Saurabh Aggarwal

TPF Developer – Delta Air Lines

Mar, 2017

Next Generation data Sharing concept between TPF and Open Systems

+

Alternate approach of Data transfer between TPF Labs

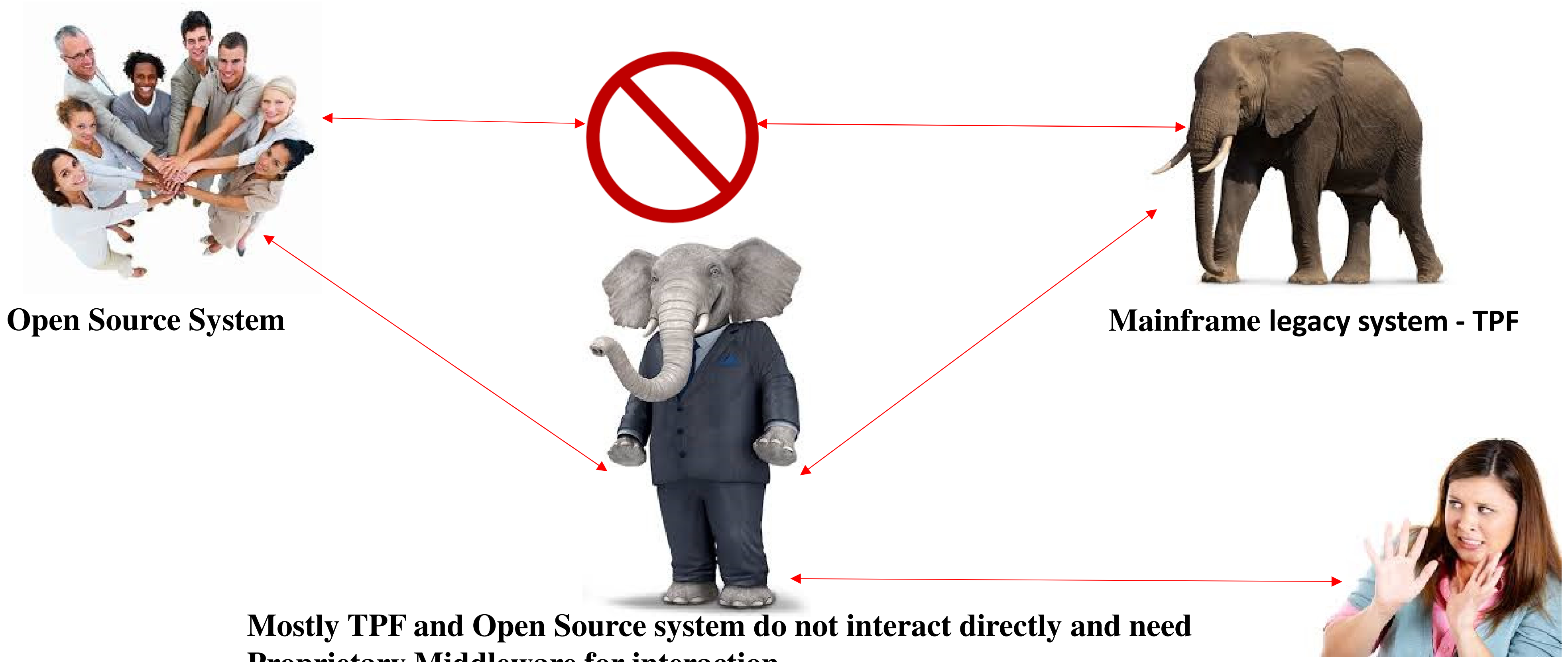
New way of TPF Report Storage and sharing

Another way to store & share TPF Tape data

Easiness in plug in TPF Data with open source Tools

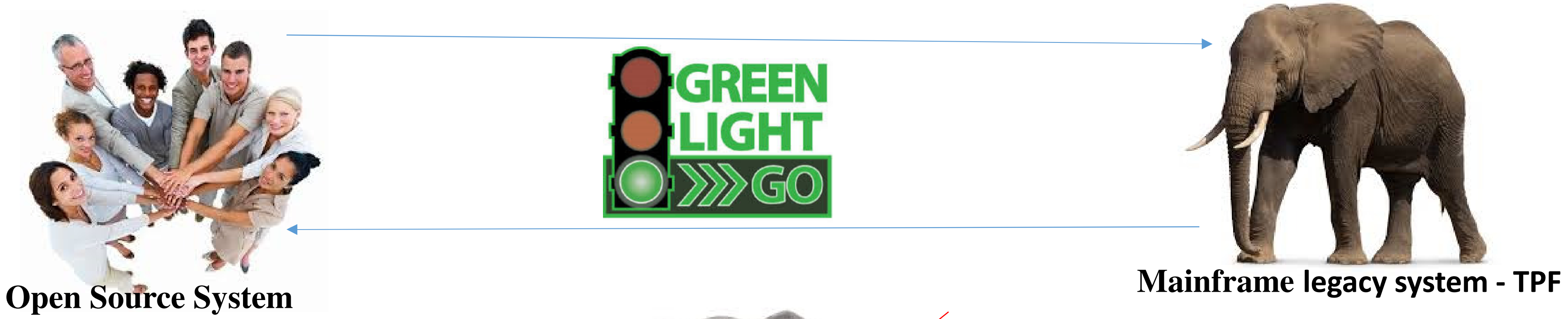
Disclaimer:

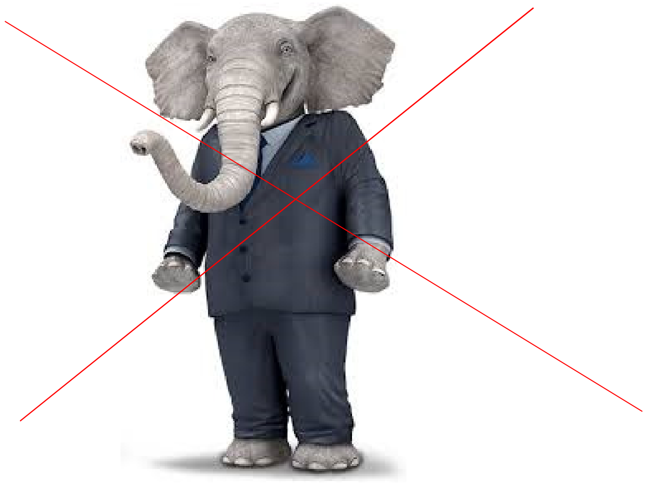
This proof of concept is to demonstrate z/TPF's ability to directly read and write data into z/TPF File System which can be accessed directly via Open Source via FTP. Security and performance of this concept will have to be evaluated in detail before it can be used in a production environment.



**Mostly TPF and Open Source system do not interact directly and need Proprietary Middleware for interaction.**

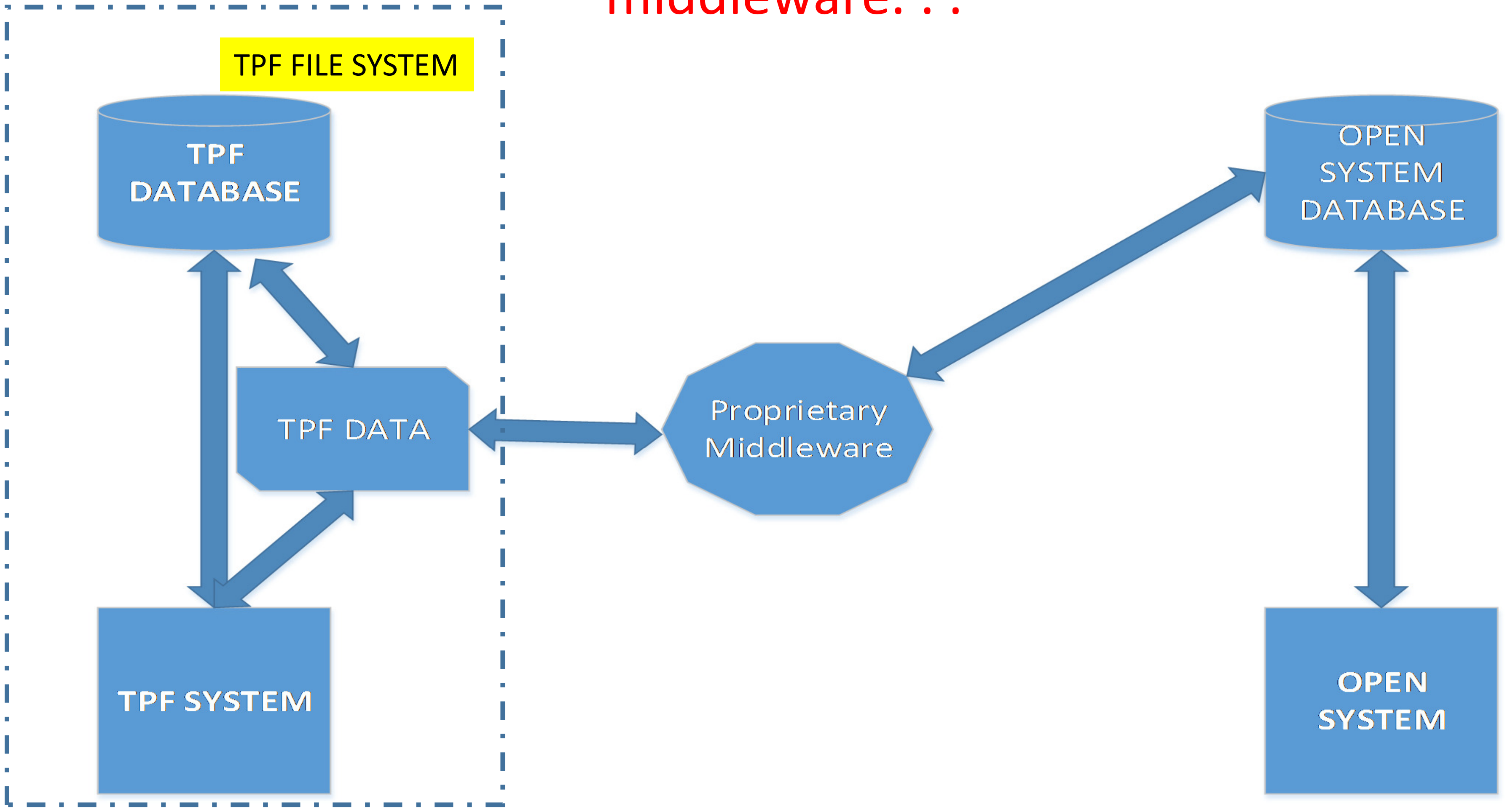
**This Proprietary Middleware is creating lots of limitation between TPF and Open source interaction.**



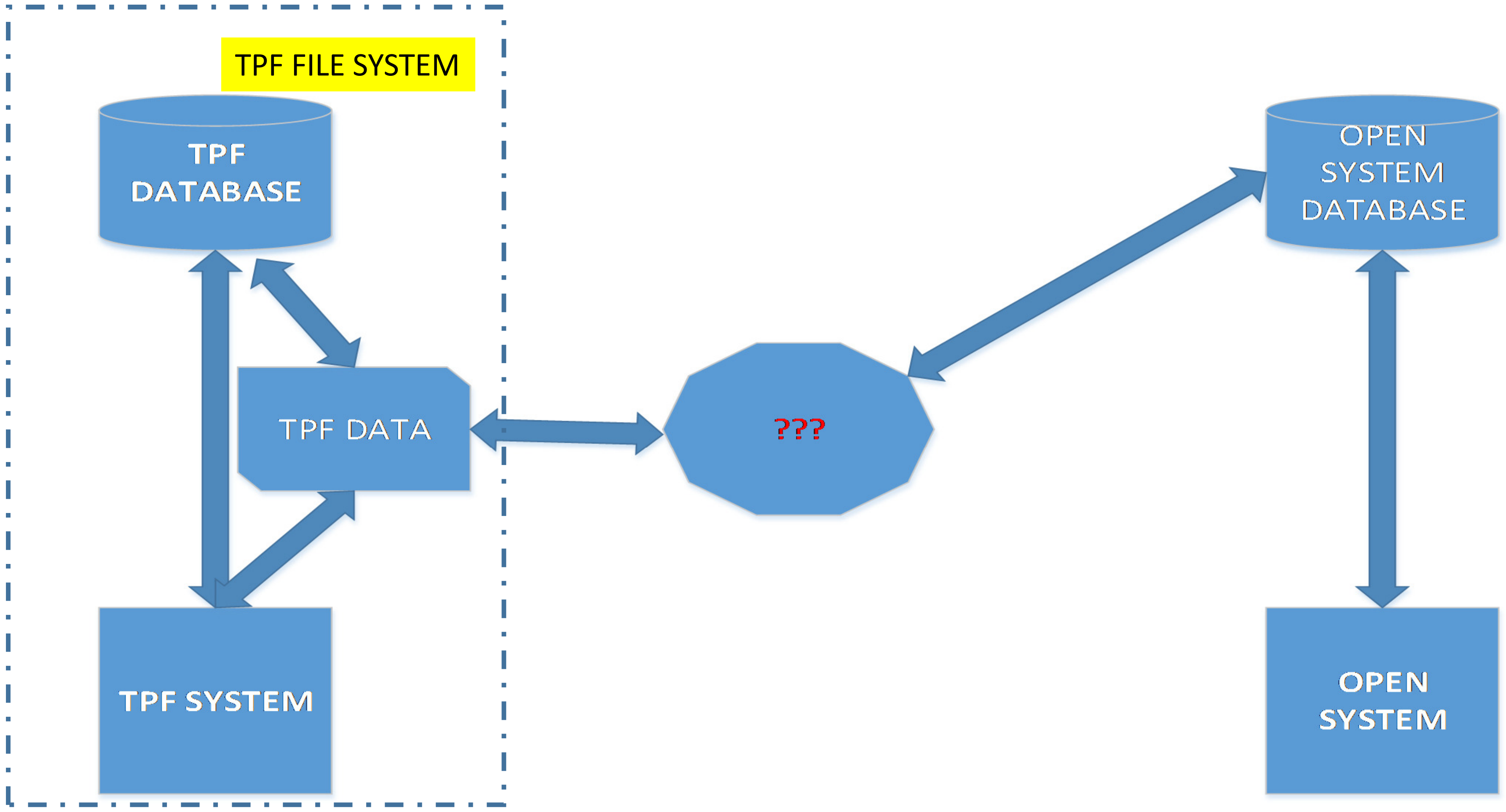


So if TPF can start interaction with open source directly – without Proprietary Middleware – TPF and open source can develop many more new possibilities together.

## In most of the scenarios TPF communicate data with open source via middleware. . .



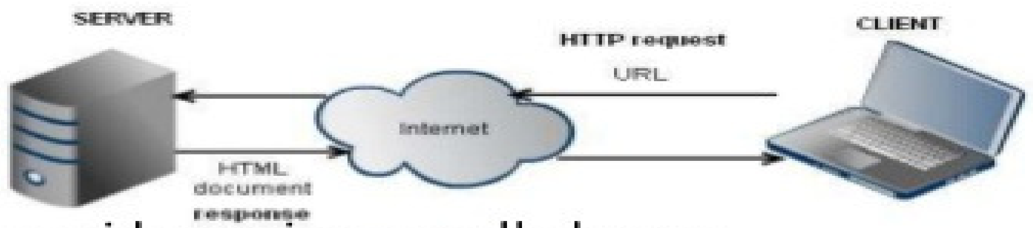
## So question is - If no middleware then what will transfer directly?



Can we transfer directly via File Transfer Protocol (FTP) ?

