



アプリケーションノート
μNet3 サンプルプログラム
エコーサーバ

Rev. 1.0

April 2, 2019

改訂記録

Rev.	発行日	改訂内容	
		ページ	内容
1.0	2019.04.02	—	初版

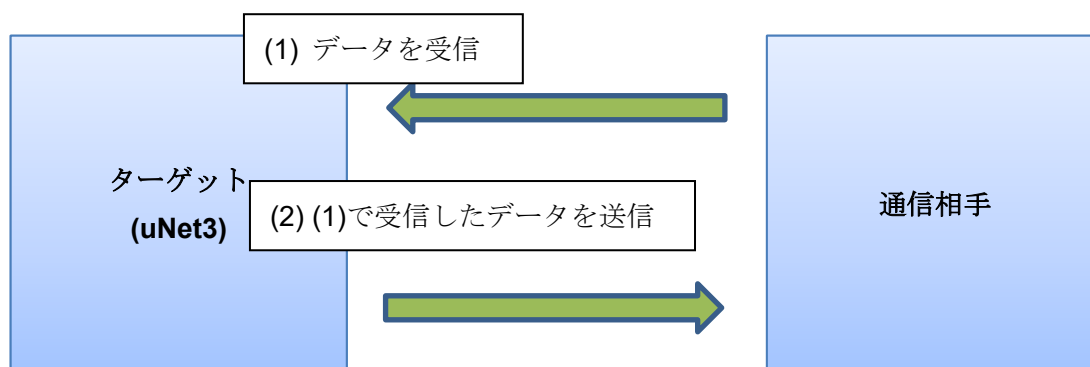
目次

改訂記録	2
目次 3	
1. 概要	4
2. ソースファイル	5
3. プロジェクトへ組み込む手順	6
3.1. プロジェクトへファイルを追加する。	6
3.2. コンフィグレータ・ソースファイル編集	7
3.2.1. タスクの作成	7
3.2.2. ソケットの作成	10
3.2.3. タスクの起動	15
3.2.4. ソースコード生成	15
4. ソースコードの解析	17
4.1. UDP 版	17
4.2. TCP 版	17
5. 動作確認	18
5.1. UDP 版	18
5.2. TCP 版	21

1. 概要

本ドキュメントはμ Net3 のソケットを使用した簡単なサンプル「エコーサーバ」を組み込む手順、使用方法について説明します。

エコーサーバは通信相手から受信したデータを、通信相手へ送信します。



UDP で動作する UDP 版、TCP で動作する TCP 版を用意しています。

エコーサーバのプログラム構造については簡単なプログラムであり、またソースコード上のコメントで解説しているので、ソースコードを参照してください。

また、μ Net3 の各 API の詳細はμ Net3 のユーザズガイドを参照してください。

μ C3 はイー・フォース株式会社の登録商標です。

μ Net3 はイー・フォース株式会社の登録商標です。

本書で記載されている内容は予告無く変更する場合があります。

い。

2. ソースファイル

エコーサーバは次のソースファイルを使用します。

表 2.1 ソースファイル

ファイル名	内容
sample_socket_echo_server_tcp.c	uNet3 のソケットを使用したエコーサーバ のサンプルプログラム (TCP 版)
sample_socket_echo_server_udp.c	uNet3 のソケットを使用したエコーサーバ のサンプルプログラム (UDP 版)

