

<https://cloud.google.com/apis/design/custom_methods>

### REST（RESTful）

#### 概要

Webアプリ通信における設計思想の１つ。（プロトコルでは無い）

GET、POST、PUT、DELETE等のHTTP標準のメソッドを使う。

セッション管理を行わない

参考クエリ

URLに関数のパラメータにあたる文字列を入れる。（厳密にはURIと言うらしい）

さらに細かい引数はクエリ文字列で指定。（すべてURIの場合もある？）

http://192.168.1.1:8080/nagoya/floor2/plc1/D100/?type=singed16

補足：ここでの192.168.1.1はPLC中継サーバー（MXコンポーネントの代用サーバー）

#### 問題点と対策

##### 複数リソース(bulk)対応

問題点

１セッションで１データしか取得／送信できない。

postでは requestボディにjsonを入れる事で対応できるが、GETはどうしようもない。

対策１：検索クエリを用いる

GET /member:search ＋クエリ文字列で条件を提示

:verbで示すのはgoogleの方式らしい。割と推奨されている様子。

対策２：ドットで区切る

例：GET /member/1,2,3/

例：GET /member/?id=1,2,3 クエリ文字列の場合

対策３：複数リソースを取得するメソッドを用意

⇒ 事前に準備しないといけないようでは使いにくい

例：GET /member:bulk　GET /member/bulk　 GET /member**s**/

対策４：単一読み込みを繰り返す

これが一番シンプル。

対策５：Graph APIにする

学習コストもかかるし、あまりお勧めできない

注意点

・GETで複数リソースを取得するのはRESTっぽくないので、必ず書類を残す。

・

補足

・bulkでなく、batchという言葉で表現される事もあるらしい。

#### ベストプラクティス

##### URI名称

名詞を使う

GET https://api.example.com/v1/get\_items 悪い例

GET https://api.example.com/v1/items 良い例

POST https://api.exmaple.com/v1/items/search 微妙

ただし、searchなど許可されるものもある。結局のところ、整合性がとれるならば良い。

ネスト化しない

https://api.example.com/v1/sports/1/players/2/friends 悪い例

https://api.example.com/v1/sports/1 良い例

https://api.example.com/v1/players/2/friends

APIである事を明確に

・URIを見るだけでWeb APIである事が分かるようにする

・バージョン情報を入れる

API https://example.com/**api**/**v1**/items

API https://**api**.example.com/**v1**/items

例

https://service.name/v1/some/resource/name:customVerb # Googleで紹介されているもの

##### レスポンスデータはフラットに

#### 関連ソフトなど

Swagger RESTful APIを構築するためのオープンソースのフレームワーク

なんでもできるのは便利だが、その分学習コストもかかる　というコメントも

API Blueprint APIの仕様をMarkdown拡張記法で記述できる言語

\*.apib というファイルにMarkdown方式で記述する

主にnode.jsで使用する？

#### 資料

REST命名規則：<https://note.com/yamarkz/n/n41e9ac83c896#8IF9v>

RESTの参考資料：

<https://qiita.com/NagaokaKenichi/items/0647c30ef596cedf4bf2>

### MQTT

#### 概要

Message Queueing Telemetry Transport；エムキュウティティ

WebSocketを使った256byte以下の超軽量通信。センサーなどで使用。

軽量なソケット通信のようなプロトコル。独自の通信ポート（TCP1883，TLS8883）を用いる。

ローカル環境でセンサ情報のような、軽く、イベント性のある電文を扱うのに向く。

送信者（Publisher）、受信者（Subscriber）、中継者（Broker）の３つでネットワークを構成。

Publisher→Broker→Subscriberとデータは送られる。一般的にはBrokerはクラウドサーバーが担う事が多い。

Publisherは送信先、Subscriberは送信元を**一切気にせずに通信を行う**のが特徴。

データとしてはtopicと値のペア。 例：topic：’/nagoya/room1/temp’ 値：27.5

パッケージ例

mosquito, uMQTTBroker

（参考）<https://www.silex.jp/blog/wireless/2016/09/mqtt.html>

#### 用語（MQTT）

Topic

メッセージ種別の識別子で「/」で区切られた階層構造をしている。（例）sensor/temp

Subscriberは受信したいトピックを指定することで、欲しいメッセージだけを手に入れることができる。

$SYS/brokerなどで始まる、システム用のトピックも存在する（らしい）。

先頭にスラッシュ(/)をつけない

大文字小文字を区別しない

小文字、数字、ダッシュ(-)だけを使う

ワイルドカード：一階層の場合は「+」。複数階層の場合は「#」。

Retain

Topicごとに最後にPublishされたメッセージをbrokerが保持しておく機能。

例えば10分にpublishされる情報をSubscriberがいつでも取りに来られるようにする。

Will

Publisherが切断されてサーバーとの通信ができなくなったときに指定されたTopicとメッセージをSubscriberに送信する機能。補足：willは英語で遺言

QoS

通信品質。Subscriberに確実にメッセージを届ける為の機能。（比較）Retain

QoS0しか実装されていないBrokerも多いらしいので注意

0 メッセージを受信の保証なしに送信。

1 受信の保証付きで送信。重複受信の可能性あり。

2 受信の保証付きで送信。重複受信の可能性なし。

ws: WebSocker 1884 が使われる？

wss: WebSocket Secure

#### 詳細

##### ソフトウェア

##### paho

python用のパッケージ

from paho.mqtt import client as paho\_client

def on\_connect(client, userdata, flags, rc):

client.subscribe("#", 0)

print(f'Connected with result code rc: {rc}')

def on\_message(client, obj, msg):

print(msg.topic+" "+str(msg.qos)+" "+str(msg.payload))

if \_\_name\_\_ == '\_\_main\_\_':

client = paho\_client.Client()

client.on\_message = on\_message

client.on\_connect = on\_connect

# client.on\_publish = on\_publish

client.connect(host='192.168.11.92', port=1883, keepalive=60)

client.loop\_forever()

参考：https://qiita.com/udai1532/items/9e4cca55f75b2c213173

##### ポート番号

1883

8883 MQTT over SSL

##### Apache Kafka

MQTTに似た仕様らしい

#### mosqitto

##### 概要

最も有名なMQTT「ブローカー」のソフトウェア。クライアント版もある。

sudo apt install mosquitto # brokerのみ？

sudo systemctl status mosquitto # インストール確認（動作状況の確認）

sudo apt install mosquitto-clients # mosquitto\_sub、mosquitto\_pubなど

sudo apt install mosquitto # brokerのみ？

sudo systemctl status mosquitto # インストール確認（動作状況の確認）

mosquitto\_sub subscriber

mosquitto\_pub publisher

/usr/sbin/mosquitto に入るらしい

##### チュートリアル

他のPCからの接続

[ etc/mosquitto/mosquitto.conf ]

listener 1883 # デフォルトのポート

protocol websockets

allow\_anonymous true # パスワード認証しない場合はこれが必要

パスワード認証？

##### 共通オプション

-t メッセージトピック

-d デバッグメッセージを有効

-r Retain機能を使う

-q QoSの指定

##### mosquitto\_pub

mosquitto\_pub -h localhost -t test -m "Hello"

##### mosquitto\_sub

mosquitto\_sub -h localhost -t sub

（公式）<https://mosquitto.org/man/mosquitto_sub-1.html>

##### 設定ファイル

場所：etc/mosquitto/mosquitto.conf

listener

listener 1883 # デフォルトのポート；他のPCから接続する際には必要

listener 9001 # WebSocket用

listener 8883 # TLS接続用

protocol websockets

allow\_anonymous true # パスワード認証しない場合はこれが必要

補足　TLS；Transport Layer Security 公開鍵証明書；WANでの通信時に必要となる？

ソケット通信9001？

#### MQTT.js

##### 概要

MQTT.js JavaScripで使う事ができるクライアントソフトウェア。

基本的にはNode.jsで使う

[ インストール ]

npm install mqtt -g

[ sample.js ]

const mqtt = require('mqtt') // require mqtt

const client = mqtt.connect('est.mosquitto.org') // create a client

但し完全JavaScript版もあり、htmlで利用が可能

[ sample.html ]

<head>

<title>MQTT.js Test</title>

<script src="https://unpkg.com/mqtt/dist/mqtt.min.js"></script>

<script>

var client = mqtt.connect('ws://192.168.1.1:9001');

client.subscribe('hoge');

</script>

</head>

公式Github：<https://github.com/mqttjs/MQTT.js/>

参考（英語）：https://www.emqx.com/en/blog/mqtt-js-tutorial

補足：意外と情報が少ないので、英語でも貴重な情報源

##### チュートリアル

ポイントは

・ブローカーでソケット通信のポートを開く事

・Webブラウザでは直接MQTTプロトコルを扱えない為、プロトコルをws:（ウェブソケット）にする事。

ブローカー側（仮想サーバーやラズパイなど）

[ etc/mosquitto/mosquitto.conf ]

listener 1883 # デフォルトのポート

listener 9001 # ソケット通信用ポート

protocol websockets

allow\_anonymous true # パスワード認証しない場合はこれが必要

[ terminal@MQTTブローカー ]

mosquitto\_sub -h localhost -t test/topic # サブスクライブ開始

[ index.html @htmlサーバー]

<!DOCTYPE html>

<html>

<head>

<meta charset="UTF-8">

<title>MQTT.js Test</title>

<script src="https://unpkg.com/mqtt/dist/mqtt.min.js"></script>

</head>

<body>

<script>

var client = mqtt.connect('ws://192.168.1.1:9001'); // ソケット通信用のポートを選択する

client.subscribe("test/topic");

client.on('message', function (topic, message) { // message is Buffer

console.log(message.toString());

});

function OnButtonClick() {

console.log('onClick');

client.publish('test/topic', 'message from html!');

}

</script>

<input type="button" value="Publish" onclick="OnButtonClick()"/>

</body>

</html>

##### 詳細

connect

書式：mqtt.connect([url], options)

publish

書式：mqtt.publish(topic, message, [options], [callback])

var client = mqtt.connect('ws://192.168.1.1:9001');

client.publish('test/topic', 'messege from html!', { retain: true })

subscribe

書式：mqtt.subscribe(topic, [options], [callback])

書式：mqtt.subscribe(topicのarray, [options], [callback])

client.subscribe(['test1/#', 'test2/subtopic/+']); // 複数のtopicをsubscribeする場合

client.on('message', function (topic, message) {

if(topic == 'mustbe/full/topic'){

console.log(`full topic:${message}`);

} else{

console.log(message); // 補足：messege.toString() でも可能

}

}

#### クラウドサービス

##### Beebotte

5万メッセージまでは無料（2023年1月現在）

https://beebotte.com/

### HTTP API

APIキャッシュ

### GraphQL

概要

RESTの欠点に対して作られた。（長所ばかりでもないらしい）

文法は全く異なるがSQLのような感覚に近く、新しいデータ接続言語といった感じ。

RESTに比べて多くの決まりごとが定められている。

参考クエリ

リクエストする項目について書かれたjson？を投げる

curl http://127.0.0.1:5000/ --data query='query { users{ id, name}}'

※まだ良くわかってないので間違っているかも

公式：<https://graphql.org/>

#### 概要

型のミューテーションが煩雑な気がする → FastAPIであれば問題ない？

参考：RESTとGraphQLの対比：<https://tech.jxpress.net/entry/graphql-vs-rest>

### WebSocket API

概要

断続的なデータ転送を行う

### gRPC

#### 概要

HTMLを使うというよりは、Web経由で別のマシン上でメソッドを実行するイメージ。

C++, python, Ruby, Objective-C, PHP, C#に対応

Java, Goなども対応。

メソッド名を公開し、即座に実行される？為、セキュリティ対策が難しい。

<https://www.grpc.io/docs/guides/>

### Webhook

#### 概要

Web上でのユーザーの行動をトリガーに、あらかじめ登録してあるURLのPOSTメソッドを実行する。

言い換えると、自由に定義できるWeb経由のコールバック関数提供のしくみ（らしい）

・フォーマットには、通常JSONが利用される。

・リクエストは、HTTP POSTリクエストとして行われる。

参考：https://qiita.com/soarflat/items/ed970f6dc59b2ab76169

#### 具体例

　Githubのリポジトリでpushされた際にDiscordで通知メッセージの表示ができる？

フォーマット

Payload URL

Content type

Secret

参考：<https://www.contentstack.com/docs/developers/set-up-webhooks/webhook-data-format/>

#### 対応サービス

IFTTT

### その他

RPC（Remote Procedure Call）

プログラム演算処理において、遠隔で実行結果を返す仕組みの総称。

SOAP 古くに使われていたが、かなり複雑なものだったらしい。医療分野では現役？

OpenAPI ？

XML-RPC

データ形式にxmlを採用したRPC

JSON-RPCも存在する

Atom

拡張子：.Atom

・XMLベースのフォーマット

・フィード形式でデータを提供

・ウェブ上のコンテンツを編集する

Atom Syndication Format

Atom Publishing Protocol

通称Atom APIまたはAtomPPとも呼ばれることがある。

例

<?xml version="1.0" encoding="UTF-8"?>

<feed xmlns="http://www.w3.org/2005/Atom">

<title>最新ニュース</title>

<link href="https://example.com/news/latest" rel="self"/>

<updated>2022-01-01T10:00:00Z</updated>

<entry>

<title>新しい機能のリリース</title>

<link href="https://example.com/news/1234"/>

<published>2022-01-01T10:00:00Z</published>

<content type="html">

<![CDATA[

<p>新しい機能がリリースされました。詳細は <a href="https://example.com/news/1234">こちら</a></p>

]]>

</content>

</entry>

</feed>

補足：<https://qiita.com/saka2jp/items/a027f2f18f926d975683>

## その他通信系プロトコル

### AMQP

Advanced Message Queuing Protocol

UDPまたはTCPの5672番。MQTTに似ている。あまり人気がない？

### DLNA；Digital Living Network Alliance

通信プロトコルでは無く「取り決め」

### DNS

アプリケーション層

リゾルバがDNSサーバに行うドメイン名からのIPアドレスの名前解決にはUDP（ポート番号53）を使用。

DNSサーバとDNSサーバとがゾーン情報のファイル転送を行う場合 TCP(ポート番号53）を使用。

### MTP；Media Transfer Protocol

メディア転送プロトコル

python:pyMTP

### NFS；Network File System

主にUNIXで利用される分散ファイルシステムおよびそのプロトコル

### iSCSI；Internet Small Computer System Interface

ストレージで使われているSCSIプロトコルを、TCP/IP上で使用する規格。

データ転送にTCP/IPを使う。ストレージエリアネットワーク(SAN)の基盤であるファイバーチャネルと違い汎用なイーサネット（またはTCP/IPが使用可能なネットワーク）があればよい。