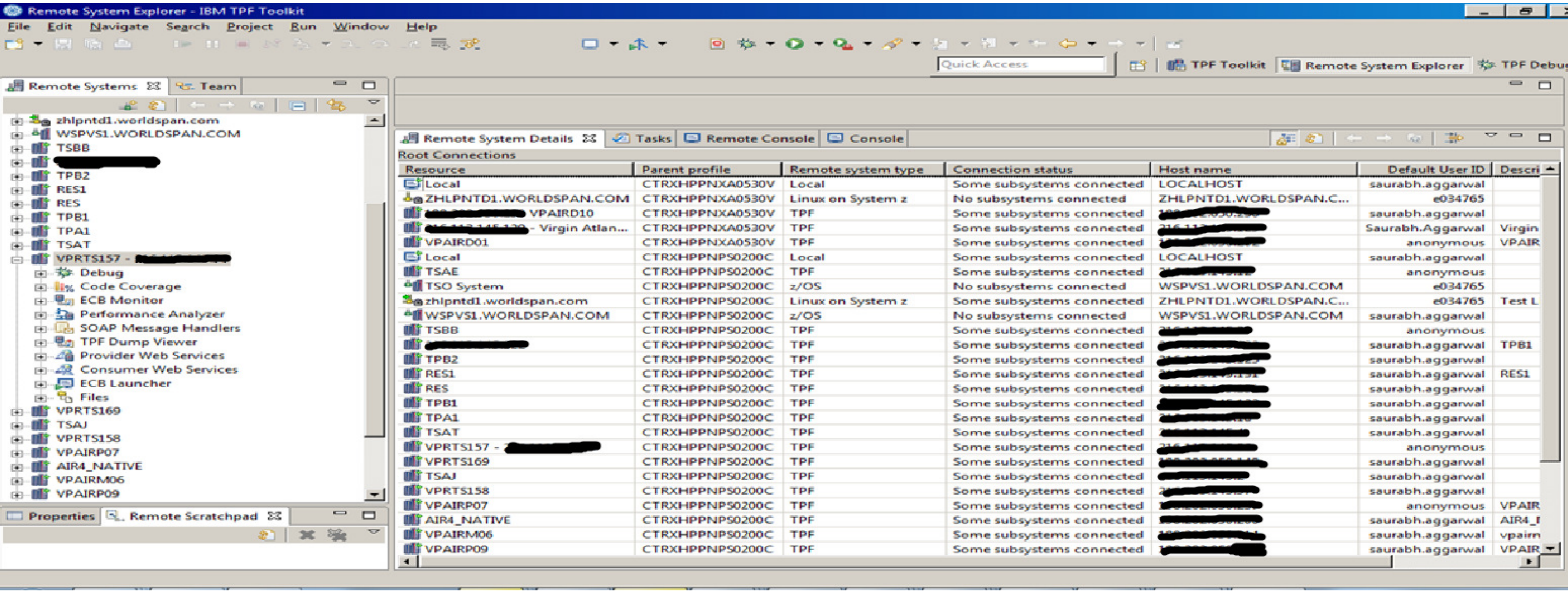
But FTP needs - Client/Server Model ? Is that possible in TPF ??

## CLIENT SERVER MODEL



- Request service, called clients and provide service are called sever
- Server is often designed to be a centralized system that serves many clients.
- Clients and servers communicate over a computer network on separate hardware
- Clients and servers exchange messages in a request response messaging pattern
- To prevent abuse and maximize uptime, the server's software limits how a client can use the server's resources.
- A denial of service attack exploits a server's obligation to process requests by bombarding it with requests incessantly

Yes - There is an area associated to each TPF Labs(Test/Prod) on z/TPF File System and they have internal and External I.P. so technically this area can be used under Client Server model and we can transfer the data via FTP.

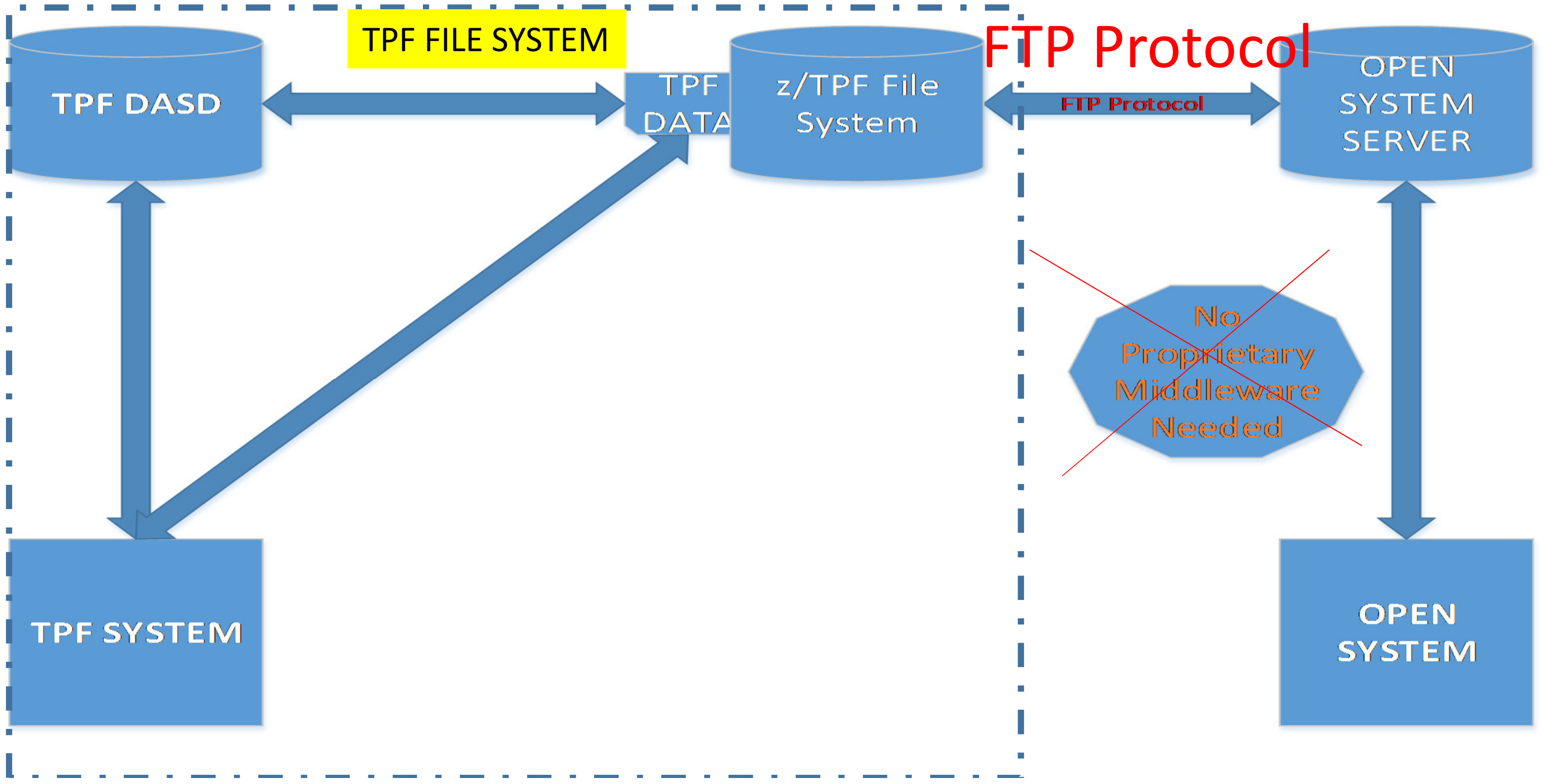


- zhlpntd1.worldspan.com
- WSPVS1.WORLDSPAN.COM
- TSBB
- [redacted]
- TPB2
- RES1
- RES
- TPB1
- TPA1
- TSAT
- VPRTS157 - [redacted]
  - Debug
  - Code Coverage
  - ECB Monitor
  - Performance Analyzer
  - SOAP Message Handlers
  - TPF Dump Viewer
  - Provider Web Services
  - Consumer Web Services
  - ECB Launcher
  - Files
- VPRTS169
- TSAJ
- VPRTS158
- VPAIRP07
- AIR4_NATIVE
- VPAIRM06
- VPAIRP09

Root Connections

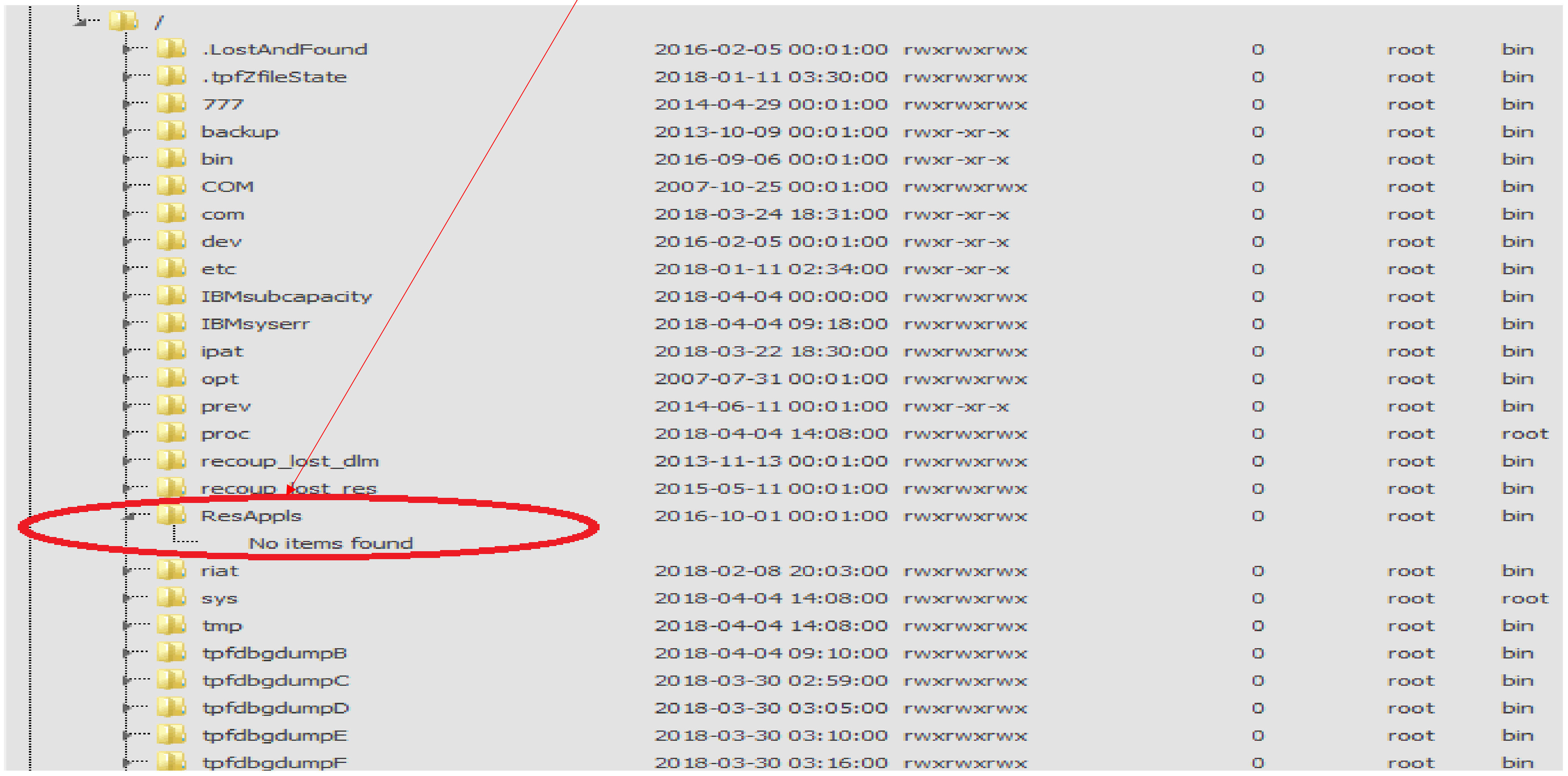
| Resource | Parent profile | Remote system type | Connection status | Host name | Default User ID | Descri |
|---|---|---|---|---|---|---|
| Local | CTRXHPPNXA0530V | Local | Some subsystems connected | LOCALHOST | saurabh.aggarwal | |
| ZHLPNTD1.WORLDSPAN.COM | CTRXHPPNXA0530V | Linux on System z | No subsystems connected | ZHLPNTD1.WORLDSPAN.C... | e034765 | |
| [redacted] VPAIRD10 | CTRXHPPNXA0530V | TPF | Some subsystems connected | [redacted] | saurabh.aggarwal | |
| [redacted] - Virgin Atlan... | CTRXHPPNXA0530V | TPF | Some subsystems connected | [redacted] | Saurabh.Aggarwal | Virgin |
| VPAIRD01 | CTRXHPPNXA0530V | TPF | Some subsystems connected | [redacted] | anonymous | VPAIR |
| Local | CTRXHPPNPS0200C | Local | Some subsystems connected | LOCALHOST | saurabh.aggarwal | |
| TSAE | CTRXHPPNPS0200C | TPF | Some subsystems connected | [redacted] | anonymous | |
| TSO System | CTRXHPPNPS0200C | z/OS | No subsystems connected | WSPVS1.WORLDSPAN.COM | e034765 | |
| zhlpntd1.worldspan.com | CTRXHPPNPS0200C | Linux on System z | Some subsystems connected | ZHLPNTD1.WORLDSPAN.C... | e034765 | Test L |
| WSPVS1.WORLDSPAN.COM | CTRXHPPNPS0200C | z/OS | No subsystems connected | WSPVS1.WORLDSPAN.COM | saurabh.aggarwal | |
| TSBB | CTRXHPPNPS0200C | TPF | Some subsystems connected | [redacted] | anonymous | |
| [redacted] | CTRXHPPNPS0200C | TPF | Some subsystems connected | [redacted] | saurabh.aggarwal | TPB1 |
| TPB2 | CTRXHPPNPS0200C | TPF | Some subsystems connected | [redacted] | saurabh.aggarwal | |
| RES1 | CTRXHPPNPS0200C | TPF | Some subsystems connected | [redacted] | saurabh.aggarwal | RES1 |
| RES | CTRXHPPNPS0200C | TPF | Some subsystems connected | [redacted] | saurabh.aggarwal | |
| TPB1 | CTRXHPPNPS0200C | TPF | Some subsystems connected | [redacted] | saurabh.aggarwal | |
| TPA1 | CTRXHPPNPS0200C | TPF | Some subsystems connected | [redacted] | saurabh.aggarwal | |
| TSAT | CTRXHPPNPS0200C | TPF | Some subsystems connected | [redacted] | saurabh.aggarwal | |
| VPRTS157 - [redacted] | CTRXHPPNPS0200C | TPF | Some subsystems connected | [redacted] | anonymous | |
| VPRTS169 | CTRXHPPNPS0200C | TPF | Some subsystems connected | [redacted] | saurabh.aggarwal | |
| TSAJ | CTRXHPPNPS0200C | TPF | Some subsystems connected | [redacted] | saurabh.aggarwal | |
| VPRTS158 | CTRXHPPNPS0200C | TPF | Some subsystems connected | [redacted] | saurabh.aggarwal | |
| VPAIRP07 | CTRXHPPNPS0200C | TPF | Some subsystems connected | [redacted] | anonymous | VPAIR |
| AIR4_NATIVE | CTRXHPPNPS0200C | TPF | Some subsystems connected | [redacted] | saurabh.aggarwal | AIR4_I |
| VPAIRM06 | CTRXHPPNPS0200C | TPF | Some subsystems connected | [redacted] | saurabh.aggarwal | vpairn |
| VPAIRP09 | CTRXHPPNPS0200C | TPF | Some subsystems connected | [redacted] | saurabh.aggarwal | VPAIR |

If we can write TPF Files directly to z/TPF File System – Then it is available to open source via FTP Protocol – And No Proprietary Middleware needed for those scenarios!!



