

# Cefore トランスポートプラグイン

---

# コンテンツ

---

1. Ceforeトランスポートプラグイン
2. サンプルトランスポートプラグイン (samptp プラグイン)
3. 新たなトランスポートプラグインの追加方法

# 1. Cefore トランスポートプラグイン

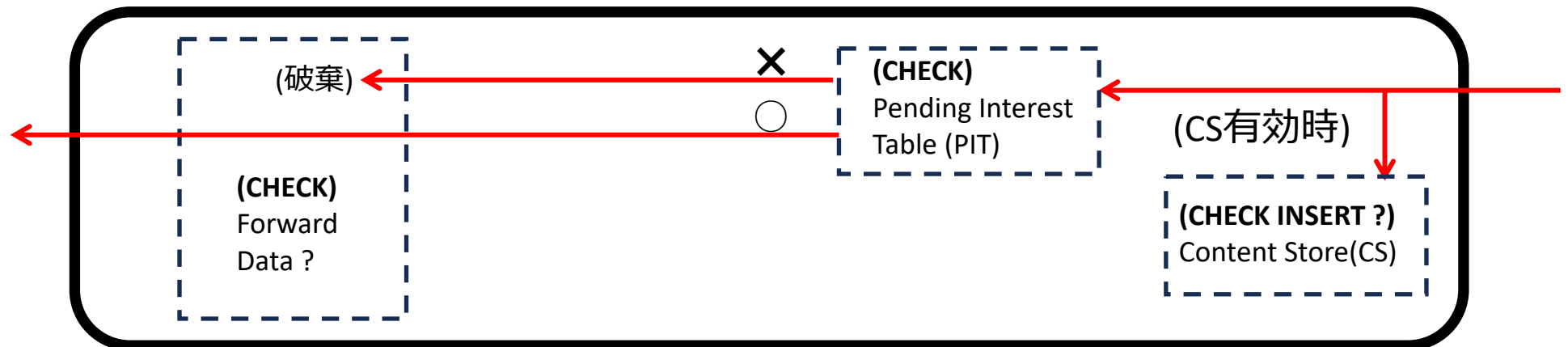
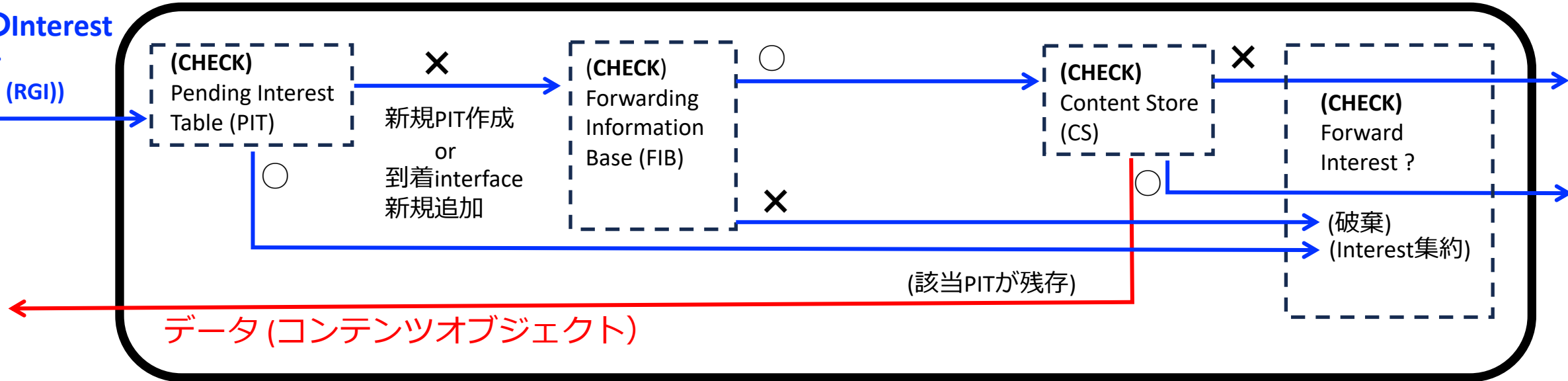
---

# 通常のトランスポート実装

## Cefore Forwarder

○ 検索ヒット × 検索ミス

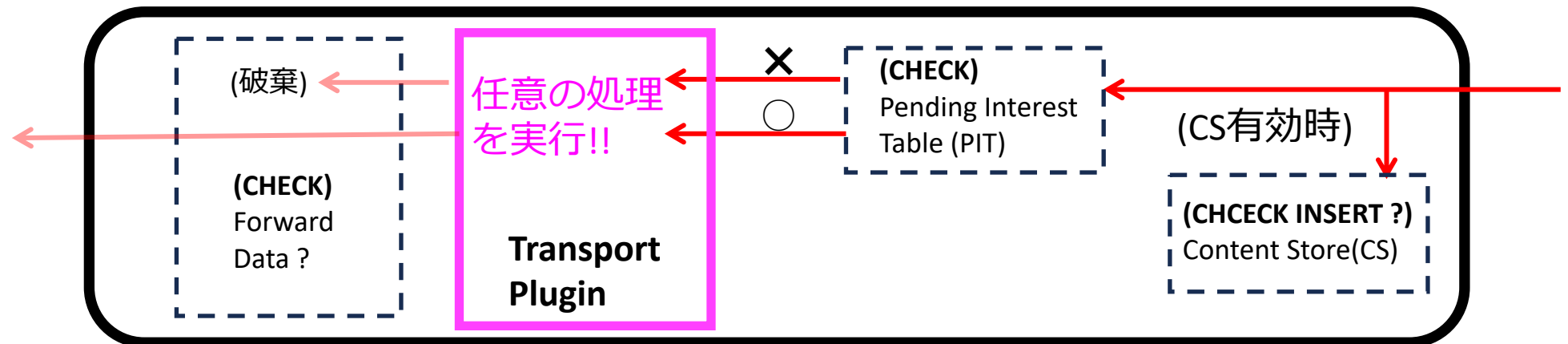
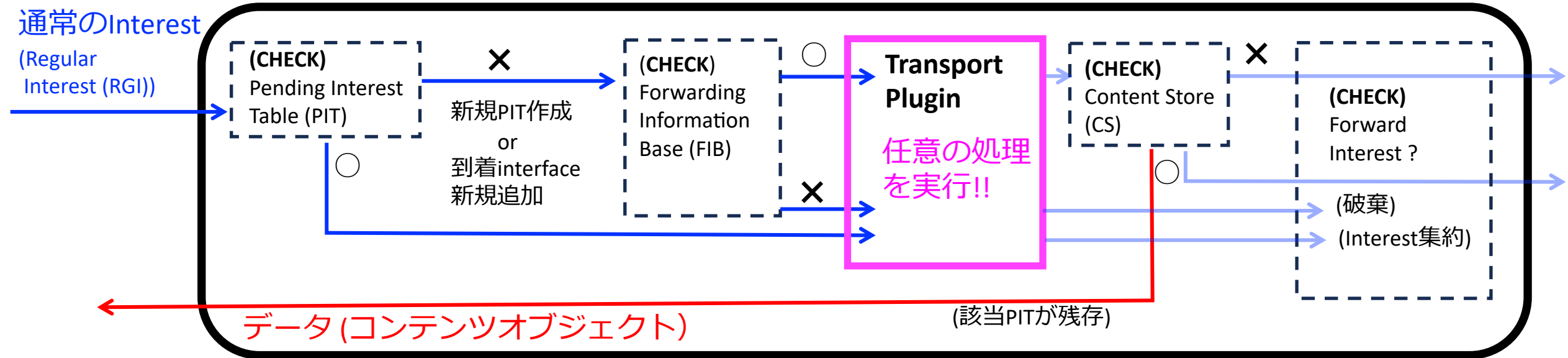
通常のInterest  
(Regular Interest (RGI))