### PTP

写真転送プロトコル

### RTP；Real-time Transport Protocol

### RTSP；Real-Time Streaming Protocol

メトリック

ネットワーク内のルーター（ゲート）数を示す数値の総称。

代表的なものにRIPのホップ数がある。

### WOL；Wake-on-LAN

#### 概要

（厳密にはプロトコルではない）

LAN経由でコンピュータの電源を投入する技術。

マザーボード、ネットワークカード、BIOS、オペレーティングシステムなどが Wake On LAN に対応している必要がある。

#### 条件

・マザーボードがWOLに対応していること

・BIOS/UEFIでWOLの有効化を行う。ACPI、APMなど名称はメーカーそれぞれ。

・同じくBIOS/UEFIで電源OFF時にもネットワークアダプタに電源が供給される設定にする

具体的に：Deep Sleep、ErP といった機能をOff

補足

ものによってはLANドライバーの更新が必要らしい。

#### マジックパケット

FF:FF:FF:FF:FF:FFに続けて起動したい装置のMACアドレスを16回繰り返したデータパターンがペイロードのどこかに含まれているようなパケットの事。

通常ブロードキャストとして送信される為、ルーター超えが出来ない。（同一セグメント内の装置のみ対象）

関連：リブーター

### バトンパス式通信方法

環状に接続された子機に、送信権である「N-バトン」を接続順に渡していき、送信を行いたい子機が１つずつ通信を行う方式。

あまり一般的ではないようだが、三菱のPLCや無線通信などで使われている場合がある。

# シリアル

## 共通情報

### UART

本来の意味はシリアル信号をパラレル信号に変換したり、その逆方向の変換を行うための集積回路（チップ）を意味する。誤用ではあるが、「調歩同期方式によるシリアル信号全般」，特にRS-232Cを指す事がある。

#### RS-232C

##### 概要

25pin，9pinの物があるが、実際に使用するのはRxD，TxD，GNDの３ピンのみの場合がほとんど。

これも誤用だが、「シリアル通信」は狭義にはRS-232C通信を意味する。

（実際にはRS-485やI2Cなどもシリアル通信）

##### 仕様

・電圧：±3V の不感帯。±7VをHighとみなす。

・通信距離は15m 以下

補足：特殊なノイズ対策ケーブルで150mの通信が可能らしい

<https://www.fukunishi.com/products/article148/>

・伝送速度は20kbit/sec 以下

ピン

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| No. | 信号 | 方向 | 説明 | 日本語 |
| 1 | DCD | in | Date Carrier Detect | キャリア検出 |
| 2 | RxD | in | Receive Data | 受信データ |
| 3 | TxD | out | Transmit Data | 送信データ |
| 4 | DTR | out | Data Terminal Ready | データ端末レディ |
| 5 | GND | - | Signal Ground | 信号用接地または共通帰線 |
| 6 | DSR | in | Data Set Ready | データセットレディ |
| 7 | RTS | out | Request To Send | 送信要求 |
| 8 | CTS | in | Clear To Send | 送信許可 |
| 9 | RI | in | Ring Indicator | 被呼表示　またはバスパワー対応品の場合は電源出力 |

※25ピンはほとんど用いられない。

通信設定

|  |  |  |
| --- | --- | --- |
| 設定項目 | 取りうる値 | 概要 |
| ボーレート | 1200～115200 | 232Cでは大体9600； 2のn乗 |
| ストップビット | 1,1.5,2.0 | スタートビット長は、1ビットに固定 |
| パリティビット | even,odd,none | 簡易的な通信確認に使われるビットの設定。 |
| データビット数 | 7or8 | 5~8が設定可能らしい |

https://www.contec.com/jp/support/basic-knowledge/daq-control/serial-communicatin/

<https://www.japansensor.co.jp/faq/940/index.html>

.NET

System.IO.Ports

ボーレート int

ストップビット Enum；System.IO.Ports.StopBits

パリティビット Enum；System.IO.Ports.Parity

データビット数 int

ポート string

python

pyserialを使う

import serial

ser = serial.Serial('/dev/ttyAMA0', 19200, timeout=None)

#### RS-485(RS-422)

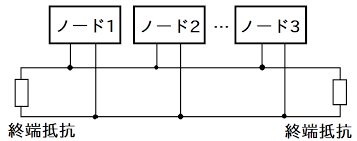
##### 概要

・電圧：+3.3VをHighとみなす。

・10m以内であれば10Mbps。　90kbps以下で良ければ**1200mまで**延長可能。

・１対ごとにツイストされたシールド線を用いる。（イーサネットケーブルが適当）

##### 終端抵抗

通信を向上させるため、最初と最後のノードの並列で終端抵抗を入れる。（下図参照）これにより過剰気味の電流を消耗する事ができ、余分な返り信号がなくなる。実際には、特に短距離や１対１通信ではつけなくても上手く行く事が多い。具体的に抵抗は100～120Ω 1/4w が良い。ケーブルによって300Ωあたりを選択する場合もある。大体のレシーバには選択する為のDIPスイッチなんかが付いている。

##### 422と485の違い

RS485はRS422を改良したもの。

|  |  |  |
| --- | --- | --- |
|  | RS485 | RS422 |
| 信号線 | 半二重  （但し4線式485も有り，互換性がある） | 全二重（SDA,SDB,RDA,RDB） |
| 通信台数 | マルチドロップ（n対32）に対応。  ※実際には64台以上の通信例も存在する。 | 基本的には1対1（常に送信状態）  ※ブロードキャスト的になら複数可 |
|  |  |  |

##### フェライトコア

ノイズフィルターとしての用途が有名で，特に基盤や電気回路を変更することなく使用できる為安易に使用できる。

中に通す導線とフェライトの輪っかでコイルを構成する為，コイルによるノイズ除去と同じ効果が得られる。

他にはインダクタのコアや変圧器としても使われている。

### USB

#### 概要

データの通信はRS422/485のような差動方式で，USB2.0までは半二重方式な為信号線は２本，USB3.0では従来のD+/D-信号線を残したまま，送受信のペア（全二重）＋GNDの計5本が加えられた。

（補足）D+/D- が残ってるのは2.0⇔3.0の互換性の為と思われる。

|  |  |  |  |
| --- | --- | --- | --- |
|  | USB 1.1 | USB 2.0 | USB 3.0 |
| ホストコントローラー | OHCI/UHCI | EHCI | xHCI |
| 転送速度 | 1.5Mbps/12Mbps | 480Mbps | 5Gbps |
| 信号線 | 平衡/半二重 | | 平衡/全二重 |
| トランザクション | ホスト制御、ポーリング、ブロードキャスト | | ホスト制御、非同期転送、ユニキャスト |
| 信号線 | 2本 | | 6本 |
| 供給電力(最大) | 500mA | | 900mA |

|  |  |  |
| --- | --- | --- |
| ピン | 信号名 |  |
| 1 | VBUS | +5VDC |
| 2 | D － | データマイナス　DMと表現される事もある |
| 3 | D ＋ | データプラス　DPと表現される事もある |
| 4 | NC | miniA/B にしかないピン。 |
| 5 | GND | A/B ではピン番号が4番 |

#### TypeC

##### 概要

##### USB PD

PDはPower Deliveryの略。最大100W（20V・5A）の電力供給を行うことが可能。

|  |  |  |
| --- | --- | --- |
| 区分 | 電力量 | 備考 |
| 標準 | 500mA、900mA、1.5A、3.0A（全て5V） |  |
| Power Delivery規格 | 3A@5V、3A@9V、3A@15V、3A@20V、5A@20V | 通称USB PD |

## 工業向きシリアル通信

### Modbus

主に産業機器で使われる。物理層はRS-485またはEthernet。RS-232Cが使われる事もあるらしい。

マイコンのI2CやSPIのように、局番号、レジスタ番号を指定する方式

伝送フォーマットは局番号、データ、チェックコードで構成

マスタ／スレーブ方式で常にマスタが通信開始をトリガする。１対１通信も可能。

### CC Link

#### 概要

CC-Linkは1996年に三菱電機によって開発されたフィールドバス。グローバルスタンダードになっている唯一の日本発オープンネットワーク。

マスタとリモートデバイス（≒スレーブ）に分かれる通信方式。

マイコンのI2CやSPIなどのようにメモリアドレスを指定して読み書きする方式。

485などと同じように、末端に終端抵抗が必要。110Ω±5%

|  |  |
| --- | --- |
| 通信方式 | ブロードキャストボーリング方式 |
| 通信速度 | 10M／5M／2.5M／625K／156Kbps |
| 通信ケーブル | シールド付き３線ツイストペアケーブル |

最大伝達距離

Ver.1.10 対応 CC-Link 専用ケーブル（特性インピーダンス110Ωタイプ）

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 通信速度 | 156kbps | 625kbps | 2.5Mbps | 5Mbps | 10Mbps |
| 局間ケーブル長 | 20cm以上 | 20cm以上 | 20cm以上 | 20cm以上 | 20cm以上 |
| 最大伝送距離 | 1200m | 900m | 400m | 160m | 100m |

種類：

|  |  |
| --- | --- |
|  |  |
| CC-Link | 10Mbps |
| CC-Link/LT | ？？？ |
| CC-Link IE | CC-Link IE ControlとCC-Link IE Fieldがある？ |
| SLMP | 通信プロトコル。物理層ではCC-LinkIE(Ethernet) を使う |

信号線

|  |  |  |
| --- | --- | --- |
|  | 概要 | 備考 |
| DA | ※485などと異なり、DA同士を接続する。  マスター側送信、スレーブ側受信？ | 信号線は通常青色。 |
| DB | マスター側受信、スレーブ側送信？ | 信号線は通常白色。 |
| DG | 232CのGND的な役割 | 信号線は通常黄色。 |
| SLD |  |  |
| FG | そのままそれぞれの筐体のグラウンドに落とす。 |  |

グラウンドが２もある？？

（補足）

EtherCAT，PROFINET など、ethernetベースのFAネットワークプロトコルは他にも存在する。

ノイズなどの影響で，オフィスで使われるようなEthernetはそのままでは工業用途では使えない。

#### 三菱

FX3U-16CCL-M FX3U用マスタ局。

FX3U-64CCL FX3U用リモートデバイス（≒スレーブ）

どうやらメモリアドレスは，どのような機器であれ大まかに用途が決まっているらしい 2021-05-18

|  |  |  |
| --- | --- | --- |
| BFM番号 | 項目 | 備考 |
| 0x000～0x01F | 基本パラメータ | ・リモート機との通信が確立していなくても読み書き可能 |
| 0x020～0x0DF | データリンク設定 |  |
| 0x0E0～0x0FF | リモート入力 | シーケンサのXにあたる。  X400 などに割り当てる事が多い？ |
| 0x160～0x17F | リモート出力 | シーケンサのYにあたる。  Y400 などに割り当てる事が多い？ |
| 0x1E0～0x21F | リモートレジスタ | ワード単位  W100などのWデバイスなるものが存在する？  0x2E0～0x2E3　1局目入力　　0x2E4～0x2E7　　2局目入力…  0x1E0～0x1E3　1局目出力　　0x1E4～0x1E7　　2局目出力… |
|  |  |  |

2021-05-18 詳細不明。どうやらこうらしい、といった程度の情報なので注意。

#### 用語（CC Link）

拡張サイクリック伝送

復列 異常局が正常になったときに，データリンクを再開する処理。

局番 個々のローカル局のIDを示すアドレス番号。マスタの局番は必ず0で0～64

占有局数 ローカルの機器一台に対し局番を何個与えるか。１局 = 4word(=8byte; 128bit)

インテリジェントデバイス局 マスタ局と１：ｎのサイクリック伝送及びトランジェンと伝送が可能な局

リモートネットモード

|  |  |
| --- | --- |
| リモートネットVer.1 | サイクリック点数の拡張が必要ない場合や既存システムの置換えの場合に選択。 |
| リモートネットVer.2 | サイクリック点数の拡張が必要な新規システムを構築する場合に選択。 |
| リモートネット追加 | 既存システムにVer.2対応子局を追加して、サイクリック点数を拡張する場合に選択。 |
| リモートＩ／Ｏネット | マスタ局とリモートＩ／Ｏ局のみのシステム構成の場合に選択。 |

CCLinkのバージョン

|  |  |  |
| --- | --- | --- |
| バージョン | 概要 | 備考 |
| Ver.1.10 | 局間ケーブル長さが20cm以上。 |  |
| Ver.2.00 | １局あたりのデータ量を最大8倍に |  |

