Linux 服务器与终端加密机的交互报文格式

1. 阅读说明

设备：指终端加密机
服务器：指运行 Server 端的 Linux 服务器
表格的每行为 4 字节，按长、中、短三种形式分别表示 4、2、1 字节，均为无符号
进制：表格中所有数据均为 16 进制
数据格式：报文传输时所有 2/4 字节均为网络序，报文解释时已转为主机序
（例：2 字节 0x003c 表示报文长度，则长度为 60 字节）
报文头：每个报文的前 8 字节（表格前 2 行）是报文头，最短的数据包只含报文头
随机值：因为是模拟程序，如果某项写随机值，则发送数据时，该项用随机数填充即可
（例：“设备的 FLASH 大小”，大小是 2 字节，要求是随机数，则发送时用 rand()
产生一个 0-65535 间数字即可，如果是字符串，产生一个字母+数字组合
即可，检查时，看 Client 产生的随机值是否与写入数据库的对应项的值
一致来判定正确性）

2. 设备连接

功能：设备向 Server 端的指定端口发起 TCP 连接
方向：设备=>服务器

3. 服务器向设备发送认证请求

- 功能：设备 TCP 连接成功后，服务器端向设备发送认证串，只有通过认证的设备才能向服务器发送信息
- 方向：服务器=>设备

0 7 8 15 16 23 24 31

0x11	0x01	报文的总长度（含 8 字节的报文头）	
0x0000		报文的数据长度（不含 8 字节的报文头）	
服务器端主版本号		次 1 版本号	次 2 版本号
设备连接服务器失败后的再次连接间隔（秒）		设备传输成功后的再次传输时间间隔（秒）	
是否允许空终端(0/1)	pad	pad	Pad
认证串（32 字节）			
Step1 生成的 random_num			
Step1 生成的 svr_time			

【版本号】：表示形式为 x.x.x，依次为主版本号、次 1 版本号、次 2 版本号
0x0000（0）及 0xFFFF（65535）另有含义，不能做为 ID 使用

【设备连接服务器失败后的再次连接间隔（秒）】：Client 如果连接服务器失败，则间隔的此值表示的秒数后再次连接服务器

【设备传输成功后的再次传输时间间隔（秒）】：Client 端发送完成后，如果 ts.conf 中的“进程接收成功后退出”为 0，则间隔的此值表示的秒数后再次连接服务器并发送数据

【是否允许空终端】：1-允许/0-不允许

【认证串】：必须是"yzmond:id*str&to!tongji@by#Auth^",将该认证串进行简单的加密后再放入报文中，发送给设备,加密的步骤如下：

Step1：
```
u_int random_num, svr_time;
int   pos;
random_num = (u_int)rand();
svr_time   = (u_int)time(0);
svr_time   =  svr_time ^ (u_int)0xFFFFFFFF;
pos        = (random_num % 4093);
```

Step2：将 32 字节的认证串与密钥数组（u_char secret[4096]，具体内容见附件）进行 32 字节的按位异或操作（从 secret 数组的第 pos 位置开始，pos=++pos%4093 即可），做一个简单加密

Step3：加经过 Step2 加密的认证串放入报文的认证串部分（32 字节）

- 设备收到后的检查规则

1、 如果收到的版本号<2.0.0，则发送“最低版本要求”报文

2、 如果收到的服务器时间<2017.1.1 00:00:00，则提示“数字证书过期”，关闭连接

3、 设备收到后对认证串进行解密（相同位置异或操作），如果认证串内容不匹配则提示“认证非法”，关闭 TCP 连接

4. 设备向服务器发送最低版本要求

- 功能：设备 TCP 连接成功后，服务器端向设备发送认证串，只有通过认证的设备才能向服务器发送信息
- 方向：设备=>服务器

0x91	0x00	报文的总长度（含 8 字节的报文头）	
0x0000		报文的数据长度（不含 8 字节的报文头）	
要求的最低服务器端主版本号		次 1 版本号	次 2 版本号