# トランスポートプラグインを適用した場合

## Cefore Forwarder

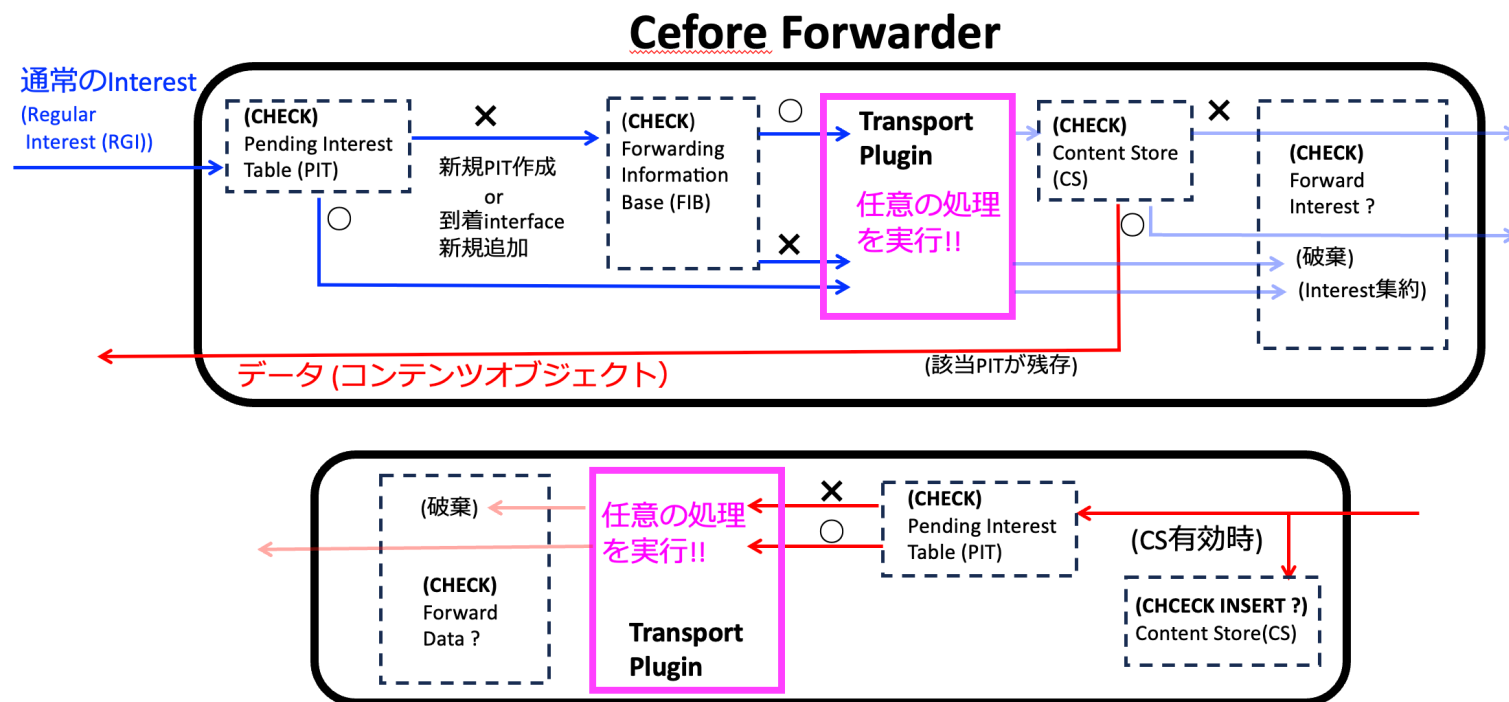
○ 検索ヒット × 検索ミス



# トランスポートプラグインの使用例

## ■ トランスポートに独自の「拡張機能」を組み入れる

- 統計収集機能
- 輻輳制御機能（インタレスト/データの送信レートを制御）
- 冗長データ転送
- 再送処理機能
- フィルター機能
- etc.