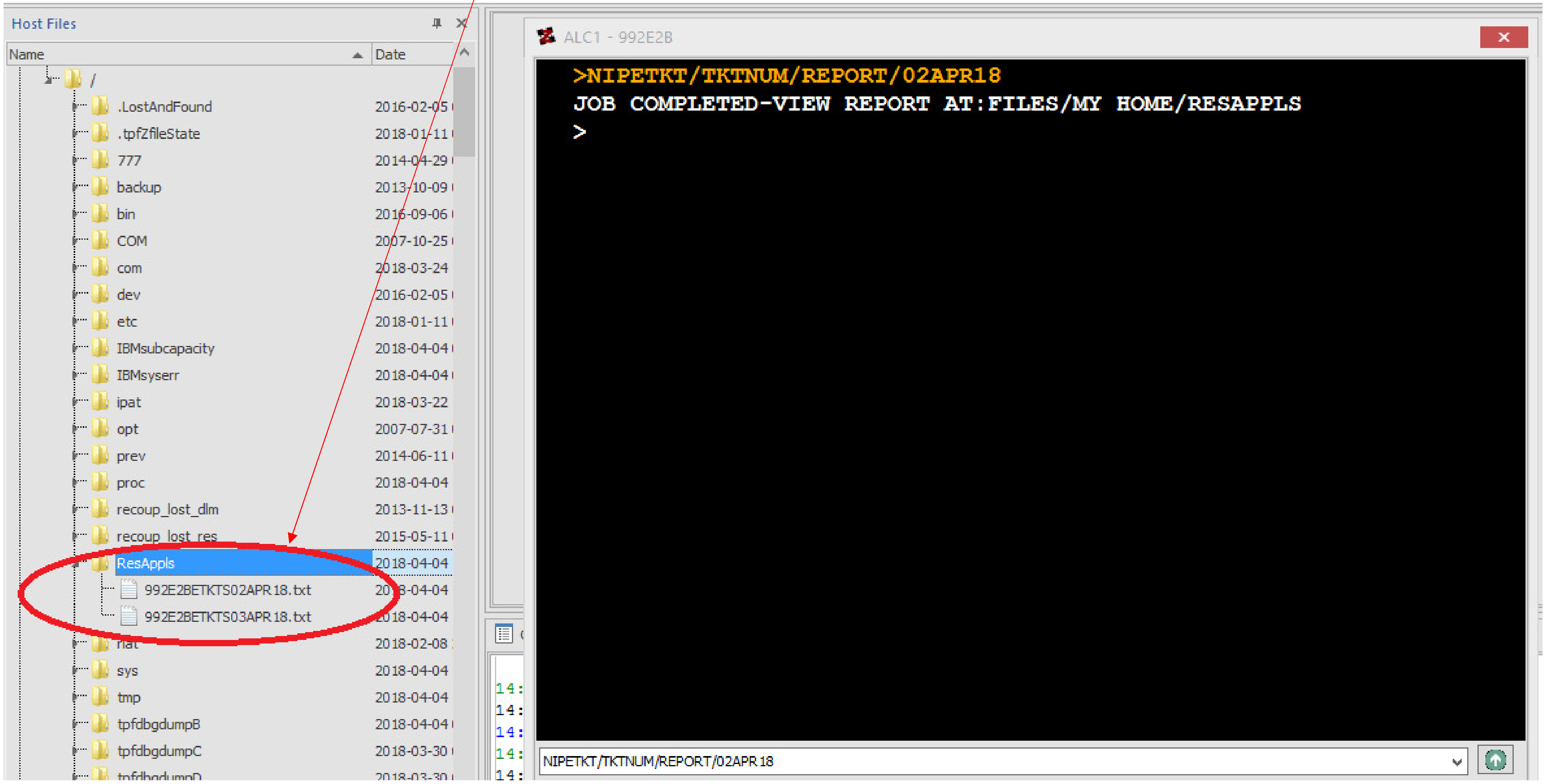
## How complex is to write TPF data in Z/TPF file system?

For our POC - We created one folder in z/TPF File System to test whether we can Write/Read/Update/Delete TPF data there.



```
/
  .LostAndFound          2016-02-05 00:01:00  rwxrwxrwx   0   root   bin
  .tpfZfileState         2018-01-11 03:30:00  rwxrwxrwx   0   root   bin
  777                    2014-04-29 00:01:00  rwxrwxrwx   0   root   bin
  backup                 2013-10-09 00:01:00  rwxr-xr-x   0   root   bin
  bin                    2016-09-06 00:01:00  rwxr-xr-x   0   root   bin
  COM                    2007-10-25 00:01:00  rwxrwxrwx   0   root   bin
  com                    2018-03-24 18:31:00  rwxr-xr-x   0   root   bin
  dev                    2016-02-05 00:01:00  rwxr-xr-x   0   root   bin
  etc                    2018-01-11 02:34:00  rwxr-xr-x   0   root   bin
  IBMsubcapacity         2018-04-04 00:00:00  rwxrwxrwx   0   root   bin
  IBMsyserr              2018-04-04 09:18:00  rwxrwxrwx   0   root   bin
  ipat                   2018-03-22 18:30:00  rwxrwxrwx   0   root   bin
  opt                    2007-07-31 00:01:00  rwxrwxrwx   0   root   bin
  prev                   2014-06-11 00:01:00  rwxr-xr-x   0   root   bin
  proc                   2018-04-04 14:08:00  rwxrwxrwx   0   root   root
  recoup_lost_dlm        2013-11-13 00:01:00  rwxrwxrwx   0   root   bin
  recoup_lost_res        2015-05-11 00:01:00  rwxrwxrwx   0   root   bin
  ResAppls               2016-10-01 00:01:00  rwxrwxrwx   0   root   bin
      No items found
  riat                   2018-02-08 20:03:00  rwxrwxrwx   0   root   bin
  sys                    2018-04-04 14:08:00  rwxrwxrwx   0   root   root
  tmp                    2018-04-04 14:08:00  rwxrwxrwx   0   root   bin
  tpfdbgdumpB            2018-04-04 09:10:00  rwxrwxrwx   0   root   bin
  tpfdbgdumpC            2018-03-30 02:59:00  rwxrwxrwx   0   root   bin
  tpfdbgdumpD            2018-03-30 03:05:00  rwxrwxrwx   0   root   bin
  tpfdbgdumpE            2018-03-30 03:10:00  rwxrwxrwx   0   root   bin
  tpfdbgdumpF            2018-03-30 03:16:00  rwxrwxrwx   0   root   bin
```

And we were able to do so just via using simple TPFC commands – No New Infrastructure needed!!!



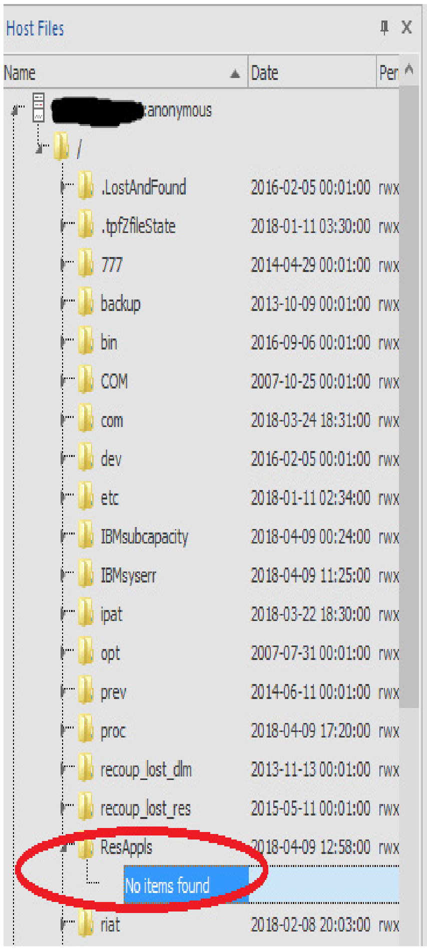
```
/
  .LostAndFound        2016-02-05
  .tpfzfileState       2018-01-11
  777                  2014-04-29
  backup               2013-10-09
  bin                  2016-09-06
  COM                  2007-10-25
  com                  2018-03-24
  dev                  2016-02-05
  etc                  2018-01-11
  IBMsubcapacity       2018-04-04
  IBMsyserr            2018-04-04
  ipat                 2018-03-22
  opt                  2007-07-31
  prev                 2014-06-11
  proc                 2018-04-04
  recoup_lost_dlm      2013-11-13
  recoup_lost_res      2015-05-11
  ResAppls             2018-04-04
    992E2BETKTS02APR18.txt   2018-04-04
    992E2BETKTS03APR18.txt   2018-04-04
  rlat                 2018-02-08
  sys                  2018-04-04
  tmp                  2018-04-04
  tpfdbgdumpB          2018-04-04
  tpfdbgdumpC          2018-03-30
  tpfdbgdumpD          2018-03-30
```

```
>NIPETKT/TKTNUM/REPORT/02APR18
JOB COMPLETED-VIEW REPORT AT:FILES/MY HOME/RESAPPLS
>
```

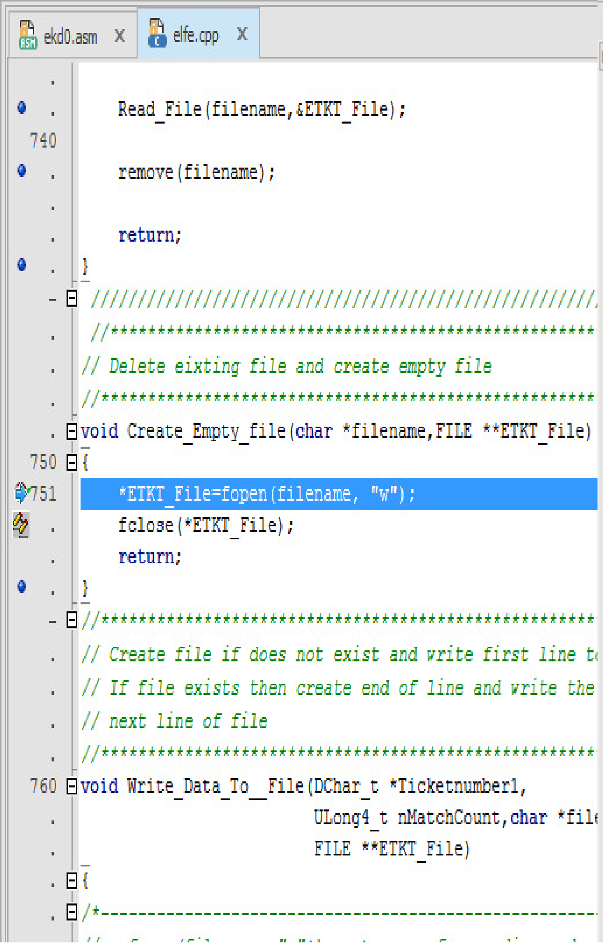
```
NIPETKT/TKTNUM/REPORT/02APR18
```

## Create Empty File on z/TPF File System:

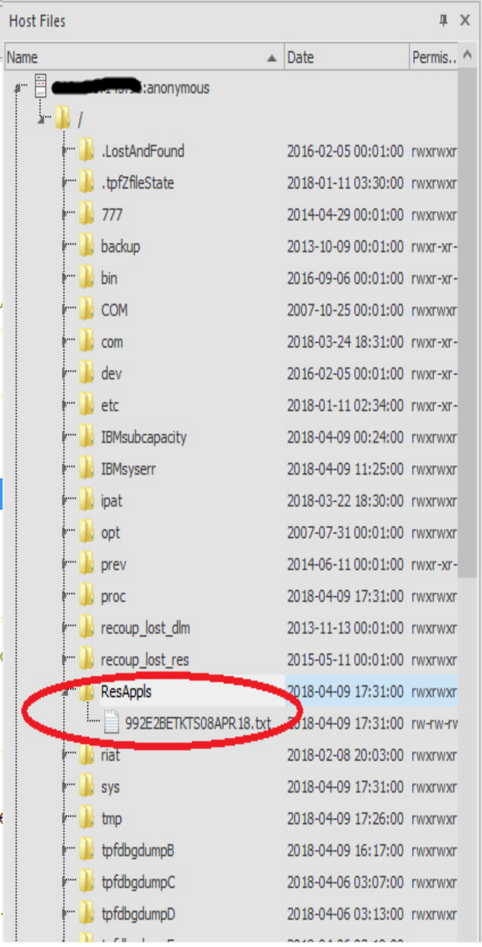
```
char          *path="/ResAppls";
char          filename[33]="/ResAppls/XXXXXXETKTSXXXXXXX.txt";
*ETKT_File=fopen(filename, "w");
fclose(*ETKT_File);
```



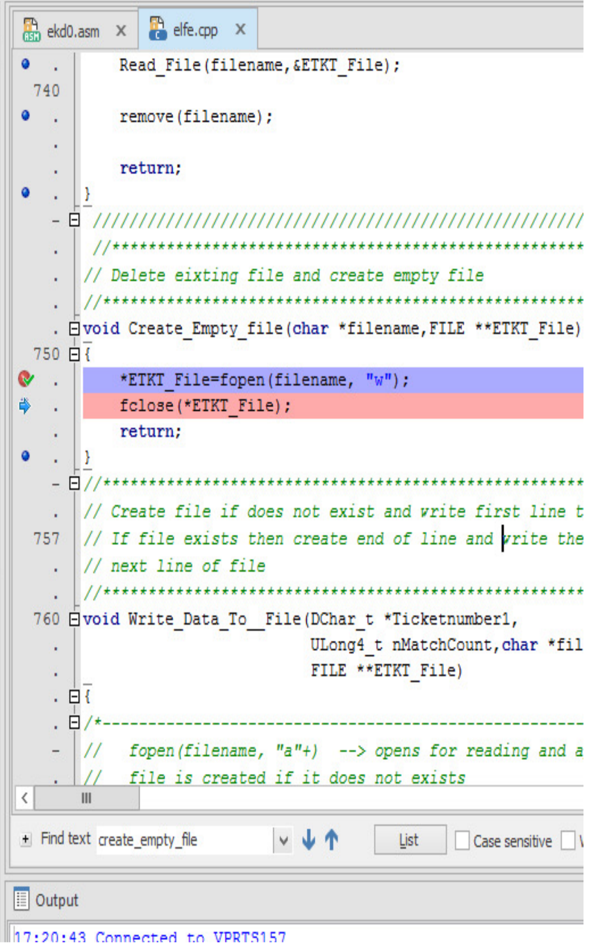
```
Host Files
Name              Date                 Per
[redacted]:anonymous
  /
    .LostAndFound      2016-02-05 00:01:00  rwx
    .tpfZfileState     2018-01-11 03:30:00  rwx
    777                2014-04-29 00:01:00  rwx
    backup             2013-10-09 00:01:00  rwx
    bin                2016-09-06 00:01:00  rwx
    COM                2007-10-25 00:01:00  rwx
    com                2018-03-24 18:31:00  rwx
    dev                2016-02-05 00:01:00  rwx
    etc                2018-01-11 02:34:00  rwx
    IBMsubcapacity     2018-04-09 00:24:00  rwx
    IBMsyserr          2018-04-09 11:25:00  rwx
    ipat               2018-03-22 18:30:00  rwx
    opt                2007-07-31 00:01:00  rwx
    prev               2014-06-11 00:01:00  rwx
    proc               2018-04-09 17:20:00  rwx
    recoup_lost_dlm    2013-11-13 00:01:00  rwx
    recoup_lost_res    2015-05-11 00:01:00  rwx
    ResAppls           2018-04-09 12:58:00  rwx
      No items found
    riat               2018-02-08 20:03:00  rwx
```



```
ekd0.asm X   elfe.cpp X

    Read_File(filename,&ETKT_File);

    remove(filename);

    return;
}
/////////////////////////////////////////////////////
//*************************************************
// Delete eixting file and create empty file
//*************************************************
void Create_Empty_file(char *filename,FILE **ETKT_File)
{
    *ETKT_File=fopen(filename, "w");
    fclose(*ETKT_File);
    return;
}
//*************************************************
// Create file if does not exist and write first line t
// If file exists then create end of line and write the
// next line of file
//*************************************************
void Write_Data_To__File(DChar_t *Ticketnumber1,
                         ULong4_t nMatchCount,char *fil
                         FILE **ETKT_File)
{
/*-----------------------------------------------
```



```
Host Files
Name                       Date                 Permis..
[redacted]:anonymous
  /
    .LostAndFound            2016-02-05 00:01:00  rwxrwxr
    .tpfZfileState           2018-01-11 03:30:00  rwxrwxr
    777                      2014-04-29 00:01:00  rwxrwxr
    backup                   2013-10-09 00:01:00  rwxr-xr-
    bin                      2016-09-06 00:01:00  rwxr-xr-
    COM                      2007-10-25 00:01:00  rwxrwxr
    com                      2018-03-24 18:31:00  rwxr-xr-
    dev                      2016-02-05 00:01:00  rwxr-xr-
    etc                      2018-01-11 02:34:00  rwxr-xr-
    IBMsubcapacity           2018-04-09 00:24:00  rwxrwxr
    IBMsyserr                2018-04-09 11:25:00  rwxrwxr
    ipat                     2018-03-22 18:30:00  rwxrwxr
    opt                      2007-07-31 00:01:00  rwxrwxr
    prev                     2014-06-11 00:01:00  rwxr-xr-
    proc                     2018-04-09 17:31:00  rwxrwxr
    recoup_lost_dlm          2013-11-13 00:01:00  rwxrwxr
    recoup_lost_res          2015-05-11 00:01:00  rwxrwxr
    ResAppls                 2018-04-09 17:31:00  rwxrwxr
      992E2BETKTS08APR18.txt 2018-04-09 17:31:00  rw-rw-rv
    riat                     2018-02-08 20:03:00  rwxrwxr
    sys                      2018-04-09 17:31:00  rwxrwxr
    tmp                      2018-04-09 17:26:00  rwxrwxr
    tpfdbgdumpB              2018-04-09 16:17:00  rwxrwxr
    tpfdbgdumpC              2018-04-06 03:07:00  rwxrwxr
    tpfdbgdumpD              2018-04-06 03:13:00  rwxrwxr
```



```
ekd0.asm X   elfe.cpp X

    Read_File(filename,&ETKT_File);

    remove(filename);

    return;
}
/////////////////////////////////////////////////////
//*************************************************
// Delete eixting file and create empty file
//*************************************************
void Create_Empty_file(char *filename,FILE **ETKT_File)
{
    *ETKT_File=fopen(filename, "w");
    fclose(*ETKT_File);
    return;
}
//*************************************************
// Create file if does not exist and write first line t
// If file exists then create end of line and write the
// next line of file
//*************************************************
void Write_Data_To__File(DChar_t *Ticketnumber1,
                         ULong4_t nMatchCount,char *fil
                         FILE **ETKT_File)
{
/*-----------------------------------------------
//   fopen(filename, "a"+)  --> opens for reading and a
//   file is created if it does not exists

Find text create_empty_file    List   Case sensitive

Output
17:20:43 Connected to VPRTS157
```