#### 参考資料

<https://plckouza.com/st3/st3_15.html>

## マイコン系シリアル通信

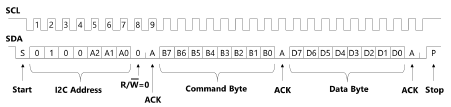
### I2C；Inter-Integrated Circuit

#### 概要

主にマイコン間通信で使われる。RS-232Cなどに比べると高速だが，通信距離は比較的短い。

２本線通信でGNDは無し。複数デバイス間でも2本の信号線しか使わない為，ピン数の少ないマイコンで使用すると良い。

具体的には以下のように通信される



#### 仕様

・信号線

SDA 　Serial Data データそのもの。

SCL 　Serial Clock クロック信号。

・7bitモードと10bitモードがある。前者が一般的。

・クロック周波数は100kHzか400kHzのみ。

・通信パケット内に7bitのSlave Addressを含む。アドレスは1～247で，0はブロードキャスト。

・基本的には通信先アドレスを指定した，瞬間的な1対1通信を行う。

プルアップ

SDAとSCLは**それぞれ**プルアップ抵抗が必要。大体1~10kΩ程度。

具体的には，各チップで使っている寄生コンデンサの静電容量によって選ぶらしい。オシロなどで波形を見ながら選ぶ？

アドレス

I2C通信では2種類のアドレスを指定する為少し分かり難い。

1つ目のアドレスは通信相手（チップ；Slave Address）を選択する為の物。上にも記したが1～247の範囲で選択。

2つ目のアドレスは相手チップのレジスタメモリアドレス。通信相手の説明書をよく読んで指示に従う。

I2C General Call

マスタ側から全スレーブに一斉に送信を行う通信。ethernetのブロードキャストの様なもの。

世の中にはGeneral Callに応答しない仕様を持つI2C通信のチップも存在するらしい。

#### Pythonで行う場合

##### python-smbus

python標準モジュールで、単純なセンサーやデバイスとのやりとりに使用。

インストール（最初から入っている事が多い）

sudo apt install python3-smbus

write\_i2c\_block\_data

read\_i2c\_block\_data

##### smbus2

サードパーティ製のモジュールで、高度なカスタムの通信プロトコルや高速な通信が必要な場合に使用。

インストール

pip install smbus2

使用方法

from smbus2 import SMBus

##### python-periphery

#### terminalで行う場合

##### I2c-tools

インストール（最初から入っている場合が多い）

sudo apt install -y i2c-tools

使用方法

# bus1番に接続されたデバイスアドレスの一覧を表示する

sudo i2cdetect -y 0x01

# bus1番のアドレス20番を読み取る

sudo i2cget -y 0x01 0x20

-y これをつけないと毎回警告文が出力される。ほぼ必須オプション、

### SPI；Serial Peripheral Interface

主にマイコン間通信で使われる。１つのマスターと複数のスレーブで通信する方式。

電圧（B接；LOWしたチップを選択）で通信相手を選び，瞬間的な1対1通信を行う。

I2Cに比べると通信速度が早い。

4W(4線式)と3W(3線式)がある。

SCLK、MISO、MOSI、SSの4種類の信号線からなり，3線式の場合はMOSIをMOSI/MISOとして使う。

SCK Serial Clock（SCLKの場合もある)

MOSI Master Out Slave In 　マスターからスレーブ。（スレーブ専用ではSDI；Serial Data In)

MISO Master In Slave Out 　スレーブからマスター。（スレーブ専用ではSD0；Serial Data Out)

SS Slave Select　通信相手（スレーブ）を選択する。（スレーブ専用ではCS；Chip Select）

-CSB；CS Bar　不論理で通信相手を選択する。（ ；バーは不論理を示す）SSBはあまり見られない（らしい）。

（補足）通信相手が１つであれば相手のCSBをGNDと短絡すれば良い事もあるらしい。

（補足）RS-232C, RS-485に慣れた人にとっては，例えばMOSI同士を接続するのは紛らわしいかもしれない。

転送モード

|  |  |  |  |
| --- | --- | --- | --- |
| mode | クロック極性  (CPOL) | クロック位相  (CPHA) | 備考 |
| 0(0b00) | 0:正論理 | 0:クロックが0→1の時 | ほとんどのデバイスはmode0で通信できる（らしい） |
| 1(0b01) | 0:正論理 | 1:クロックが1→0の時 |  |
| 2(0b10) | 1:負論理 | 0:クロックが0→1の時 |  |
| 3(0b11) | 1:負論理 | 1:クロックが1→0の時 |  |

（補足）CPOL：Clock Polarity CPHA ：Clock Phase

具体的にはCSがLOWかつCLKがHIGH（※）になる時にデータを1bit転送する。　*※転送モードによる*

その他の通信仕様はSPI特有の仕様と言うより，通信相手の仕様に則る事になる。（この辺は232Cなどでも同様）

LSBFirst

各バイトのLSBからMSBにかけてbitを送信していく事。設定でMSBFirstに変える事もできる。

### SWD

マイコン開発でデバッグを行う時に使用する，ARM社が定めたシリアル通信の規格。

SWCLK 　クロック

SWDIO 　通信データそのもの

なおパソコンを用いて開発する際はUSB ⇔ SWDの変換（デバッガ）が必要。

また，SWDのプロトコルにはバージョン1とバージョン2があり，2はマルチドロップ（複数台）接続が可能。

### JTAG

SWDとほぼ同じ用途で用いられるシリアル通信規格。

もともとはプリント基板検査の為の規格だったが，後にデバッグに用いられるようになった。

TCK（クロック）

TDI（データ入力）

TDO（データ出力）

TMS（状態制御）

Compact JTAGと言う2本線規格も存在するが，SWDとは対照的でほとんど使われていない。

### I2S

音声や信号などに使われる？

### CAN

主に自動車向きのシリアル通信。

### DICOM

Digital Imaging and Communications in Medicine ダイコム

CTやMRI、CRの医用画像フォーマット、医用画像機器間で用いる通信プロトコル

## シリアル関連の用語

RS Recommended Standard の略らしい。

TxD(Transmit Data) 送信に用いる線やピン。相手のRD;Recieve Data と対応する。

RxD(Receive Data) 受信に用いる線やピン。相手のSD;Send Dataと対応する。

DTR(Data Terminal Ready) 通信が可能である事を示す。相手のDR; Data Set Readyと対応する。

RTS (Requet To Send) 送信側が送信を開始する際に送る信号。相手のCS; Clear to Sendと対応する。

CTS(Clear to Send) 送信可。相手のRS; Request to Sendと対応する。

bytesize

通信１ブロックに何ビット含むか。7bitにすると0x80以降の数字を送れなくなる。

（127までしか使わないアスキー文字しか送れない，と言う事）

startbit 1bit固定。待機状態では信号線にはHigh電圧が印加。Lowになった時点を通信開始とみなす。

stopbit データ終端を表すビットの長さ。1, 1.5, 2から選択。

parity パリティ。簡易的な通信確認。通常odd。（参考）CRC

# 通信ケーブル

## USB TypeC

### Thunderbolt

USB TypeCを使ったデータ伝送規格。MacBook Proに採用されたのが始まり。



もともと別のコネクタ形状だったが、2015年の「Thunderbolt 3」でUSB Type-Cコネクタを採用。

対応品は右図のような印がある。（最も、2022年現在、ほとんどのケーブルが対応済）

### TypeCの電流量

|  |  |  |
| --- | --- | --- |
| 区分 | 電力量 | 備考 |
| 標準 | 500mA、900mA、1.5A、3.0A（全て5V） |  |
| Power Delivery規格 | 3A@5V、3A@9V、3A@15V、3A@20V、5A@20V | 通称USB PD |

# 接続方式

## デイジーチェーン

PC-機器1-機器2　といった形で数珠繋ぎにする方式。

（右図参照）

# 負荷分散

## SLB；Server Load Blancer

レイヤ３方式 IP

レイヤ４方式 IP+ポート

レイヤ７方式 クッキー

# Web系

## ドメイン

### 概要

1つの名前空間をTLD；Top Level Domain ごとに分割し、それぞれのTLDの特色に合った、利用者の多様なニーズに応えるサービスを運用する、などといった柔軟な対応が可能

補足

www.example.comの場合、TLDは.com、2LDはexample、3LDはwwwとなる。

### FQDN；Fully Qualified Domain Name

別名：完全修飾ドメイン名，絶対ドメイン名

www.abc.com と言った形で，ホスト名とドメイン名をつなげた、ルートまですべてのノードを示す文字列。

正引き（resolve）：FQDN からIPアドレスを検索する事　(opp) 逆引き(reverse)

関連用語

ホスト名 FQDN前半部

ドメイン名 ドメインのグループを識別するための名前。www.example.comの場合example.comの部分

パス名（pathname）ドメイン名より後の部分。（クエリ文字列を除く）

サブドメイン sub.example.comはexample.comのサブドメインにあたる

### Zone APEX

別名：ゾーンエイペックス、ネイキッドドメイン（naked domain）、エイペックスドメイン（Apex Domain）

サブドメイン(wwwなど)を含まないドメイン名 例：example.jp

特徴

・CNAMEが設定できない

（RFC1912 からZone APEXとCNAMEレコードはプロトコルエラーになる）

## MIME

Web電文のドキュメントの種類の情報。ファイルの拡張子のようなもの。

type/subtype で構成され、text/htmlやvideo/mp4などと表記される。

## S/MIME

MIMEを拡張した、電子メールの暗号化とデジタル署名に関する標準規格

ディジタル署名の付加などセキュリティ対策を取り入れたもの

メール本文の暗号化に利用した共通鍵を更に暗号化した上でメールに添付する。

（ハイブリッド暗号）

# サーバー

サーバーの役割：

## Webサーバー（Html）

### MVC

Mode データベースのテーブルなどに該当。ロジックなどを処理。

View ユーザーが目にする出力の部分

Control クライアントからのリクエストに対応する。

### Html

#### httpメソッド

get：リソースの取得。urlとして値を受け取る。

post：新規データを追加。本文として値を送信。

put：リソースを更新。リソースの新規作成。

delete：リソースの削除

patch ：一部のみ更新

head：リソースのメタデータを取得。

options：リソースがサポートしているメソッドの取得

trace：プロキシ動作の確認

connect：プロキシ動作のトンネル接続への変更

#### form

action

method

formのメソッドではgetとpostしか使えない。

#### CSS

#### PHP

#### java.script

## DNSサーバ

### 概要

リソースレコード

ドメイン名に対応するIPv6アドレスを登録するレコード

|  |  |  |
| --- | --- | --- |
| 名前 | Full Name | 定義するもの |
| SOA | Start of Authority | ゾーンに関する情報 |
| NS | Name Server | ゾーンの情報を管理するDNSサーバー名 |
| MX | Mail Exchange | ゾーンで管理するドメイン宛のメールサーバー |
| A、AAAA | Address | FQDNに対応するIPアドレス  AレコードはIPv4  AAAAレコードはIPv6用 |
| CNAME | Canonical Name | 名前（FQDN）の別名 |
| PTR | Pointer | IPアドレスに対するFQDN  IPアドレスはin-addr.arpa 形式で表記 |

※ゾーン：１つのDNSサーバーが管理しているドメインの範囲

※FQDN；Fully Qualified Domain Name 別名：完全修飾ドメイン名，絶対ドメイン名

関連コマンド

dig

### 用語（DNS）

ゾーン

権威DNSサーバーが管理しているドメインの範囲

ゾーン転送

１つのDNSサーバーが管理している情報全体（ゾーン）を一括して別のDNSサーバに転送すること。

通常は、セカンダリDNSサーバがプライマリDNSサーバの持つデータの複製を作るために利用する。

権威DNS

他のサーバーに問い合わせることなく応答を返すことができるDSNサーバーのこと。

キャッシュサーバー

別名：DNSサーバー フルサービスリゾルバ 。

ゾーン情報を管理せず、IPアドレスやドメイン名の問い合わせ結果を一定期間保存しておき、問い合わせに対応するサーバー。

関連：DNSキャッシュ

リゾルバ

広義には、DSNからIPアドレスに変換するソフトウェア全般。

DNSレコード

別名：ゾーンファイル

ドメイン名とIPアドレスの対応表など、ドメインに関する登録情報

参考資料：<https://jprs.jp/tech/material/iw2012-lunch-L3-01.pdf>

## メールサーバ

## ファイルサーバ

SMB

FTP

## データベースサーバ

### 用語

クライアント/サーバモデル

サービスを提供するノードが固定（サーバー）

Peer To Peer；P2P

サーバーにあたるノードが常に入れ替わりながら、常に１対１で通信を行う。

|  |  |  |
| --- | --- | --- |
|  | クライアント・サーバ・モデル | P2P・モデル |
| モデル | 中央集権型 | 自律分散型 |
| システム拡張性 | 利用者規模に合わせたシステム増強 | 柔軟なスケーラビリティ |
| 情報の所在 | サーバへの情報の集中 | 情報はネットワーク上に遍在 |
| 耐障害性 | サーバへのリスクの集中 | 障害時に他ノードによる代替 |

## プロキシサーバー

あるネットワーク内にインターネット接続させないクライアントPCがある場合、代理としてインターネット接続を行うサーバの事。

例えば、企業内LANにプロキシサーバーを置いてパケットを監視する事で、不正な通信がないか監視することができる。

クライアントPCはまずプロキシサーバと接続を確立させる。次にプロキシサーバとWebサーバが接続を確立させる。

## アプリケーションサーバー

Webサーバー（html）からのリクエストに応じて処理を行う。

通常WebサーバーとDBサーバーの間に立ってミドルウェア的な処理を行う

|  |  |  |
| --- | --- | --- |
| 名前 | バックエンド言語 | 概要／特徴 |
|  |  |  |
| Glass Fish | Java | ライセンス料が100万円以上だとか |
| Gunicorn | Python | WSGI準拠。シンプル。 |
| mod\_wsgi | Python | WSGI準拠。Django標準 |
| NGINX Unit | Python, PHP, Go, Perl, Ruby |  |
| Puma | Ruby |  |
| Tomcat | Java | 20年以上使われ続けている |
| Unicorn | Ruby |  |
| uWSGI | Ruby, PHP | WSGI準拠。高速。 |
| waitress | Python | WSGI準拠はWindows環境ではこれしか選択肢が無い |

補足

・PHPの場合はアプリケーションサーバーを使う必要がない（らしい）

FastCGIとは、Webサーバ上でユーザプログラムを動作させるためのインタフェース仕様の一つ

## Webフレームワーク

先に結論：

JavaScript，(Rails)，Django，PHPあたりを抑えておけば大丈夫そう。

### Python系

以下の２つがメジャー

Django

がっつりフル装備のアプリを作りたい (例: 認証や DB を使う)

ウェブフレームワーク、MVC、正規表現の知識がすでにある

Flask

ウェブフレームワークの勉強を始めた初心者。シンプルなウェブアプリをちゃちゃっと作りたい

必要最低限の機能しかなく，それ以外の事は追加する必用がある

近年はあまり人気がない？

### その他

Ruby on Rails

文字通りRubyで動く。Webフレームワークでは一番人気。

ただし，近年人気が下降気味。

Vue.js

JavaScripのフレームワーク

CakePHP

Spring

React.js

Facebookが開発

Angular.js

Googleが開発

## レンタルサーバー

サポート機能は結構重要なので，無駄に質問などしてみる。 20.10.14

### VPS

近年のレンタルサーバーの内，最も機能の少ないもの。

特徴

・中度のデータベース利用。（Word Press対応の為，大体MySQLとMariaDB）

・PHP，Ruby, などのスクリプトに対応

Pythonは対応していないところもあるかも？

逆に言うと，それ以外の言語の使用は基本的には難しい

・

用途

・ブログやWebページ（Html）

## Webツール

### データ分析

Elastic Stack

データ分析，可視化ツール

# Web系ツール（コマンド）

## IP系

arp

-d # 履歴を削除

netstat

-r # ルーティングテーブルを表示

nbtstat

ip route #　経路情報を出力する [WIN：route print]