- 服务器收到后的检查规则

提示本程序的版本“*.*.*”低于设备要求的最低版本“*.*.*”，关闭 TCP 连接

5. 设备向服务器发送认证串及基本配置信息

- 功能：设备收到服务器发送的认证请求后，向服务器发送的应答报文
- 方向：设备=>服务器

0 7 8 15 16 23 24 31

0x91	0x01	报文的总长度（含 8 字节的报文头）	
0x0000		报文的数据长度（不含 8 字节的报文头）	
设备的 CPU 主频		设备的 RAM 大小（单位 MB）	
设备的 FLASH 大小（单位 MB）		设备的内部序列号	
设备的组序列号（任意字符组成，最长 16 字节）			
设备的型号（任意字符组成，最长 16 字节）			
设备的软件版本号（任意字符组成，最长 16 字节）			
以太口数量（0/1/2）	同步口数量（0/1/2）	异步口数量(0/8/16)	交换口数量(0/8/16/24)
USB 口数量（0/1）	打印口数量(0/1)	pad	pad
设备的机构号（9 位编码，对应数据库的 devid）			
机构内序号(定值 1)	pad	pad	
认证串（32 字节）			
Step1 生成的 random_num			

【CPU 主频】: 取 Linux 系统下 /proc/cpuinfo 文件的 cpu MHz 项的值（只取第一个 CPU 的）

```
[root@RHEL74-X64 ~]# cat /proc/cpuinfo
processor       : 0
vendor_id       : GenuineIntel
cpu family      : 6
model           : 69
model name      : Intel(R) Core(TM) i5-4210U CPU @ 1.70GHz
stepping        : 1
microcode       : 0x23
cpu MHz         : 2400.639
cache size      : 3072 KB
```

【RAM 大小】：取 Linux 系统下 /proc/meminfo 文件的首行（转 MB 后取整）

```
[root@RHEL74-X64 ~]# cat /proc/meminfo
MemTotal:        1867048 kB
```

【认证串】：必须是"yzmond:id*str&to!tongji@by#Auth^"

- 报文加密：需将本报文中部分内容进行简单的加密后再发送给服务器，加密的步骤如下：

  Step1：u_int random_num;
  int pos;
  random_num = (u_int)rand();
  pos = (random_num % 4093);

  Step2：将报文的数据部分（CPU 主频 - 认证串，共 104 字节）与密钥数组（u_char secret[4096]，具体内容见附件）进行按位异或操作（从 secret 数组的第 pos 位置开始，pos=++pos%4093 即可），做一个简单加密

  Step3：将经过 Step2 加密的认证串替换原位置的内容（104 字节）

- 服务器收到后的检查规则

  1、 对报文的加密部分进行解密（相同位置异或操作），如果认证串内容不匹配则提示“不接受数据”，关闭 TCP 连接

  2、 如果认证串匹配则提示认证成功，向日志文件中打印 CPU 主频等信息（查看是否与发送的内容一致），进入下一步

- 数据库对应表项

devstate_base_devid	本报文 - 设备的机构号
devstate_base_devno	本报文 - 机构内序号（定值 1）
devstate_base_time	本次写入数据库的时间
devstate_base_ipaddr	Client 端的 IP 地址
devstate_base_sid	本报文 - 设备的组序列号+设备的内部序列号 **（均为随机值，%s-%d 方式组合）**
devstate_base_type	本报文 - 设备的型号**（随机值）**
devstate_base_version	本报文 - 设备的软件版本号**（随机值）**
devstate_base_cpu	本报文 - 设备的 CPU 主频
devstate_base_sdram	本报文 - 设备的 SDRAM 大小
devstate_base_flash	本报文 - 设备的 FLASH 大小**（随机值）**
devstate_base_ethnum	本报文 - 以太口数量
devstate_base_syncnum	本报文 - 同步口数量
devstate_base_asyncnum	本报文 - 异步口数量
devstate_base_switchnum	本报文 - 交换机数量
devstate_base_usbnum	本报文 - USB 口数量(将 0/1 转换为不存在/存在)
devstate_base_prnnum	本报文 - 打印口数量(将 0/1 转换为不存在/存在)