## All File open Mode options: Read, Write, Append etc. etc.

***Can find all format details over IBM help.

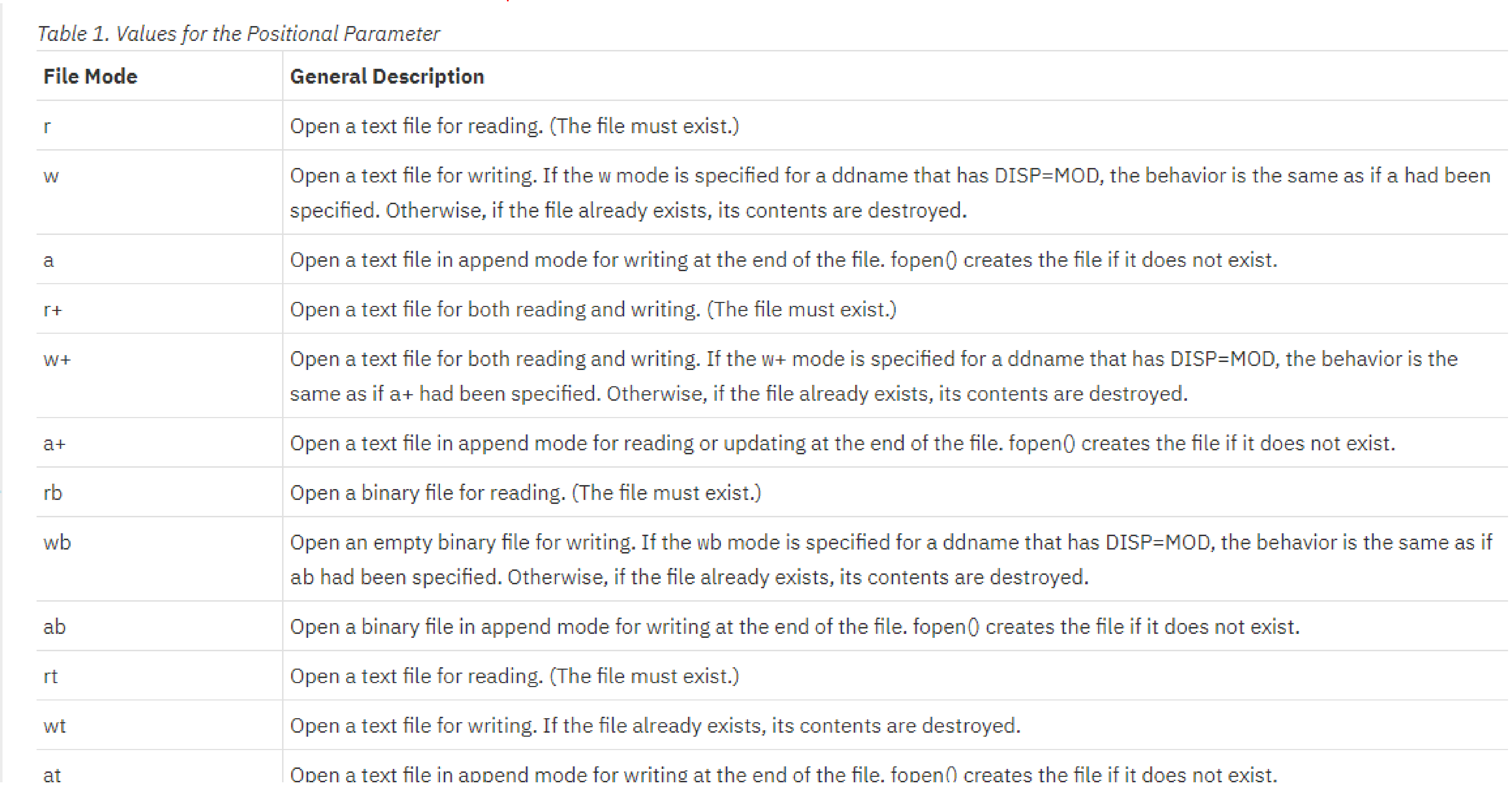
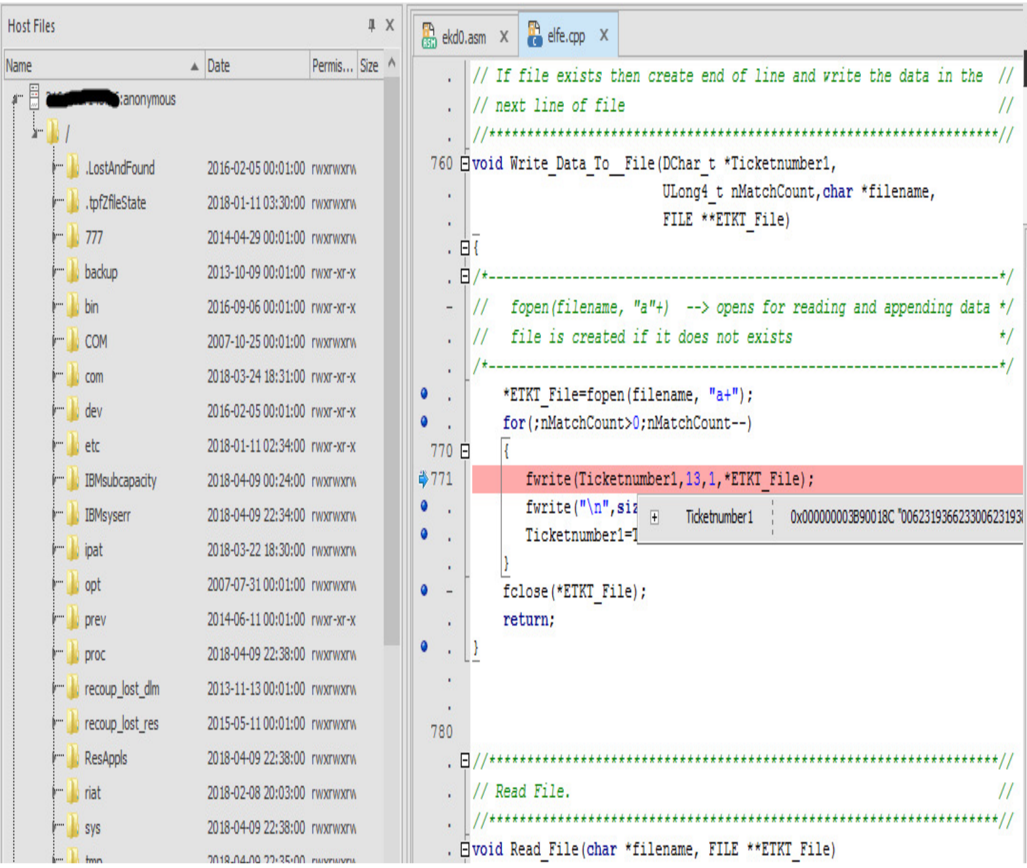


*Table 1. Values for the Positional Parameter*

| File Mode | General Description |
|---|---|
| r | Open a text file for reading. (The file must exist.) |
| w | Open a text file for writing. If the w mode is specified for a ddname that has DISP=MOD, the behavior is the same as if a had been specified. Otherwise, if the file already exists, its contents are destroyed. |
| a | Open a text file in append mode for writing at the end of the file. fopen() creates the file if it does not exist. |
| r+ | Open a text file for both reading and writing. (The file must exist.) |
| w+ | Open a text file for both reading and writing. If the w+ mode is specified for a ddname that has DISP=MOD, the behavior is the same as if a+ had been specified. Otherwise, if the file already exists, its contents are destroyed. |
| a+ | Open a text file in append mode for reading or updating at the end of the file. fopen() creates the file if it does not exist. |
| rb | Open a binary file for reading. (The file must exist.) |
| wb | Open an empty binary file for writing. If the wb mode is specified for a ddname that has DISP=MOD, the behavior is the same as if ab had been specified. Otherwise, if the file already exists, its contents are destroyed. |
| ab | Open a binary file in append mode for writing at the end of the file. fopen() creates the file if it does not exist. |
| rt | Open a text file for reading. (The file must exist.) |
| wt | Open a text file for writing. If the file already exists, its contents are destroyed. |
| at | Open a text file in append mode for writing at the end of the file. fopen() creates the file if it does not exist. |

## Write TPF data on File in z/TPF File System:

```
   *ETKT_File=fopen(filename, "a+");
   for(;nMatchCount>0;nMatchCount--)
   {
       fwrite(Ticketnumber1,13,1,*ETKT_File);
        fwrite("\n",sizeof(char),1,*ETKT_File);
   }
 fclose(*ETKT_File);
```

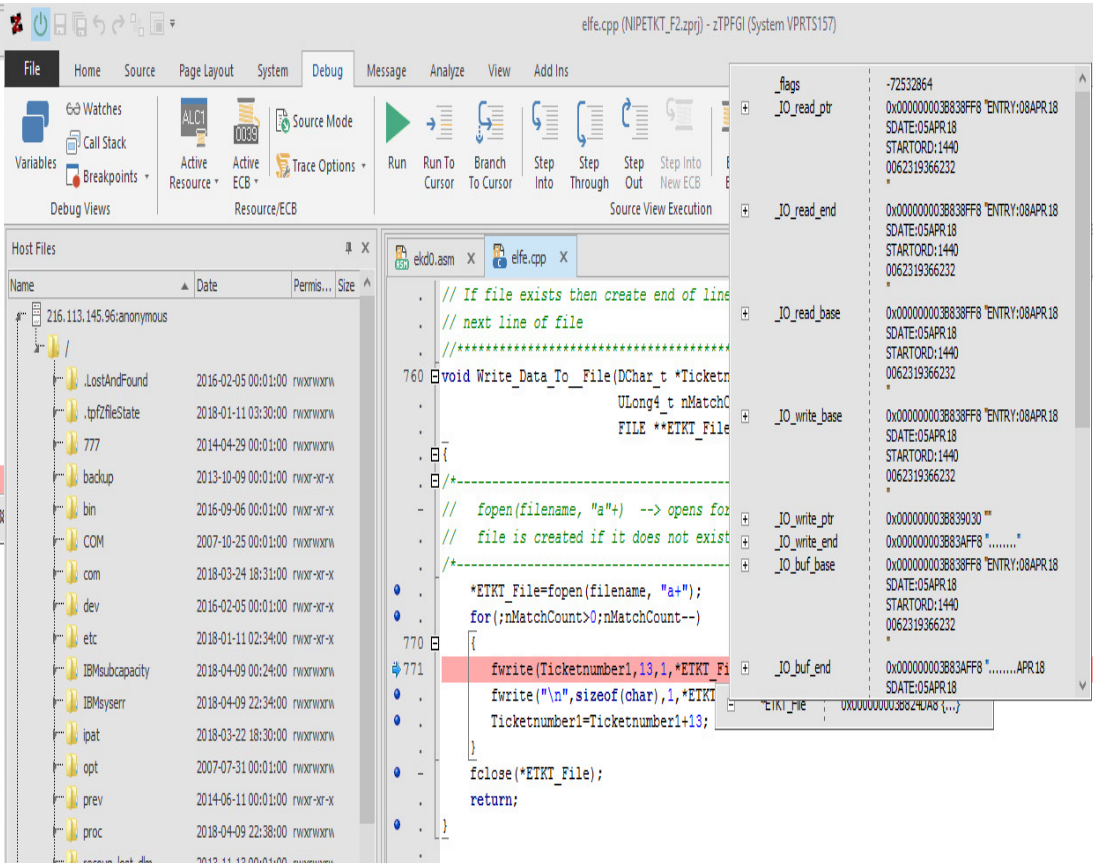


[redacted]:anonymous

```
Name                Date                 Permis...  Size
/
  .LostAndFound     2016-02-05 00:01:00  rwxrwxrw
  .tpfZfileState    2018-01-11 03:30:00  rwxrwxrw
  777               2014-04-29 00:01:00  rwxrwxrw
  backup            2013-10-09 00:01:00  rwxr-xr-x
  bin               2016-09-06 00:01:00  rwxr-xr-x
  COM               2007-10-25 00:01:00  rwxrwxrw
  com               2018-03-24 18:31:00  rwxr-xr-x
  dev               2016-02-05 00:01:00  rwxr-xr-x
  etc               2018-01-11 02:34:00  rwxr-xr-x
  IBMsubcapacity    2018-04-09 00:24:00  rwxrwxrw
  IBMsyserr         2018-04-09 22:34:00  rwxrwxrw
  ipat              2018-03-22 18:30:00  rwxrwxrw
  opt               2007-07-31 00:01:00  rwxrwxrw
  prev              2014-06-11 00:01:00  rwxr-xr-x
  proc              2018-04-09 22:38:00  rwxrwxrw
  recoup_lost_dlm   2013-11-13 00:01:00  rwxrwxrw
  recoup_lost_res   2015-05-11 00:01:00  rwxrwxrw
  ResAppls          2018-04-09 22:38:00  rwxrwxrw
  riat              2018-02-08 20:03:00  rwxrwxrw
  sys               2018-04-09 22:38:00  rwxrwxrw
```

```
// If file exists then create end of line and write the data in the  //
// next line of file                                                  //
//*******************************************************************//
void Write_Data_To__File(DChar_t *Ticketnumber1,
                         ULong4_t nMatchCount,char *filename,
                         FILE **ETKT_File)
{
/*------------------------------------------------------------------*/
//   fopen(filename, "a"+)  --> opens for reading and appending data */
//   file is created if it does not exists                           */
/*------------------------------------------------------------------*/
    *ETKT_File=fopen(filename, "a+");
    for(;nMatchCount>0;nMatchCount--)
    {
      fwrite(Ticketnumber1,13,1,*ETKT_File);
      fwrite("\n",siz
      Ticketnumber1=T
    }
    fclose(*ETKT_File);
    return;
}


//*******************************************************************//
// Read File.                                                         //
//*******************************************************************//
void Read_File(char *filename, FILE **ETKT_File)
```

```
+  Ticketnumber1  0x000000003B90018C "006231936623300623193
```



