Pcapyzer数据包辅助分析程序技术原理

**修订记录**

|  |  |  |  |
| --- | --- | --- | --- |
| 版本号 | 修订日期 | 修订人员 | 更新说明 |
| V.1.0 | 20191011 | 王子翰 | 建立初版 |
|  |  |  |  |

目录

[1. 概述 4](#_Toc22287300)

[1.1. 介绍 4](#_Toc22287301)

[1.2. 开发环境 4](#_Toc22287302)

[1.3. 程序说明 4](#_Toc22287303)

[2. 程序原理 4](#_Toc22287304)

[2.1. xml基本结构 4](#_Toc22287305)

[2.2. xml结构说明 5](#_Toc22287306)

[2.2.1. 元素说明 5](#_Toc22287307)

[2.2.2. 元素规则 6](#_Toc22287308)

[2.3. 属性说明 6](#_Toc22287309)

[2.3.1. 行为元素 6](#_Toc22287310)

[2.3.2. filter元素 6](#_Toc22287311)

[2.3.3. DatalinkLayer元素 6](#_Toc22287312)

[2.3.4. NetworkLayer元素 7](#_Toc22287313)

[2.3.5. TransportLayer元素 7](#_Toc22287314)

[2.3.6. ApplicationLayer元素 8](#_Toc22287315)

[2.3.7. CodeFilter元素 8](#_Toc22287316)

[2.3.7.1. Type元素 9](#_Toc22287317)

[2.3.7.2. Length元素 9](#_Toc22287318)

[2.3.7.3. Value元素 9](#_Toc22287319)

[2.3.8. key/value元素 9](#_Toc22287320)

[3. 使用说明 10](#_Toc22287321)

[3.1.1. 加载xml 10](#_Toc22287322)

[3.1.2. 生成数据包 12](#_Toc22287323)

[3.1.3. 建立连接 12](#_Toc22287324)

[3.1.4. 发送数据 13](#_Toc22287325)

[3.1.5. 接收数据 13](#_Toc22287326)

[3.1.6. 断开连接 13](#_Toc22287327)

[4. 爬虫程序 13](#_Toc22287328)

[4.1. 程序说明 13](#_Toc22287329)

[4.2. 插件代码说明 14](#_Toc22287330)

1. 概述
   1. 介绍

Pcapyzer数据包辅助分析程序为本人在协议分析工作中，当需要对数据包中具有某些特征的数据进行过滤和统计时，使用现有的工具如wireshark等不能满足需求，所以开发此软件，意在提高协议分析工作效率。

* 1. 开发环境

系统：Windows 7 旗舰版 32/64

编译器：Microsoft Visual Studio 2015

语言：MFC/C++

* 1. 程序说明

../config/\*.xml：记录数据包的过滤方式的配置文件。

../Pcapyzer.exe:即主程序。

Packetyzer.dll：开源的数据包解析程序。

Packet.dll：winpcap第三方库。

wpcap.dll：winpcap第三方库。

1. 程序原理
   1. xml基本结构

<?xml version="1.0" encoding="UTF-8"?>

<config>

<weixin\_android\_chat\_pic\_receive \*>

<filter \*>

<DatalinkLayer>

<CodeFilter/>

……

</DatalinkLayer>

<NetworkLayer>

<CodeFilter/>

……

<NetworkLayer>

<TransportLayer>

<CodeFilter/>

……

</TransportLayer>

<ApplicationLayer/>

<CodeFilter/>

……

<ApplicationLayer/>

</filter>

……

</weixin\_android\_chat\_pic\_receive>

……

</config>

* 1. xml结构说明
     1. 元素说明

config：根元素，元素名称不可更改。

weixin\_android\_chat\_pic\_receive：行为元素，元素名称为用户自定义。如此标识微信安卓聊天图片接收行为。

filter：过滤规则元素，元素名称不可更改。标识行为下的一条过滤规则。

DatalinkLayer：数据链路层元素，元素名称不可更改。标识数据链路层协议的过滤规则。

NetworkLayer：网络层元素，元素名称不可更改。标识网络层协议的过滤规则。

TransportLayer：传输层元素，元素名称不可更改。标识传输层协议的过滤规则。

ApplicationLayer：应用层元素，元素名称不可更改。标识应用层私有协议的过滤规则。

* + 1. 元素规则

1. 一个行为元素下可包含多个filter子元素。
2. 一个filter元素下一般最多只包含一个DatalinkLayer/ NetworkLayer/ TransportLayer/ApplicationLayer子元素。
3. 一个ApplicationLayer元素下可包含多个CodeFilter元素。
   1. 属性说明
      1. 行为元素