6. 服务器向设备发送各种取信息的请求

- 功能：认证通过后，服务器向设备发送取信息请求，设备根据请求信息向服务器发送相应的应答信息
- 包含若干种信息，具体如下

6.1. 取系统信息

- 方向：服务器=>设备

0 7 8 15 16 23 24 31

0x11	0x02	报文的总长度（含 8 字节报文头）
0x0000		0x0000：数据的长度（0 字节）

- 方向：设备=>服务器

0 7 8 15 16 23 24 31

0x91	0x02	报文的总长度（含 8 字节报文头）
0x0000		数据的长度（不含报文头）
user CPU time:		
nice CPU time		
system CPU time		
idle CPU time		
freed_memory		

【前 4 个 CPU time】：取 Linux 系统下 /proc/stat 文件第 1 行的前 4 项即可

```
[root@RHEL71-X64 ~]# cat /proc/stat
cpu  5797 0 13013 5101426 315 3 662 0 0 0
```

【freed_memory】：取 Linux 系统下 /proc/meminfo 文件的第 2/4/5 项之和

```
[root@RHEL71-X64 ~]# cat /proc/meminfo
MemTotal:        1870512 kB
MemFree:         1003784 kB
MemAvailable:    1327724 kB
Buffers:             888 kB
Cached:           429436 kB
```

- 数据库对应表项

devstate_base_cpu_used	CPU 占用率(user+system)/(user+nice+system_idle)
devstate_base_sdram_used	内存使用情况(freed_memory/RAM 大小)，注意单位转换

6.2. 取配置信息

- 方向：服务器=>设备

0 7 8 15 16 23 24 31

0x11	0x03	报文的总长度（含 8 字节报文头）
0x0000		0x0000：数据的长度（0 字节）

- 方向：设备=>服务器

0 7 8 15 16 23 24 31

0x91	0x03	报文的总长度（含 8 字节报文头）
0x0000		数据的长度（不含报文头）
设备的配置信息（长度不定）		

【注:】实际设备（嵌入式 Linux）会取真实配置信息，模拟程序只需要取附件给出的 config.dat 文件中的内容，最后加一个\0 即可（可以自行构造 config.dat，若文件长度超过 8191 字节，则只取前 8191 个字节+\0）

- 数据库对应表项

devstate_base_config	设备的配置信息

6.3. 取进程信息

- 方向：服务器=>设备

0 7 8 15 16 23 24 31

0x11	0x04	报文的总长度（含 8 字节报文头）
0x0000		0x0000：数据的长度（0 字节）

- 方向：设备=>服务器

0 7 8 15 16 23 24 31

0x91	0x04	报文的总长度（含 8 字节报文头）
0x0000		数据的长度（不含报文头）
设备的进程信息（长度不定）		

【注:】实际设备（嵌入式 Linux）会取真实进程信息，模拟程序只需要取附件给出的 process.dat 文件中的内容，最后再加一个\0 即可（可以自行构造 process.dat，若文件长度超过 8191 字节，则只取前 8191 个字节+\0）

- 数据库对应表项

devstate_base_process	设备的进程信息

6.4. 取以太口信息

- 方向：服务器=>设备

0 7 8 15 16 23 24 31

0x11	0x05	报文的总长度（含 8 字节报文头）
0x0000/0x0001（Ethernet0/1 口）		0x0000：数据的长度（0 字节）

- 方向：设备=>服务器

0 7 8 15 16 23 24 31

0x91	0x05	报文的总长度（含 8 字节报文头）	
0x0000/0x0001（Ethernet0/1 口）		数据的长度（不含报文头）	
是否存在（1/0）	是否配置（1/0）	UP（1）/Down（0）	pad
MAC[0]	MAC[1]	MAC[2]	MAC[3]
MAC[4]	MAC[5]	options	
IP 地址			
子网掩码			
IP 地址（子接口 1）			
子网掩码（子接口 1）			
IP 地址（子接口 2）			