3. プロジェクトへ組み込む手順

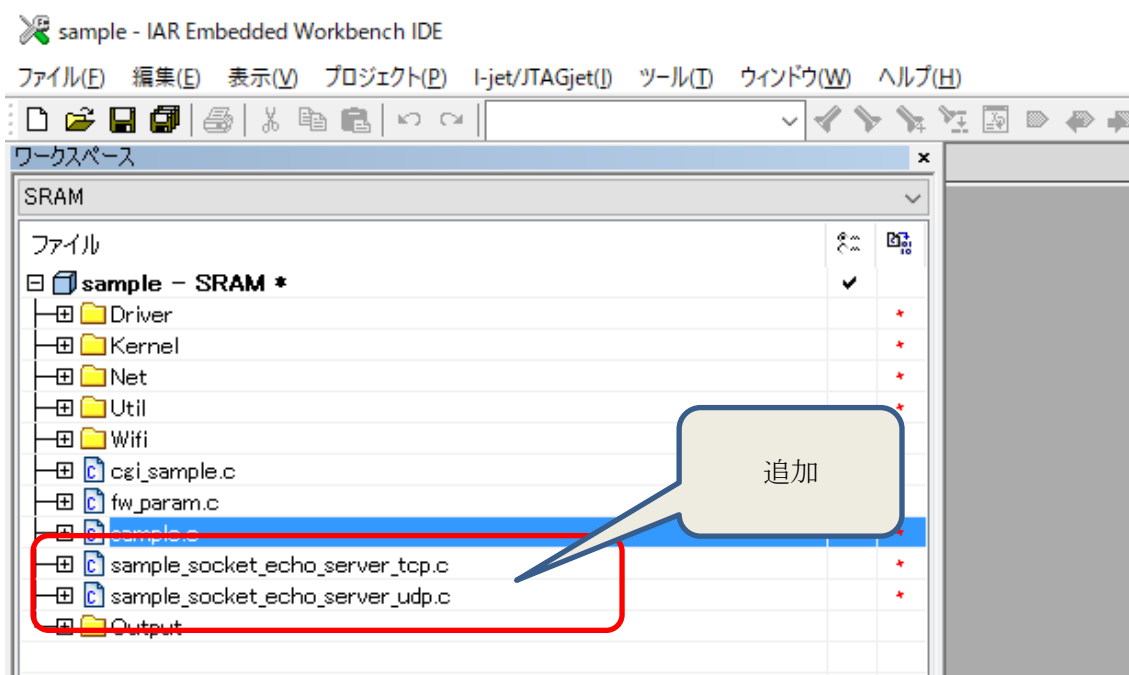
パッケージに収録されているネットワークサンプルプログラムへエコーサーバを追加する手順を説明します。

3.1. プロジェクトへファイルを追加する。

「2 ソースファイル」のファイルをサンプルプログラムが収録されているフォルダへコピーしてください。

その後、コピーしたファイルをプロジェクトへ追加してください。

IAR Embedded Work Bench の場合は以下のような形になります。



他のビルドツールをご使用の場合は、そのツール向けに読み替えてください。

3.2. コンフィグレータ・ソースファイル編集

エコーサーバはタスクから実行するので、タスクとソケットを作成、起動してください。

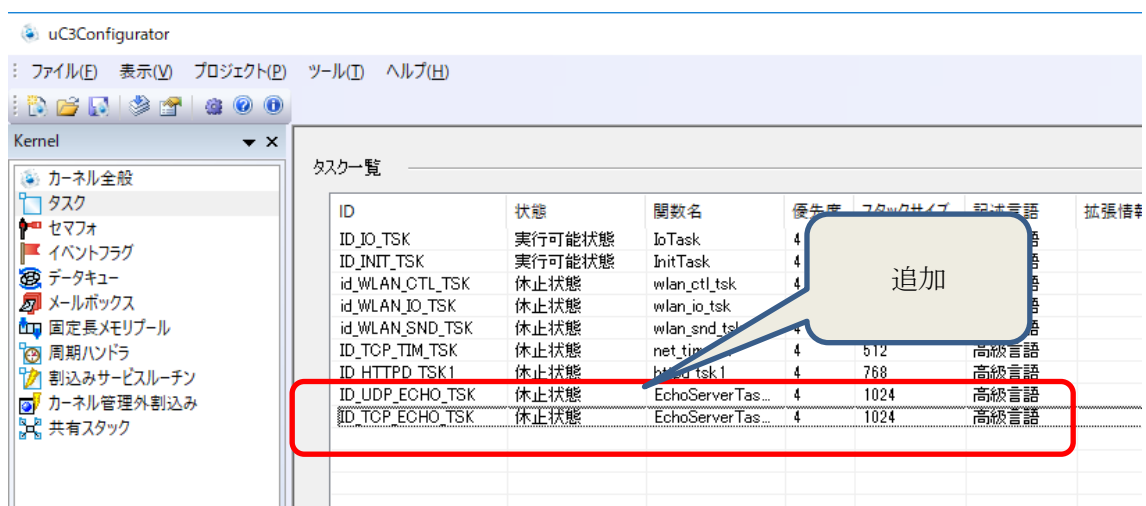
タスクの作成、ソケットの作成は uC3/Compact と uC3/Standard で異なります。

ご利用のパッケージが uC3/Compact か uC3/Standard か確認してください。

3.2.1. タスクの作成

■uC3/Compact の場合

コンフィグレータにタスクを追加します。



ID_UDP_ECHO_TSK (UDP 版のタスク)