216.113.145.96:anonymous

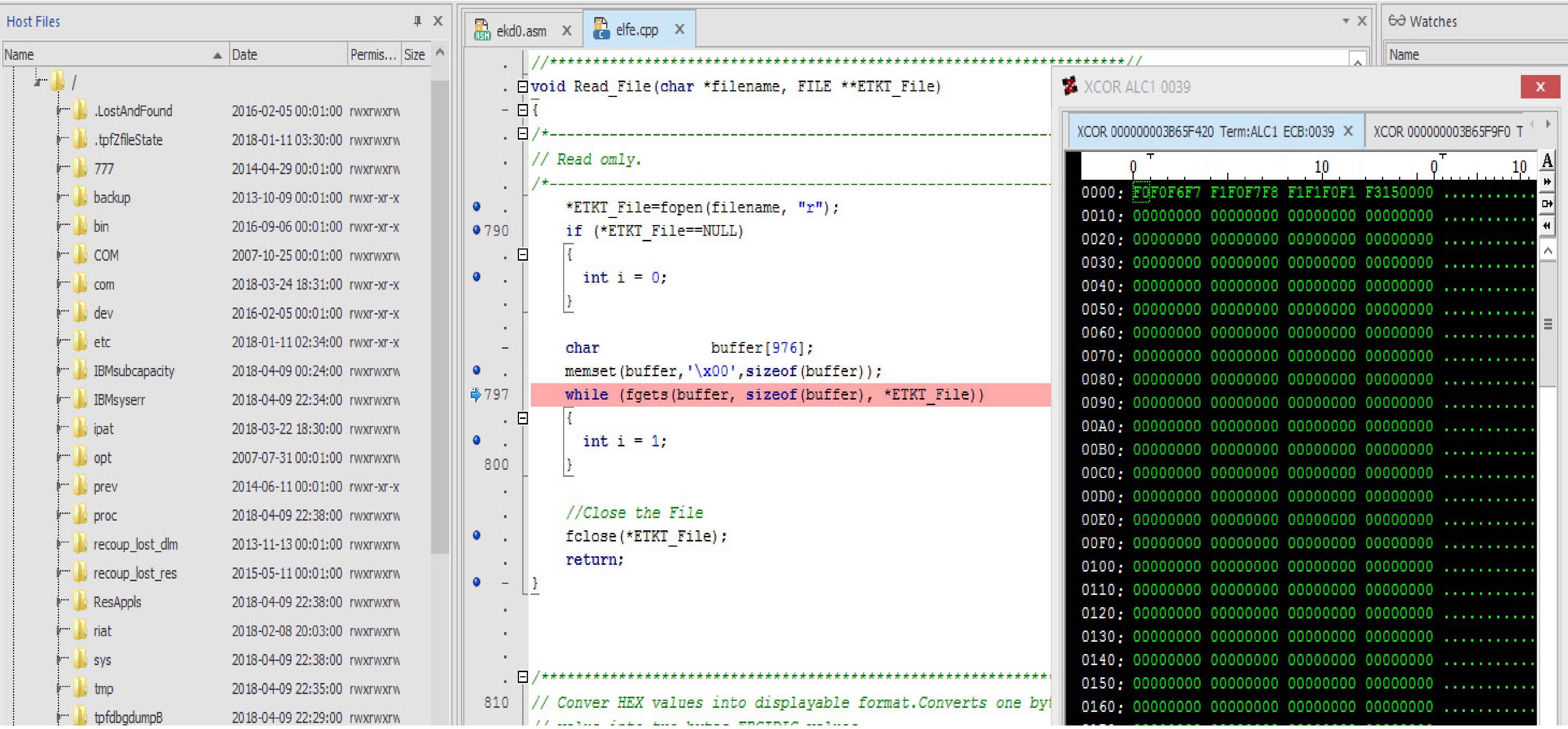
```
Name                Date                 Permis...  Size
/
  .LostAndFound     2016-02-05 00:01:00  rwxrwxrw
  .tpfZfileState    2018-01-11 03:30:00  rwxrwxrw
  777               2014-04-29 00:01:00  rwxrwxrw
  backup            2013-10-09 00:01:00  rwxr-xr-x
  bin               2016-09-06 00:01:00  rwxr-xr-x
  COM               2007-10-25 00:01:00  rwxrwxrw
  com               2018-03-24 18:31:00  rwxr-xr-x
  dev               2016-02-05 00:01:00  rwxr-xr-x
  etc               2018-01-11 02:34:00  rwxr-xr-x
  IBMsubcapacity    2018-04-09 00:24:00  rwxrwxrw
  IBMsyserr         2018-04-09 22:34:00  rwxrwxrw
  ipat              2018-03-22 18:30:00  rwxrwxrw
  opt               2007-07-31 00:01:00  rwxrwxrw
  prev              2014-06-11 00:01:00  rwxr-xr-x
  proc              2018-04-09 22:38:00  rwxrwxrw
```

```
// If file exists then create end of line
// next line of file
//***********************************
void Write_Data_To__File(DChar_t *Ticketn
                         ULong4_t nMatchC
                         FILE **ETKT_File
{
/*-----------------------------------
//   fopen(filename, "a"+)  --> opens for
//   file is created if it does not exist
/*-----------------------------------
    *ETKT_File=fopen(filename, "a+");
    for(;nMatchCount>0;nMatchCount--)
    {
      fwrite(Ticketnumber1,13,1,*ETKT_Fi
      fwrite("\n",sizeof(char),1,*ETKT
      Ticketnumber1=Ticketnumber1+13;
    }
    fclose(*ETKT_File);
    return;
}
```

```
   _flags           -72532864
+  _IO_read_ptr     0x000000003B838FF8 "ENTRY:08APR18
                    SDATE:05APR18
                    STARTORD:1440
                    0062319366232
                    "
+  _IO_read_end     0x000000003B838FF8 "ENTRY:08APR18
                    SDATE:05APR18
                    STARTORD:1440
                    0062319366232
                    "
+  _IO_read_base    0x000000003B838FF8 "ENTRY:08APR18
                    SDATE:05APR18
                    STARTORD:1440
                    0062319366232
                    "
+  _IO_write_base   0x000000003B838FF8 "ENTRY:08APR18
                    SDATE:05APR18
                    STARTORD:1440
                    0062319366232
                    "
+  _IO_write_ptr    0x000000003B839030 ""
+  _IO_write_end    0x000000003B83AFF8 "........"
+  _IO_buf_base     0x000000003B838FF8 "ENTRY:08APR18
                    SDATE:05APR18
                    STARTORD:1440
                    0062319366232
                    "
+  _IO_buf_end      0x000000003B83AFF8 "........APR18
                    SDATE:05APR18
-  *ETKT_File       0x000000003B824DA8 {...}
```

## Read TPF Data from z/TPF File System:

```
 *ETKT_File=fopen(filename, "r");
while (fgets(buffer, sizeof(buffer), *ETKT_File))
fclose(*ETKT_File);
```


Host Files

| Name | Date | Permis... | Size |
|---|---|---|---|
| / | | | |
| .LostAndFound | 2016-02-05 00:01:00 | rwxrwxrw | |
| .tpfZfileState | 2018-01-11 03:30:00 | rwxrwxrw | |
| 777 | 2014-04-29 00:01:00 | rwxrwxrw | |
| backup | 2013-10-09 00:01:00 | rwxr-xr-x | |
| bin | 2016-09-06 00:01:00 | rwxr-xr-x | |
| COM | 2007-10-25 00:01:00 | rwxrwxrw | |
| com | 2018-03-24 18:31:00 | rwxr-xr-x | |
| dev | 2016-02-05 00:01:00 | rwxr-xr-x | |
| etc | 2018-01-11 02:34:00 | rwxr-xr-x | |
| IBMsubcapacity | 2018-04-09 00:24:00 | rwxrwxrw | |
| IBMsyserr | 2018-04-09 22:34:00 | rwxrwxrw | |
| ipat | 2018-03-22 18:30:00 | rwxrwxrw | |
| opt | 2007-07-31 00:01:00 | rwxrwxrw | |
| prev | 2014-06-11 00:01:00 | rwxr-xr-x | |
| proc | 2018-04-09 22:38:00 | rwxrwxrw | |
| recoup_lost_dlm | 2013-11-13 00:01:00 | rwxrwxrw | |
| recoup_lost_res | 2015-05-11 00:01:00 | rwxrwxrw | |
| ResAppls | 2018-04-09 22:38:00 | rwxrwxrw | |
| riat | 2018-02-08 20:03:00 | rwxrwxrw | |
| sys | 2018-04-09 22:38:00 | rwxrwxrw | |
| tmp | 2018-04-09 22:35:00 | rwxrwxrw | |
| tpfdbgdumpB | 2018-04-09 22:29:00 | rwxrwxrw | |

ekd0.asm   elfe.cpp

```
//***************************************************************//
void Read_File(char *filename, FILE **ETKT_File)
{
/*------------------------------------------------------------------
// Read only.
/*------------------------------------------------------------------
    *ETKT_File=fopen(filename, "r");
    if (*ETKT_File==NULL)
    {
      int i = 0;
    }

    char             buffer[976];
    memset(buffer,'\x00',sizeof(buffer));
    while (fgets(buffer, sizeof(buffer), *ETKT_File))
    {
      int i = 1;
    }

    //Close the File
    fclose(*ETKT_File);
    return;
}


/*****************************************************************
// Conver HEX values into displayable format.Converts one byt
```

Watches

Name

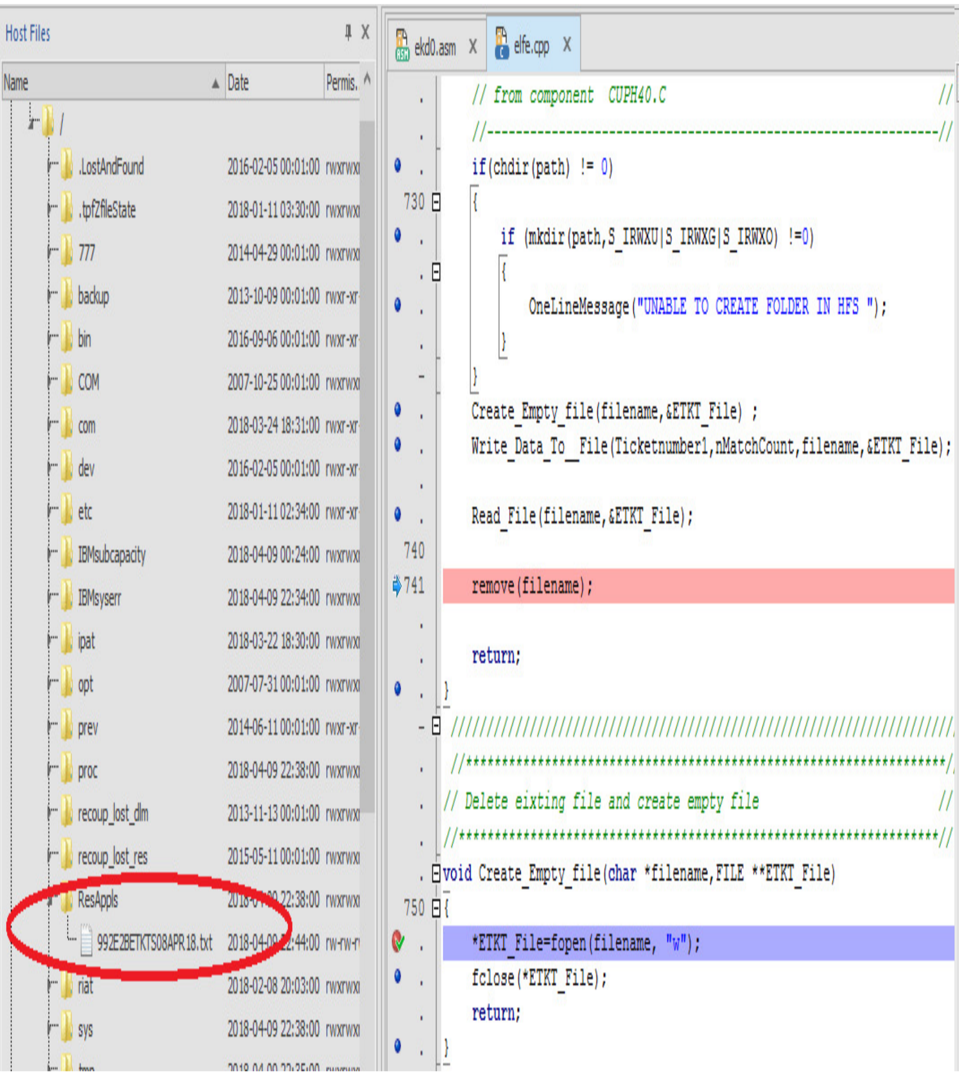
XCOR ALC1 0039

XCOR 000000003B65F420 Term:ALC1 ECB:0039   XCOR 000000003B65F9F0 T

```
0000: F0F0F6F7 F1F0F7F8 F1F1F0F1 F3150000 ...........
0010: 00000000 00000000 00000000 00000000 ...........
0020: 00000000 00000000 00000000 00000000 ...........
0030: 00000000 00000000 00000000 00000000 ...........
0040: 00000000 00000000 00000000 00000000 ...........
0050: 00000000 00000000 00000000 00000000 ...........
0060: 00000000 00000000 00000000 00000000 ...........
0070: 00000000 00000000 00000000 00000000 ...........
0080: 00000000 00000000 00000000 00000000 ...........
0090: 00000000 00000000 00000000 00000000 ...........
00A0: 00000000 00000000 00000000 00000000 ...........
00B0: 00000000 00000000 00000000 00000000 ...........
00C0: 00000000 00000000 00000000 00000000 ...........
00D0: 00000000 00000000 00000000 00000000 ...........
00E0: 00000000 00000000 00000000 00000000 ...........
00F0: 00000000 00000000 00000000 00000000 ...........
0100: 00000000 00000000 00000000 00000000 ...........
0110: 00000000 00000000 00000000 00000000 ...........
0120: 00000000 00000000 00000000 00000000 ...........
0130: 00000000 00000000 00000000 00000000 ...........
0140: 00000000 00000000 00000000 00000000 ...........
0150: 00000000 00000000 00000000 00000000 ...........
0160: 00000000 00000000 00000000 00000000 ...........
```

## Delete TPF Files from z/TPF File System:

remove(filename);