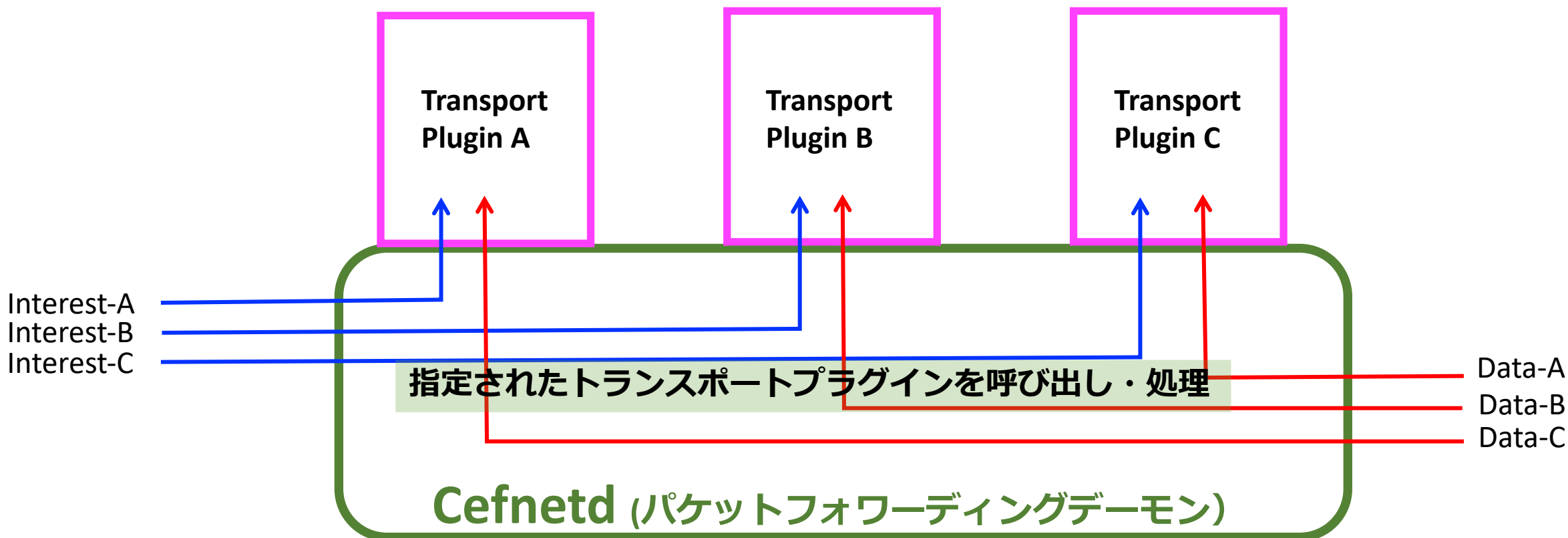
インタレスト and/or データに対して、所望の処理を実装!!

# トランスポートプラグイン実装

## ■ 柔軟に追加・削除、使用可能

※ **Cefnetd**: パケットフォワーディングデーモン

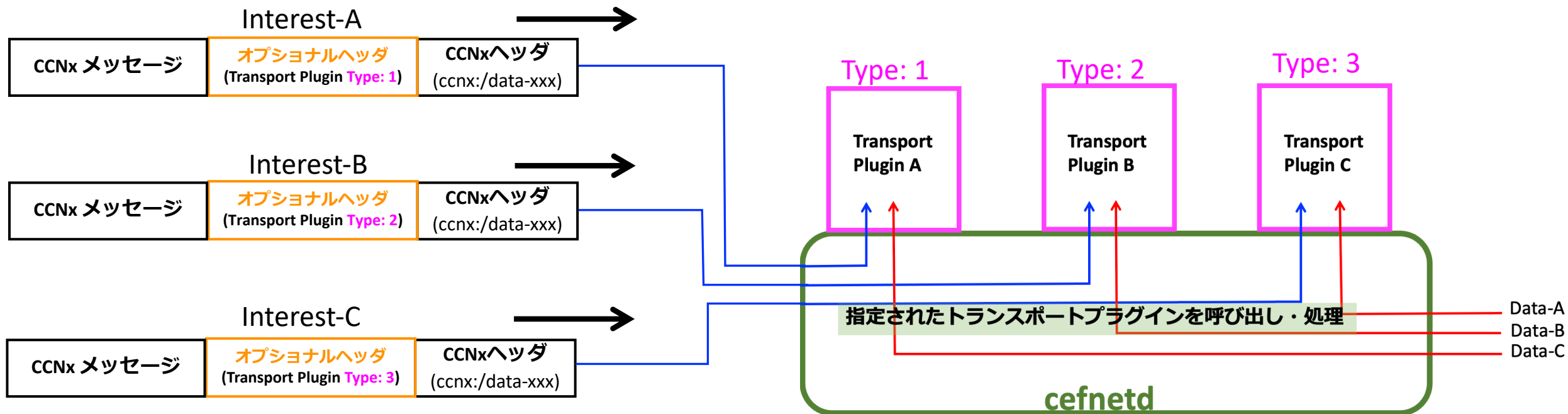
- **Cefnetd**からコールバック関数として呼び出し
- 複数のトランスポートプラグインが追加されている場合、**Cefnetd**から任意のプラグイン呼び出しが可能



# Ceforeのトランスポートプラグイン実装

## ■ 利用するトランスポートプラグインの指定方法

- オプションルヘッダを利用
  - インタレストおよびデータに対して、利用するトランスポートプラグインのType番号を指定 (Transport Variant)





# トランスポートプラグイン実装

## ■ Type番号定義

- cefore-0.10.0e/src/include/cefore/cef\_plugin.h

```
/*-----  
  Transport  
-----*/  
  
/***** TLVs for use in the CefC_T_OPT_TRANSPORT TLV *****/  
#define CefC_T_OPT_TP_NONE          0x0000      /* Invalid */  
#define CefC_T_OPT_TP_SAMPTP        0x0001      /* Default Transport */  
#define CefC_T_OPT_TP_L4C2          0x0002      /* L4C2 */  
#define CefC_T_OPT_TP_NUM           0x0003
```

## ■ Type番号（Transport Variant）指定

- 下記構造体で定義されているTransport Plugin Variantを利用して、Type番号付きインタレスト/データを生成

```
/*-----*/  
/* Parameter of the vender-specific info of the option header */  
/*-----*/  
typedef struct _CefT_HdrOrg_Params_t {  
    ~省略~  
    /***** Transport Plugin Variant *****/  
    uint16_t    tp_variant;      /* Transport Variant */  
    uint8_t     tp_len;          /* length of Transport value field */  
    uchar_t     tp_val[CefC_Max_Header_Size]; /* Transport value field */  
} CefT_HdrOrg_Params;
```

cefore-0.10.0e/src/include/cefore/cef\_frame.h

# トランスポートプラグイン実装

- Cefnetdからトランスポートプラグインをコールバック関数として呼び出すための関数定義 cefore-0.10.0e/src/include/cefore/cef\_plugin.h

```
typedef struct _CefT_Plugin_Tp {  
  
    /*** transport variant ***/  
    int                variant;  
  
    /*** callback to init the transport plugin ***/  
    int (*init)(struct _CefT_Plugin_Tp*, void*);  
  
    /*** callback to process the received cob ***/  
    int (*cob)(struct _CefT_Plugin_Tp*, CefT_Rx_Elem*);  
  
    /*** callback to process the received interest ***/  
    int (*interest)(struct _CefT_Plugin_Tp*, CefT_Rx_Elem*);  
  
    /*** callback to signal the PIT entries delete ***/  
    void (*pit)(struct _CefT_Plugin_Tp*, CefT_Rx_Elem_Sig_DelPit*);  
  
    /*** callback to post process ***/  
    void (*destroy)(struct _CefT_Plugin_Tp*);  
  
    /*** tx queue ***/  
    CefT_Rngque*    tx_que;  
    CefT_Mp_Handle  tx_que_mp;  
  
} CefT_Plugin_Tp;
```