タスク

IDの定義名

ID_UDP_ECHO_TSK

タスクの関数名

EchoServerTask_UDP

優先度の初期値

4

拡張情報

タスク属性

☐ 実行可能状態(TA_ACT)
☐ 制約タスク(TA_RSTR)

記述言語

☒ 高級言語(TA_HLNG)
☐ アセンブリ言語(TA_ASM)

スタック

☒ ローカルスタックを使用

スタックサイズ

1024

☐ 共有スタックを使用

共有スタックID

OK

キャンセル

ID_TCP_ECHO_TSK (TCP 版のタスク)

タスク

IDの定義名

ID_TCP_ECHO_TSK

タスクの関数名

EchoServerTask_TCP

優先度の初期値

4

拡張情報

タスク属性

☐ 実行可能状態(TA_ACT)
☐ 制約タスク(TA_RSTR)

記述言語

☒ 高級言語(TA_HLNG)
☐ アセンブリ言語(TA_ASM)

スタック

☒ ローカルスタックを使用

スタックサイズ

1024

☐ 共有スタックを使用

共有スタックID

OK

キャンセル

■uC3/Standard の場合

uC3/Standard の API `acre_tsk()` を使用してタスクを作成してください。

`acre_tsk` の引数の値

・UDP 版

tskatr	タスク属性
TA_HLNG	
exinf	タスクの拡張情報
0	
task	タスクの起動番地
EchoServerTask_UDP	
itskpri	タスクの起動時優先度
4	
stksz	タスクのスタック領域のサイズ
1024	
stk	タスクのスタック領域の先頭番地
0	

・TCP 版

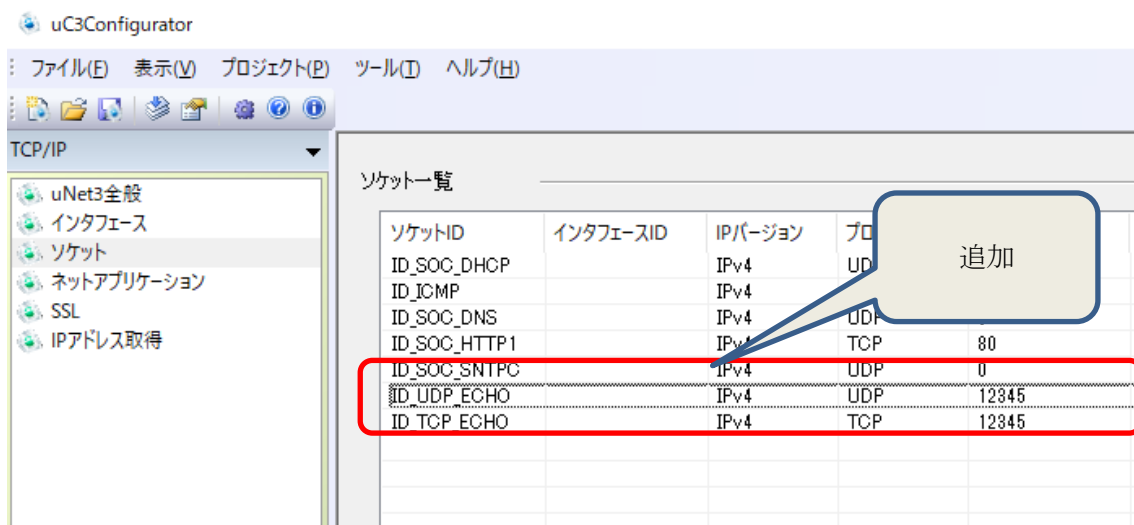
tskatr	タスク属性
TA_HLNG	
exinf	タスクの拡張情報
0	
task	タスクの起動番地
EchoServerTask_UDP	
itskpri	タスクの起動時優先度
4	
stksz	タスクのスタック領域のサイズ
1024	
stk	タスクのスタック領域の先頭番地
0	

3.2.2. ソケットの作成

■uC3/Compact の場合

コンフィグレータにソケットを追加します。

ソケットはUDP 版向けに UDP ソケットを 1 つ、TCP 版向けに TCP ソケットを 1 つ追加します。



ID_UDP_ECHO のソケット(UDP 版で使用するソケット)

ソケット

IDの定義名

インターフェースのバインディング

IPバージョン

プロトコル

ローカルポート

タイムアウト設定

snd_socのタイムアウト(ミリ秒)

rcv_socのタイムアウト(ミリ秒)

con_socのタイムアウト(ミリ秒)

cls_socのタイムアウト(ミリ秒)