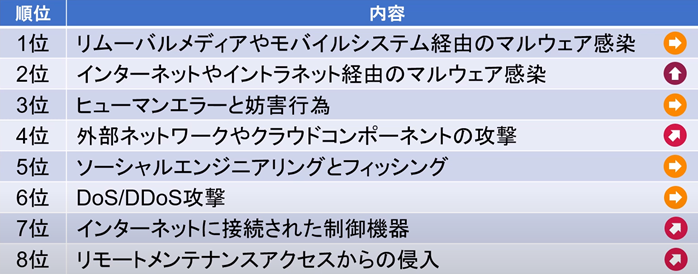
サブコマンド

show #

# セキュリティ

## 概要

2022年ドイツBSI発表による産業用システム(ICS)の10大脅威　より抜粋



## 保護系

### IDS；Intrusion Detection System

侵入検知システム。不正なアクセスを検出し、通知する。

IDPSは誤検知するのでチューニングする必要がある。NIDS（ネットワーク全般監視）とIDS（ホスト監視型）がある。

ヘッダだけでなく検知内容を自由に設定できる。

|  |  |  |
| --- | --- | --- |
|  | メリット | デメリット |
| NIDS | ネットワーク全体を監視できる  ホストに負荷がかからない | 暗号化されたパケットやファイルの改ざんなど、検知できない攻撃がある。 |
| HIDS | ファイルの改ざんなど、きめ細かい監視ができる  暗号化ファイルも復号して検査できる | 全てのホストに導入する必要がある  ホストに負荷がかかる |

### DMZ

DeMilitarized Zone

ルーターを介してWANとLANの間に設ける独立セグメント。

公開サーバーをそのセグメントに置くことで、本命のセグメント保護につながる。

但し、万能ではない事を頭に置いておく。Webサーバーなどの公開サーバーとデータベースなどのサーバーは基本的に分離するのがセキュリティの基本。

### ファイアウォール(FW)

#### パケットフィルタリング型

通信をパケット単位で解析し、決められたルールに基づいて通過の許否を判断。

送り元と宛先のIPやport番号などを参考にフィルタリングする。

#### ゲートウェイ型

内部コンピューターを覆うプロキシを用いて通信を監視する。

http, smtp などアプリごとに判断する。

#### WAF(Web Application Firewall)

Webアプリに特化したファイアウォール。SQLインジェクションやクロスサイトスクリプティングなどWebに特化。

ブラックリスト方式とホワイトリスト方式がある。ブラックリストは通常WAFのベンダが提供。

## 攻撃系

### MACアドレスの偽装

無線LANの正規通信中のパケットの傍受

→正規につながっている機器を切断させるのは簡単。再接続させて認証パケットを傍受する。

暗号解読に使われる「鍵」はパスワードとSSIDから生成される。

解析ツールを用いれば英小文字8文字をランダムに試しても解読に1日かからない。

### マルウェアの種類

#### ワーム

・ネットワークやリムーバルメディアを媒介として自己増殖する。

#### トロイの木馬

・特定の条件になるまで活動せずに待機する。

### 攻撃の種類

#### Buffer Overflow

別名：バッファオーバフロー攻撃

概要

プログラムが確保するメモリ領域に，領域の大きさを超える長さの文字列を入力してあふれさせ，ダウンさせる攻撃

防ぎ方：

入力の全体の長さが制限を超えているときは受け付けない。

#### CSRF；Cross Site Request Forgeries

別名：クロスサイトリクエストフォージェリ

正規のリクエストとは別のサイトからリクエストを送ることによってデータの不正更新を行うもの。

具体的には、

１　あるサイトに通常の手順でログインを行い、セッションIDを発行する

２　ユーザーはしばらくの間ログイン状態を保持

３　ログイン状態を保持したまま、攻撃者が用意したURLを踏む

４　攻撃者が利用者のアカウントを使ってサイトのpostフォームを用いてデータを改ざん

対策

ユーザー

ログイン状態を必要以上に長く保持しない

サービス提供側

postフォームを利用する際はトークンを利用。（その際hiddenにしておく）

重要な処理を行うまえにはパスワードの再入力を求める

WAFで防ぐことができる（らしい）

#### Directory Traversal

別名：ディレクトリトラバーサル攻撃

概要

「../etc/passwd」のように上層に遡ってパス指定して、不正にファイルにアクセスする。

防ぎ方：

・入力に上位ディレクトリを指定する文字(../)を含むときは受け付けない。

・ファイル名のハッシュ化。

#### DNSキャッシュポイズニング

クライアントPCが参照するDNSサーバに偽のキャッシュ情報を登録させ、その汚染されたDNSサーバの利用者を、攻撃者の設定した別のサイトに誘導させようとする攻撃手法

#### DNSリフレクション攻撃

反射型DoS攻撃の一種

攻撃対象のサービスを妨害するために，攻撃者がDNSサーバーを踏み台に利用して再帰的な問合せを大量に行う

#### Footprinting

別名：フットプリンティング

サイバー攻撃に先立ち、攻撃者が標的の弱点を探る為の下調べ。

DNSサーバーのソフトウェアのバージョン情報を入手して，DNSサーバーのセキュリティホールを特定する、など。

#### Host Header Injection

通信対象の「Host」ヘッダを改ざん

別名：hostヘッダインジェクション

Hostヘッダーを基にアプリ用のURIを発行するWebサイトがあった時、HTTPリクエストのHostヘッダーに改ざんを行う事で攻撃者が該当のURIを取得するもの。

防ぎ方：

Web アプリ側で、Host および X-Forwarded-Host を信頼しない。

SERVER\_NAMEを使用する。

#### HTTP Header Injection

別名：HTTPヘッダインジェクション

クッキーのsessionIDの値を書き換えるように細工したURLをユーザーに踏ませるユーザーのセッションを乗っ取る手法。

#### SEOポイズニング

検索エンジンの検索結果上位に悪意のあるサイトが表示されるように細工する攻撃。

#### SQLインジェクション攻撃

具体例

「SELECT \* FROM mytable WHERE uname='$uname'」のunameをクエリ文字列で指定できるとして、

「$uname」に、「”‘;DELETE FROM mytable –“」と入力された場合、

「SELECT \* FROM mytable WHERE uname='';DELETE FROM mytable--'」となってしまう。

防ぎ方：

入力中の文字が，データベースへの問合せや操作において，特別な意味をもつ文字として解釈されないようにする。

#### クロスサイトスクリプティング(XSS)

概要

特定のサイトに別の攻撃用サイトへの誘導URLを埋め込み、閲覧者が攻撃用サイトにアクセスした際にマルウェアを仕込んだり、クッキーを取得されたりするもの。

防ぎ方：

入力にHTMLタグが含まれていたら，HTMLタグとして解釈されない他の文字列に置き換える。

#### サイドチャネル攻撃

暗号化装置の動作を電磁波から解析することによって解読する。

#### ゼロデイ攻撃

別名：Zero-Day

セキュリティホールが発見された日から対処方法が確立される日まで（ゼロデイ）の間に攻撃をする事。

#### ソーシャルエンジニアリング

身分詐称、人のミスや心理的な隙に付け込む、などでパスワードなどの秘密情報を不正に取得する方法の総称。

#### パスワードリスト攻撃

あるサイトの攻撃の結果得られたIDとパスワードのリストを用いて別のサイトへの不正ログインを試みる攻撃。

#### ブルートフォース攻撃

別名：総当り攻撃，Brute Force Attack

与えられた1組の平文と暗号文に対し，総当たりで鍵を割り出す。

またはidを固定し、パスワードを総当たりで割り出す

#### リバースブルートフォース攻撃

パスワードを１つ固定し，idを総当たりで割り出す。

パスワード桁数が固定されていたり、adminなど安易なパスワードを設定していると被害にあいやすい。

#### レインボー攻撃

別名：レインボーテーブル攻撃

予め、想定され得るパスワードとハッシュ値の対のテーブルを用意しておく。

攻撃の際は、入手した解読前のデータ（ハッシュ値）とテーブル内の値を照らし合わせる事でパスワードを効率的に解析する。

#### 辞書攻撃

Dictionary attack

一般的な単語などのリストを作成し、それを元にパスワードのクラックを試行する。

#### 水飲み場攻撃

別名：Watering Hole Attack

標的の組織が頻繁に利用するWebサイトを改ざんして攻撃コード、マルウェアを埋め込んでおく。

### 攻撃対策の種類

#### URLフィルタリング

利用者に有害なWebサイトを見せないようにする仕組み。

#### キーワードフィルタリング

利用者に特定のキーワードを含むWebサイトを見せないようにする仕組み。

## 暗号系

### デジタル証明書

別名：ディジタル証明書

Web上での偽造やなりすましを防ぎ正当性を保証するもの。

具体的には秘密鍵を持っている事を証明する事が目的。

証明作業の手順

・データのハッシュ値を求める。

・送信者は自身の秘密鍵を用いてハッシュ値を暗号化。

・受診者は受信データのハッシュ値を求める。

・その後、送信者が送ってきた暗号化されたハッシュ値を送信者の公開鍵で復号し、先程のハッシュ値と比較する。

ITU-T X.509

デジタル証明書のデータ形式を定めた標準規格。

公開鍵の配送に必要な証明書データの記録形式や署名方式、無効な証明書を周知するリストのデータ形式などを定めている。

認証レベル

DV；Domain Validation

OV；Organization Validation

運営者の実在性審査が行われる。1990年台に登場。

EV；Extended Validation

Webサーバー運営組織に対する厳格な審査と確認が行われた上で発行される。

OVと異なり、認証の仕方が規格化されている。

補足：暗号の種類とはまた別の話。

DV のリスク

DV 証明書ではWeb サイトの「本物」の運営者であることまでが関連付けられていない。

まだ誰も取得（登録）していないドメインであれば、類似の紛らわしいドメインも登録できてしまう。

補足：認証局と認証レベルはブラウザ上の南京錠をクリックして証明書の詳細を確認すればわかる

### SNI；Server Name Indication

暗号化通信で使用するサーバー証明書をパブリックIPアドレスではなくドメイン名によって判断する技術。

### ハッシュ化

データを復元する必要が無い場合の暗号化。

#### SHA-1

#### SHA-2

#### SHA-256

Secure Hash Algorithm 256bit

#### MD5

### FIPS 140-2

別名：Federal Information Processing Standard：米国連邦情報処理規格

FIPS 140-2は、暗号化ハードウェアの有効性を検証するためのベンチマーク

### SSH

#### 概要

Port：TCP22（ただし実際にはセキュリティの観念で別のポート番号を使用する事が多い）

リモートシェルに特化した暗号技術。

なお、OpenSSHはOpenSSLのライブラリを使用してはいるが、暗号化の部分だけ。

#### configファイル

Linux：[~/.ssh/config] windows：[C:\ Users\tuser\_name\.ssh\config ]

Host test

HostName srv\_url\_or\_ip # リモートの場所。URLやIPアドレスなど

User srv\_user\_name # リモートOSでのユーザー名

IdentityFile ~/.ssh/test/gehogeho\_a.key # 任意：パスワード認証も可能

Port 2222 # 任意：def.=22

IdentitiesOnly yes # 任意：yes or no[def.]

上記のように設定した場合、以下のようにして接続できる。

[ terminal ]

Host test # これで接続できるようになる

#### sshd\_configファイル

パス：~/.ssh/sshd\_config

SSHデーモンの設定ファイル

|  |  |  |
| --- | --- | --- |
| 項目 | 値 | 説明 |
| AllowUsers | 略 | ログインユーザー等の制限。user@ip\_addr となる。 |
| Match User |  |  |
| Port | 数値 | 接続を受け付けるポート番号。 |
| PermitRootLogin | yes/no | rootユーザーでのログインの可否。  without-password, forced-commands-onlyの値も有効 |
| PasswordAuthentication | yes/no | パスワードでのログインの可否。 |
| PubkeyAuthentication | yes/no | 公開鍵でのログインの可否。 |
| Protocol | 1/2 | 2が推奨らしい |
| ChallengeResponseAuthentication | yes/no | パスワード認証が否なのでこちらも否 |

#### SCPとSFTP

概要

・共にSSHプロトコルを使用する。

・共にUNIX 系OS 向きの暗号通信

・共に秘密鍵を用いた通信が可能

|  |  |  |
| --- | --- | --- |
|  | SFTP | SCP |
| 意味 | SSH File Transfer Protocol | Secure Copy |
| 送受信方法 | ログイン後にデータ転送。 | １コマンドにつき１回認証、１ファイル転送 |
| 特徴 | ・ファイル転送の再開が可能 | ・比較的シンプルに作られている  ・フォルダの転送はできない  ・比較的高速 |
|  |  |  |

#### トラブルシューティング

##### Bad owner or permissions on C:\Users\xxxx\.ssh\config

現象

Windows環境で.ssh/configを使ったsshコマンドを実行すると表題の権限エラーが発生する

対策

以下のいずれかで対応

・.ssh/configの場所を変える（例 d: /ssh/configなど）

・「グループ名またはユーザー名」を編集

操作：configを右クリック > プロパティ > 「セキュリティ」タブ > 編集

SYSTEM、Administrators、自分自身　だけにする

情報元：<https://kurage-worker.com/2022/windows-ssh-permission-error>

詳細

OpenSSHの最新版ではセキュリティが強化されているため、設定ファイルが他のユーザから見えないようにしていないと接続ができない。

#### X11 Forwarding

##### チュートリアル

サーバー側

・ssh実行時に-Xオプションを付ける

・または設定ファイルを利用する

[ /etc/ssh/sshd\_config ]

X11Forwarding yes

クライアント側

[ /etc/ssh\_config ]

Forward X11 yes

・X11（TC6000）の設定を行う

補足

DISPLAY環境変数はログインの際に自動で設定されるが、念のため以下のコマンドで確認できる。

printenv DISPLAY

> 0 # :0または :0.0 となっている事が多い。これはローカルホストを示す。

xauthがない場合はインストール

sudo apt-get install xauth

設定ファイルを変更

[ /etc/ssh/sshd\_config ]