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 属性 | 必填 | 缺省值 | 规则影响 | 备注 |
|  |  |  |  |  |

示例：暂无。

* + 1. filter元素

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 属性 | 必填 | 缺省值 | 规则影响 | 备注 |
| name | 是 |  | 否 | 过滤规则的标识名称，用于软件中显示。 |
|  |  |  |  |  |

示例：<filter name="1">

* + 1. DatalinkLayer元素

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 属性 | 必填 | 缺省值 | 规则影响 | 备注 |
| LayerType | 是 |  | 是 | 标识网络分层协议类型  Ethernet：数据链路层以太网协议  IPv4：网络层IPV4协议  IPv6：网络层IPV6协议  TCP：传输层TCP协议  UDP：传输层UDP协议  PrivateProtocol：应用层私有协议 |
| isFilter | 否 | 0 | 是 | 标识规则是否（1/0）进行过滤 |
|  |  |  |  |  |

示例：<DatalinkLayer LayerType="Ethernet" isFilter="0">

* + 1. NetworkLayer元素

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 属性 | 必填 | 缺省值 | 规则影响 | 备注 |
| LayerType | 是 |  | 是 | 标识网络分层协议类型  Ethernet：数据链路层以太网协议  IPv4：网络层IPV4协议  IPv6：网络层IPV6协议  TCP：传输层TCP协议  UDP：传输层UDP协议  PrivateProtocol：应用层私有协议 |
| isFilter | 否 | 0 | 是 | 标识规则是否（1/0）进行过滤 |
|  |  |  |  |  |

示例：<NetworkLayer LayerType="IPv4" isFilter="0">

* + 1. TransportLayer元素

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 属性 | 必填 | 缺省值 | 规则影响 | 备注 |
| LayerType | 是 |  | 是 | 标识网络分层协议类型  Ethernet：数据链路层以太网协议  IPv4：网络层IPV4协议  IPv6：网络层IPV6协议  TCP：传输层TCP协议  UDP：传输层UDP协议  PrivateProtocol：应用层私有协议 |
| isFilter | 否 | 0 | 是 | 标识规则是否（1/0）进行过滤 |
|  |  |  |  |  |

示例：<TransportLayer LayerType="TCP" isFilter="0">

* + 1. ApplicationLayer元素

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 属性 | 必填 | 缺省值 | 规则影响 | 备注 |
| LayerType | 是 |  |  | 标识网络分层协议类型  Ethernet：数据链路层以太网协议  IPv4：网络层IPV4协议  IPv6：网络层IPV6协议  TCP：传输层TCP协议  UDP：传输层UDP协议  PrivateProtocol：应用层私有协议 |
| isFilter | 否 | 0 | 是 | 标识规则是否（1/0）进行过滤 |
| style | 否 | packet | 是 | packet:标识数据为报文过滤  stream:标识数据为流过滤 |
|  |  |  |  |  |

示例：<ApplicationLayer LayerType="PrivateProtocol" isFilter="1" style="packet">

* + 1. CodeFilter元素

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 属性 | 必填 | 缺省值 | 规则影响 | 备注 |
| codetype | 是 |  | 是 | 标识过滤数据类型  SourceMAC：源MAC  DestinationMAC：目的MAC  Protocol：协议类型  SourcePort：源端口  DestinationPort：目的端口  CODE：特征码数据  BACK\_FILL：回填数据，如长度  GRAY：未知数据，此时元素值标记了跳过解析的数据长度  TLV：TLV结构数据，此时解析TLV子元素 |
| valuetype | 是 |  | 是 | 标识元素值解析类型  VALUE\_BYTE：1字节数据  VALUE\_WORD：2字节数据  VALUE\_DWORD：4字节数据  VALUE\_DDWORD：8字节数据  VALUE\_STRING：字符串数据  VALUE\_HEXSTR：16进制串数据  VALUE\_CHILD：嵌套子元素 |
| offsettype | 是 |  | 是 | 标识数据偏移类型  OFFSET\_FRONT：起始偏移  OFFSET\_ORDER：顺序偏移  OFFSET\_FIXED：固定位置偏移  OFFSET\_BACK：尾部反序偏移 |
| offset | 否 |  | 是 | 当offsettype为OFFSET\_FRONT，OFFSET\_BACK，OFFSET\_FIXED时，标识偏移位置 |
| endian | 否 | 1 | 是 | 标识值的大小（1/0）端序 |
| isFilter | 否 | 0 | 是 | 标识规则是否（1/0）进行过滤 |
|  |  |  |  |  |