子网掩码（子接口 2）
IP 地址（子接口 3）
子网掩码（子接口 3）
IP 地址（子接口 4）
子网掩码（子接口 4）
IP 地址（子接口 5）
子网掩码（子接口 5）
统计数据 - 收到的字节数
统计数据 - 收到的包数
统计数据 - 收到的错误包数
统计数据 - 收到的丢弃包数
统计数据 - 收到的 fifo 包数
统计数据 - 收到的帧数
统计数据 - 收到的压缩包数
统计数据 - 收到的广播包数
统计数据 - 发送的字节数
统计数据 - 发送的包数
统计数据 - 发送的错误包数
统计数据 - 发送的丢弃包数
统计数据 - 发送的 fifo 包数
统计数据 - 发送的帧数
统计数据 - 发送的压缩包数
统计数据 - 发送的广播包数

【注】：两个以太网口的信息分两个包取得

【optons】：以下三种信息按 bit 位或操作，任意组合均可

0x0001 - 100MB 速率，不置位就是 10MB 速率

0x0002 - 全双工模式，不置位就是半双工

0x0004 - 自动协商方式，不置位就是非自动协商

例：options = 0x0007 100MB + 全双工 + 自动协商

0x0000 10MB + 半双工 + 非自动协商

【MAC 地址】：模拟程序任意设置均可

【IP 地址及掩码】：模拟程序任意设置均可

【所有统计数据】：Ethernet0 的信息取 /proc/net/dev 文件中的第 1 张网卡的对应数据

Ethernet1 的信息可取第 2 张网卡（lo:环回口）的对应数据

```
[root@RHEL71-X64 ~]# cat /proc/net/dev
Inter-|   Receive                                                |  Transmit
 face |bytes    packets errs drop fifo frame compressed multicast|bytes    packets errs drop fif
o colls carrier compressed
eno16777728:  751500   11367    0 2667    0     0          0         0   137838    2125    0
0     0       0       0          0
    lo:  136720     488    0    0    0     0          0         0   136720     488    0    0
0     0       0          0
eno33554968: 6562019   21964    0    0    0     0          0         0  3961750   15947    0
0     0       0       0          0
```

- 数据库对应表项

devstate_base_eth0_ip	eth0 的 ip 地址

devstate_base_eth0_mask	eth0 的 mask
devstate_base_eth0_mac	eth0 的 mac 地址
devstate_base_eth0_state	eth0 的状态（up/down）
devstate_base_eth0_speed	eth0 的速度（100MB/10MB）
devstate_base_eth0_duplex	eth0 的工作方式（全双工/半双工）
devstate_base_eth0_autonego	eth0 的是否自动协商（是/否）
devstate_base_eth0_txbytes	eth0 发送的字节数
devstate_base_eth0_txpackets	eth0 发送的包数
devstate_base_eth0_rxbytes	eth0 接收的字节数
devstate_base_eth0_rxpackets	eth0 接收的包数
devstate_base_eth1_ip	eth1 的 ip 地址
devstate_base_eth1_mask	eth1 的 mask
devstate_base_eth1_mac	eth1 的 mac 地址
devstate_base_eth1_state	eth1 的状态
devstate_base_eth1_speed	eth1 的速度（100/10）
devstate_base_eth1_duplex	eth1 的工作方式（全双工/半双工）
devstate_base_eth1_autonego	eth1 的是否自动协商
devstate_base_eth1_txbytes	eth1 发送的字节数
devstate_base_eth1_txpackets	eth1 发送的包数
devstate_base_eth1_rxbytes	eth1 接收的字节数
devstate_base_eth1_rxpackets	eth1 接收的包数