X11DisplayOffset 10

X11Forwarding yes

X11UseLocalhost no

XauthLocation /usr/bin/xauth

##### クライアントがWindows/Macの場合

X11サーバーが入っていない為、要インストール

Windows

VcXsrv <https://sourceforge.net/projects/vcxsrv/>

（参考）<https://dev.classmethod.jp/articles/wsl-x-window/>

Xming <http://www.straightrunning.com/XmingNotes/>

など

さらにサーバー側で

terminal@サーバー

export DISPLAY=192.168.1.1:0.0 # IPアドレス：画面番号で指定

ファイアウォールの6000番ポートを開ける？ 2022-11-30再確認

補足：Windows Subsystem for Linux を有効にしてもダメだった。

### 鍵暗号

#### 概要

リモート機へのアクセスの際，パスワードの代わりに1対の「秘密鍵と公開鍵」を利用する方式。

一方で暗号化された通信はもう一方の鍵で復元される（以下のどちらか）

１　公開鍵で暗号化を行い、秘密鍵で復元。（クライアント → サーバー）

２　秘密鍵で暗号化を行い、公開鍵で復元。（サーバー → クライアント）

・ログイン先（通常サーバー）に公開鍵を置き，クライアントは秘密鍵でアクセスを行う。

・クライアント，サーバーどちら側で鍵を作成しても構わない。（公開鍵の方に作成したユーザーの署名が入る）

SSHの認証に先立ち、SSLによるホスト認証（サーバー認証）を行う（らしい）

クライアントが接続要求→SSL公開鍵がクライアントへ→クライアント側でデータを暗号化

→暗号化されたデータをサーバーへ → サーバーの秘密鍵で複合→そのハッシュ値をクライアントへ。

補足

SSH認証の時はクライアント側が秘密鍵を持つが、SSL認証の時はサーバーで秘密鍵を保管するのでややこしい。（どちらも流出するとまずい方が秘密鍵を持つ、と考えると良いかも）

オプション

-v デバッグ情報を表示

-X ForwardX11を行う

#### チュートリアル（編集中）

Terminal（またはcmdなど）

ssh-keygen #ホームディレクトリ下の .ssh/ に鍵ペアが作成される。

#### 詳細

・秘密鍵は「誰にも渡さない」が原則。

・公開鍵の置き場は，ホームディレクトリ直下 .sshフォルダ（~/.ssh）が一般的。

・鍵作成の際にパスフレーズの設定ができる。このパスフレーズはサーバーへのログインパスワードとは無関係で、ログインの際はこのパスフレーズ＋鍵で認証となる。空入力でパスフレーズなし。パスフレーズは11文字以上が良い。

（参考）パスフレーズのクラックに必要な時間（単純計算の場合）

8文字 20日 ； 9文字 3.3年； 10文字 197年

鍵の特徴のまとめ

|  |  |  |  |
| --- | --- | --- | --- |
| 項目 | 公開鍵 | 秘密鍵 | 備考 |
| ssh-keygen直後の名前 | id\_<暗号タイプ>.pub | id\_<暗号タイプ> | （詳細）id\_rsa など |
| 一般的なリネーム | authorized\_keys | 特に決まっていない | 複数の公開鍵を利用する場合は追記。 |
| ファイル権限 | 600 | 600 | この権限と**完全に一致しないと使用できない**。 |

#### SSHクライアント

OpenSSH を使う時は，この例のように -i オプションを使用する。

ssh user@192.168.1.100 -i ~/ssh/keys/id\_rsa # -i オプションではフルパスを指定する。

面倒ならばssh\_configで設定しても良い。項目名はIdentityFile

IdentityFile ~/ssh/keys/id\_rsa # 同じくフルパスを指定する。デリミタは空白。

備考

・公開鍵から秘密鍵の割り出しは事実上不可能。

・万一，秘密鍵が盗まれてしまう事もある。また，盗まれても通常気が付きにくいのが問題。

・必ずしも鍵認証の方が安全，とは限らない。十分な長さと複雑なパスワードの管理さえできれば、同等以上のセキュリィティ確保は可能。パスワード認証＋ホスト制限，というのも有効な手。

特に，秘密鍵を置くホスト自体が信頼できない場合などは秘密鍵と言うシステム自体を使うべきでない。

#### 公開鍵認証

通信先（通信相手）側の公開鍵を使用して暗号化。受信した側は自身の秘密鍵を使って復号する。

どちらかというとSSL/TLS通信などで使われる。

#### ファイル形式

##### 概要

|  |  |  |
| --- | --- | --- |
| 形式 | ソフト | 概要 |
| .pem | OpenSSL | X.509証明書、RSA秘密鍵、DSA秘密鍵など多種多様な情報を保存する |
| .key | OpenSSH | OpenSSH独自のフォーマット |
| .jks | Java | Javaのアプリで使われる鍵と証明書を保存する |
| BKS | Java - BouncyCastle |  |
| .der / .crt / .cer | ？ | 主にX.509証明書。バイナリ形式の証明書 |
| .p12 / .pfx | WindowsOS | WindowsOSで使われる |
| .ppk | PuTTY | WindowsユーザーかつPuTTYを使う場合のみ、と思って問題ない  WinSCPを使う時にも用いる？ |

補足

PuTTY Windowsでよく使われるSSHクライアント

##### pem

別名：Privacy Enhanced Mail

#### 共通鍵

少し古い？方式で共通鍵という方式もあるらしい

|  |  |  |
| --- | --- | --- |
|  | 共通鍵暗号方式 | 公開鍵暗号方式 |
| 暗号化と複合の鍵 | 同じ（ペアで作る） | 異なる |
| 鍵の配送 | 必要（手間がかかる） | 必要なし |
| 処理速度 | 早い | 遅い |
| n人で利用する際に必要となる鍵の数 | n(n-1)/2 | 2n |
| 管理すべき鍵 | 両方 | 秘密鍵 |
| アルゴリズム | AES, 3DES, Blowfish | RSA |

### SSL

#### 概要

Port：TCP443(https)、TCP465 (SMTP over SSL/TLS)など、各プロトコルにより異なる

基本的にはWebサーバー（https）で使用されるが、SMTPやPOPでも使用可能。

但し、SSL、特にSSL2.0までには深刻な脆弱性があったため、SSL3.0の新しいバージョンとしてTLS1.0が生まれた。現在TLSが主流だが、SSLの知名度が高かった為にSSL/TLSと表現される事が多い。

サーバー側に認証局から発行された証明書の入手が必要。（ただし独自証明書の場合もあり）

有名な認証局と言えばverisignとかglobalsign？

ブラウザにはあらかじめ信頼出来るルート認証局の証明書が登録されている。

暗号化するのはOSI参照モデルの上位３層、つまりアプリケーション層、プレゼンテーション層、セッション層。

#### SSHとSSLの違い

・SSHでは通信相手が100人いる場合は100人分の公開鍵が必要。SSLは1つで良い。

・SSLは任意のプロトコルをトンネリング。SSHはリモートシェルに特化。

・SSLは事前に認証局発行の証明書が必要

・SSHはパスワード認証も可能だが、SSLはどうしても証明書が必要。

#### 証明書

概要

Webサーバーの身元が正当性を証明する為の電子データ。また、SSL暗号化にも必要となる。

証明書を持つことで、様々なウェブブラウザでサイトが正しく表示される信頼性を獲得できます。

作成手順

・SSL証明書の発行元（認証局）を探す

・認証局で要求文字列（CSR；Certificate Signing Request）を入力

・CSRを元に秘密鍵と公開鍵のペアを生成

openssl genrsa -out my\_private\_key.pem 2048

openssl rsa -pubout -in my\_private\_key.pem -out my\_pubkey.pem

・秘密鍵から証明書(.crtか.pem)を作成

openssl req -new -key 秘密鍵ファイル名.key -out CSRファイル名.csr

> 秘密鍵と証明書がダウンロードできるはずなのでサーバーで保管。

> 公開鍵は認証局で保管される

一つのパブリックIPアドレスを持つサーバーに対して一つのドメイン名が紐づく。

例えば「example.com」と「example.co.jp」の2つのドメイン名を持つWebサーバーの場合、ドメイン名ごとに異なるパブリックIPアドレスを取得してサーバー証明書を導入する必要がある。

#### .pfxファイル

秘密鍵をパスワードで保護しつつ証明書と一緒に格納する仕組み

OpenSSLを使って作成する事が可能

補足：拡張子は「.p12」もありうる

### 暗号の種類

#### 概要

|  |  |  |  |
| --- | --- | --- | --- |
|  | キー長 | 特徴 | 備考 |
| RSA  Rivest Shamir Adleman | 2048, 3072, 4096bit | 公開暗号方式 | ・長い歴史を持つ暗号方式  ・最も普及している |
| Ed25519 | 256bit固定 | 処理が早い | ・安全面と性能面で最良 |
| ECDSA  Elliptic Curve Digital Signature Algorithm  楕円曲線DSA | 256, 384, 512bit | 短いキーサイズで高いセキュリティ |  |
| AES  Advanced Ecryption Standard | 128, 192, 256bit | 対象暗号方式 |  |
| ElGamal |  | 公開鍵方式  OpenPGP, GnuPGなど |  |
| Kcipher-2 |  | 対象暗号方式  ハードウェア実装に適す | ・日本で開発  ・産業界で利用を推奨 |
|  |  |  |  |

#### RSA；Rivest Shamir Adleman

4096bit など、bit数の大きい鍵が作成できる。最も普及している。

なお、RSAの2048bitが推奨となる、標準基準となっている。

ssh-keygen -t rsa -b 4096 # 使用例

#### Ed25519 256bitの固定長さ

ECDSAの一形式。256bitの固定長で高いセキュリティを提供。処理速度が速い。

OpenSSHのバージョン6.5で対応。ssh -V で確認。

安全面と性能面で最良。

ssh-keygen -t ed25519 # 使用例

#### ECDSA

Elliptic Curve Digital Signature Algorithm；楕円曲線DSA

楕円曲線（ y2 = x3 + ax + b）で作成される暗号

キー長は256、384、512bitがある

ssh-keygen -t ecdsa -b 384 # 384bit

ssh-keygen -t ecdsa -b 521 -C "wnoguchi-mbp" # 512bit ※521は打ち間違いではない

～鍵暗号　暗号の種類　つづき

#### DSA

RFC 4251に規定されているのでSSH2で必ず使える。但しユーザー認証のみ対応、通信の暗号化は非対応。

ssh-keygenによっては鍵の長さが1024bitと今になっては心もとない長さしか使えない。

RSAの特許が失効するまではSSH2ではDSA公開鍵暗号を使っていた。

#### RSA1

RSAの1024bit以下の事らしい。古いRSA。

SSH v1しか使えない環境で使うが、それ以外では絶対使用しないようにする。

#### DES

1970年代にアメリカで開発。56bitのキー。

1990年代後半からブルートフォースで破る事が現実的になりAESに置き換わった。

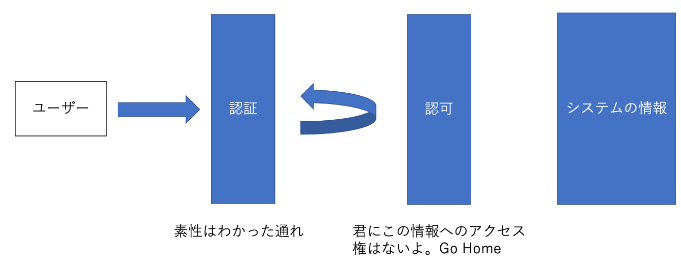
現在では使われていない。

#### 補足

OpenSSHではRSA,DSA,ECDSA,EdDSA(ed25519)が使用可能

## 認証系

### 概要（認証）



HTTPのレスポンスでの違い

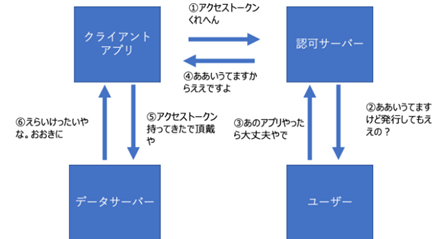
401 Unauthorized → 認証失敗 「あなた誰？」

403 Forbidden → 認可の不足 「誰かは分かったが許可できない」

### 二段階認証と二要素認証

|  |  |  |
| --- | --- | --- |
|  | 二要素認証 | 二段階認証 |
| 簡潔に言うと | ２つの要素を用いた認証 | 1度のログインに対し認証を2回行う |
| 具体例 | 要素  ・ID、パスワード  ・秘密の質問  ・ワンタイムパスワード  ・生体情報：顔、静脈 | 単に回数の事  二要素かつ二段階、というのも有り得る |

### OAuth

認可に関する仕組みなので、SAML、OpenIDなどの認証とは関係が無い。

アプリとユーザーが分かれており、アプリにはユーザに関する情報（ユーザーIDやパスワード）を渡していない点が重要

### SAML

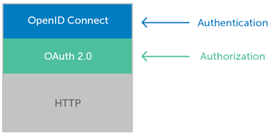
Security Assertion Markup Language

認証の為の通信プロトコルで、主にシングルサインオンやID連携で利用されるマークアップ言語。

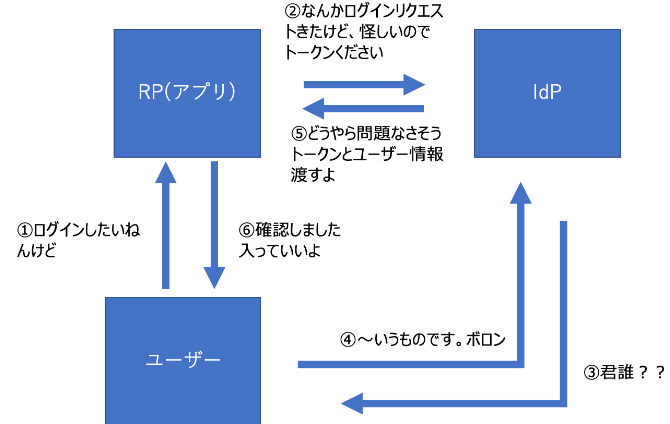
メッセージの送受信にはHTTPやSOAPが使われる。

Identityプロバイダー(IdP)　IDを管理しているところ

### OpenID Connect

OAuth 2.0をベースとする、シンプルなアイデンティティ連携プロトコル

IDPとアプリが直接やりとりしている点がSAMLと違う



### 資料

分かりやすい：<https://zenn.dev/kisukeyas/articles/910898b0bb6ad3>

## 資料（セキュリティ）

### 脆弱性レポート

#### JVN；Japan Vulnerability Notes

CVSSv2

|  |  |
| --- | --- |
| 深刻度 | スコア |
| 緊急 | 9.0~10.0 |
| 重要 | 7.0~8.9 |
| 警告 | 4.0~6.9 |
| 注意 | 0.1~3.9 |
| なし | 0 |

