NGINX
Part of F5

東京エレクトロン デバイス

「NGINX Plus」基礎解説

― NGINX configの書き方入門 ―

「NGINX Plus」はOSS（オープンソースソフトウェア）版として広く普及しているNGINXの商用製品で、APIゲートウェイやロードバランサ、Webアプリケーションファイアウォール（WAF）、Webサーバーなどの多様な用途に対応します。NGINX Plusは具体的にどのような機能を備え、どのように利用することができるのかを解説します。

NGINX Plusの概要

NGINX Plusとは？

NGINX PlusはAPIゲートウェイ、キャッシュ、ロードバランサ、Webアプリケーションファイアウォール（WAF）、Webサーバーといった機能をまとめたオールインワン・ソリューションです。

OSS版のNGINXに対して、NGINX Plusには次のような追加機能があります。

まずNGINX Plusは、商用製品としてメーカー（F5）から技術サポートが提供されます。またNGINX Plusでしか使えない機能があります。NGINXメインラインでテスト・実証されたOSS版のすべての機能に加え、セッション維持、APIによる構成、アクティブヘルスチェックなど独自のエンタープライズ向け機能が実装されています。さらに、NGINX App Protectがアドオンできるのも大きな特長です。

機　能	特　徴
アクティブヘルスチェック	負荷分散サーバーをモニターして適切なサーバーにリクエストを振り分ける機能
セッション維持	Cookieパーシステンス
DNSサービス・ディスカバリ統合	ホスト名による負荷分散サーバーのグルーピング
JWT認証	リクエストに含まれるJWT認証
ストリーミング配信	HTTP live streaming（HLS）サポート
APIサポート	APIによる設定変更、ステータス取得
有償サポート	メーカーによる技術サポート・脆弱性/不具合によるパッチ提供

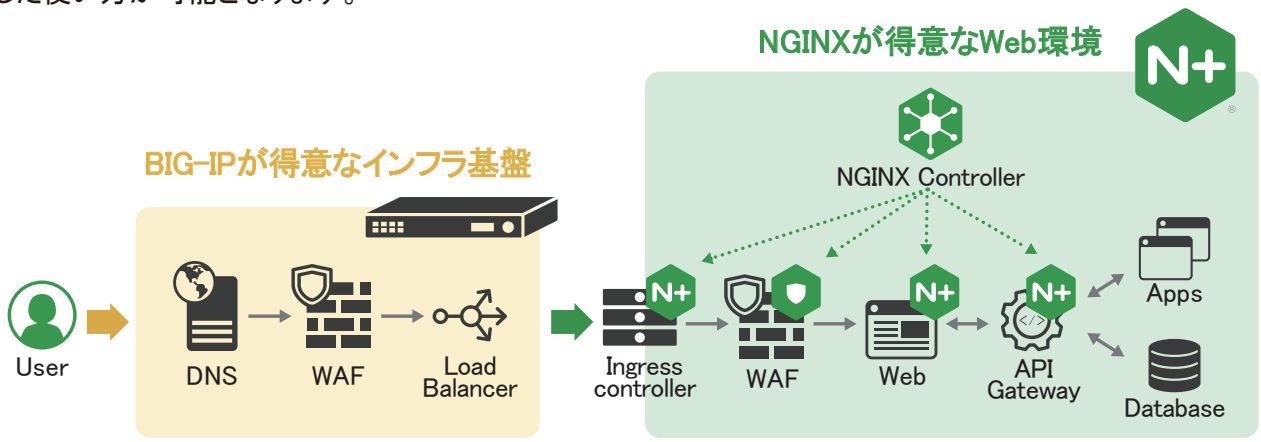
【図1】NGINX Plusが備える主な追加機能

NGINX PlusとBIG-IPの棲み分け

F5では、DNSやWAFなどの機能をアドオンできる非常に高性能なL7ロードバランサとして「BIG-IP」も提供しています。では、BIG-IPとNGINX Plusはどう使い分ければよいのでしょうか。

端的に言えば、DNSやWAF、ロードバランサといったインフラに近い部分をBIG-IPが担当し、Webなどのシステム

の入り口、いわゆるアプリケーションの領域をNGINX Plusが担当する形となります。BIG-IPにはコンテナイメージが存在していないため、さまざまなマイクロサービスのコンテナが存在する環境において、とくにNGINX Plusの特長を活かした使い方が可能となります。