TCPバッファ設定

送信バッファ(byte)

受信バッファ(byte)

OK キャンセル

待ち受けの
ポート番号を入力

ID_TCP_ECHO のソケット(TCP 版で使用するソケット)

ソケット

IDの定義名: ID_TCP_ECHO

インターフェースのバインディング: []

IPバージョン: IPv4

プロトコル: TCP

ローカルポート: 12345

タイムアウト設定

- snd_socのタイムアウト(ミリ秒): -1
- rcv_socのタイムアウト(ミリ秒): -1
- con_socのタイムアウト(ミリ秒): -1
- cls_socのタイムアウト(ミリ秒): -1

TCPバッファ設定

- 送信バッファ(byte): 1024
- 受信バッファ(byte): 1024

OK キャンセル

待ち受けの
ポート番号を入力

■uC3/Standard の場合

ソケットの作成はエコーサーバの初期化処理で実行していますので、特に何も行う必要はありません。既にソースコードに実装しています。

以下はソケットを作成している箇所です。

・UDP 版

sample_socket_echo_server_udp.c の echo_server_UDP ()

```
/* エコーサーバ処理 */
ER echo_server_UDP(void)
{
    ER ercd;
    T_NODE host;
    VB ebuf[256];

#ifdef NET_C
    ID ID_UDP_ECHO;

    /* ソケットの生成 */
    memset(&host, 0, sizeof(host));
    host.num = 1;
    host.ver = IP_VER4;
    host.ipa = INADDR_ANY;
    host.port = 12345; /* 待ち受けのポート番
    ercd = cre_soc(IP_PROTO_UDP, &host);
    if (0 >= ercd) {
        return ercd;
    }
    ID_UDP_ECHO = ercd;
#endif
}
```

ここでソケットを作成しています。

• TCP 版

sample_socket_echo_server_tcp.c の echo_server_TCP()

```
/* エコーサーバ処理 */  
ER echo_server_TCP(void)  
{  
    ER ercd;  
    T_NODE host;  
    VB ebuf[256];  
  
#ifndef NET_C  
    ID ID_TCP_ECHO;  
  
    /* ソケットの生成 */  
    memset(&host, 0, sizeof(host));  
    host.num = 1;  
    host.ver = IP_VER4;  
    host.ipa = INADDR_ANY;  
    host.port = 12345; /* 待ち受けのポート */  
    ercd = cre_soc(IP_PROTO_TCP, &host);  
    if (0 >= ercd) {  
        return ercd;  
    }  
    ID_TCP_ECHO = ercd;  
#endif
```