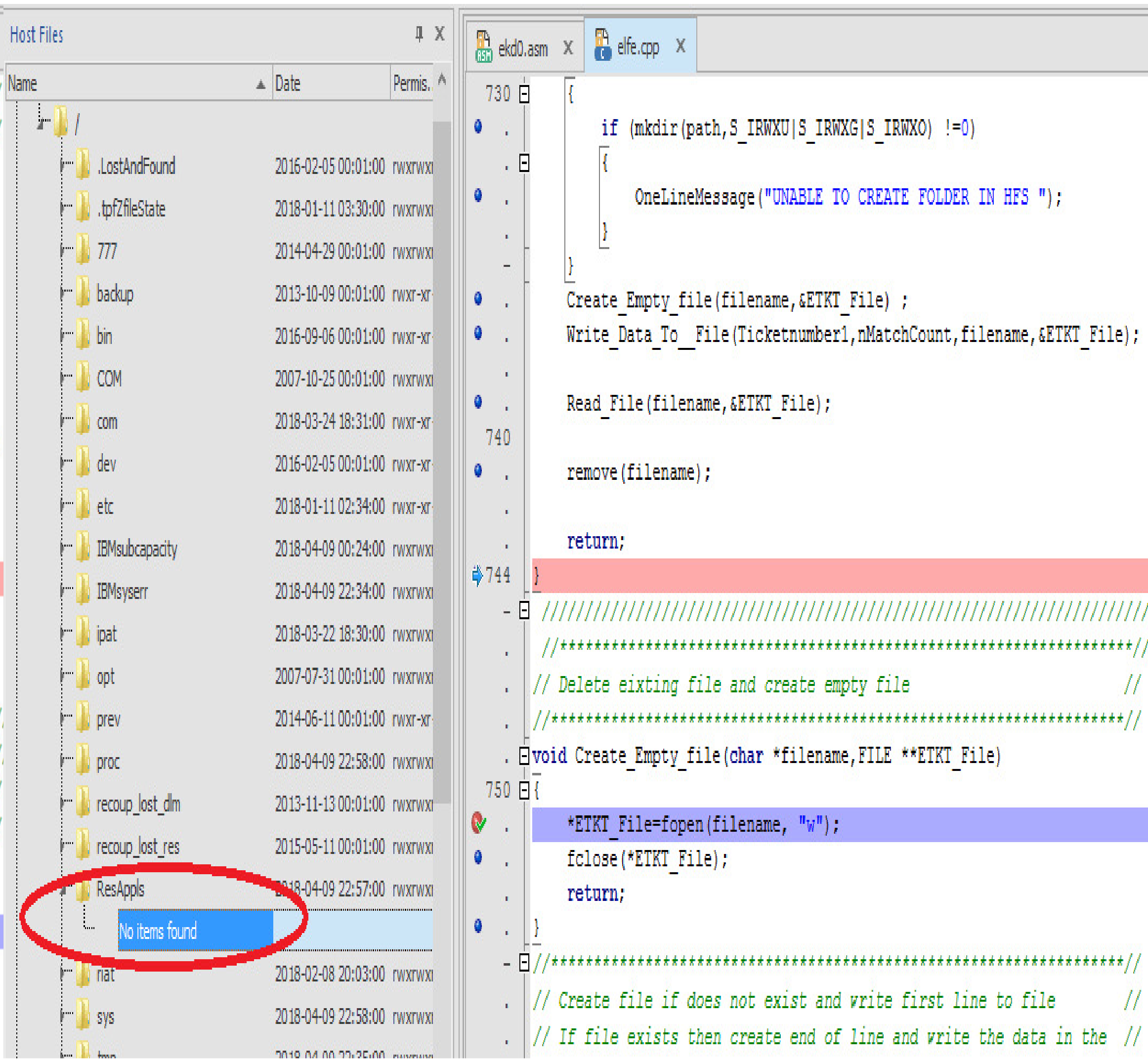


Host Files

| Name | Date | Permis. |
|---|---|---|
| / | | |
| .LostAndFound | 2016-02-05 00:01:00 | rwxrwx |
| .tpf2fileState | 2018-01-11 03:30:00 | rwxrwx |
| 777 | 2014-04-29 00:01:00 | rwxrwx |
| backup | 2013-10-09 00:01:00 | rwxr-xr |
| bin | 2016-09-06 00:01:00 | rwxr-xr |
| COM | 2007-10-25 00:01:00 | rwxrwx |
| com | 2018-03-24 18:31:00 | rwxr-xr |
| dev | 2016-02-05 00:01:00 | rwxr-xr |
| etc | 2018-01-11 02:34:00 | rwxr-xr |
| IBMsubcapacity | 2018-04-09 00:24:00 | rwxrwx |
| IBMsyserr | 2018-04-09 22:34:00 | rwxrwx |
| ipat | 2018-03-22 18:30:00 | rwxrwx |
| opt | 2007-07-31 00:01:00 | rwxrwx |
| prev | 2014-06-11 00:01:00 | rwxr-xr |
| proc | 2018-04-09 22:38:00 | rwxrwx |
| recoup_lost_dlm | 2013-11-13 00:01:00 | rwxrwx |
| recoup_lost_res | 2015-05-11 00:01:00 | rwxrwx |
| ResAppls | 2018-04-09 22:38:00 | rwxrwx |
| 992E2BETKTS08APR18.txt | 2018-04-09 22:44:00 | rw-rw-r |
| riat | 2018-02-08 20:03:00 | rwxrwx |
| sys | 2018-04-09 22:38:00 | rwxrwx |

ekd0.asm    elfe.cpp

```
    // from component  CUPH40.C                                    //
    //-------------------------------------------------------------//
    if(chdir(path) != 0)
    {
        if (mkdir(path,S_IRWXU|S_IRWXG|S_IRWXO) !=0)
        {
            OneLineMessage("UNABLE TO CREATE FOLDER IN HFS ");
        }
    }
    Create_Empty_file(filename,&ETKT_File) ;
    Write_Data_To _File(Ticketnumber1,nMatchCount,filename,&ETKT_File);

    Read_File(filename,&ETKT_File);

    remove(filename);

    return;
}
////////////////////////////////////////////////////////////////////
//**************************************************************//
// Delete eixting file and create empty file                     //
//**************************************************************//
void Create_Empty_file(char *filename,FILE **ETKT_File)
{
    *ETKT_File=fopen(filename, "w");
    fclose(*ETKT_File);
    return;
}
```



Host Files

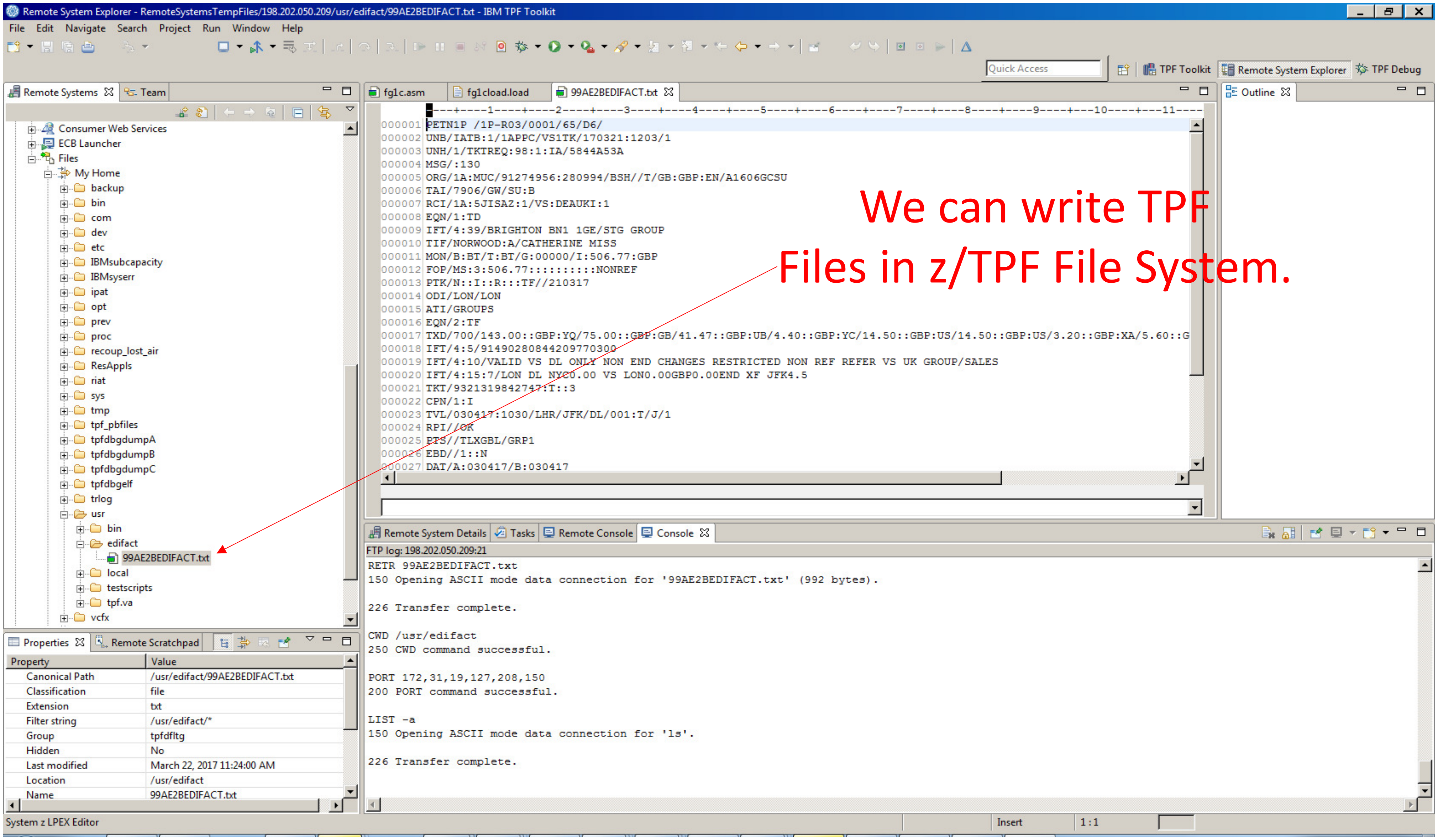
| Name | Date | Permis. |
|---|---|---|
| / | | |
| .LostAndFound | 2016-02-05 00:01:00 | rwxrwx |
| .tpf2fileState | 2018-01-11 03:30:00 | rwxrwx |
| 777 | 2014-04-29 00:01:00 | rwxrwx |
| backup | 2013-10-09 00:01:00 | rwxr-xr |
| bin | 2016-09-06 00:01:00 | rwxr-xr |
| COM | 2007-10-25 00:01:00 | rwxrwx |
| com | 2018-03-24 18:31:00 | rwxr-xr |
| dev | 2016-02-05 00:01:00 | rwxr-xr |
| etc | 2018-01-11 02:34:00 | rwxr-xr |
| IBMsubcapacity | 2018-04-09 00:24:00 | rwxrwx |
| IBMsyserr | 2018-04-09 22:34:00 | rwxrwx |
| ipat | 2018-03-22 18:30:00 | rwxrwx |
| opt | 2007-07-31 00:01:00 | rwxrwx |
| prev | 2014-06-11 00:01:00 | rwxr-xr |
| proc | 2018-04-09 22:58:00 | rwxrwx |
| recoup_lost_dlm | 2013-11-13 00:01:00 | rwxrwx |
| recoup_lost_res | 2015-05-11 00:01:00 | rwxrwx |
| ResAppls | 2018-04-09 22:57:00 | rwxrwx |
| No items found | | |
| riat | 2018-02-08 20:03:00 | rwxrwx |
| sys | 2018-04-09 22:58:00 | rwxrwx |

ekd0.asm    elfe.cpp

```
    {
        if (mkdir(path,S_IRWXU|S_IRWXG|S_IRWXO) !=0)
        {
            OneLineMessage("UNABLE TO CREATE FOLDER IN HFS ");
        }
    }
    Create_Empty_file(filename,&ETKT_File) ;
    Write_Data_To _File(Ticketnumber1,nMatchCount,filename,&ETKT_File);

    Read_File(filename,&ETKT_File);

    remove(filename);

    return;
}
////////////////////////////////////////////////////////////////////
//**************************************************************//
// Delete eixting file and create empty file                     //
//**************************************************************//
void Create_Empty_file(char *filename,FILE **ETKT_File)
{
    *ETKT_File=fopen(filename, "w");
    fclose(*ETKT_File);
    return;
}
//**************************************************************//
// Create file if does not exist and write first line to file    //
// If file exists then create end of line and write the data in the  //
```

## Proof Of Concept in Test Systems

We can write TPF Files in z/TPF File System.



```
PETN1P /1P-R03/0001/65/D6/
UNB/IATB:1/1APPC/VS1TK/170321:1203/1
UNH/1/TKTREQ:98:1:IA/5844A53A
MSG/:130
ORG/1A:MUC/91274956:280994/BSH//T/GB:GBP:EN/A1606GCSU
TAI/7906/GW/SU:B
RCI/1A:5JISAZ:1/VS:DEAUKI:1
EQN/1:TD
IFT/4:39/BRIGHTON BN1 1GE/STG GROUP
TIF/NORWOOD:A/CATHERINE MISS
MON/B:BT/T:BT/G:00000/I:506.77:GBP
FOP/MS:3:506.77:::::::::NONREF
PTK/N::I::R:::TF//210317
ODI/LON/LON
ATI/GROUPS
EQN/2:TF
TXD/700/143.00::GBP:YQ/75.00::GBP:GB/41.47::GBP:UB/4.40::GBP:YC/14.50::GBP:US/14.50::GBP:US/3.20::GBP:XA/5.60::G
IFT/4:5/91490280844209770300
IFT/4:10/VALID VS DL ONLY NON END CHANGES RESTRICTED NON REF REFER VS UK GROUP/SALES
IFT/4:15:7/LON DL NYC0.00 VS LON0.00GBP0.00END XF JFK4.5
TKT/9321319842747:T::3
CPN/1:I
TVL/030417:1030/LHR/JFK/DL/001:T/J/1
RPI//OK
PTS//TLXGBL/GRP1
EBD//1::N
DAT/A:030417/B:030417
```

FTP log: 198.202.050.209:21

```
RETR 99AE2BEDIFACT.txt
150 Opening ASCII mode data connection for '99AE2BEDIFACT.txt' (992 bytes).

226 Transfer complete.

CWD /usr/edifact
250 CWD command successful.

PORT 172,31,19,127,208,150
200 PORT command successful.

LIST -a
150 Opening ASCII mode data connection for 'ls'.

226 Transfer complete.
```

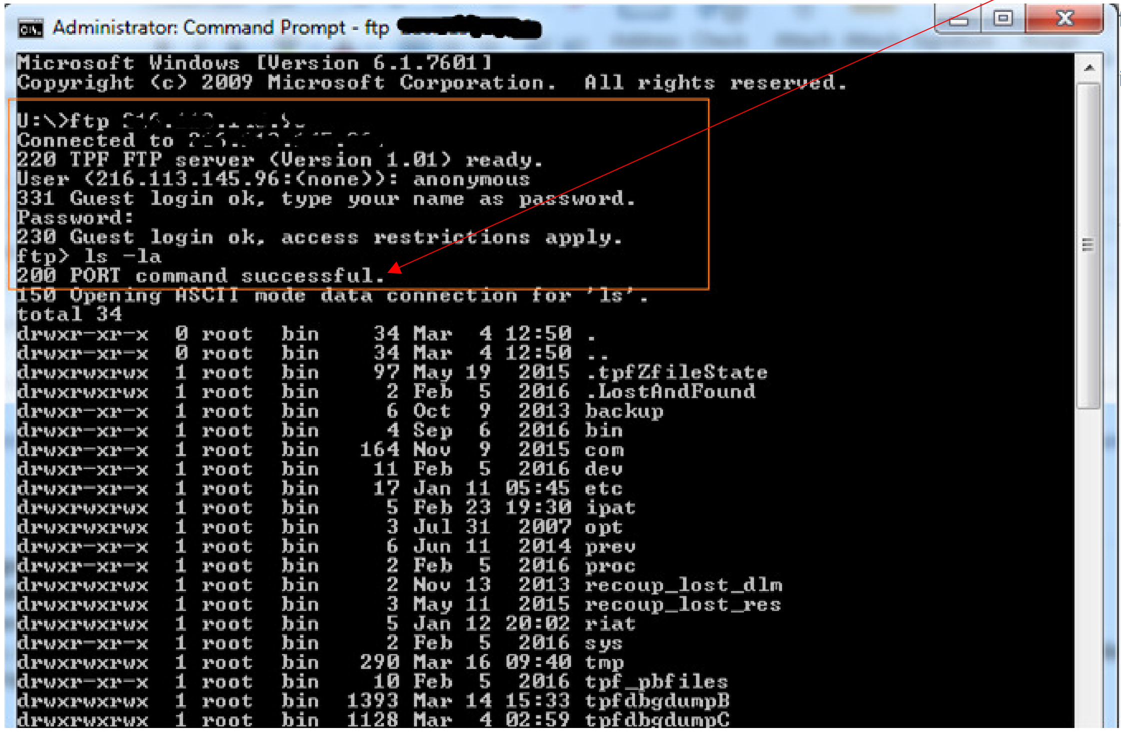
| Property | Value |
|---|---|
| Canonical Path | /usr/edifact/99AE2BEDIFACT.txt |
| Classification | file |
| Extension | txt |
| Filter string | /usr/edifact/* |
| Group | tpfdfltg |
| Hidden | No |
| Last modified | March 22, 2017 11:24:00 AM |
| Location | /usr/edifact |
| Name | 99AE2BEDIFACT.txt |

## Just via FTP on internal z/TPF File System IP xxx.xxx.xxx.xx, we are able to get mainframe's files without any Proprietary Middleware.