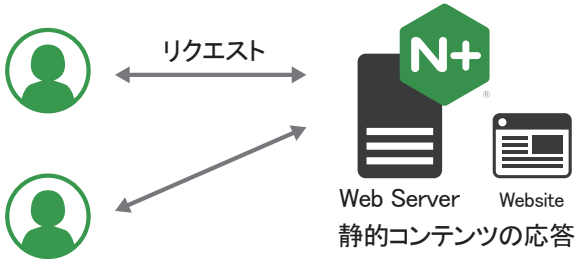


【図2】NGINX PlusとBIG-IPの使い分け

NGINX Plusの基本機能

Webサーバー

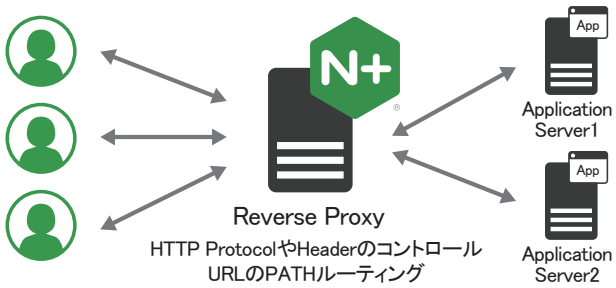
NGINX Plusは静的なコンテンツを返すシンプルなWebサーバーとして使うことが可能です。もともとApacheのC10K問題（クライアント数が約1万台に達すると、Webサーバーなどのレスポンスが悪くなる問題）を解決するためにつくられたのがNGINXであり、高速・軽量・安定なWebサーバーとして定評があります。2021年5月にはApacheのシェアを抜いて、世界一使われているWebサーバーとなりました。



【図3】Linux系のOS上で動作するWebサーバー

リバースプロキシ

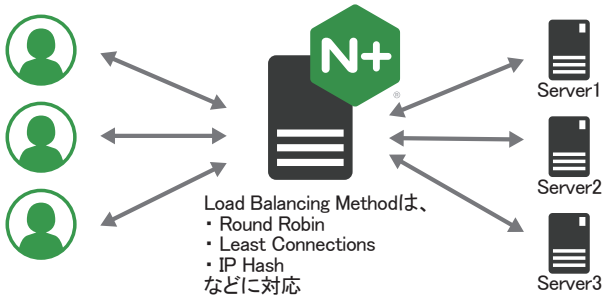
NGINX Plusをアプリケーションサーバーの前に配置することで、リバースプロキシとして利用することができます。クライアントからのトラフィックをNGINX Plusがいったん終端し、HTTPのプロトコルやヘッダなどの情報をもとに適切なアプリケーションサーバーへ中継する形をとります。



【図4】リバースプロキシの役割

ロードバランサ

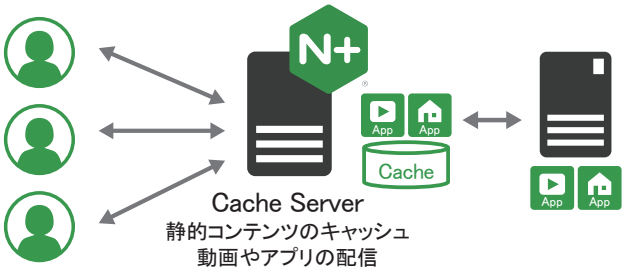
クライアントからのトラフィックを背後のサーバーに均等に渡したり、サーバーの状態を見ながらトラフィックを渡したりといったコントロールを行うことができます。



【図5】ロードバランサの役割

コンテンツキャッシュ

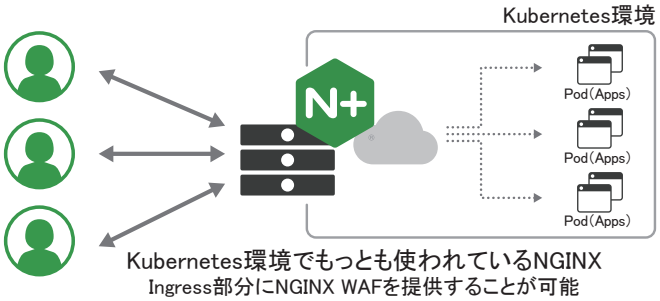
動画ファイルなどの静的コンテンツをNGINX Plusでキャッシュすることで、エンドユーザーに素早くコンテンツを返すことができます。前述のリバースプロキシやロードバランサと併用することで、より高い効果を発揮します。