ここでソケットを
作成しています。

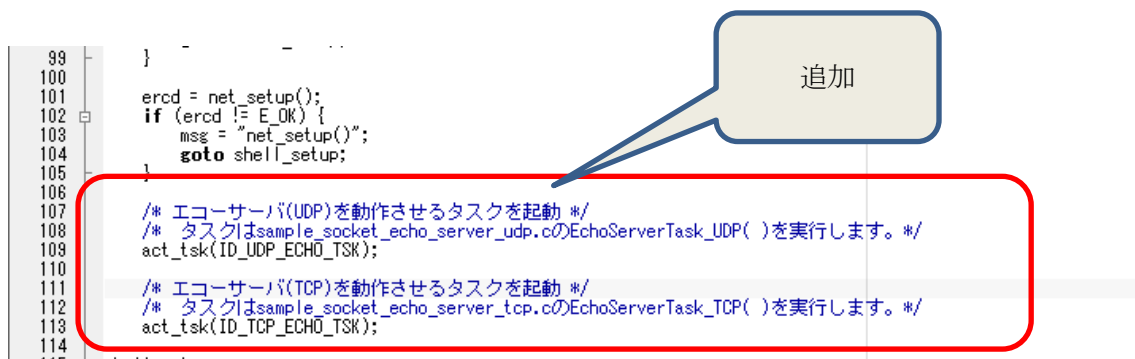
3.2.3. タスクの起動

「3.2.1 タスクの作成」で作成したタスクを起動させるにはμ C3 の API 「act_tsk()」を実行する必要があります。

これは、ソースコードをテキストエディタで編集してください。

タスクの起動はμ Net3 の初期化が完了した後に実行してください。

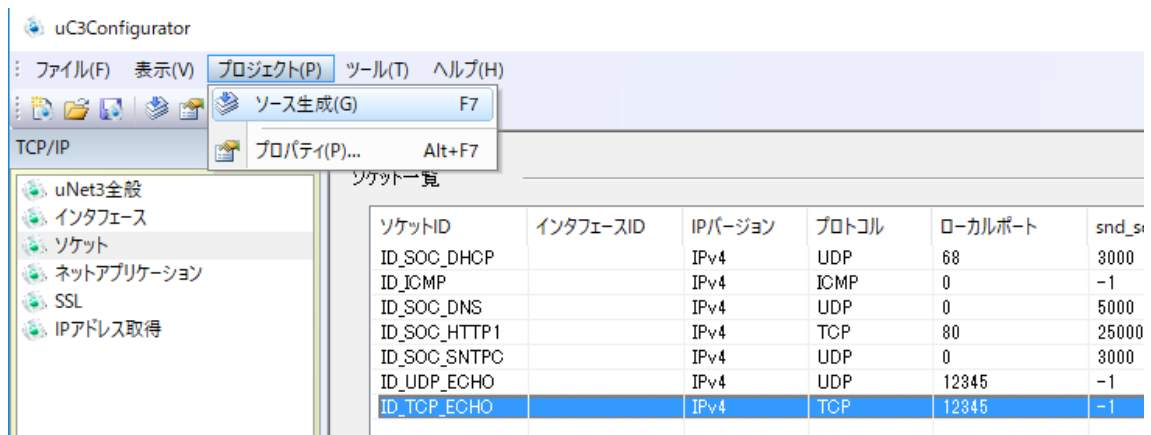
具体的には、パッケージに収録されているサンプルプログラムではnet_setup()の後になります。