【注】：以太网子接口的信息未写入数据库中，统计信息收发也仅各有 4 项被写入数据库中

6.5. 取同步口信息（为简化，Server 不发此包，数据库中也无对应存储项）

6.6. 取 USB 口上是否存在 U 盘的信息

- 方向：服务器=>设备

0 7 8 15 16 23 24 31

0x11	0x07	报文的总长度（含 8 字节报文头）
0x0000		0x0000：数据的长度（0 字节）

- 方向：设备=>服务器

0 7 8 15 16 23 24 31

0x91	0x07	报文的总长度（含 8 字节报文头）	
0x0000		数据的长度（不含报文头）	
是否插入 U 盘（1/0）	pad	pad	pad

- 数据库对应表项

devstate_base_usbstate	是否插入 U 盘（1/0），转换为“已插入”/“未插入”

- 如果收到了插入 U 盘的信息，则向 Client 端发送“取 U 盘上的文件列表信息”的报文

6.7. 取 U 盘上的文件列表信息

- 方向：服务器=>设备

0 7 8 15 16 23 24 31

0x11	0x0c	报文的总长度（含 8 字节报文头）
0x0000		0x0000：数据的长度（0 字节）

- 方向：设备=>服务器

0 7 8 15 16 23 24 31

0x91	0x0c	报文的总长度（含 8 字节报文头）
0x0000		数据的长度（不含报文头）
U 盘上的文件列表信息（长度不定）		

【注:】实际设备（嵌入式 Linux）会取 U 盘的实时信息，模拟程序只需要取附件给出的 usefiles.dat 文件中的内容，最后再加一个\0 即可（可以自行构造 usefile.dat，若文件长度超过 4095 字节，则只取前 4095 个字节+\0）

- 数据库对应表项

devstate_base_usbfiles	U 盘上的文件列表信息

6.8. 取打印口信息

- 方向：服务器=>设备

0 7 8 15 16 23 24 31

0x11	0x08	报文的总长度（含 8 字节报文头）
0x0000		0x0000：数据的长度（0 字节）

- 方向：设备=>服务器

0 7 8 15 16 23 24 31

0x91	0x08	报文的总长度（含 8 字节报文头）
0x0000		数据的长度（不含报文头）
服务是否启动（1/0）	pad	打印队列中现有任务数
打印机名称（任意字符组成，最长 32 字节）		

【打印队列中现有的任务数】: 若服务是否启动为 1，用模拟程序用 0-25 间随机数表示即可

- 数据库对应表项

devstate_base_prnname	打印机名称
devstate_base_prnstate	服务是否启动（0/1），转换为未启动/已启动

- 如果打印队列中有任务，则向 Client 端发送“取打印队列信息”的报文

6.9. 取打印队列信息

- 方向：服务器=>设备

0 7 8 15 16 23 24 31

0x11	0x0d	报文的总长度（含 8 字节报文头）
0x0000		0x0000：数据的长度（0 字节）

- 方向：设备=>服务器

0 7 8 15 16 23 24 31

0x91	0x0d	报文的总长度（含 8 字节报文头）
0x0000		数据的长度（不含报文头）
打印队列的信息（长度不定）		

【注:】为简化，可以直接填一个\0，返回长度=头长+1 即可

- 数据库对应表项

devstate_base_prnfiles	打印队列的信息

6.10. 取交换机口信息（为简化，Server 不发此包，数据库中对应项置 NULL 即可）

- 数据库对应表项

devstate_base_switchinfo	交换机配置信息，直接置 NULL 即可

6.11. 取终端服务信息

- 方向：服务器=>设备

0 7 8 15 16 23 24 31

0x11	0x09	报文的总长度（含 8 字节报文头）
0x0000		0x0000：数据的长度（0 字节）

- 方向：设备=>服务器

0 7 8 15 16 23 24 31

0x91	0x09	报文的总长度（含 8 字节报文头）	
0x0000		数据的长度（不含报文头）	
哑终端(1)是否使用	…	…	…
…	…	…	…
…	…	…	…
…	…	…	哑终端(16)是否使用
IP 终端(1)是否使用	…	…	…
…	…	…	…
IP 终端(253)是否使用	IP 终端(254)是否使用	配置的终端数量	