【図6】コンテンツキャッシュの役割

Ingressコントローラー

KubernetesのIngress環境としてNGINX Plusを利用することができます。とくにIngress環境にWAFの機能を持たせられるのは、現時点でNGINX Plusのみとなっています。



【図7】Ingressコントローラーの役割

NGINX Plusのアーキテクチャー

NGINX Plusのプロセス構成

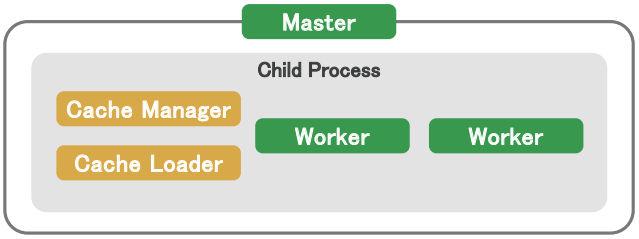
NGINX Plusには、Master ProcessとChild Processという大きく2つのプロセスがあります。
Master Processは全体のプロセスをコントロールするプロセスです。
Child Processはさらに次のような種類に分かれます。

- Cache manager：Cacheファイルの管理、Cacheサイズを定期的に監視
- Cache loader：メモリにCacheファイルを読み込む
- Worker Process：通信を制御するプロセス

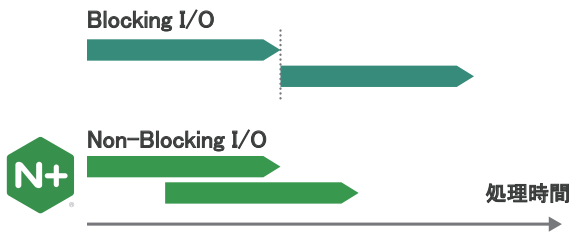
このうち、とくに重要な役割を担うWorker Processについて説明します。

Worker ProcessはCPUのコア数に応じて増えていきます。例えばCPUのコア数が2つであればWorker Processも2個となり、プロセスの数に比例して通信の処理性能も向上していきます。

また、NGINX PlusのWorker ProcessはNon-Blocking I/Oという仕組みを採用しており、1つのWorker Processで複数のリクエストを処理することも可能です。一般的なBlocking I/Oでは1つのデータを受け取った際に、その処理が完了するまで次の処理を待たなければなりません。これに対してNon-Blocking I/Oは、1つのデータを処理している途中でも次のデータの処理を開始することができ、データ処理を並列して行うことができるのです。NGINX PlusはこのNon-Blocking I/Oを採用することで、軽量かつ大量のリクエストを処理することに適したアーキテクチャーとなっています。