初期化用処理

データ(Cob)受信時処理

インタレスト受信時処理

PIT削除時処理

Cefnetd終了時処理

- 任意のトランスポートプラグインに対してコールバック関数を定義・登録すれば、cefnetdから上記の関数を用いて呼び出し可能
  - 以降、サンプルトランスポートプラグインを例に見てみよう

## 2. サンプルトランスポートプラグイン (samptp)

---

# サンプルトランスポートプラグイン (samptp)

## ■ デフォルトでCeforeに内蔵

- Type番号: 1

cefore-0.10.0e/

```
/*-----  
Transport  
-----*/  
/* ***/  
#define CefC_T_OPT_TP_NONE 0x0000 /* Invalid */  
#define CefC_T_OPT_TP_SAMPTP 0x0001 /* Default Transport */  
#define CefC_T_OPT_TP_L4C2 0x0002 /* L4C2 */  
#define CefC_T_OPT_TP_NUM 0x0003
```

## ■ samptp用のコールバック関数定義

cefore-0.10.0e/src/include/cefore/cef\_plugin\_com.h

```
/*-----  
Transport Plugin  
-----*/  
/*-----  
Default Transport  
-----*/  
int  
cef_plugin_samptp_init (  
    CefT_Plugin_Tp* tp, /* Transport Plugin Handle */  
    void* arg_ptr /* Input argument block */  
);  
  
int  
cef_plugin_samptp_cob (  
    CefT_Plugin_Tp* tp, /* Transport Plugin Handle */  
    CefT_Rx_Elem* rx_elem  
);  
  
int  
cef_plugin_samptp_interest (  
    CefT_Plugin_Tp* tp, /* Transport Plugin Handle */  
    CefT_Rx_Elem* rx_elem  
);  
  
void  
cef_plugin_samptp_delpit (  
    CefT_Plugin_Tp* tp, /* Transport Plugin Handle */  
    CefT_Rx_Elem_Sig_DelPit* info  
);  
  
void  
cef_plugin_samptp_destroy (  
    CefT_Plugin_Tp* tp /* Transport Plugin Handle */  
);
```

初期化用処理

データ(Cob)受信時処理

インタレスト受信時処理

PIT削除時処理

Cefnetd終了時処理

cefore-0.10.0e/src/include/cefore/cef\_plugin\_com.h

# サンプルトランスポートプラグイン (samptp)

## ■ samptp用のコールバック関数の処理

➤ cefore-0.10.0e/src/plugin/transport/samptp/cef\_plugin\_samptp.c

```
/*-----  
Makefile Callback for incoming interest process  
-----*/  
int cef_plugin_samptp_interest (  
    CefT_Plugin_Tp* tp, /* Transport Plugin Handle */  
    CefT_Rx_Elem* rx_elem /* Rx Element (受信要素) */  
) {  
    CefT_Tx_Elem* tx_elem;  
    int i;  
  
    /* Updates statistics */  
    m_stat_int_rx++;  
  
    /* Creates the forward object */  
    tx_elem = (CefT_Tx_Elem*) cef_mpool_alloc (tp->tx_queue_mp);  
    tx_elem->type = CefC_Elem_Type_Interest;  
    tx_elem->msg_len = rx_elem->msg_len;  
    tx_elem->faceid_num = rx_elem->out_faceid_num;  
  
    for (i = 0 ; i < rx_elem->out_faceid_num ; i++) {  
        tx_elem->faceids[i] = rx_elem->out_faceids[i];  
    }  
    memcpy (tx_elem->msg, rx_elem->msg, rx_elem->msg_len);  
  
    /* Pushes the forward object to tx buffer */  
    i = cef_rnqueue_push (tp->tx_queue, tx_elem);  
  
    if (i < 1) {  
        cef_mpool_free (tp->tx_queue_mp, tx_elem);  
    }  
  
    /* Updates statistics */  
    m_stat_int_tx += rx_elem->out_faceid_num;  
  
    return (CefC_Pi_Interest_NoSend);  
}
```

インタレスト受信時処理:  
- 受信/送信インタレストの統計値を記録  
(データ受信時処理も同様)

Cefnetd終了時処理: 統計値を出力

```
/*-----  
Callback for post process  
-----*/  
void cef_plugin_samptp_destroy (  
    CefT_Plugin_Tp* tp /* Transport Plugin Handle */  
) {  
    /* Outputs statistics */  
    if (m_stat_output_f) {  
        fprintf (stderr, "[SAMPTP STATISTICS]\n");  
        fprintf (stderr, " Rx Interests : " FMTU64 "\n", m_stat_int_rx);  
        fprintf (stderr, " Tx Interests : " FMTU64 "\n", m_stat_int_tx);  
        fprintf (stderr, " Rx Cobs : " FMTU64 "\n", m_stat_cob_rx);  
        fprintf (stderr, " Tx Cobs : " FMTU64 "\n", m_stat_cob_tx);  
    }  
  
    return;  
}
```

# サンプルトランスポートプラグイン (samptp)

## ■ samptp用のコールバック関数の登録

```
/*-----
      Inits Transport Plugin
-----*/

int
cef_tp_plugin_init (
    CefT_Plugin_Tp**    tp,           /* Transport Plugin Handle */
    CefT_Rngque*         tx_que,       /* TX ring buffer */
    CefT_Mp_Handle       tx_que_mp,    /* Memory Pool for CefT_Tx_Elem */
    void*                arg_ptr       /* Input argument block */
) {
    ~省略~

    /*-----
      Registration the callback functions
    -----*/

    if (strcmp (samptp_param, "yes") == 0) {
        work[CefC_T_OPT_TP_SAMPTP].variant = CefC_T_OPT_TP_SAMPTP;
        work[CefC_T_OPT_TP_SAMPTP].tx_que = tx_que;
        work[CefC_T_OPT_TP_SAMPTP].tx_que_mp = tx_que_mp;
        work[CefC_T_OPT_TP_SAMPTP].init = cef_plugin_samptp_init;
        work[CefC_T_OPT_TP_SAMPTP].cob = cef_plugin_samptp_cob;
        work[CefC_T_OPT_TP_SAMPTP].interest = cef_plugin_samptp_interest;
        work[CefC_T_OPT_TP_SAMPTP].pit = cef_plugin_samptp_delpit;
        work[CefC_T_OPT_TP_SAMPTP].destroy = cef_plugin_samptp_destroy;
    }
    ~省略~
}
```

cefore-0.10.0e/src/plugin/cef\_tp\_plugin.c

← samptp用コールバック関数の登録



# サンプルトランスポートプラグイン (samptp)

## ■ パラメータの指定

- /usr/local/cefore/plugin.conf. に指定
- 書式

```
[TAG]
param1=value
param2=value1,value2
```

```
#
# Plugin Configurations
#
#
# General Configurations
#
# log: Outputs log
[COMMON]
log=no
#
# Transport Plugin Configurations
#
[TRANSPORT]
samptp=no
```