攻撃対象となるホストやシステムにおいての「脆弱性による深刻度」を評価

AV：ローカル（L） / 隣接（A） / ネットワーク（N） でどこから攻撃可能か。

AC：高（H） / 中（M） / 低（L） で攻撃の複雑さ

などで評価

CVSSv3

コンポーネント単位で評価する手法を取り込んだ仕様。（仮想化などが進んできている事が理由）

CVE

共通脆弱性識別子

世界で脆弱性情報に割り振ったユニークID

### セキュリティ関連の規格

#### ISO27701

#### ISO 27017

#### ISO 27018

#### 問題

一つ以上の実際に起きた，または 潜在的なインシデント原 因のこと。

JIS Q 20000-1：2020で定義されている。

#### 既知の誤り

根本原因が特定されているか、またはサービスへの影響を低減，もしくは除去する方法がある問題。

JIS Q 20000-1：2020で定義されている。

## メモ（セキュリティ系）

リモートデスクトップはデフォルトで3389番ポートを使う。

→　攻撃対象にならないように変えておく

・無線LANのプロトコルはWPA/WPA2-PSKでAESを選択。

・SSL 通信をしていても(https)、Cookie の改変は防げない。

参考）<https://blog.tokumaru.org/2013/09/cookie-manipulation-is-possible-even-on-ssl.html>

ITサービスマネジメントの活動

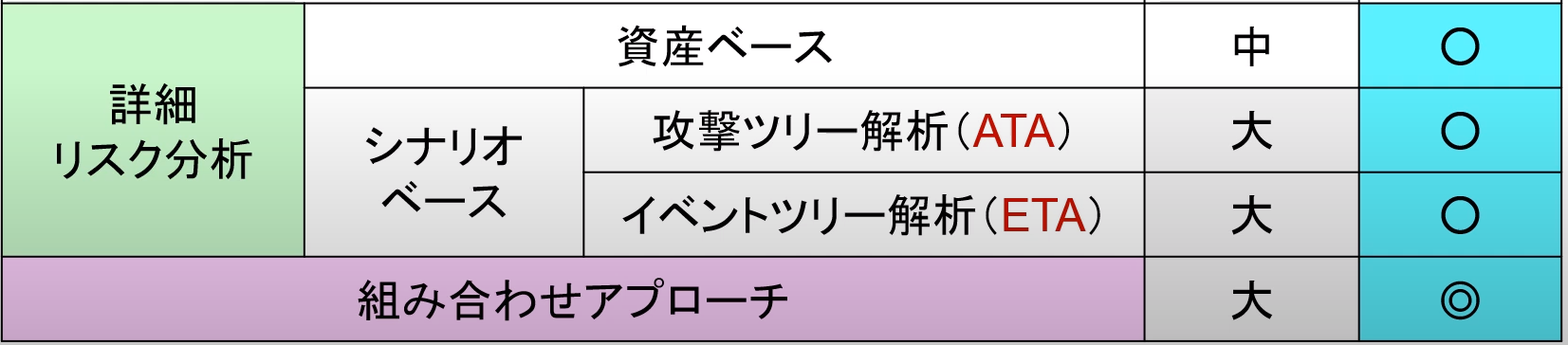
サービスレベル管理 顧客満足度が合意したサービス目標を満たしているかどうかを評価

キャパシティ管理 ディスクの空き容量がしきい値に近づいたので，対策を検討する。

### セキュリティ分析ガイド

IPAが発行している「セキュリティ分析ガイド」というpdf冊子がある。

リスク分析の手法（抜粋）



ETA（threat-oriented）アプローチ 攻撃者、攻撃ツール視点で攻撃シナリオを考える

ATA（impact-oriented）アプローチ 回避したい事業被害から攻撃シナリオを考える

この２つを相互的に使用する

#### 資産ベースのリスク分析

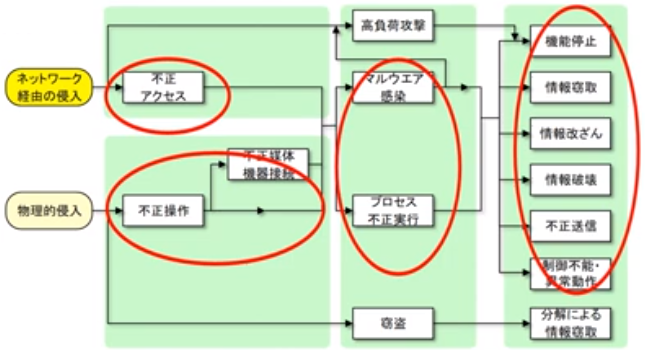
・全資産を様々な攻撃手法についてもれなく分析する

資産の重要度、脅威レベル、脆弱性レベル　から算定する

不正アクセス、経路遮断、盗聴、マルウェア感染　など

脅威は大きくネットワーク経由、物理的侵入の２種類に大別される

重要度の高い　赤丸内のリスクに限定して考えても良い



補足：見にくい部分に不正媒体（USBメモリなど）の記述あり

セキュリティ対策が資産の中で行われるべきか、外で行われるべきかを考える

　たとえば制御装置であれば

・操作盤のキーは資産の中で行われるセキュリティ対策

・工場内への入室にアクセスカードを義務付けるのは資産の外で行われるセキュリティ対策

### CTF

#### 概要

Capture The Flag

Jeopardy 形式

インターネット上で開催されているCTF では、最も定番。クイズ形式で得られる得点を競う。

最近ではダイナミックスコアリングと呼ばれる問題の解かれた数に応じて動的にスコアが変動する形式もよく見られます。この方式を用いることで、問題の作問者が意図した難易度と競技者が感じる難易度の差による配点の違和感を軽減し、結果的に適切な配点になることが多いです。

Attack & Defense形式

各チームに脆弱性のあるサービスが稼働しているサーバが与えられます。競技者は、他チームのサービスを攻撃してフラグを手に入れることができます。また、脆弱性を修正して他のチームから攻撃されないようにするといったことも重要です。Attack & Defense は主に決勝大会で行われることが多いです。

#### 体験用アプリなど

Google版：Webアプリのゲーム形式？

<https://capturetheflag.withgoogle.com/>

ksnctf：

<https://ksnctf.sweetduet.info/>

Cisco版（使い方が分からない）

<https://devnetdayjp.ciscoctf.io/>

参考：

<https://qiita.com/Lion_rion/items/16d3434619987275bbec>

### SIEM；Security Information and Event Management

Kakusy

各種ログを総合的に分析して、脅威を発見する仕組み。

クラウド、オンプレ版とあるが、近年クラウド版が優位。

（個人的には、この手のアプリはクラウド版の方が信頼できる）

Microsoft Sentinel

EDR

SWG

CASB

## 用語（セキュリティ）

GDPR

General Data Protection Regulation：一般データ保護規則

欧州委員会が欧州連合 内の全ての個人のためにデータ保護を強化し統合することを意図している規則

IDS；Intrusion Detection System

不正侵入検知システム

IPS；Intrusion Prevention System）

補足：SnortはIDSもIPSもできる。

SIEM；Security Information and Event Management

OS、データベース、アプリケーション、ネットワーク機器など多様なソフトウェアや機器が出力する大量のログデータを分析し、異常などを知らせる仕組み

ハニーポット

攻撃させるために意図的に脆弱性を持たせたネットワークやシステムの事。

攻撃者のログを取る事を主の目的とする。

ボットネット

攻撃者の支配下におかれた状態のコンピュータ群の事。

サンドボックス

システムに影響を及ぼさないよう隔離されている領域。

コンペア法

検体のハッシュ値を計算し、オンラインデータベースに登録された既知のマルウェアのハッシュ値と比較して照合する手法。

静的解析

検体をネットワーク上の通信データから抽出し、さらに逆コンパイルして取得したコードから検体の機能を調べる。

動的解析

検体をサンドボックス上で実行し、その動作や外部との通信を観測する。

ピクチャパスワード

画面上で設定した動作でアカウントにサインインする。

3Dセキュア

オンライン決済時に本人のみが知り得る情報を入力させる仕組み。

# 仮想化

## ネットワーク構成

NAT 仮想OSはNAT（ネットワークアドレス変換）を使用する

ブリッジ 仮想OSは物理NICに直接接続

ホストオンリー 仮想OSはホストOSとしか通信できない

Vagrantは、FLOSSの仮想機械を構築するためのソフトウェア

※FLOSS（Free/Libre Open Source Software）オープンソースソフトウェアとフリーソフトウェアをまとめた言葉

## 仮想デスクトップ

VNC；Virtual Network Computing)

遠隔PCを制御する仕組み。≒リモートデスクトップ。RFBプロトコルを利用する。

VDI；Virtual Desktop Infrastructure

１つのOS上で複数の仮想化されたデスクトップ環境を構築する仕組み。

ユーザーが使用するアプリケーション、データがそのサーバーに集約される。

SBC；Server Based Computing

1つのOS上でアプリケーションの仮想化を行い、遠隔のPCにアプリケーション画面イメージを転送

遠隔利用するのはアプリケーションのみ

# 無線

## 概要

|  |  |  |
| --- | --- | --- |
|  | 用途 | 詳細 |
| 315MHz帯 |  | 小出力のため通信距離が短い |
| 400MHz帯 |  | 狭帯域のため通信速度が出ない |
| 920MHz帯 | Zigbee（アメリカ） | 2012年7月から免許不要で使用可能  回り込みに強い、低消費電力。大容量の通信には向かない。  2018年3月31日迄となっている950MHz帯の代わり？ |
| 1.2GHz帯 |  | 狭帯域のため通信速度が出ない。 |
| 2.4GHz帯 | Bluetooth、wifi、  Zigbee（日本） | 人気の周波数帯で干渉が多い。  電子レンジでも使用。 |
| 5GHz帯 | wifi(802.11a, 802.11ac) | \*aのwifi専用の周波数帯 |

（参考）<https://www.m-system.co.jp/mstoday/plan/mame/2014-2015/1510/index.html>

## 無線通信全般

## Bluetooth

### バージョン

ver.5：BLE＋方向探知

5.0 通信範囲が4倍、通信容量は8倍に。メッシュネットワークにも対応

5.1 方向探知機能

ver.4：BLE；Bluetooth Low Energy

4.0 LE(Low Energy) 機能

4.1 インターネット接続機能

4.2 LEのデータ通信速度を2.5倍化

ver.3：クラシックと呼ばれる

4.0以降との互換性は無い。

3.0 HS（High Speed）最大24MHz

#### SMARTとSMART READY

SMART：クラシックのみ SMART READY：クラシック、LE両方対応

### クラス

|  |  |  |  |
| --- | --- | --- | --- |
| クラスの種類 | 最大出力 | 通信最大距離 |  |
| クラス1 | 100mW | 約100m | BLEでも100m行けるらしい。 |
| クラス2 | 2.5mW | 約10m |  |
| クラス3 | 1mW | 約1m |  |

BLE

伝送速度 1Mbps

### 2.4GHzのWifiとの干渉

Bluetoothは仕様上，周波数帯を変更する事ができない。

Wifiを5GHz帯にするか，2.4GHz帯でもch（チャンネル）を変えると良い。

更に，5GHz帯の方が使えるch数も19chと多い。

蛇足情報：

FH 方式の Bluetooth は瞬間的に占有するスペクトラムは 1MHz だけで、これが約 80MHz 幅の範囲に割り当てられた 79 チャネルの間をランダムに飛びまわる。

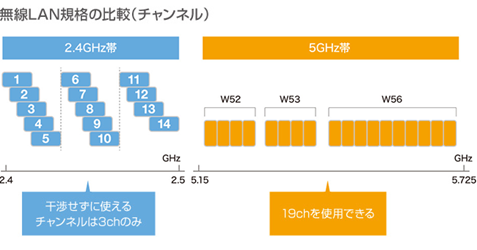
（参考）<http://www.silex.jp/blog/wireless/2013/02/bluetooth5.html>

そもそも，通信機器の再起動，受信機と送信機の距離を近づける，などで解消する場合もある。

また，電子レンジ，蛍光灯，コードレス電話機，等から遠ざけるのも大切。

## wifi

### チャンネルについて

他機器との干渉を抑える為に，

Wifiではchの選択ができる。

b/g/nの2.4GHz帯と

a/ac/n の5GHz帯とでは

選べるch数や範囲が異なる。

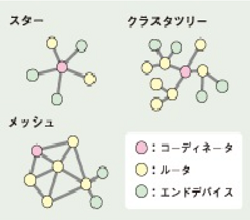
### 通信規格

|  |  |  |  |
| --- | --- | --- | --- |
| 無線LAN規格 | 周波数帯 | 最大速度 | 備考 |
| IEEE802.11a | 5GHz | 54Mbps | 最も良く使われる。 |
| IEEE802.11g | 2.4GHz | 54Mbps | Bluetoothとほとんど同じ。最も良く使われる。 |
| IEEE802.11ad | 60GHz | 6.7Gbps |  |
| IEEE802.11n | 2.4/5GHz | 300Mbps | 通称：Wi-Fi 4 |
| IEEE802.11ac | 5GHz | 6.9Gbps | 通称：Wi-Fi 5 |
| IEEE802.11b | 2.4GHz | 11Mbps |  |
| IEEE802.11ax | 2.4GHz帯/5GHz帯 | 9.6Gbps | 通称：Wi-Fi 6 |
| IEEE802.11ah |  |  | 遠距離用通信 |
| IEEE802.15.4 | 2.4GHz | 250kbps | ZigBee準拠の軽量通信  <https://mono-wireless.com/jp/tech/802_15_4.html> |