【図8】NGINX Plusのプロセス構成



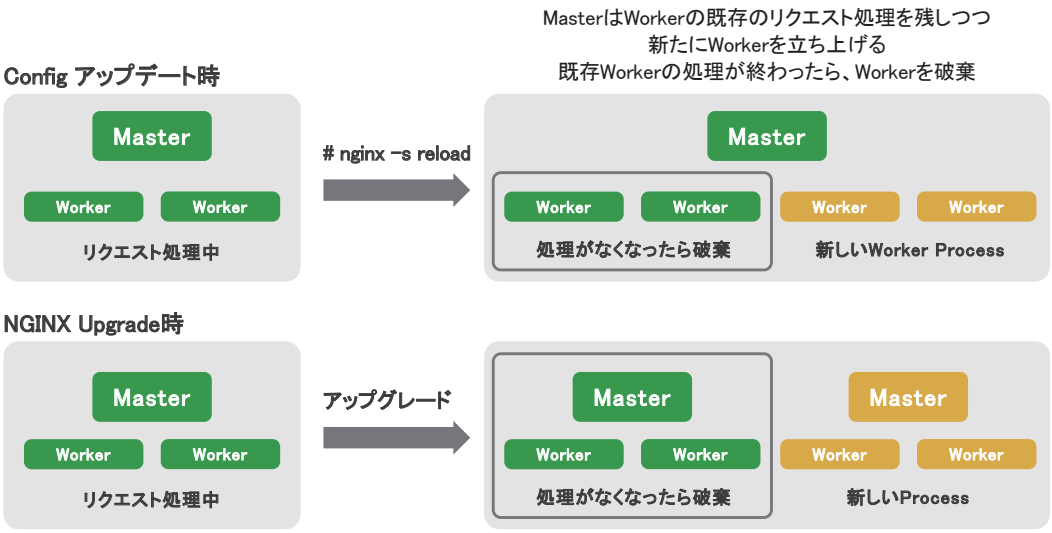
【図9】Non-Blocking I/Oの動き

No Downtime

NGINX Plusは、メンテナンス時にもユーザーのリクエスト処理になるべく影響を与えない配慮がなされています。

まずはConfigアップデート時の動作です。NGINX Plusのconfigを変更した際に「nginx –s reload」というconfigを再読み込みさせるコマンドを実行しますが、仮にこのとき、Worker Processがリクエスト処理中だった場合どうなるでしょうか。NGINX PlusはWorker Processが実行中のリクエスト処理を残しつつ、新しいWorker Processを立ち上げます。そして既存のリクエスト処理がすべて終了した時点で、古いWorker Processを破棄します。

NGINX Plusのアップグレードを行う際も同様です。上述のConfigアップグレードと少し異なるのは、Worker Processに加えてMaster Processも一緒に立ち上げる動作を行うことで、既存のリクエスト処理に与える影響を回避します。



【図10】リクエスト処理に与える影響を回避

NGINX configの書き方

初期のconfigファイルの格納場所

NGINX Plusのconfigファイルは、初期設定では次の2つの場所に配置されています。

- /etc/nginx/nginx.conf
- /etc/nginx/conf.d/default.conf

前者にはNGINX Plusの基本設定のほか、Worker Processの設定やエラーログなどの設定などが書かれています。後者にはDefaultで動作するWebサーバーの設定などが入っています。

NGINX configについての基礎知識

NGINX configを書くにあたっては、次の3つの要素を最低限理解しておく必要があります。

❶ Directives

NGINX Plusの動作を記述するもので、文末はセミコロンで終了させることがルールとなっています。例えば「Listen 443 ssl ; 」と記述した場合、NGINX Plusは port 443 ssl で通信を受けることを意味します。

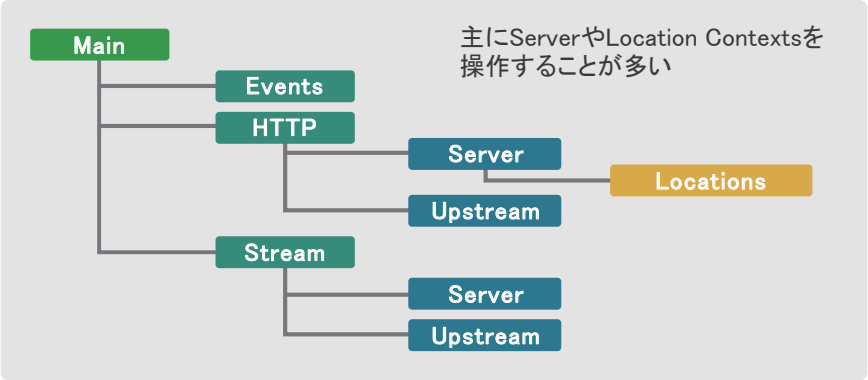
❷ Block

Directiveをまとめたもので、{}で囲まれている部分がブロックとなります。

【図11】DirectivesとBlockの書き方

❸ Configuration Contexts

NGINX Plusのconfigは、コンテキスト単位で設定を記述します。また、このコンテキストは階層構造となっています。



【図12】コンテキストの階層構造

Main Context

コンテキストの階層構造における最上位のDirectivesがMain Contextです。Worker Processの数、エラーログの出力先、エラーレベル、動的モジュールのLoad、実行ユーザー名の設定、Process IDのファイル場所などをここで定義します。

Event Context

Worker Processごとのコネクション数を記述します。ただし、通常はあまり触ることのないコンテキストであるため、基本的にはdefaultのままで使用します。

HTTP Context

HTTPのコネクションをどのように処理するか定義します。さらに、アクセスログおよびそのフォーマットの定義、Location（通信の渡し先）の指定を行うほか、include directiveという仕組みを使って、別ファイルからconfigを読み込むことも可能です。

Server Context

ユーザーからのリクエストに応答するVirtual Serverを設定します。例えばDomain name、IPアドレスやportの設定、SSL/TLS証明書および鍵の指定、SSLのプロトコルやciphersの指定などを行います。