デフォルトの/usr/local/cefore/plugin.conf

## ■ パラメータへのアクセス

- 実装されている以下の関数を利用可能

```
/*-----
Gets values of specified tag and parameters
-----*/
CefT_List* /* listed parameters */
cef_plugin_parameter_value_get (
    const char* tag, /* tag */
    const char* parameter /* parameter */
);
/*-----
Gets the size of the specified list
-----*/
int /* size of the specified list */
cef_plugin_list_size (
    CefT_List* list_ptr /* list pointer */
);
/*-----
Gets the stored data which is listed in the specified position
-----*/
void* /* pointer to the data which is */
/* listed in the specified position */
cef_plugin_list_access (
    CefT_List* list_ptr, /* list pointer */
    int pos_inde /* position in the list to access */
);
/*-----
Gets informations of specified tag
-----*/
CefT_Plugin_Tag* /* tag informations */
cef_plugin_tag_get (
    const char* tag /* tag */
);
```

cefore-0.10.0e/src/plugin/cef\_plugin.c

# サンプルトランスポートプラグイン (samptp)

## ■ パラメータへのアクセス

- cefore-0.10.0e/src/plugin/cef\_tp\_plugin.c の cef\_tp\_plugin\_init関数

```
/*-----
Registration the callback functions
-----*/
#ifdef CefC_Plugin_Samptp
{
    CefT_List* samptp_lp    = NULL;
    char* samptp_param      = NULL;

    /***** Sample Transport *****/
    samptp_lp = cef_plugin_parameter_value_get ("TRANSPORT", "samptp");

    if (samptp_lp) {
        samptp_param = (char*) cef_plugin_list_access (samptp_lp, 0);

        if (strcmp (samptp_param, "yes") == 0) {
            work[CefC_T_OPT_TP_SAMPTP].variant      = CefC_T_OPT_TP_SAMPTP;
            work[CefC_T_OPT_TP_SAMPTP].tx_que       = tx_que;
            work[CefC_T_OPT_TP_SAMPTP].tx_que_mp    = tx_que_mp;
            work[CefC_T_OPT_TP_SAMPTP].init         = cef_plugin_samptp_init;
            work[CefC_T_OPT_TP_SAMPTP].cob         = cef_plugin_samptp_cob;
            work[CefC_T_OPT_TP_SAMPTP].interest     = cef_plugin_samptp_interest;
            work[CefC_T_OPT_TP_SAMPTP].pit         = cef_plugin_samptp_delpit;
            work[CefC_T_OPT_TP_SAMPTP].destroy     = cef_plugin_samptp_destroy;
        }
    }
}
#endif // CefC_Plugin_Samptp
```

～省略～

plugin.confにて、

**[TRANSPORT]**  
**samptp=yes**

と書かれていれば、  
samptpプラグインの  
登録処理を行う



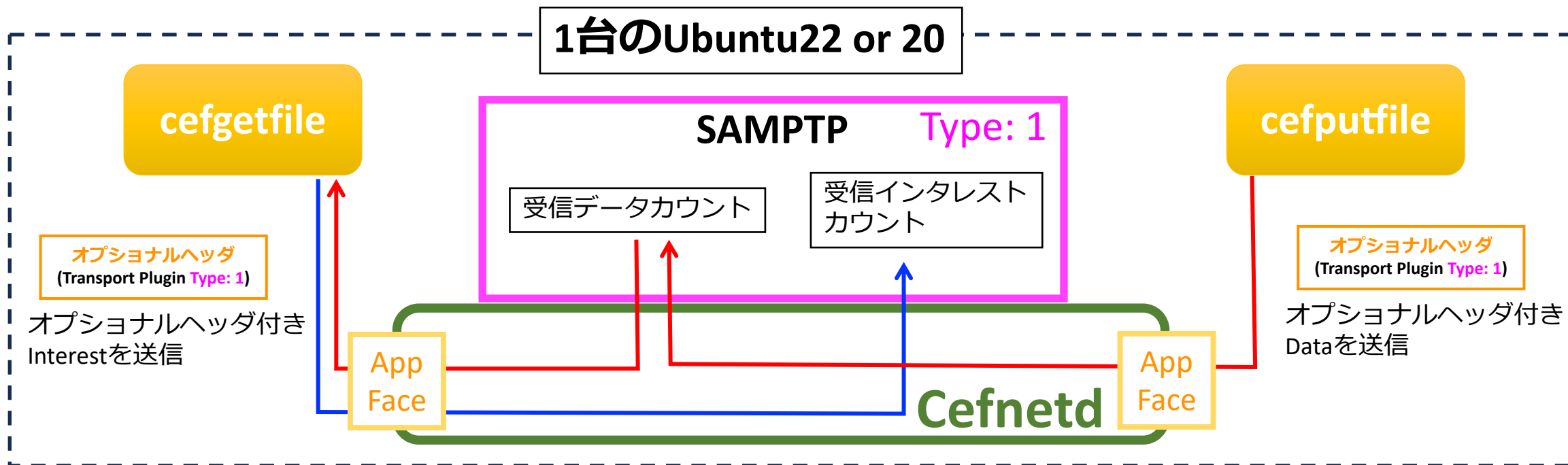
# samtpを実際に使ってみよう

## ■ 手順：

1. /user/local/cefore/plugin.confを修正
2. cefputfileとcefgetfileを改変
3. samtpを有効化し、再ビルド・再インストール