**Cc:** Deruiter, Alex <alex.deruiter@travelport.com>; Allister, Lance <Lance.Allister@delta.com>; Lei, Rachel <rachel.lei@delta.com>; Jordan, Scott <scott.jordan@travelport.com>
**Subject:** RE: File Maintenance - VPRTS157/Files/My Home/ResAppls/992E2BETKTS13MAR17.txt

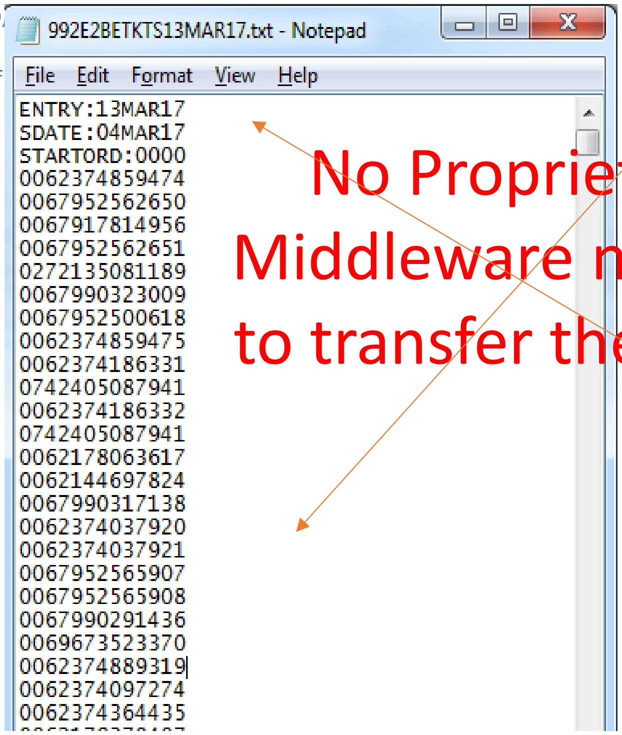
I was able to FTP that file from the TPF VPARS VPRTS157 onto my PC.

Successful – Means FTP completed.



Administrator: Command Prompt - ftp [redacted]

```
Microsoft Windows [Version 6.1.7601]
Copyright (c) 2009 Microsoft Corporation.  All rights reserved.

U:\>ftp [redacted]
Connected to [redacted]
220 TPF FTP server (Version 1.01) ready.
User (216.113.145.96:(none)): anonymous
331 Guest login ok, type your name as password.
Password:
230 Guest login ok, access restrictions apply.
ftp> ls -la
200 PORT command successful.
150 Opening ASCII mode data connection for 'ls'.
total 34
drwxr-xr-x  0 root  bin    34 Mar  4 12:50 .
drwxr-xr-x  0 root  bin    34 Mar  4 12:50 ..
drwxrwxrwx  1 root  bin    97 May 19  2015 .tpfZfileState
drwxrwxrwx  1 root  bin     2 Feb  5  2016 .LostAndFound
drwxr-xr-x  1 root  bin     6 Oct  9  2013 backup
drwxr-xr-x  1 root  bin     4 Sep  6  2016 bin
drwxr-xr-x  1 root  bin   164 Nov  9  2015 com
drwxr-xr-x  1 root  bin    11 Feb  5  2016 dev
drwxr-xr-x  1 root  bin    17 Jan 11 05:45 etc
drwxrwxrwx  1 root  bin     5 Feb 23 19:30 ipat
drwxrwxrwx  1 root  bin     3 Jul 31  2007 opt
drwxr-xr-x  1 root  bin     6 Jun 11  2014 prev
drwxr-xr-x  1 root  bin     2 Feb  5  2016 proc
drwxrwxrwx  1 root  bin     2 Nov 13  2013 recoup_lost_dlm
drwxrwxrwx  1 root  bin     3 May 11  2015 recoup_lost_res
drwxrwxrwx  1 root  bin     5 Jan 12 20:02 riat
drwxr-xr-x  1 root  bin     2 Feb  5  2016 sys
drwxrwxrwx  1 root  bin   290 Mar 16 09:40 tmp
drwxr-xr-x  1 root  bin    10 Feb  5  2016 tpf_pbfiles
drwxrwxrwx  1 root  bin  1393 Mar 14 15:33 tpfdbgdumpB
drwxrwxrwx  1 root  bin  1128 Mar  4 02:59 tpfdbgdumpC
```

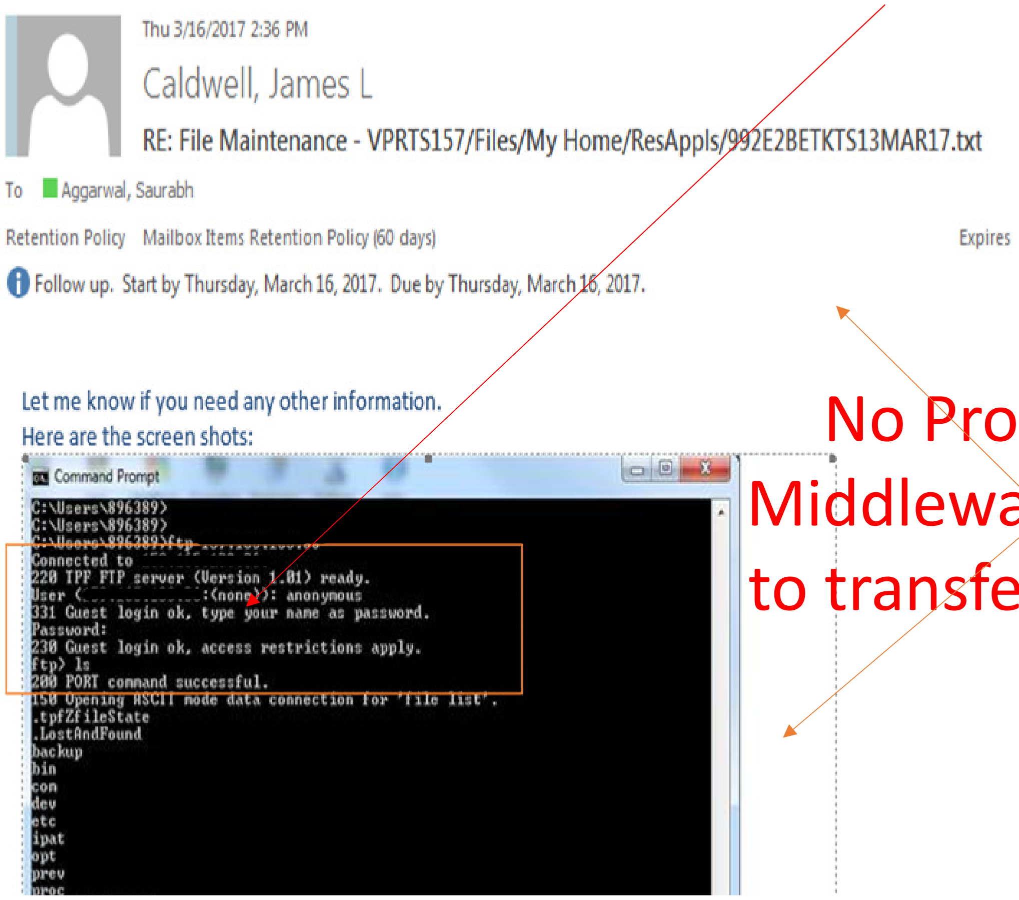


```
ENTRY:13MAR17
SDATE:04MAR17
STARTORD:0000
0062374859474
0067952562650
0067917814956
0067952562651
0272135081189
0067990323009
0067952500618
0062374859475
0062374186331
0742405087941
0062374186332
0742405087941
0062178063617
0062144697824
0067990317138
0062374037920
0062374037921
0067952565907
0067952565908
0067990291436
0069673523370
0062374889319
0062374097274
0062374364435
```

No Proprietary Middleware needed to transfer the data.

Just via FTP on External IP: xxx.xxx.xxx.xx, we are able to get these files without any Proprietary Middleware.

Successful – Means FTP completed.



Thu 3/16/2017 2:36 PM

Caldwell, James L

RE: File Maintenance - VPRTS157/Files/My Home/ResAppls/992E2BETKTS13MAR17.txt

To Aggarwal, Saurabh

Retention Policy Mailbox Items Retention Policy (60 days) Expires

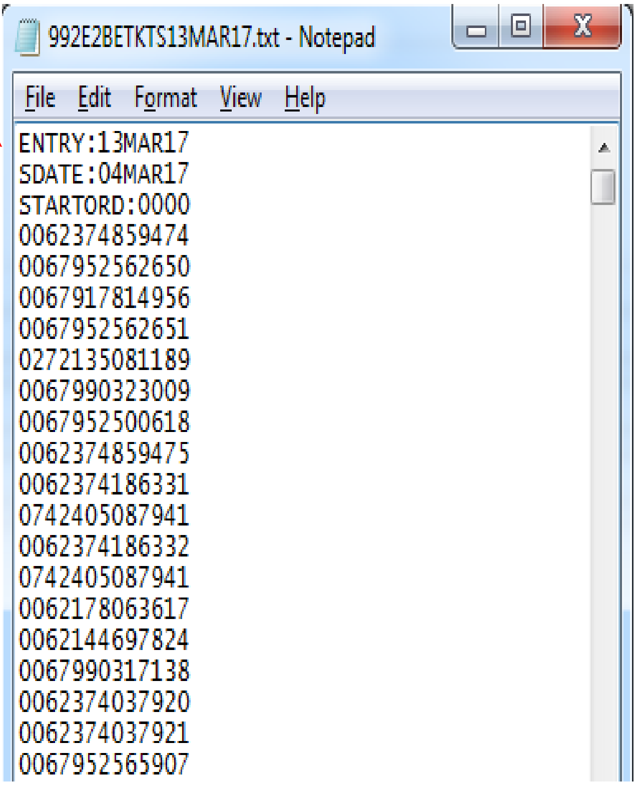
Follow up. Start by Thursday, March 16, 2017. Due by Thursday, March 16, 2017.

Let me know if you need any other information.

Here are the screen shots:

```
C:\Users\896389>
C:\Users\896389>
C:\Users\896389>ftp [redacted]
Connected to [redacted]
220 TPF FTP server (Version 1.01) ready.
User ([redacted]:(none)): anonymous
331 Guest login ok, type your name as password.
Password:
230 Guest login ok, access restrictions apply.
ftp> ls
200 PORT command successful.
150 Opening ASCII mode data connection for 'file list'.
.tpfZfileState
.LostAndFound
backup
bin
con
dev
etc
ipat
opt
prev
proc
```

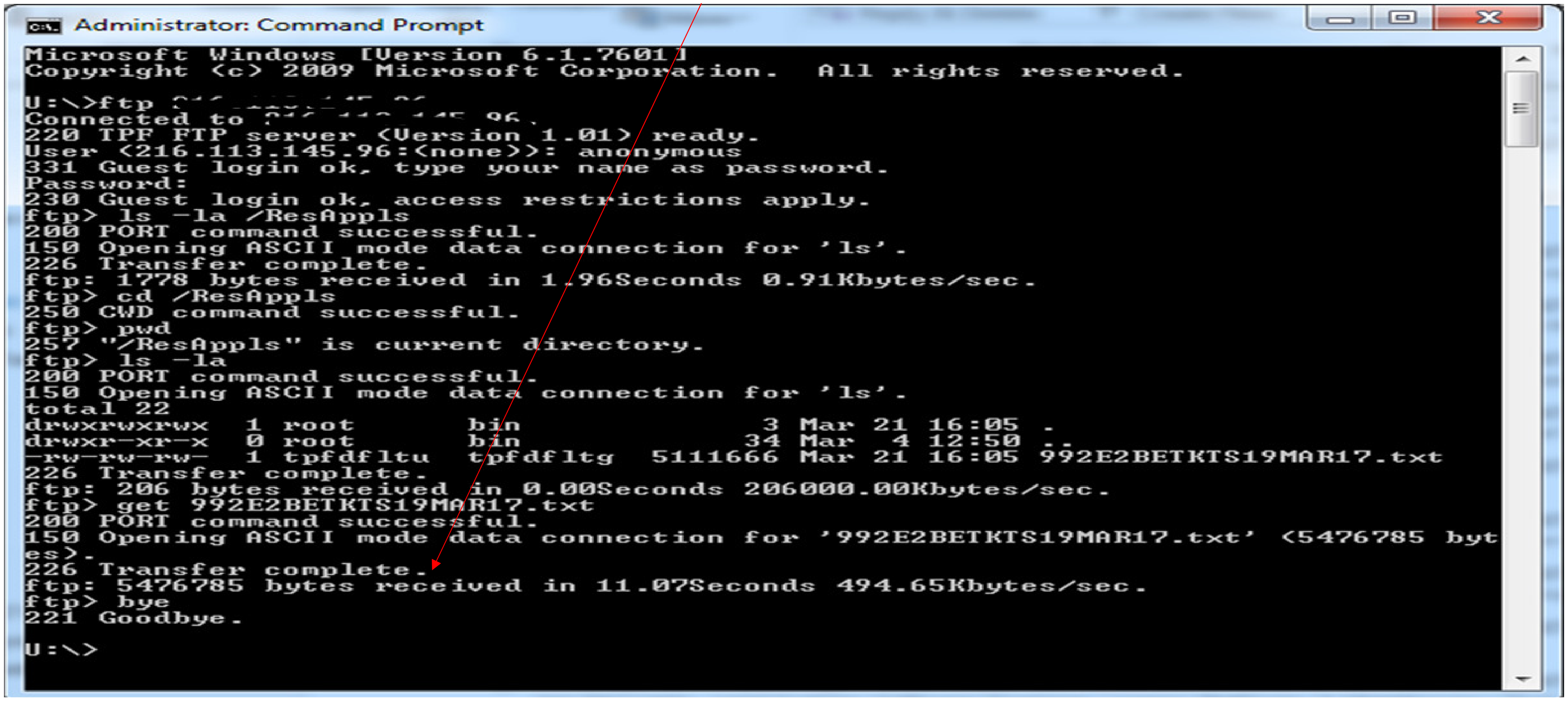
No Proprietary Middleware needed to transfer the data.



```
ENTRY:13MAR17
SDATE:04MAR17
STARTORD:0000
0062374859474
0067952562650
0067917814956
0067952562651
0272135081189
0067990323009
0067952500618
0062374859475
0062374186331
0742405087941
0062374186332
0742405087941
0062178063617
0062144697824
0067990317138
0062374037920
0062374037921
0067952565907
```