（参考）<https://www.iodata.jp/product/network/info/base/kikaku.htm>

### wifi6

## ZigBee

ネットワークのトポロジー（右図参照）もスター、クラスタツリーと、メッシュの3種に対応。

通信可能距離は約30m

データ転送速度は20Kbps-250kbps。使用する無線周波数帯によって異なり、2.4GHzでは250Kbps、902-928MHzでは40Kbps、868-870MHzでは20Kbpsとなる。

但し、日本国内では電波法の影響により2.4GHz帯のみ使用可能。

## RFID

### 概要

Radio Frequency IDentification。無線通信による個体識別の技術の一種。≒ICタグと考えて良い。

RFIDの技術を用いて作られたICタグは，厳密にはRFIDタグと呼ばれる。

対象がICタグの場合はデバイス側が発する電波からの電磁誘導で発生する電力で通信やデータの書き換えを行う為，電池などが不要。距離は10mまで可能。13.56MHzの電波を用い、標準では100～400kbps程度で通信。

現時点で思いつく使用用途：

・社員の勤怠管理

・ライン製造の個体情報管理

### 種類

#### UHF

通信距離が長い。

#### NFC

近距離無線通信　Near Field Communication。

RFIDの技術体系の一つで，距離が10cm程度の近距離のもの。

近年比較的安価になってきている（らしい）；１枚50~100円程度。

スマホなどにはリーダー／ライタが標準で搭載されている。

電話をかける、SMS送信、メール送信、WebブラウザでURLを開く、特定のアプリを起動　などができる。

#### まとめ

|  |  |  |  |
| --- | --- | --- | --- |
|  | 使用例 | メーカー | 備考 |
| MIFARE  （TypeA） | Taspo | NXP  （元フィリップスの１部門） |  |
| （TypeB） | 免許書、マイナンバーカード、キャッシュカード | モトローラ |  |
| FeliCa | Suica、Edy、電子マネー | ソニー | 電子マネー系と交通系ICはまた種類が異なるらしい |
|  |  |  |  |

MIFAREの中でTypeA/B がある？

具体的な種類

TypeA

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | 会社 | 容量 | 転送速度 |  |
| Lite-S | NXP | 1kb | 106kb/s | 低価格 |
| NTag | NXP | 888byte | 424kb/s |  |
|  |  |  |  |  |
| iCODE SLIX | NXP | 4kb |  | 最大1メートルの読み取り距離 |

FeliCa

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | 会社 | 容量 | 転送速度 |  |
| Lite-S | NXP | 26byte |  | 非常に小型。NXP Lite-Sとは別物 |
|  |  |  |  |  |

### 詳細

#### 機能

カードエミュレーション機能

モバイル端末による、Suicaなどのカードのエミュレーション。

端末間通信機能（P2P）

モバイル⇔PCなど２つのデバイス間でデータ転送を行う。

補足

NFCのライタは，SonyのFeliCaのおかげか結構存在する

SONY-RC-S380S（パソコン向き；USB）

<https://www.switch-science.com/catalog/1060/>

PN7120 NFC Controller SBC Kit；NXP Semiconductors NV（マイコン向き）

<https://akizukidenshi.com/catalog/g/gM-10012/>

反対に，RFIDのライタはほとんど見られない。リーダーならある程度存在する。

Parallax Inc. RFIDリーダー

<https://akizukidenshi.com/catalog/g/gM-06826/>

### ICカード

### 使い方

#### NFC

・スマホアプリを使用

iOS/Android 対応

NFCタグマティック

iOS

NFC Tools

Android

MacroDroid マクロ設定ツール？

・カードリーダー製造元のアプリを使用

Sanwa Supply：NFCかんたん設定アプリ

#### 使用するためのモジュールなど

winscard.h （winscard.dll）WindowsでICカードを使うためのドライバ（標準インストール）

PCSC .NETでwinscard.hを使うためのラッパー。Nugetで管理。

nfcpy pythonでNFCタグを使うためのモジュール

## LPWA

### 概要

別名：LPWAN

Low Power Wide Area Network

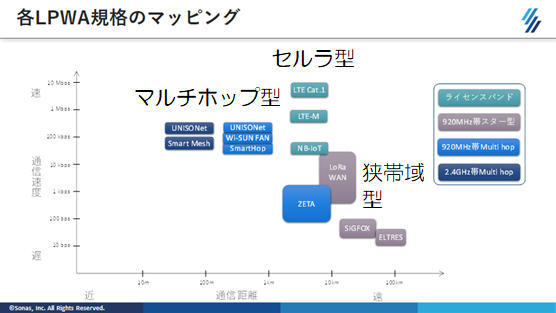
サブGHz帯（866 MHz帯・915 MHz帯・920 MHz帯）を用いる。

broadcastingによるため、コネクションの必要がない。

「アンライセンス系」と「ライセンス系」とに分かれ、ライセンス系は無線局免許が必要。

セルラーLPWA 携帯キャリアの運営するネットワークを用いたLPWA規格

通常、通信はAES-128で暗号化されている



情報元：<https://www.sonas.co.jp/media/iotdx-lab/lpwa001>

製品概要（UNISONet）

LoRaとZETAの中間をとったような無線通信。

距離はLoRaより短く、データ転送速度はZETAより早い。

送信データ量は静止画程度なら送信可能、との事。

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | LoRa | ZETA | UNISONet leap | UNISONet metro |
| １ホップでの  通信可能距離 | 10~20km | 10~20km | ~2km | ~5km |
| １通信のデータ量 | ～242byte | 8～50byte | 数Mbyte | 数Mbyte |
| 転送速度 | 73bps〜22kbps | 300~600bps | 240kbps | 20kbps |
| 基地局の値段 | 20万円程度 | 26万円程度 | 親機：5万程度 | |
| 中継局の値段 | 要調査 | 7万円程度 | 子機：4万程度 | |

### 比較

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | 速度 | 距離 | ペイロード | 特徴 | 備考 |
| RoLa | ～250 kbps | 10km | - | 物理層の規格 | 利用者が一番多く、結果として安い。 |
| RoLaWAN | ～50kbps | ~242byte |  |
| SIGFOX | ～100bps | 50 km | ~12byte | 低電力  専用クラウドがある | ポリシーとして１国１社 |
| UNISONet | ～240kbps | ~2km | ~数MByte | 送信データ量が多め | 独自マルチホップによる安定通信 |
| ～20kbps | ~5km | ~数MByte |
| Wi-Fi HaLow  (IEEE 802.11ah) | ～20Mbps | 1km |  | Wi-Fiの通信距離拡張 | 物理層はacの1/10クロックダウン  Wi-Fiでなくてもよくね？の意味でいまいち |
| Wi-SUN | ～300 kbps | ~3 km |  |  | 家庭の電力計で使われている  海外ではほとんど使われない |
| ZETA | ~2.4kbps | 10~20km | 80byte |  | 中国企業ZiFiSenseの技術 |
| ELTRES | 1分間隔・10分間隔・1時間間隔 | 100km以上 | 16byte | 高速移動通信（100km/h以上）に対応 | ソニーが開発した。  電波強度が強い。  みちびきにも使われている |

### LoRa

#### 概要

Long Range

LoRa Alliance” （世界のIoT関連480社以上が加盟）で仕様化

ライセンス不要の通信方式とライセンスが必要な通信キャリアを使った無線方式の2種類に分けられる。

通信モジュール例：

<https://www.switch-science.com/products/8671>

モジュールメーカー

|  |  |  |  |
| --- | --- | --- | --- |
|  | 所在地 | 採用会社 | 備考 |
| SEMTEC | アメリカ |  |  |
| Kiwi Technology | 台湾 | IIJ、SORACOM |  |
| EASEL | 横浜 | 東京デバイセズ |  |
|  |  |  |  |

補足：JRC、村田製作所なども作っているらしい

アンテナ例：

<https://www.switch-science.com/products/8676>

#### LoRaWAN

LoRaを採用したLPWANプロトコルで、ライセンス不要の無線方式

1送信のペイロードサイズはわずか11~242byte

補足：LoRaは変調方式（データ⇔電波への変換方式）

単三電池2本で1年間は動かし続けることが可能

IIJにLoRaWAN製品があるらしい？

<https://akizukidenshi.com/catalog/goods/search.aspx?maker=kiwi&style=T&search.x=true>

ClassA/B/C が存在する

#### デバイス

ゲートウェイ

|  |  |  |  |
| --- | --- | --- | --- |
| 製品名 | URL | 値段 |  |
| SenseCAP M2 | <https://akizukidenshi.com/catalog/g/gM-18036/> | 16280 |  |
| SenseCAP M2 | <https://www.marutsu.co.jp/pc/i/2720302/> | 14800 |  |
|  |  |  |  |
|  |  |  |  |

### UNISONet

・機器の値段が比較的安価

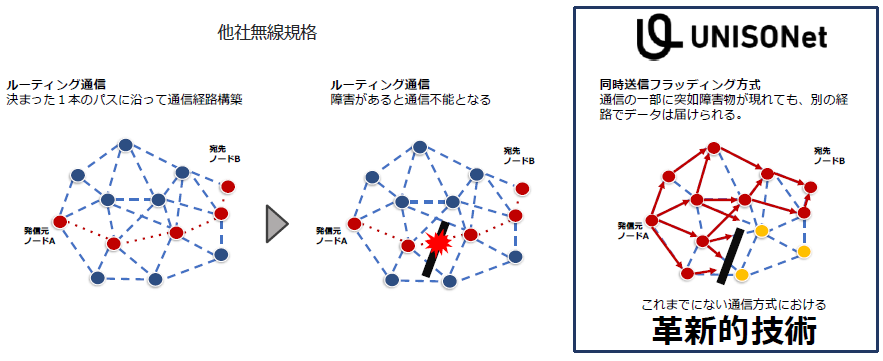
・40F の建物でも使える（5Fごとに中継器を挟む；LoRa だと3F くらいまでが限界）

・高頻度、大容量の通信が可能

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | LoRa | ZETA | UNISONet leap | UNISONet metro |
| １ホップ範囲 | 10~20km | 10~20km | ~2km | ~5km |
| １通信のデータ量 | ～242byte | 8～50byte | 数Mbyte | 数Mbyte |
| 転送速度 | 73bps〜22kbps | 300~600bps | 240kbps | 20kbps |
| 基地局の値段 | 20万円程度 | 26万円程度 | 親機：5万程度（従来機） | |
| 中継局の値段 | 要調査 | 7万円程度 | 子機：4万程度（従来機） | |
| 用途 | 一日数回 | | 複数ノード | |

・「同時送信フラッティング方式」というルーティングの一部に障害が起こっても通信できるしくみが強み。

「複数のノードが次のノードに同時に無線通信しても問題がない」というところにすごさがある。



### SigFOX

専用クラウドがあり、Webhookによるコールバックなどが可能

### Wi-SUN

Wi-SUNアライアンス（アメリカ合衆国に本拠を置く業界団体）

サブGHz帯を用いるIEEE 802.15.4-2020の技術。

日本では、家庭の電力計に使われている。

### ZETA

ZiFiSense社の独自規格

## UWB

UWB（Ultra-Wideband）

日本国内では8.5GHz～9.5GHz程度の周波数帯域を利用

・高精度な位置取得用途で使われるが、屋内に30m～40m間隔でセンサーを設置する必要がある。

・iPhoneのAirTAGはBluetoothと併用して？

（個々のiOS端末から匿名でTAG情報を拾ってiCloudで共有しているらしい）

## バーコード

### 概要

通信技術的には無線でも何でもないが、RFIDなどと比較される事が多いため、ここにまとめる。

## QRコード

rMQRコード

最大17×139セル 最小7×73セル

で従来のQRコードと同等のデータを保持できる。

誤り訂正レベル

## スターリンク

人工衛生を利用したインターネットの仕組み。

屋外（理論的に、"空を見られる"ところ）で通信可能。

## その他

### 無線に関連するルール

免許無しで使えるのは10mW以下

#### 技適

#### 微弱無線局の規定

<https://www.tele.soumu.go.jp/j/ref/material/rule/>

## 用語

PER；Packet Error Rate）

パケット誤り率。送信したパケット数に対し，エラーとなり受信できなかったパケット数の割合をパーセンテージで示した物。

# その他通信系情報

## Well-Knownポート

0 TCP/UDP (予約済み)

**20 TCP FTP (File Transfer Protocol) (データ)**

**21 TCP FTP (File Transfer Protocol) (制御)**

22 TCP/UDP ssh (secure shell)

23 TCP Telnet

**25 TCP/UDP SMTP (Simple Mail Transfer Protocol)**

42 TCP/UDP WINS (Windows Internet Name Service)

43 TCP WHOIS

53 TCP/UDP DNS (Domain Name System)

67 UDP BOOTP (Bootstrap Protocol) (サーバー)

68 UDP BOOTP (Bootstrap Protocol) (クライアント)

69 UDP TFTP (Trivial File Transfer Protocol)

**80 TCP/UDP HTTP (Hypertext Transfer Protocol)**

88 TCP/UDP Kerberos

109 TCP POP2 (Post Office Protocol version 2)

110 TCP POP3 (Post Office Protocol version 3)

113 UDP ident

119 TCP NNTP (Network News Transfer Protocol)

123 UDP NTP (Network Time Protocol)

135 TCP Microsoft RPC (Remote Procedure Call)

137 TCP/UDP NetBIOS (名前解決)

138 TCP/UDP NetBIOS (データ転送)

139 TCP/UDP NetBIOS (セッション制御)

**143 IMAP2/4 (Internet Message Access Protocol version 2/4)**

161 SNMP (Simple Network Management Protocol)

162 SNMP (Simple Network Management Protocol) (トラップ)

179 TCP BGP (Border Gateway Protocol)

194 IRC (Internet Relay Chat)

220 IMAP3 (Interactive Mail Access Protocol version 3)

389 LDAP (Lightweight Directory Access Protocol)

**443 TCP/UDP HTTPS (HTTP over SSL/TLS)**

445 TCP/UDP Microsoft Directory Service (SMB/CIFS)

465 TCP SMTPS (SMTP over SSL/TLS)

514 UDP syslog

520 UDP RIP (Routing Information Protocol)

546 TCP/UDP DHCPv6 (Dynamic Host Configuration Protocol version 6) (クライアント)

547 TCP/UDP DHCPv6 (Dynamic Host Configuration Protocol version 6) (サーバー)

554 TCP/UDP RTSP (Real-Time Streaming Protocol)

587 TCP SMTP Message Submission (メッセージ投稿)

636 TCP/UDP LDAPS (LDAP over SSL/TLS)

843 TCP Adobe Flash (ソケット通信)

860 TCP/UDP iSCSI

873 TCP rsync

888 TCP CDDB (CD DataBase)

**989 TCP/UDP FTPS (FTP over SSL/TLS) (データ)**

**990 TCP/UDP FTPS (FTP over SSL/TLS) (制御)**

992 TCP/UDP Telnets (Telnet over SSL/TLS)

**993 TCP IMAPS (IMAP over SSL/TLS)**

**995 TCP POP3S (POP3 over SSL/TLS)**

3389 TCP リモートデスクトップ

追記：19.08.12

591 FileMaker社のポート。但しHTTP Alternateとしても使われる

8000 http proxy へのアクセス。irdmi 特定の用途はないが、80番を連想するが為に使われる？

8080 http proxy へのアクセス。80ではない別のhttp用途で用いられる　HTTP Alternate

8008 HTTP Alternateの定番

その他のHTTP Alternate候補：

880, 980, 3280, 4380, 5180, 5380, 5480, 5880, 5980, 6180-6380, 6680, 6780 6980, 7180, 7380-7580, 8180, 8480-8780, 9180, 9480-9780, 10180-10780, 11080-20380