示例：

<CodeFilter codetype="CODE" valuetype="VALUE\_BYTE" offsettype="OFFSET\_FRONT" offset="0" isFilter="1" endian="1">3e</CodeFilter>

<CodeFilter codetype="GRAY" valuetype="VALUE\_DWORD" offsettype="OFFSET\_ORDER" isFilter="1" endian="1">23</CodeFilter>

<CodeFilter codetype="TLV" valuetype="VALUE\_CHILD" offsettype="OFFSET\_ORDER" isFilter="1">

<Type valuetype="VALUE\_DWORD" isFilter="0"></Type>

<Length valuetype="VALUE\_DWORD" isFilter="1"></Length>

<Value valuetype="VALUE\_STRING" isFilter="1">uin</Value>

</CodeFilter>

* + - 1. Type元素

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 属性 | 必填 | 缺省值 | 规则影响 | 备注 |
| valuetype | 是 |  | 是 | 标识元素值解析类型  VALUE\_BYTE：1字节数据  VALUE\_WORD：2字节数据  VALUE\_DWORD：4字节数据  VALUE\_DDWORD：8字节数据  VALUE\_STRING：字符串数据  VALUE\_HEXSTR：16进制串数据  VALUE\_CHILD：嵌套子元素 |
| isFilter | 否 | 0 | 是 | 标识规则是否（1/0）进行过滤 |
| endian | 否 | 1 | 是 | 标识值的大小（1/0）端序 |
|  |  |  |  |  |

示例：<Type valuetype="VALUE\_HEXSTR" isFilter="0" endian="1">0102</Type>

* + - 1. Value元素

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 属性 | 必填 | 缺省值 | 规则影响 | 备注 |
| valuetype | 是 |  | 是 | 标识元素值解析类型  VALUE\_BYTE：1字节数据  VALUE\_WORD：2字节数据  VALUE\_DWORD：4字节数据  VALUE\_DDWORD：8字节数据  VALUE\_STRING：字符串数据  VALUE\_HEXSTR：16进制串数据  VALUE\_CHILD：嵌套子元素 |
| isFilter | 否 | 0 | 是 | 标识规则是否（1/0）进行过滤 |
| isPreLen | 是 | 0 | 是 | 标识元素值是否（1/0）有前置长度。 |
| preLenType | 否 | c | 是 | 当isPreLen值为1时，此字段必填。  c:前置长度占1字节。  w:前置长度占2字节。  dw:前置长度占4字节。  dww前置长度占8字节。 |
| preLenEndian | 否 | 1 | 是 | 当isPreLen值为1时，此字段必填。  标识前置长度的大小（1/0）端序。 |
| isSelfinLen | 是 |  | 是 | 当isPreLen值为1时，此字段必填。  标识前置长度是否（1/0）包含长度字段本身长度。 |
| endian | 否 | 1 | 是 | 标识值的大小（1/0）端序 |
|  |  |  |  |  |

示例：<Value valuetype="VALUE\_STRING" isFilter="1" isPreLen="1" preLenType="w" preLenEndian="1" isSelfinLen="1" endian="1">uin</Value>