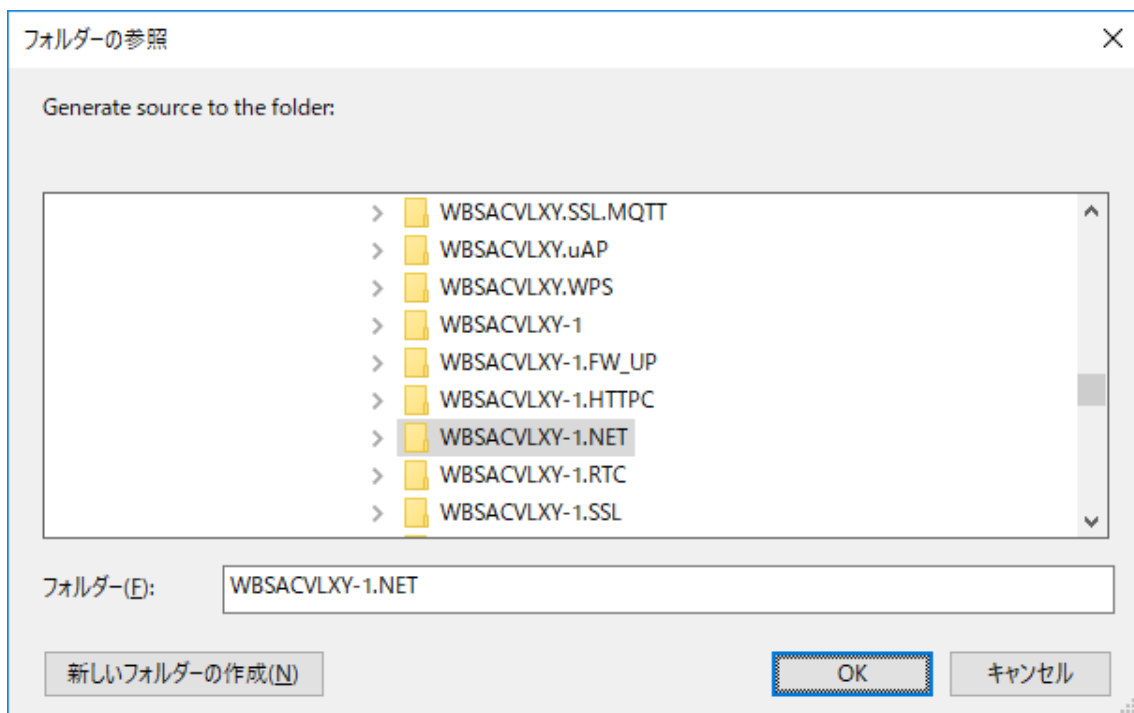
3.2.4. ソースコード生成

コンフィグレータを使用している場合は、タスク、ソケットを追加したので、ソースコードを生成してください。出力先のフォルダは元にしたサンプルプログラムと同じです。

メニューから「プロジェクト」→「ソース生成」



ダイアログが起動するので、生成先を入力してください。



4. ソースコードの解析

4.1. UDP 版

次の順に実行されます。

- (1) 「3.2 コンフィグレータ・ソースファイル編集」で追加した `act_tsk()` でエコーサーバを実行するタスクが起動します。
- (2) タスクは「`sample_socket_echo_server_udp.c`」の関数 `echo_server_UDP()` を実行します。

上記の順にソースコードを参照してください。

ソースコードには解説のたえに、コメントを書いております。

また、このプログラムは、μ Net3 の API を実行しています。

μ Net3 の API の詳細は μ Net3 のユーザズガイドを参照してください。

4.2. TCP 版

次の順に実行されます。

- (1) 「3.2 コンフィグレータ・ソースファイル編集」で追加した `act_tsk()` でエコーサーバを実行するタスクが起動します。
- (2) タスクは「`sample_socket_echo_server_tcp.c`」の関数 `echo_server_TCP()` を実行します。

上記の順にソースコードを参照してください。

ソースコードには解説のたえに、コメントを書いております。

また、このプログラムは、μ Net3 の API を実行しています。

μ Net3 の API の詳細は μ Net3 のユーザズガイドを参照してください。

5. 動作確認

ターゲットへのダウンロード、ネットワーク接続の手順、使用する機材についてはパッケージのサンプルプログラムのドキュメントを参照してください。

ここでは、追加機能「エコーサーバ」の動作確認を説明します。

5.1. UDP 版

PC とターゲットを同じネットワークへ接続して、PC から UDP データを送信してください。
ターゲットは UDP データを受信して同じデータを PC へ送信します。

適当なツールを使用して PC からターゲットへ UDP データを送信して、その後、受信データを確認してください。

ここでは、netcat を使用した場合の手順を説明します。

netcat は UNIX 系 OS のコマンドラインツールです。Linux や Cygwin 等で利用できます。

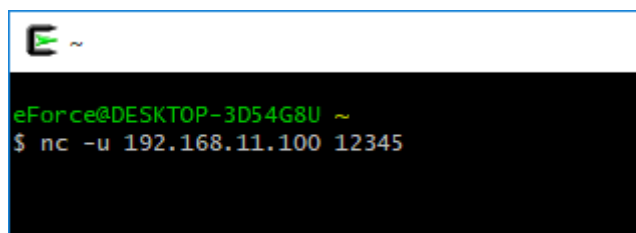
netcat のコマンド名は「nc」としていますが、ご利用の環境に合わせて読み替えてください。

(1) 次のコマンドを入力してください。

```
nc -u IP アドレス ポート番号
```

IP アドレスはターゲットの IP アドレス

ポート番号はターゲットが待ち受けているポート番号

A terminal window with a black background and green text. The prompt is 'eForce@DESKTOP-3D54G8U ~'. The command entered is '\$ nc -u 192.168.11.100 12345'.

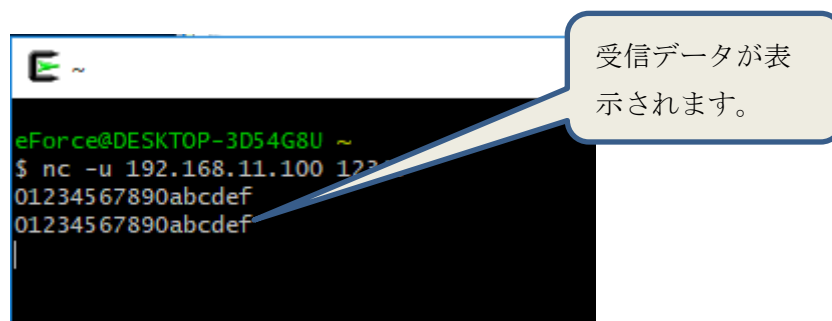
```
eForce@DESKTOP-3D54G8U ~  
$ nc -u 192.168.11.100 12345
```