【哑终端 1-16】：在“设备向服务器发送认证串及基本配置信息”中，异步口（用于连接连接哑终端的借口）数量为 0/8/16

【IP 终端】：数量为 0-254 个

- 本报文填写方法

  1、 在“ts.conf”中，有两项分别是“最小配置终端数量”/“最大配置终端数量”，模拟程序首先产生一个在[最小...最大]之间的随机数 total（例：范围 3-50，产生的随机数 total=31 / 范围 5-28，产生的随机数 totla=17）

  2、 在“设备向服务器发送认证串及基本配置信息”报文中，要求异步口填写的值为 0/8/16（用于连接连接哑终端的借口），如果是 8/16，则随机产生 1-8/1-16 间的随机数 async_term_num（例：异步口数量 8，产生随机数 async_term_num=3）

  3、 本报文的哑终端 1-16 对应的字节中，随机挑选 async_term_num 个位置置 1，其余位置置 0（注：若 async_term_num > total，则 total = async_term_num）

  4、 令：ipterm_num = total - async_term_num，在本报文的 IP 终端 1-254 对应的字节中，随机挑选 ipterm_num 个位置置 1，其余置 0

【配置的终端数量】：填入一个在[total-270]间的随机数即可

- 数据库对应表项

devstate_base_tty_configed	配置的终端数量

6.12. 取哑终端/IP 终端的配置信息及对应虚屏的配置信息

- 方向：服务器=>设备

0 7 8 15 16 23 24 31

0x11	0x0a/0x0b：哑/IP	报文的总长度（含 8 字节报文头）
0x0001-0x00FE（1-16/1-254 代表终端编号）		0x0000：数据的长度（0 字节）

【注】：1、多个终端信息的获取请求分多个包发送设备

2、设备发来的“取终端服务信息”的应答包中哑终端 1-16/IP 终端 1-154 中对应位置置 1 的，才向设备发送本数据包

- 方向：设备=>服务器

0 7 8 15 16 23 24 31

0x91	0x0a/0x0b：哑/IP	报文的总长度（含 8 字节报文头，**含虚屏信息**）	
0x0001-0x00FE（1-16 代表哑终端编号）		数据的长度（不含报文头，**含虚屏信息**）	
端口号（1-254）	配置端口号（1-254）	当前活动虚屏编号	虚屏的总量
终端的 IP 地址			
终端类型：哑终端-字符串“串口终端”/IP 终端-“IP 终端”/“IP 代理”（共占用 12 字节）			
终端状态：字符串“正常”/“菜单”（共占用 8 字节，随机选择其一即可）			
此处跟 screen_num 个虚屏信息，每个虚屏信息的结构见后...(本行不计算在报文内!!!)			

【注】：多个终端的信息分多个包发送给服务器

【虚屏的总量】：在“ts.conf”中，有两项分别是“每个终端最小虚屏数量”/“每个终端最大虚屏数量”，模拟程序首先产生一个在[最小...最大]之间的随机数 screen_num（例：范围 3-16，产生的随机数 screen_num=9 / 范围 1-6，产生的随机数 screen_num=3）

【当前活动虚屏编号】：填入一个在[0 - screen_num-1]间的随机数即可

【终端的 IP 地址】：哑终端 - 填 0

IP 终端 - 任意一个符合 IP 地址要求的值即可

- 数据库对应表项

字段名	说明
devstate_ttyinfo_devid	同 devstate_base_devid
devstate_ttyinfo_devno	同 devstate_base_devno
devstate_ttyinfo_ttyno	哑终端：终端编号(1-16)+900 IP 终端：1-254
devstate_ttyinfo_time	本条记录写入数据库的时间
devstate_ttyinfo_readno	配置端口号
devstate_ttyinfo_type	终端类型
devstate_ttyinfo_state	终端状态
devstate_ttyinfo_ttyip	终端的 IP 地址
devstate_ttyinfo_scrnum	虚屏的总量