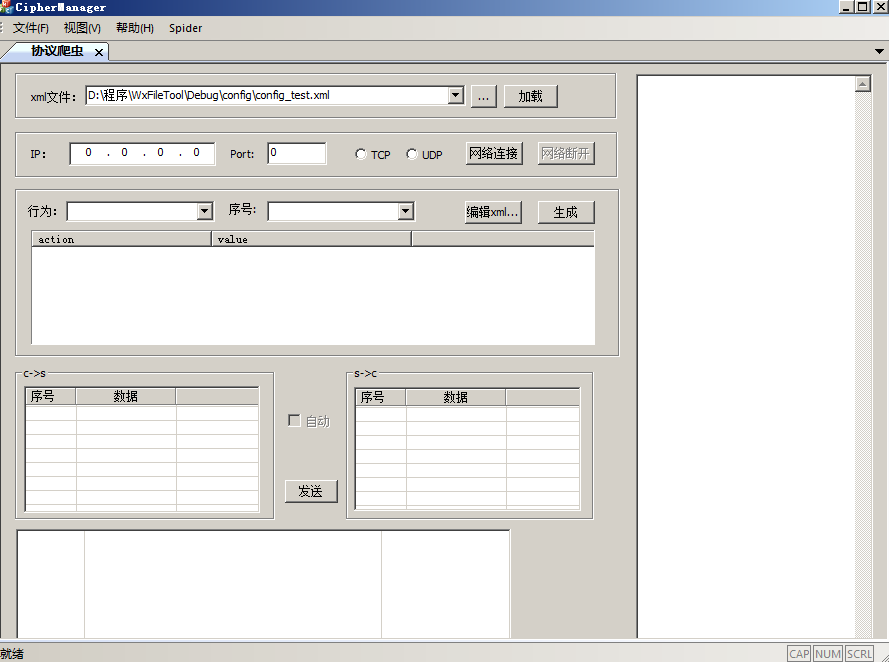
* + - 1. BackFill元素

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 属性 | 必填 | 缺省值 | 规则影响 | 备注 |
| valuetype | 是 |  | 是 | 标识元素值解析类型  VALUE\_BYTE：1字节数据  VALUE\_WORD：2字节数据  VALUE\_DWORD：4字节数据  VALUE\_DDWORD：8字节数据  VALUE\_STRING：字符串数据  VALUE\_HEXSTR：16进制串数据  VALUE\_CHILD：嵌套子元素 |
| FillType | 是 |  | 是 | 标识回填数据类型  LEN：长度回填字段 |
| isFilter | 否 | 0 | 是 | 标识规则是否（1/0）进行过滤 |
| endian | 否 | 1 | 是 | 标识值的大小（1/0）端序 |
|  |  |  |  |  |

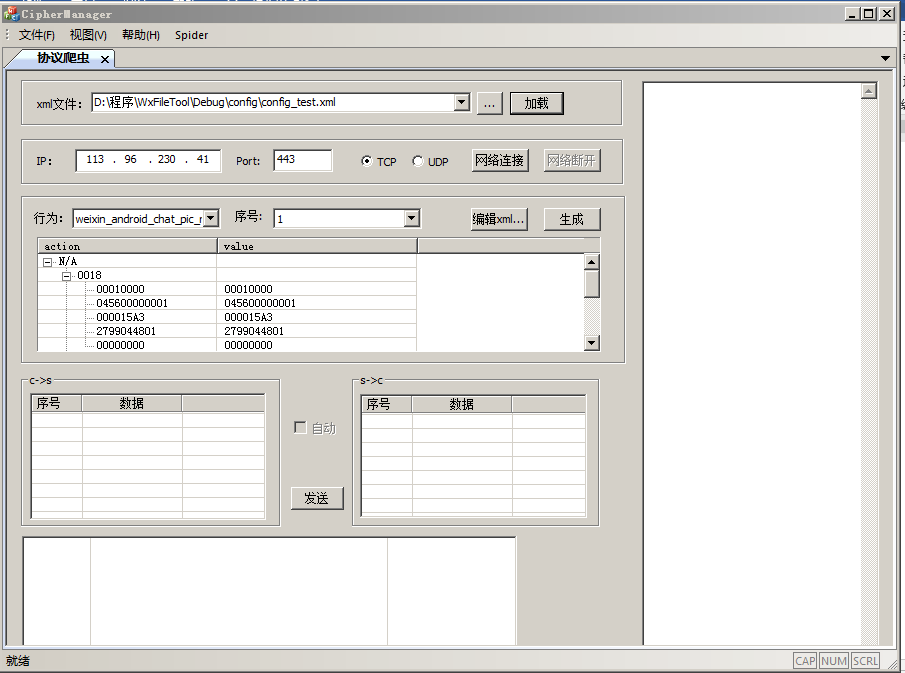
示例：<Value valuetype="VALUE\_STRING" isFilter="1" isPreLen="1" preLenType="w" preLenEndian="1" isSelfinLen="1" endian="1">uin</Value>