(2) 送信する文字列を入力後、Enter を押してください。

下の画像では「01234567890abcdef」を送信します。

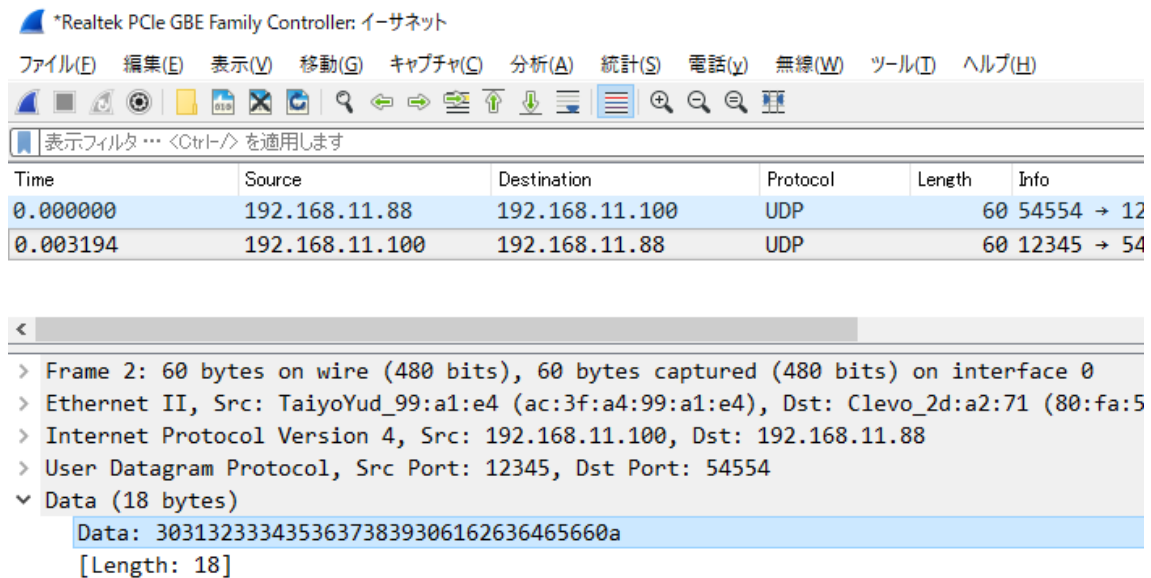


(3) データを受信すると、受信データが表示されます。



(4) Ctrl-C を入力することで、netcat は終了します。

通信のパケットを Wireshark で取得すると、PC からの送信とターゲットからの応答があることがわかります。



PC の IP アドレス 192.168.11.88

ターゲットの IP アドレス 192.168.11.100

5.2. TCP 版

PC とターゲットを同じネットワークへ接続して、PC から TCP でデータを送信してください。
ターゲットはデータを受信して同じデータを PC へ送信します。

適当なツールを使用して PC からターゲットへ TCP 接続してデータを送信して、その後、受信データを確認してください。

ここでは、netcat を使用した場合の手順を説明します。

netcat は UNIX 系 OS のコマンドラインツールです。Linux や Cygwin 等で利用できます。

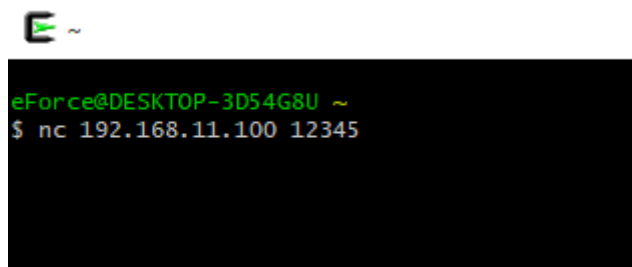
netcat のコマンド名は「nc」としていますが、ご利用の環境に合わせて読み替えてください。

(1) 次のコマンドを入力してください。

```
nc IP アドレス ポート番号
```

IP アドレスはターゲットの IP アドレス

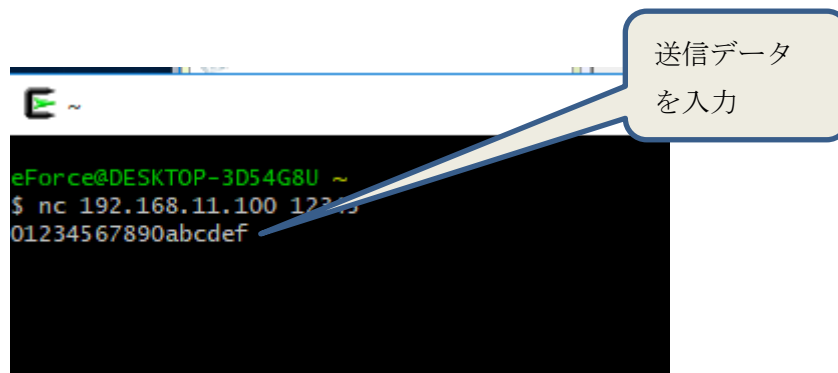
ポート番号はターゲットが待ち受けているポート番号

A terminal window with a black background and green text. The prompt is 'eForce@DESKTOP-3D54G8U ~'. The command entered is '\$ nc 192.168.11.100 12345'.