## ■ ゴール：samtpを用いて、cefnetd終了後に下記の統計値を出力させる

- 受信/送信インタレスト合計数、受信/送信データ(Cob)数



# samptpを実際に使ってみよう

- 手順 1 : /user/local/cefore/plugin.confを下記のように修正

```
#
# Plugin Configurations
#
# General Configurations
#
# log: Outputs log
[COMMON]
log=no
#
# Transport Plugin Configurations
#
[TRANSPORT]
samtp=yes
```

noからyesに変更

```
[SAMPTP]
stat=yes
```

統計値を出力するために追記

# samptpを実際に使ってみよう

## ■ 手順 2 : cefgetfileとcefputfileを改変

```
if (nsg_flag) {
    Cef_Int_Symbolic(params);
    opt.lifetime = 4000;
} else {
    Cef_Int_Regular(params);
    opt.lifetime = CefC_Default_LifetimeSec * 1000;
    params.chunk_num = 0;
    params.chunk_num_f = 1;
}
```

```
opt.org.tp_variant = CefC_T_OPT_TP_SAMPTP;
opt.org.tp_val[0] = 0x01;
opt.org.tp_val[1] = 0x12;
opt.org.tp_val[2] = 0x23;
opt.org.tp_val[3] = 0x34;
opt.org.tp_val[4] = 0xA1;
opt.org.tp_val[5] = 0xB2;
opt.org.tp_val[6] = 0xC3;
opt.org.tp_val[7] = 0xD4;
opt.org.tp_len = 8;
```

オプションヘッダ  
を指定

cefore-0.10.0e/tools/cefgetfile/cefgetfile.c

```
if (res < 0) {
    fprintf (stdout, "ERROR: Invalid URI is specified.\n");
    exit (1);
}
fprintf (stdout, "OK\n");
```

```
params.name_len = res;
params.chunk_num_f = 1;
```

```
opt.org.tp_variant = CefC_T_OPT_TP_SAMPTP;
opt.org.tp_val[0] = 0x51;
opt.org.tp_val[1] = 0x62;
opt.org.tp_val[2] = 0x73;
opt.org.tp_val[3] = 0x84;
opt.org.tp_val[4] = 0x11;
opt.org.tp_val[5] = 0x22;
opt.org.tp_val[6] = 0x33;
opt.org.tp_val[7] = 0x44;
opt.org.tp_len = 8;
```

オプションヘッダ  
を指定

cefore-0.10.0e/tools/cefputfile/cefputfile.c

# samptpを実際に使ってみよう

## ■ 手順 3 : samptpを有効化し、再ビルド・再インストール

```
cefore:~/cefore-0.10.0e$ ./configure --enable-samptp --enable-cache --enable-csmgr --enable-debug
cefore:~/cefore-0.8.1$ make clean && make
cefore:~/cefore-0.8.1$ sudo make install
```

サンプルトランスポートの有効化      上記オプションは任意（指定しなくてもOK!!）

## ■ 実行手順：（3つのターミナルを立ち上げて使用する場合（logが読みやすくなるため））

### 1. terminal-A

```
cefore:~/cefore-0.10.0e$ cefnetdstart
```

### 2. terminal-B

```
cefore:~/cefore-0.10.0e$ cefgetfile ccnx:/hoge -f ./outputfile -z sg
```

### 3. terminal-C

*inputfile/outputfileは任意*

```
cefore:~/cefore-0.10.0e$ cefputfile ccnx:/hoge -f ./inputfile
```

### 4. terminal-A

```
cefore:~/cefore-0.10.0e$ cefnetdstop
```

# Samptpの統計出力結果例

## ■ cefnetd終了後

- 下記は、--enable-debugを指定してビルドし、  
/usr/local/cefore/cefnetd.confにて、CEF\_LOG\_LEVEL=2, CEF\_DEBUG\_LEVEL=3  
を指定しているとき

```
cefore:~/cefore-0.10.0e$ cefnetdstop
```

```
2023-08-17 08:39:44.936 [cefctrl] DEBUG: operation is kill
```

```
2023-08-17 08:39:44.936 [cefctrl] INFO: [client] Config directory is /usr/local/cefore
```

```
2023-08-17 08:39:44.936 [cefctrl] INFO: [client] Local Socket Name is /tmp/cef_9896.0
```

```
2023-08-17 08:39:44.936 [cefctrl] INFO: [client] Listen Port is 9896
```

```
2023-08-17 08:39:44.936 [cefnetd] DEBUG: [face] Creation the new Face#18 (FD#10) for local peer
```

```
[SAMPTP STATISTICS]
```

```
RX Interests : 4
```

```
Tx Interests : 0
```

```
Rx Cobs      : 435
```

```
Tx Cobs      : 0
```

注) アプリケーション (App Face) に転送されたInterest/Cobは  
カウントされない。  
(ネットワークに転送した時にカウントされる)

# 3. 新たなトランスポートプラグインの追加方法

---

# ディレクトリ構成

- “**xxntp**”は追加するトランスポートプラグイン名（自由に置き換え可能）

- **cefore-0.10.0e**

- **configure.ac** (→ 手順 2 ), **Makefile.am** (→ 手順 1 )
- **src**
  - **cefnetd, conpubd, csmgrd, dlplugin, lib**
  - **include**
    - **cefore**
      - **cef\_plugin.h** (→ 手順 5 ), **cef\_plugin\_xxntp.h** (→ 手順 3 ), **Makefile.am** (→ 手順 3 ),
  - **plugin**
    - **cef\_plugin.c, cef\_tp\_plugin.c, Makefile.am** (→ 手順 4 )
    - **transport**
      - **xxntp** (→ 手順 4 )
      - **cef\_plugin\_xxntp.c** (→ 手順 4 )
- **tools**
  - **cefgetfile**
    - **cefgetfile.c**
  - **cefputfile**
    - **cefputfile.c**

※ 青文字は変更を行う必要があるファイル名

※ 赤文字は追加するディレクトリ or ファイル名

# 手順のまとめ

1. configureオプションの変更
2. 1. で追加したconfigureオプションのチェック処理を追加
3. ヘッダファイル追加
4. ソースファイルを追加
5. パラメータ指定
6. プラグイン登録
7. cefgetfileとcefputfileのopt.org.tp\_variantを変更
8. ビルド

samptpと同様の処理を行うプラグインを追加してみよう。  
(できた人は、name-prefix毎に任意の統計を集計するなど、  
自身の処理を加えてみよう)



# 1. configureオプションの変更

- cefore-0.10.0e/Makefile.amに対し、configure実行時にプラグインを有効/無効にするためのオプションチェックを追加
  - “**--enable-xxtp**” を “DISTCHECK\_CONFIGURE\_FLAGS”の最後に追加

```
# set distcheck configure flag
DISTCHECK_CONFIGURE_FLAGS=--enable-debug --enable-csmgr --enable-cache --enable-samtp --enable-cefping -
-enable-conpub --enable-interest-return --enable-xxtp

EXTRA_DIST=apps doc LICENSE Readme.md

# load sub directory
SUBDIRS = src tools utils config
```