- 报文续：虚屏信息结构

0 7 8 15 16 23 24 31

虚屏编号（1-16）	pad	远端服务器的 TCP 端口号
远端服务器的 IP 地址		
虚屏协议：对应的通信协议，类似“SSH”/“专用 SSH”等任意构造即可（共占用 12 字节）		
虚屏状态：字符串“开机”/“关机”/“已登录”（共占用 8 字节，随机选择其一即可）		
该虚屏的提示串，类似“储蓄系统”/“基金开户”等任意构造即可（共占用 24 字节）		
该虚屏对应的终端类型，类似“vt100”/“vt220”等任意构造即可（共占用 12 字节）		
终端连接上来的时间		
发送给终端的字节数		
收到终端发来的字节数		
发送给远端服务器的字节数		
从远端服务器收到的字节数		
Ping 包的最小值		
Ping 包的平均值		
Ping 包的最大值		

【注】：一个终端有多个虚屏，因此这个报文会重复多次，接在终端信息报文的后面（终端报文中的长度需要包含若干虚屏的长度）

【终端连接上来的时间】：取当前系统时间即可（注意：time(0)的返回不是 4 字节）
【四个统计信息】：随机值即可
【三个 ping 包信息】：[0..123456]间的随机值即可

- 数据库对应表项

● 字段名	说明
devstate_scrinfo_devid	同 devstate_base_devid
devstate_scrinfo_devno	同 devstate_base_devno
devstate_scrinfo_ttyno	同 devstate_ttyinfo_ttyno
devstate_scrinfo_scrno	虚屏编号
devstate_scrinfo_time	本条记录写入数据库的时间
devstate_scrinfo_is_current	若 虚屏编号=当前活动虚屏编号+1，则填入*，否则为 NULL
devstate_scrinfo_protocol	虚屏协议
devstate_scrinfo_serverip	远端服务器的 IP 地址
devstate_scrinfo_serverport	远端服务器的 TCP 端口号
devstate_scrinfo_state	虚屏状态
devstate_scrinfo_ttytype	该虚屏对应的终端类型
devstate_scrinfo_tx_server	发送给远端服务器的字节数
devstate_scrinfo_rx_server	从远端服务器收到的字节数
devstate_scrinfo_tx_terminal	发送给终端的字节数
devstate_scrinfo_rx_terminal	收到终端发来的字节数
devstate_scrinfo_ping_min	Ping 包的最小值/10.0
devstate_scrinfo_ping_avg	Ping 包的平均值/10.0
devstate_scrinfo_ping_max	Ping 包的最大值/10.0

- 在多个终端的信息全部接收完成后，更新 devstate_base 中的对应项

devstate_base_tty_connected	本次接收到的信息的终端总数

- 特别说明：devstate_base 中的 devstate_base_sendreq 填 0 即可

7. 所有包均收到（双向）

- 功能：服务器向设备发送的所有取信息请求均得到了应答，本次接收完成，则双向发送此报文并完成本次连接，具体流程如下：
  Step1：服务器=>设备发送此报文
  Step2：设备收到后，应答此本报文（不关闭连接）
  Step3：服务器收到应答报文，关闭 TCP 连接
  Step4：设备判断 TCP 断开（read()<0），则关闭 TCP 连接

- 方向：服务器=>设备

0 7 8 15 16 23 24 31

0x11	0xff	报文的总长度（含 8 字节报文头）
0x0000		0x0000：数据的长度（0 字节）

- 方向：设备=>服务器

0 7 8 15 16 23 24 31

0x91	0xff	报文的总长度（含 8 字节报文头）
0x0000		数据的长度（不含报文头）

【注】：1、在“ts.conf”中有“进程接收成功后退出”项，置 1 则 Client 退出；置 0 则 Client 不退出，间隔一定时间后再次连接服务器（开始下一轮发送）

2、每次连接成功后的第一个报文（“服务器向设备发送认证请求”）中的“设备传输成功后的再次传输时间间隔（秒）”即指定了此时间间隔