Location Context

Server コンテキストで受け取ったリクエストをどのように処理するかを定義します。httpのバージョンはどうするのか、リクエストを受け取ったあとで行うURIの書き換え、PATHによる制御、HTTPヘッダの制御、受け渡すupstreamなどを記述します。

configの継承

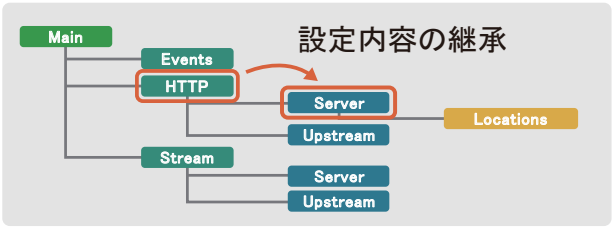
NGINX Plusでは、上位の階層で設定したconfigが下位の階層に引き継がれます。例えば上位階層のhttpコンテキストにindex.htmlを設定すると、その下位階層のServer Contextではindexのを記述する必要がありません。ただし、下位階層で明示的にindexのを記述した場合は、その内容で上書きされた動作を行います。

```
http{
    index index.html
    server {
        listen 80;
        root conf.d/html;
    }
    server{
        listen 8080;
        root/etc/nginx/conf.d/html;
        index main.html;
```

HTTP contextにindexのDirectiveあり

1つ目のServer contextに indexのDirectivesはなし → Listen 80 リクエストを送信すると index.htmlが返ってくる

2つ目のServer contextに indexのDirectivesあり → Listen 8080 リクエストを送信すると main.htmlが返ってくる



上位の設定(HTTP Context)が
下位の設定(Server Context)に引き継がれる

【図13】上位階層の設定が下位階層に引き継がれる

Include Directive

http Directiveで別ファイルを参照し、その内容を読み込むことができます。このInclude Directiveは、/etc/nginx/nginx.confの中にデフォルトで設定されています。

それが【図14】に示したinclude /etc/nginx/conf.d/*.conf;の部分です。ここでは*.confと記述しているため、/etc/nginx/conf.d/にある.confで終わっているすべてのファイルを読み込む設定となります。例えば/etc/nginx/conf.d/loadbalancer.confというファイルがあり、そこにロードバランシングの設定が記述されていれば、その内容がそのままHTTP Contextの中に読み込まれます。

このように1つのファイルの中にすべての設定を記述する必要はなく、ある程度まとまった設定をファイルごとに記述することが可能です。

指定された別ファイルを「＊」＝「全て」で指定し、拡張子.confのファイルをすべて読み込む

ファイルパス：/etc/nginx/nginx.conf

```
http {
    include /etc/nginx/conf.d/*.conf;
}
```

ファイルパス：
/etc/nginx/conf.d/loadbalancer.conf

```
upstream app-pool {
    server 10.1.2.150;
    server 10.1.3.150;
}
server {
    listen       80;
    server_name  web01;
    root         /var/www/html;
    location / {
        proxy_pass http://app-pool;
----------<省略>----------
```

【図14】設定を別ファイルから読み込む

NGINX Plusの基本コマンド

NGINX Plusでよく使われる基本的なコマンドをまとめておきます。

・nginx -v：バージョンの確認

```
$ nginx -v
nginx version: nginx/1.21.3 (nginx-plus-r25)
```

・nginx -V：バージョン詳細およびモジュールの確認

```
$ nginx -V
nginx version: nginx/1.21.3 (nginx-plus-r25)
built by gcc 7.5.0 (Ubuntu 7.5.0-3ubuntu1~18.04)
built with OpenSSL 1.1.1  11 Sep 2018
TLS SNI support enabled
configure arguments: --prefix=/etc/nginx --sbin-path=/usr/sbin/nginx --modules-path=/usr/lib/nginx/modules --conf-
path=/etc/nginx/nginx.conf --error-log-path=/var/log/nginx/error.log --http-log-path=/var/log/nginx/access.log --pid-
path=/var/run/nginx.pid --lock-path=/var/run/nginx.lock --http-client-body-temp-path=/var/cache/nginx/client_temp --
http-proxy-temp-path=/var/cache/nginx/proxy_temp --http-fastcgi-temp-path=/var/cache/nginx/fastcgi_temp --
http-uwsgi-temp-path=/var/cache/nginx/uwsgi_temp --http-scgi-temp-
path=/var/cache/nginx/scgi_temp --user=nginx --group=nginx --with-compat --with-file-aio --with-threads --with-
http_addition_module --with-http_auth_request_module --with-http_dav_module --with-http_flv_module --with-
http_gunzip_module --with-http_gzip_static_module --with-http_mp4_module --with-http_random_index_module --with-
http_realip_module --with-http_secure_link_module --with-http_slice_module --with-http_ssl_module --with-
http_stub_status_module --with-http_sub_module --with-http_v2_module --with-mail --with-mail_ssl_module --with-stream --
with-stream_realip_module --with-stream_ssl_module --with-stream_ssl_preread_module --build=nginx-plus-r25 --with-
http_auth_jwt_module --with-http_f4f_module --with-http_hls_module --with-http_session_log_module --
with-cc-opt='-g -O2 -fdebug-prefix-map=/data/builder/debuild/nginx-plus-1.21.3/debian/debuild-
base/nginx-plus-1.21.3=. -fstack-protector-strong -Wformat -Werror=format-security -Wp,-D_FORTIFY_SOURCE=2 -fPIC' --
with-ld-opt='-Wl,-Bsymbolic-functions -Wl,-z,relro -Wl,-z,now -Wl,--as-needed -pie'
```