**追記箇所**

## 2. (1.)で追加したconfigureオプションのチェック処理を追加

- cefore-0.10.0e/configure.acに対し、下記のチェック処理を追加  
(check samptpのチェック処理部分をコピー・追加して変更すると楽かも)

```
dn1
dn1  check xxxtp
dn1
AC_ARG_ENABLE(
  xxxtp,
  AS_HELP_STRING([--enable-xxxtp], [xxx transport (default no)]),
  [enable_xxxtp=yes],
  [enable_xxxtp=no]
)
AM_CONDITIONAL(XXXTP_ENABLE, test x"${enable_xxxtp}" = xyes)
```

注) 大文字

# 3. ヘッダファイル追加 (1/2)

- cefore-0.10.0e/src/include/ceforeに以下の“cef\_plugin\_xxntp.h”を追加

```
#ifndef __CEF_PLUGIN_XXNTP_HEADER__
#define __CEF_PLUGIN_XXNTP_HEADER__

/*=====
Transport Plugin
=====*/

/*=====
XXXX Transport
=====*/

int
cef_plugin_xxntp_init (
    CefT_Plugin_Tp* tp,      /* Transport Plugin Handle */
    void* arg_ptr           /* Input argument block */
);

int
cef_plugin_xxntp_cob (
    CefT_Plugin_Tp* tp,      /* Transport Plugin Handle */
    CefT_Rx_Elem* rx_elem
);

int
cef_plugin_xxntp_interest (
    CefT_Plugin_Tp* tp,      /* Transport Plugin Handle */
    CefT_Rx_Elem* rx_elem
);

void
cef_plugin_xxntp_delpit (
    CefT_Plugin_Tp* tp,      /* Transport Plugin Handle */
    CefT_Rx_Elem_Sig_DelPit* info
);

void
cef_plugin_xxntp_destroy (
    CefT_Plugin_Tp* tp      /* Transport Plugin Handle */
);

#endif // __CEF_PLUGIN_XXNTP_HEADER__
```

※ cef\_plugin\_xxntp.hは、gitからdownload可能  
(追加するトランスポートプラグイン名に応じて、  
ソースファイル名およびソース内容を適切に変更する  
こと (“xxntp”, “XXXXTP”を適切に変更する))

(追加するトランスポートプラグイン名がxxntpそのま  
まの場合、変更なしでOK)

# 3. ヘッダファイル追加 (2/2) XXXX

- 追加したヘッダファイルを読み込むか否かは、追加プラグインをビルドするか否かで切り分けられるように、cefore-0.10.0e/src/include/cefore/Makefile.amを下記のように変更

```
# specify the include file
CEF_HEADER=cef_client.h cef_csmgr.h cef_csmgr_stat.h cef_ccninfo.h \
    cef_define.h cef_face.h cef_fib.h cef_frame.h cef_hash.h cef_mpool.h \
    cef_pit.h cef_log.h cef_print.h cef_rngque.h cef_plugin.h cef_plugin_com.h cef_valid.h \
    cef_mem_cache.h

if CONPUB_ENABLE
CEF_HEADER+=cef_conpub.h
endif # CONPUB_ENABLE

if XXXTP_ENABLE
CEF_HEADER+=cef_plugin_xxntp.h ← 追記
endif #XXXTP_ENABLE

includedir=$(CEFORE_DIR_PATH)/include/cefore
include_HEADERS=$(CEF_HEADER)
```

※ “XXXTP\_ENABLE”は、「2. (1.)で追加したconfigureオプションのチェック処理を追加」で指定した変数

## 4. ソースファイル追加

- コールバック関数などを記述したソースファイル(cef\_plugin\_xxtp.c)をcefore-0.10.0e/src/plugin/transport/xxtpの下に追加(xxtpというディレクトリを作成し、その下に配置)
  - cef\_plugin\_samtp.cを改変したcef\_plugin\_xxtp.cは、gitからdownload可能
- cefore-0.10.0e/src/plugin/Makefile.amにビルド対象となるソースファイルおよびマクロを追加

```
AM_CFLAGS=-I$(top_srcdir)/src/include -Wall -O2 -fPIC
AM_CSOURCES=cef_plugin.c cef_tp_plugin.c
AUTOMAKE_OPTIONS=subdir-objects

if SAMPTP_ENABLE
AM_CFLAGS+=-DCefC_Plugin_Samtp
AM_CSOURCES+=transport/samtp/cef_plugin_samtp.c
endif # SAMPTP_ENABLE

if XXXTP_ENABLE
AM_CFLAGS+=-DCefC_Plugin_Xxtp
AM_CSOURCES+=transport/xxtp/cef_plugin_xxtp.c
endif # XXXTP_ENABLE

lib_LIBRARIES=libcef_plugin.a
libcef_plugin_a_CFLAGS=$(AM_CFLAGS)
libcef_plugin_a_SOURCES=$(AM_CSOURCES)
```