1. 使用说明
   * 1. 加载xml

双击CipherManager.exe程序，点击菜单“Spider->协议爬虫”,如下图所示：

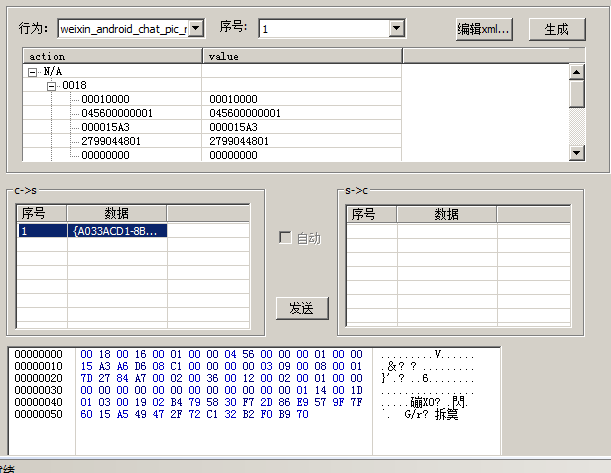


点击界面“…”按钮，选择xml文件，并点击“加载”按钮，如下图所示：



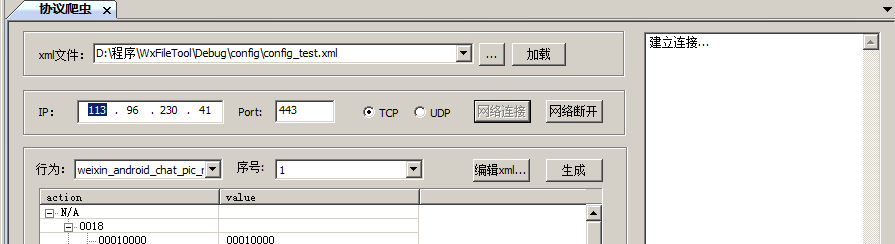
* + 1. 生成数据包

在行为中选择“weixin\_android\_chat\_pic\_receive”后，在序号中选择数据包1，点击“生成按钮”，此时在“c->s”控件中将显示生成的数据包，当选中某个数据包后，在下方显示16进制的数据，如下图所示：



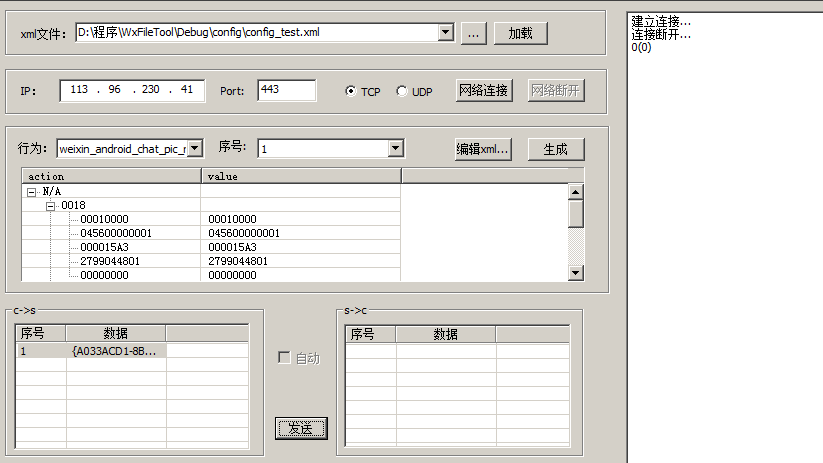
* + 1. 建立连接

点击界面中的“网络连接”按钮，右方控件中显示日志信息，如下所示：



* + 1. 发送数据

选择“c->s”控件中的数据包，并点击“发送”按钮，发送数据包，如下图所示：



* + 1. 接收数据
    2. 断开连接

1. 爬虫程序
   1. 程序说明

微信程序主要功能代码在WeChatWin.dll中，程序可能采用了VMP函数保护插件API进行开发，也就是说对部分关键的函数或者代码段进行了虚拟机保护。

* 1. 插件代码说明

暂无。