・nginx -t：Configuration チェック（エラーがあればその内容を表示）

問題なし

```
$ sudo nginx -t
nginx: the configuration file /etc/nginx/nginx.conf syntax is ok
nginx: configuration file /etc/nginx/nginx.conf test is successful
```

問題あり

```
$ sudo nginx -t
nginx: [emerg] unknown directive "www" in /etc/nginx/conf.d/lb.conf:3
nginx: configuration file /etc/nginx/nginx.conf test failed
```

・nginx -T：Configuration チェック（問題がなかった場合は設定内容を表示）

```
$ sudo nginx -T
nginx: the configuration file /etc/nginx/nginx.conf syntax is ok
nginx: configuration file /etc/nginx/nginx.conf test is successful
# configuration file /etc/nginx/nginx.conf:

user  nginx;
worker_processes  auto;

error_log  /var/log/nginx/error.log notice;
pid        /var/run/nginx.pid;

events {
    worker_connections  1024;
}


http {
    include       /etc/nginx/mime.types;
--------<省略>--------
```

・nginx -s reload：Configuration の読み込み

正常に読み込めた場合は、何も表示されない

```
$ sudo nginx -s reload
```

読み込みに失敗した場合は、以下のように失敗した箇所についてメッセージが出力される

```
$ sudo nginx -s reload
nginx: [emerg] unknown directive "www" in /etc/nginx/conf.d/lb.conf:3
```

設定変更した際は、以下のようにconfigurationチェックをした後で、設定を読み込む

```
$ sudo nginx -t
$ sudo nginx -s reload
```

NGINX PlusなどのF5製品をトータルサポート

東京エレクトロンデバイスはF5製品に関して、日本法人が設立される以前から代理店として長年にわたる実績を重ねており、現在ではF5国内販売額7年連続No.1の一次代理店となっています。

NGINX Plusはもとより、WAF製品のNGINX App Protect、統合管理製品のNGINX Controller、Kubernetes環境のIngressリソースを管理するNGINX Ingress Controller、コンテナ環境のトラフィックを管理するNGINX Service Meshといった幅広いラインナップを取り揃えています。

さらに東京エレクトロンデバイスでは、これらのF5製品に関して専任体制で営業・技術支援も行っています。

NGINX Plusの「30日間無償トライアル」をご用意しています。
ぜひ東京エレクトロンデバイスのホームページ
（https://cn.teldevice.co.jp/product/f5-nginx/）
からお問い合わせください。

東京エレクトロンデバイス NGINX 紹介ページ

東京エレクトロン デバイス株式会社
CN BU
https://cn.teldevice.co.jp

新宿：〒163-1034 東京都新宿区西新宿 3-7-1 新宿パークタワー S34 階
Tel.03-5908-1990 Fax.03-5908-1991

大阪：〒540-6033 大阪府大阪市中央区城見 1-2-27 クリスタルタワー 33 階
Tel.06-4792-1908 Fax.06-6945-8581

名古屋：〒451-0045 愛知県名古屋市西区名駅 2-27-8 名古屋プライムセントラルタワー 8 階
Tel.052-562-0826 Fax.052-561-5382

つくば：〒305-0033 茨城県つくば市東新井 15-4 関友つくばビル 7 階
Tel.029-848-6030 Fax.029-848-6035

お問い合わせは、Web サイトの下記フォームよりお願いします。
https://cn.teldevice.co.jp/product/f5/form.html

Mar 2022