## z/TPF File System -> Local Computer
## So that can be processed by Open Source Tool if needed.

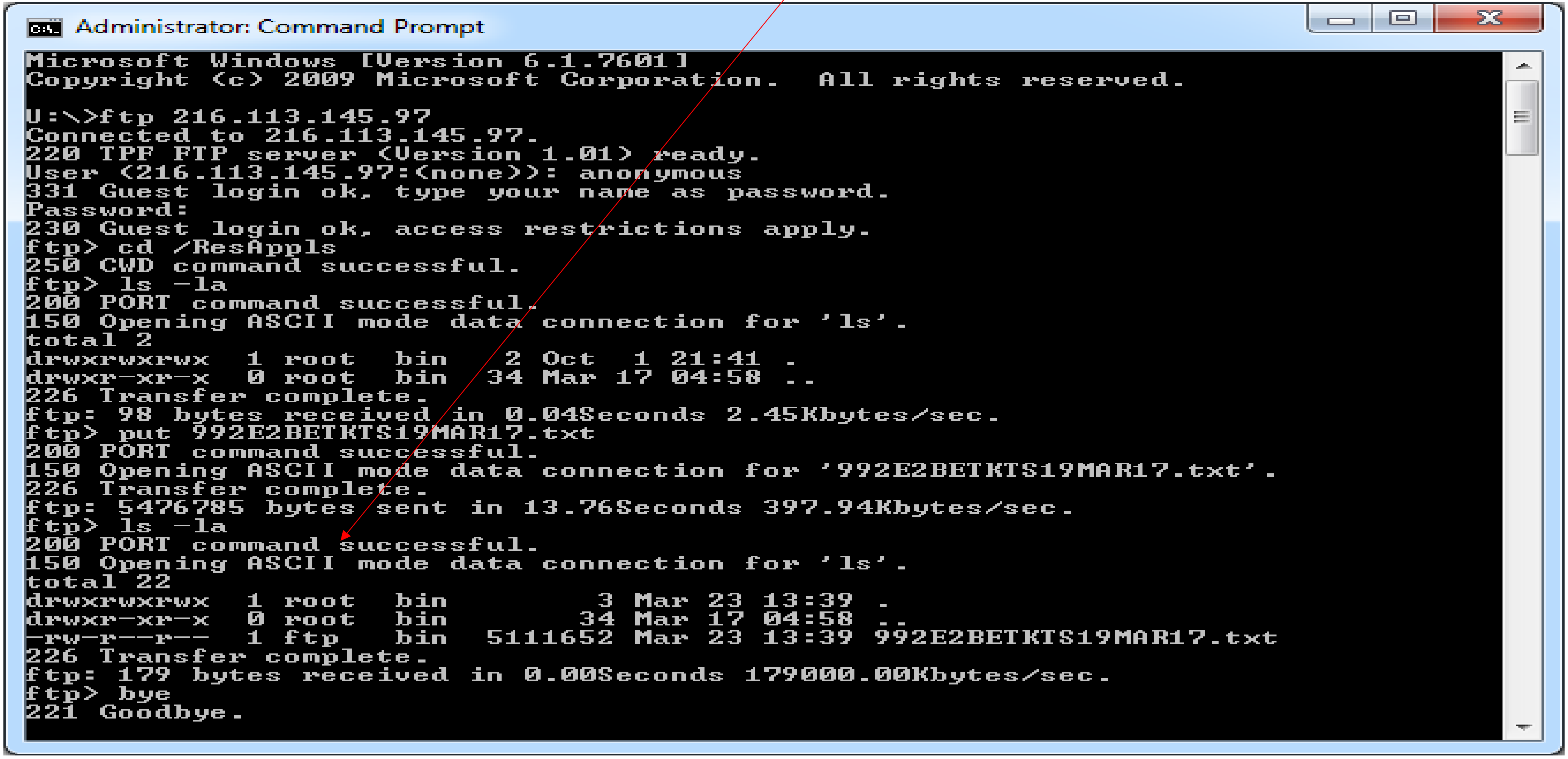
Transfer Complete – Means FTP completed.



Administrator: Command Prompt

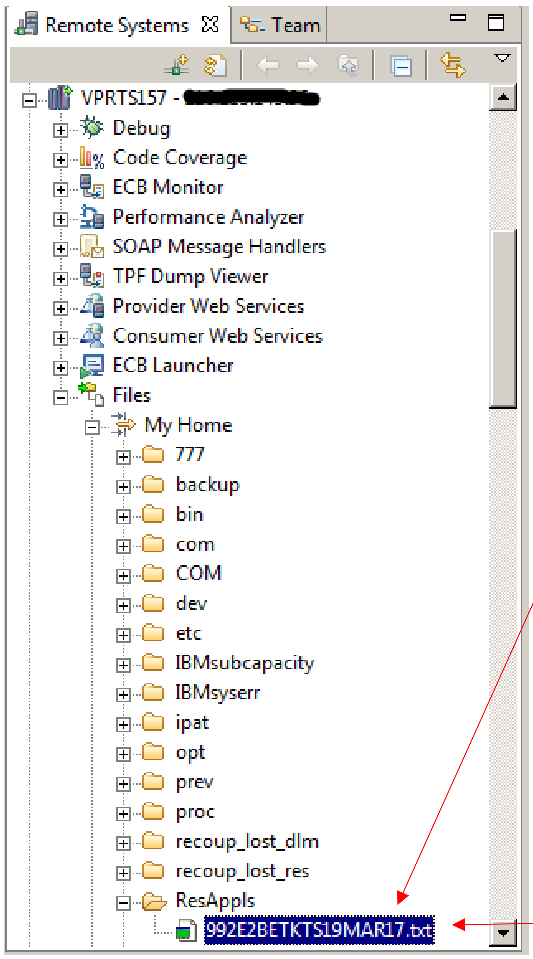
```
Microsoft Windows [Version 6.1.7601]
Copyright (c) 2009 Microsoft Corporation.  All rights reserved.

U:\>ftp [redacted]
Connected to [redacted].
220 TPF FTP server (Version 1.01) ready.
User (216.113.145.96:(none)): anonymous
331 Guest login ok, type your name as password.
Password:
230 Guest login ok, access restrictions apply.
ftp> ls -la /ResAppls
200 PORT command successful.
150 Opening ASCII mode data connection for 'ls'.
226 Transfer complete.
ftp: 1778 bytes received in 1.96Seconds 0.91Kbytes/sec.
ftp> cd /ResAppls
250 CWD command successful.
ftp> pwd
257 "/ResAppls" is current directory.
ftp> ls -la
200 PORT command successful.
150 Opening ASCII mode data connection for 'ls'.
total 22
drwxrwxrwx  1 root     bin             3 Mar 21 16:05 .
drwxr-xr-x  0 root     bin            34 Mar  4 12:50 ..
-rw-rw-rw-  1 tpfdfltu tpfdfltg  5111666 Mar 21 16:05 992E2BETKTS19MAR17.txt
226 Transfer complete.
ftp: 206 bytes received in 0.00Seconds 206000.00Kbytes/sec.
ftp> get 992E2BETKTS19MAR17.txt
200 PORT command successful.
150 Opening ASCII mode data connection for '992E2BETKTS19MAR17.txt' (5476785 byt
es).
226 Transfer complete.
ftp: 5476785 bytes received in 11.07Seconds 494.65Kbytes/sec.
ftp> bye
221 Goodbye.

U:\>
```

Local Computer -> z/TPF File System


Administrator: Command Prompt

```
Microsoft Windows [Version 6.1.7601]
Copyright (c) 2009 Microsoft Corporation.  All rights reserved.

U:\>ftp 216.113.145.97
Connected to 216.113.145.97.
220 TPF FTP server (Version 1.01) ready.
User (216.113.145.97:(none)): anonymous
331 Guest login ok, type your name as password.
Password:
230 Guest login ok, access restrictions apply.
ftp> cd /ResAppls
250 CWD command successful.
ftp> ls -la
200 PORT command successful.
150 Opening ASCII mode data connection for 'ls'.
total 2
drwxrwxrwx   1 root   bin     2 Oct   1 21:41 .
drwxr-xr-x   0 root   bin    34 Mar  17 04:58 ..
226 Transfer complete.
ftp: 98 bytes received in 0.04Seconds 2.45Kbytes/sec.
ftp> put 992E2BETKTS19MAR17.txt
200 PORT command successful.
150 Opening ASCII mode data connection for '992E2BETKTS19MAR17.txt'.
226 Transfer complete.
ftp: 5476785 bytes sent in 13.76Seconds 397.94Kbytes/sec.
ftp> ls -la
200 PORT command successful.
150 Opening ASCII mode data connection for 'ls'.
total 22
drwxrwxrwx   1 root   bin          3 Mar 23 13:39 .
drwxr-xr-x   0 root   bin         34 Mar 17 04:58 ..
-rw-r--r--   1 ftp    bin    5111652 Mar 23 13:39 992E2BETKTS19MAR17.txt
226 Transfer complete.
ftp: 179 bytes received in 0.00Seconds 179000.00Kbytes/sec.
ftp> bye
221 Goodbye.
```

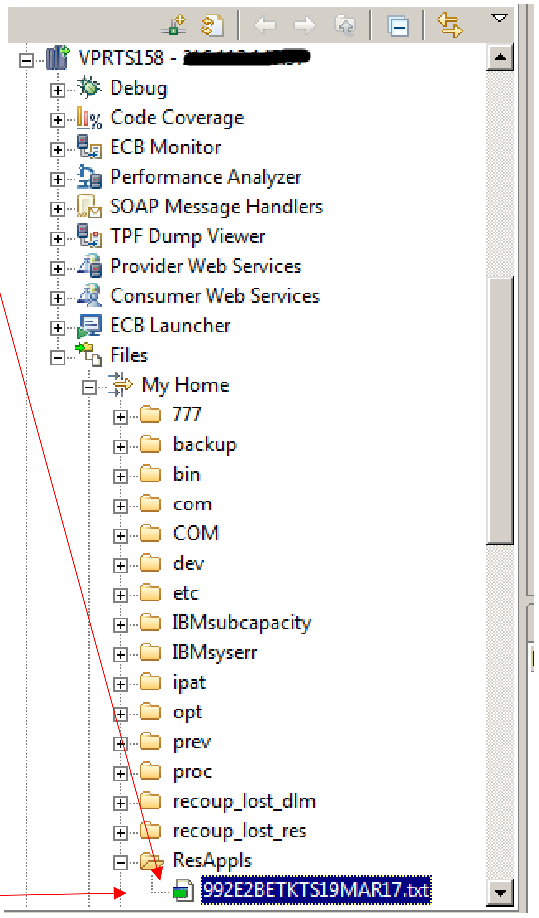
One TPF Lab -> Another TPF Lab.



Remote Systems | Team

- VPRTS157 - [redacted]
  - Debug
  - Code Coverage
  - ECB Monitor
  - Performance Analyzer
  - SOAP Message Handlers
  - TPF Dump Viewer
  - Provider Web Services
  - Consumer Web Services
  - ECB Launcher
  - Files
    - My Home
      - 777
      - backup
      - bin
      - com
      - COM
      - dev
      - etc
      - IBMsubcapacity
      - IBMsyserr
      - ipat
      - opt
      - prev
      - proc
      - recoup_lost_dlm
      - recoup_lost_res
      - ResAppls
        - 992E2BETKTS19MAR17.txt

We can further test it for PNR and TKT data transfer from one system to another system



- VPRTS158 - [redacted]
  - Debug
  - Code Coverage
  - ECB Monitor
  - Performance Analyzer
  - SOAP Message Handlers
  - TPF Dump Viewer
  - Provider Web Services
  - Consumer Web Services
  - ECB Launcher
  - Files
    - My Home
      - 777
      - backup
      - bin
      - com
      - COM
      - dev
      - etc
      - IBMsubcapacity
      - IBMsyserr
      - ipat
      - opt
      - prev
      - proc
      - recoup_lost_dlm
      - recoup_lost_res
      - ResAppls
        - 992E2BETKTS19MAR17.txt

## Other Benefits if suitable after feasibility study!!!

## Generating TPF reports via New approach - Will process it fast(Save IO), MIPS Saver and more Flexible.

1. If a report is around 6.5 MB of data – Current TPF will save it in around 1600 4K block – means around 1600 IO operation.
2. Then another challenge is to display that report via multiple MDs and capture it.
3. Then another challenge is to send it.

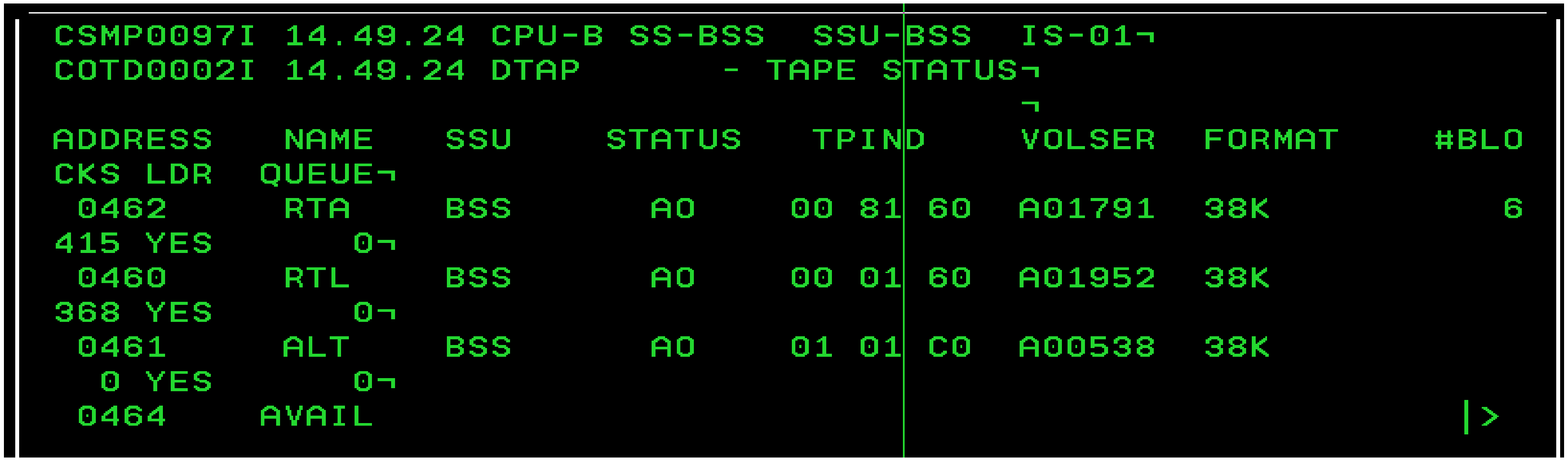
Via new approach:

Take big work area, Keep writing it there – then at the end write whole TPF data in Z/TPF File Systems.

1. Its faster – saved IO.
2. Saved MIPS while not doing multiple MDs.
3. We can FTP OR directly copy/Paste/delete from there.

## Tape:

- Currently we write Bulky TPF data on Tape and process them via traditional ways – We can write data directly on z/TPF File System and can FTP or read directly from there and delete it as per utilization.



```
CSMP0097I 14.49.24 CPU-B SS-BSS   SSU-BSS   IS-01¬
COTD0002I 14.49.24 DTAP         - TAPE STATUS¬
                                            ¬
ADDRESS   NAME    SSU     STATUS    TPIND      VOLSER   FORMAT     #BLO
CKS LDR  QUEUE¬
 0462     RTA     BSS       A0     00 81  60   A01791   38K           6
415 YES     0¬
 0460     RTL     BSS       A0     00 01  60   A01952   38K
368 YES     0¬
 0461     ALT     BSS       A0     01 01  C0   A00538   38K
  0 YES     0¬
 0464    AVAIL                                                      |>
```

This is a concept of writing the data at common area(z/TPF File System ) – which can be read by TPF, Open source, within the system and outside of System – and this concept can be used in multiple many other ways as per the need.

## Precaution – Storage data cleanup

z/TPF file system is not on a separate File System, it resides on same TPF File System itself.

So whenever we plan to write TPF data there, we need to consider current TPF File System capabilities and procedure to delete newly added data as soon they are not needed otherwise storage can full up quickly.

NEXT STEPS

We have tested this in Test System only – There is a need to evaluate security and performance parameters in Production before using this concept.

Questions