最初の1文字目は  
大文字

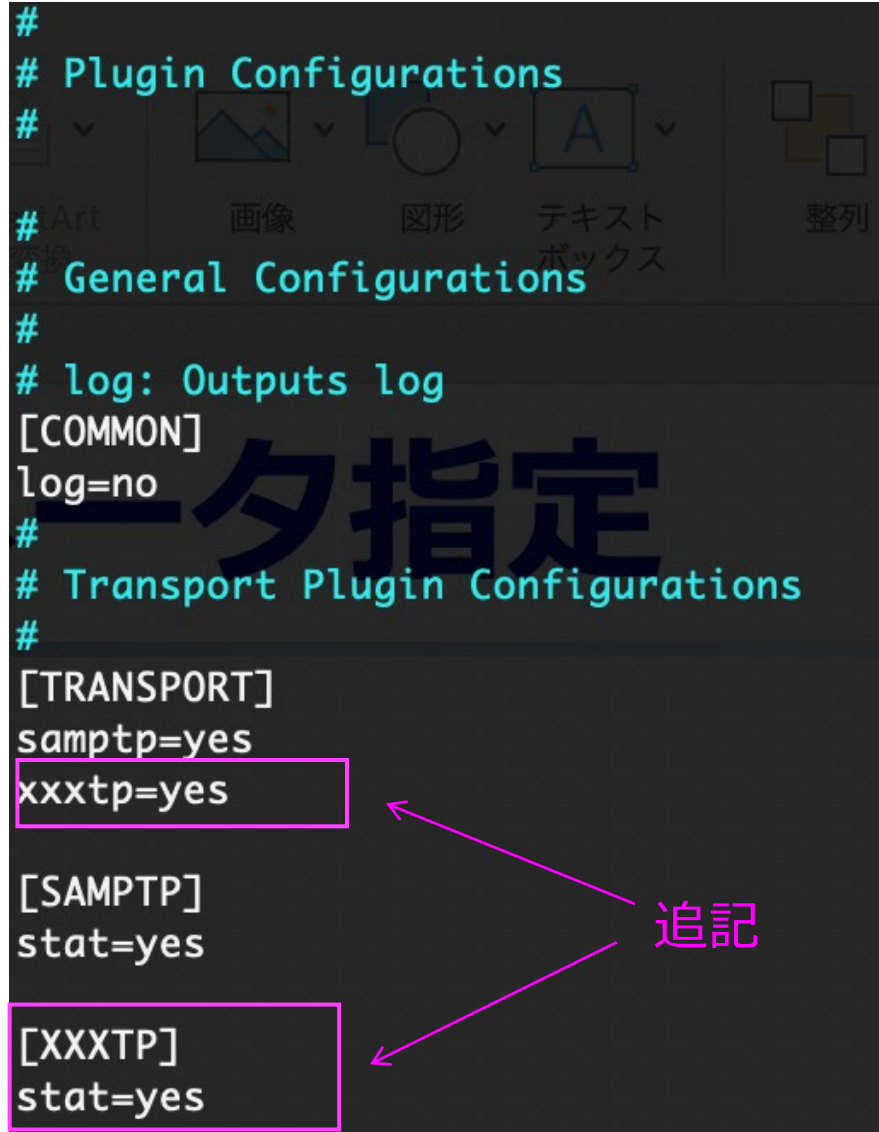
# 5. パラメータ指定

- /usr/local/cefore/plugin.conにパラメータを追記・指定

```
#
# Plugin Configurations
#
# Art 画像 図形 テキスト 整列
#   描き ボックス
# General Configurations
#
# log: Outputs log
[COMMON]
log=no
#
# Transport Plugin Configurations
#
[TRANSPORT]
samptp=yes
xxxtp=yes

[SAMPTP]
stat=yes

[XXXTP]
stat=yes
```





## 6. プラグイン登録

- 追加プラグインのType番号を  
cefore-0.10.0e/src/include/cefore/cef\_plugin.hに定義
  - CefC\_T\_OPT\_TP\_NUMの値は、(追加プラグインのType番号 + 1) とする

```
/*-----  
-----Transport-----*/  
-----*/  
  
/***** TLVs for use in the CefC_T_OPT_TRANSPORT TLV *****/  
#define CefC_T_OPT_TP_NONE      0x0000 /* Invalid */  
#define CefC_T_OPT_TP_SAMPTP    0x0001 /* Default Transport */  
#define CefC_T_OPT_TP_L4C2      0x0002 /* L4C2 */  
#define CefC_T_OPT_TP_XXTP      0x0003 /* Xxx Transport */  
#define CefC_T_OPT_TP_NUM      0x0004
```

追記 & 変更

# 7. cefgetfileとcefputfileのopt.org.tp\_variantを変更

- xxxtpを使う場合、tp\_variantを適切に変更する

```
if (nsg_flag) {
    Cef_Int_Symbolic(params);
    opt.lifetime = 4000;
} else {
    Cef_Int_Regular(params);
    opt.lifetime = CefC_Default_LifetimeSec * 1000;
    params.chunk_num = 0;
    params.chunk_num_f = 1;
}
```

```
opt.org.tp_variant = CefC_T_OPT_TP_SAMPTP;
opt.org.tp_val[0] = 0x01;
opt.org.tp_val[1] = 0x12;
opt.org.tp_val[2] = 0x23;
opt.org.tp_val[3] = 0x34;
opt.org.tp_val[4] = 0xA1;
opt.org.tp_val[5] = 0xB2;
opt.org.tp_val[6] = 0xC3;
opt.org.tp_val[7] = 0xD4;
opt.org.tp_len = 8;
```

CefC\_T\_OPT\_TP\_XXXTP  
に変更する

オプションヘッダ  
を指定

cefore-0.10.0e/tools/cefgetfile/cefgetfile.c

※ samptpのType番号を指定したままであれば、  
samptpが有効な場合、samptpがコールバックされる

```
if (res < 0) {
    fprintf (stdout, "ERROR: Invalid URI is specified.\n");
    exit (1);
}
fprintf (stdout, "OK\n");
```

```
params.name_len      = res;
params.chunk_num_f   = 1;
```

```
opt.org.tp_variant      = CefC_T_OPT_TP_SAMPTP;
opt.org.tp_val[0]       = 0x51;
opt.org.tp_val[1]       = 0x62;
opt.org.tp_val[2]       = 0x73;
opt.org.tp_val[3]       = 0x84;
opt.org.tp_val[4]       = 0x11;
opt.org.tp_val[5]       = 0x22;
opt.org.tp_val[6]       = 0x33;
opt.org.tp_val[7]       = 0x44;
opt.org.tp_len          = 8;
```

CefC\_T\_OPT\_TP\_XXXTP  
に変更する

オプションヘッダ  
を指定

cefore-0.10.0e/tools/cefputfile/cefputfile.c



## 8. cef\_tp\_plugin.cを改変する

- cefore-0.10.0e/src/plugin/cef\_tp\_plugin.cにxxxtpの処理を加える
  - xxxtp用のコールバック関数の登録
  - Cefnetd終了時にxxxtpの統計値出力処理を追加

- **改変したcef\_tp\_plugin.cは、gitからdownload可能**

- 追加するトランスポートプラグイン名に応じて、cef\_tp\_plugin.cに記述されている“xxxtp”, “XXXTP”, “Xxxtp”を適切に置き替えて、cef\_tp\_plugin.cを改変する

(追加するトランスポートプラグイン名がxxxtpそのままの場合、変更なしでOK)

## 9. 再ビルド

```
cefore:~/cefore-0.10.0e$ autoconf
cefore:~/cefore-0.10.0e$ automake
cefore:~/cefore-0.10.0e$ ./configure --enable-samptp --enable-xxtp --enable-debug --enable-cache
cefore:~/cefore-0.10.0e$ make clean
cefore:~/cefore-0.10.0e$ make
cefore:~/cefore-0.10.0e$ sudo make install
```

- 再ビルド後、samptpの実行手順と同様にして、xxtpを動かしてみよう。