# LTE

編集中

## PIX―MT100

<https://www.pixela.co.jp/products/network/pix_mt100/>

SIMフリーを謳うわりにauの電波がバンド1（2GHz帯）しか使えない

らしい

情報元：<https://www.makkyon.com/2017/01/23/gh-udg-mcltec/>

## FSM04UJ-B02

グリーンハウス

## GH-UDG-MCLTEC

富士ソフト

# AllJoyn

# ソフトウェア

## OpenSSL

SSL／TLS対応のWebサイトを構築したり，セキュアな通信路を必要とするアプリケーションをセットアップしたりする場合，電子証明書の取得などで使われている。オープンソースで開発・提供。

オンラインマニュアル：<https://euske.github.io/openssh-jman/>

## OpenSSH

### 概要

win10 は標準で入っている。cmdかps でssh と入力すると使える。

ssh [user]@[host]

### 設定ファイル

#### 概要

|  |  |  |
| --- | --- | --- |
|  |  | 備考 |
| ssh\_config | クライアント側で，接続HOSTの設定を書くファイル | linux /home/ {user} /.ssh/ssh\_config  win　C:\Users\ {user} \.ssh\ssh\_config |
| sshd\_config | サーバー側で， |  |
| config | デフォルト値用のファイル |  |

#### ssh\_config

|  |  |  |
| --- | --- | --- |
| 項目 | 値 | 説明 |
| ConnectionAttempt | 3など | 接続試行回数 |
| Compression | yes/no | ファイル転送で圧縮するかどうか。gitなどのファイル転送にも適用される。 |
| GSSAPIAuthentication | yes/no |  |
| HashKnownHosts | yes/no |  |
| Host | elias\_name | 接続ホストの設定に別名を付ける事ができる  ssh elias\_name で接続できるようになる。 |
| HostName | host.com | リモートのホストのURLやアドレス。 |
| IdentitiesOnly | yes/no | IdentityFileでのみ認証するかどうか |
| IdentityFile | ~/.ssh/authorized\_keys など | 秘密鍵の場所をフルパスで記述する。 -i オプションをつける必要が無くなる。 |
| Port | 1234など | ポート番号の指定。-p オプションをつける必要が無くなる。 |
| SendEnv |  | LANG LC ？？？ |
| ServerAliveCountMax | 3など | 連続で失敗したら切断される接続回数を指定。 |
| ServerAliveInterval | 60など | サーバーからの応答に対するタイムアウト値。単位は sec |
| TCPKeepAlive | yes/no | 接続状態を継続したい場合：yes　継続しない場合：no |
| User | user\_name | 接続先OSでのユーザー名 |
|  |  |  |

（参考資料：最強設定）<https://qiita.com/keiya/items/dec9a1142ac701b19bd9>

#### config

Include ~/.ssh/ssh\_config # これによって config → ssh\_config の順に適用

ssh\_configとconfigで重複した項目は，**先に出た方が適用**される。

#### sshd\_config

|  |  |  |
| --- | --- | --- |
| 項目 | 値 | 説明 |
| AllowUsers | 略 | ログインユーザー等の制限。user@ip\_addr となる。 |
| Port | 数値 | 接続を受け付けるポート番号。 |
| PermitRootLogin | yes/no | rootユーザーでのログインの可否。 |
| PasswordAuthentication | yes/no | パスワードでのログインの可否。 |
| PubkeyAuthentication | yes/no | 公開鍵でのログインの可否。 |
| Protocol | 1/2 | 2が推奨らしい |
| ChallengeResponseAuthentication | yes/no | パスワード認証が否なのでこちらも否 |

### 各ファイルのパス

Windows

実行ファイル ％WINDIR%\System32\OpenSSH

config C:\Users\user\_name\.ssh\config（自分で作成する）

sshd\_config %PROGRAMDATA%\ssh\sshd\_config（自分で作成する）

authorized\_keys %USERPROFILE%\.ssh\authorized\_keys（自分で作成する）

Linux

config $HOME/.ssh/config（自分で作成する）

ssh\_config $HOME/.ssh/ssh\_config（自分で作成する）

sshd\_config /etc/ssh/sshd\_config

authorized\_keys $HOME/.ssh/authorized\_keys

再インストールなどでホスト鍵が変わった場合

$HOME/.ssh/known\_hostsの該当レコードを削除

※windows だと　users\[ユーザー名]\.ssh\known\_hosts

## Stunnel

汎用TLS/SSLトンネリングサービス用アプリ。

クライアントサーバシステムがTLSやSSLにネイティブで対応していない場合に，安全な暗号化されたコネクションを提供することができる

## WireShark

プロトコル。

例えば検索バーに nbns と入力すると，プロトコルがNBNSのパケットだけを抽出できる。

## vsftpd

FTPサーバーを提供する。

## WebRTC

WebのAPI経由でリアルタイム通信を提供するソフトウェア。

BSDライセンス。

## Apache Pulsar

非同期通信（ストリーミング）を提供するメッセージミドルウェア。

Splunkの内部でも使用されている。

# そのほか

## BBR-4HG設定 (20.05.21)

最重要ポイントとして、win10にルート記述があるかはしっかり確認する事。

### WAN：plcnet LAN：sugino.localの場合

・管理→PINGテスト　ブラウザはLAN(sugino.local)側からしか開けない。WAN側へPINGテストして返ってきた。WAN：sugino.local LAN：plcnetの時は返って来なかった。

・通信できなかった → WAN側RIP送信を RIP1とRIP2両方　にしたら通信するようになった。

ICMPリダイレクト：使用しない

はっきりとしている事

・WANからのPING「応答する」にする必要がある事。

### WAN：sugino.local LAN：plcnetの場合

DD2AX(10.4.1.112) からルーター(10.4.1.251) にping が届かない時点でおかしい。

ファイヤーウォールを停止してもpingは通らなかった。

Apache Spark オープンソースの分散処理システムです。

Apache Storm オープンソースのリアルタイム分散処理システムです。

Apache Kafka オープンソースの分散メッセージングシステムです。

Apache Hadoop 多数のサーバで構成された大規模な分散ファイルシステム機能を提供し，MapReduceによる大規模データの分散処理を実現する

### WINS サーバー

NetBIOS名での通信を可能にさせるサーバー。

### VPN (Virtual Private Network)

仮想で保護された専用線で通信を行う（リモートデスクトップ等）。

・ルーターが対応していなくてもサーバーで構築可能。

「VPNリモートアクセス」という規格があるらしい。

・NTTなどが運用する、実際の独立ケーブルを利用したサービスもある。

#### vSwitch；Virtual Switch

サーバー上のソフトウェアによって仮想的に作成するL2スイッチで、仮想サーバ同士、仮想サーバーと物理ネットワークを接続する為に使う。仮想スイッチ、仮想ブリッジとも呼ばれる。

VPNプロトコル：

PPTP ：iOSやmacOS、v6プラス(MAP-E)

transix ：DS-Lite)

## NetAcad

Ciscoが出しているエミュレータがある。Ciscoのアカウントを作る必要がある。

https://www.netacad.com/

# 用語

## 一般用語

CDN；Content Delivery Network

動画などの大容量コンテンツを複数のキャッシュサーバーで分散配置し、大量配信や高速配信に耐えうるようにする仕組み。

CDN提供の証明書を使用して、ネットワーク内でホストされているプロパティの訪問者にセキュリティを提供するという事もできる？

CDP；Cisco Discovery Protocol

データリンク層で動作するCisco独自のプロトコル。

L3機器（IPアドレスを管理）に依存せずに、隣接するCisco機器の情報が得られる。

CRUD

Create, Read, Update, Deleteの頭文字で、永続的なデータを取り扱うソフトウェアに要求される4つの基本機能。

CRC；Cyclic Redundancy Check

誤り検出符号の一種。パリティやチェックサムに比べ検出精度が高い。

DMA メインメモリと入出力装置の間で、CPUを介さずに直接データ転送を行う

メモリインタリーブ

メインメモリをバンクと呼ばれる独立して動作可能な部分に分割し、分割されたバンクごとにバスにつなげ、CPUからのアクセス要求を並列的に処理する

LDAP

Lightweight Directory Access Protocol

マスタとなるサーバーに認証情報が登録されていれば、クライアント機での個人設定なしにユーザー認証ができる仕組みの事。

IETF

Internet Engineering Task Force

インターネットで利用される技術の標準化を推進する組織。「RFC2616」のようにRFC；Request For Comeents と呼ばれる文書でインターネット上に公開され、誰もが自由に閲覧できる。

Jitter

電気通信などの分野において、時間軸方向での信号波形の揺らぎの事

また，それによって生じる映像などの乱れ

### NIST；National Institute of Standards and Technology

アメリカ国立標準技術研究所

アメリカの技術革新や産業競争力を強化するために、経済保障を強化して生活の質を高めるように計測学、規格、産業技術を促進することを目的とする機関。

### SPOF：Single Point Of Failure

単一障害点

ある箇所が故障すると、システム全体が機能不全になる箇所

アプライアンス型

ハードウェアとソフトウェアをセットで提供する装置

スループット

機器や通信路などの性能を表す特性の一つで、単位時間あたりに処理できる量

ライトスルー

書き込み命令が実行されたときに、キャッシュメモリと主記憶の両方に書き込む

ポートスキャナ

インターネットに公開されているサーバのTCPポートの中からアクティブになっているポートを探して，稼働中のサービスを特定するためのツール

ミラーポート

ネットワーク上の通信パケットを取得して通信内容を見るために設けられたスイッチのLANポート

ライトバック

キャッシュメモリにだけデータを書き込む。主記憶への書き込むはブロックの入れ替え維持に行う

スケールイン 物理サーバーの台数を減らす。

スケールアウト 物理サーバーの台数を増やす。

スケールダウン CPU、メモリのスペックを下げる。

スケールアップ CPU、メモリのスペックを上げる。

ディレクトリ

「フォルダ」（コンピューター）とほぼ同意語だが、以下の特徴、傾向がある。

・UNIX 系OSで好んで使われる

・CUIで使われる。一方、GUIでは「フォルダ」が好まれる。

・厳密にはブートドライブなどの保存場所を指し、メモリ上だけで存在するファイルなどは対象ではない。

スイッチ

近年使用されている、スイッチングハブの事。L2スイッチとも呼ばれる。

従来のハブでは受信したパケットを、接続する全ての端末に送信していたため、効率が悪かった。

スイッチのポートとPCを1対1で接続した場合は全二重通信が可能。

半二重通信

RS422，485，イーサネット通信などで用いられている方式で、送受信に１対の信号線を共有する方式。

※全二重では送信用と受信用に２対の信号線が必要となる。

CSMA/CD方式

通信路の使用状況を監視し、伝送路の空きを見つけてデータ伝送を行う方式。ハブ（半二重）で使われる。

衝突（コリジョン）を検知すると、乱数時間待って再送（リトライ）を行う。イーサネットの特徴。

但し、スイッチングハブの登場により、CSMA/CDの使用頻度は減っている、との事。

OpenPGP

テキストメッセージと電子メールの暗号化と復号化に使用される暗号化技術PGPを広く利用されるようにする為の暗号化規格。通信を暗号化するSSL や SSH とは異なり、OpenPGP はアップロードする前にファイルを暗号化する。そのため 標準の FTP、SSL、SSH 接続と組み合わせて使用​​。

PGP；Pretty Good Privacy （関連）GnuPG

PPAP

Password付ZIP、Password、暗号化、プロトコル、の意味らしい。命名は2017年頃らしく、無理やり付けた感がある。メールにパスワード付zipファイルを添付し，のちにパスワードを送る方式。

メールをクラックされていたら簡単に資料の閲覧ができるのでセキュリティ面では無意味。

2020年に政府からPPAP利用の廃止の発表があった。

PKI；Public Key Infrastructure

公開鍵基盤（こうかいかぎきばん）

公開鍵暗号やデジタル署名によりWeb通信を安全に運用する為の基盤。

SSID；Service Set Identifier)

ネットワーク内の無線LANを識別する為の名前

svchost.exe Windows OS(NT系)の複数のサービスを管理するシステムプロセス。

UPnP

ユニバーサルプラグアンドプレイ。DHCPを更に強化したような機能。

機器を接続しただけでネットワークに参加。対応のアプリを使用する時のみ利用する。

バックエンド

特にWeb開発において、ユーザーの目に触れない部分。

フロントエンド

クライアント端末で、ユーザーが直接触る部分。

マイクロセグメンテーション

スイッチのポートごとにコリジョンドメインを分割（小さくセグメント化）する事。

ハイパーバイザ

仮想マシン(VM) を構成するプログラムの事。

（補足）hyper-v という仮想化ソフトがあったような

サイクリック通信

FA通信における周期的なデータ伝送の事。Ethernet/IPではImplicit通信と呼ばれる。

1:NのUDPブロードキャスト（全ノード）通信。

デジタル署名

別名：ディジタル署名

電子文書に付与され、署名したのが本人である、文書が改ざんされていない、という事を証明する。

「ハッシュ関数」、「公開鍵暗号方式」、「公開鍵暗号基盤（PKI）」を組み合わせる。