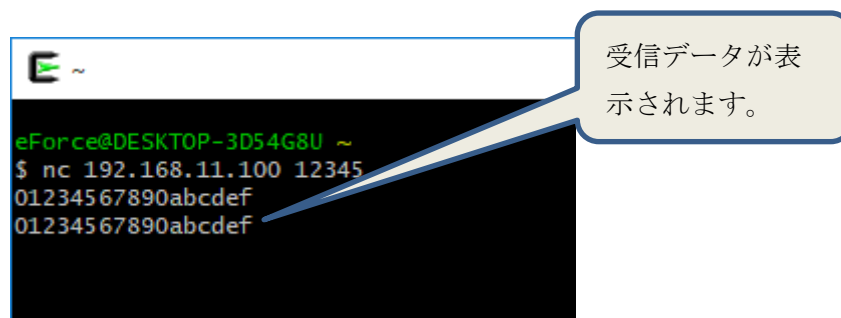
```
eForce@DESKTOP-3D54G8U ~  
$ nc 192.168.11.100 12345
```

(2) 送信する文字列を入力後、Enter を押してください。

下の画像では「01234567890abcdef」を送信します。



(3) データを受信すると、受信データが表示されます。



(4) Ctrl-C を入力することで、netcat は終了します。

通信のパケットを Wireshark で取得すると、PC からの送信とターゲットからの応答があることがわかります。

*Realtek PCIe GBE Family Controller: イーサネット

ファイル(E) 編集(E) 表示(V) 移動(G) キャプチャ(C) 分析(A) 統計(S) 電話(y) 無線(W) ツール(T) ヘルプ(H)

ip.addr==192.168.11.100

Time	Source	Destination	Protocol	Length	Info
16.107057	192.168.11.88	192.168.11.100	TCP	66	64640 → 12345 [SYN
16.110411	192.168.11.100	192.168.11.88	TCP	62	12345 → 64640 [SYN
16.110482	192.168.11.88	192.168.11.100	TCP	54	64640 → 12345 [ACK
53.279801	192.168.11.88	192.168.11.100	TCP	72	64640 → 12345 [PSH
53.283031	192.168.11.100	192.168.11.88	TCP	60	12345 → 64640 [ACK
53.291881	192.168.11.100	192.168.11.88	TCP	72	12345 → 64640 [PSH
53.291978	192.168.11.88	192.168.11.100	TCP	54	64640 → 12345 [ACK
99.379555	192.168.11.88	192.168.11.100	TCP	54	64640 → 12345 [FIN
99.386411	192.168.11.100	192.168.11.88	TCP	60	12345 → 64640 [ACK
99.386412	192.168.11.100	192.168.11.88	TCP	60	12345 → 64640 [FIN
99.386504	192.168.11.88	192.168.11.100	TCP	54	64640 → 12345 [ACK

<

> Frame 23: 72 bytes on wire (576 bits), 72 bytes captured (576 bits) on interface 0

> Ethernet II, Src: TaiyoYud_99:a1:e4 (ac:3f:a4:99:a1:e4), Dst: Clevo_2d:a2:71 (80:fa:5b:2d:a2:71)

> Internet Protocol Version 4, Src: 192.168.11.100, Dst: 192.168.11.88

> Transmission Control Protocol, Src Port: 12345, Dst Port: 64640, Seq: 1, Ack: 19, Len: 18

▼ Data (18 bytes)

Data: 30313233343536373839306162636465660a

[Length: 18]

PC の IP アドレス 192.168.11.88

ターゲットの IP アドレス 192.168.11.100

アプリケーションノート **uNet3** サンプルプログラム エコーサーバ

2019 年 04 月 02 日

Rev.1.0

イー・フォース株式会社 <http://www.eforce.co.jp/>

お問い合わせ support@eforce.co.jp

Copyright (C) 2019 eForce Co.,Ltd. All Rights Reserved.