APT

黑白通吃：Glutton木马潜伏主流PHP框架，隐秘侵袭长达1年

Alex.Turing, Acey9

2024年12月10日 • 17 min read

简介

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。进一步溯源发现，该 IP 曾于2023年12月20日传播一个VirusTotal 0检测的恶意PHP文件`init_task.txt`，这一线索为我们后续的调查提供了重要切入点。

以 init_task 为线索，我们进一步发现了一系列关联的恶意 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，也可以通过 task_loader 作为入口，逐步加载其他 payload，形成一个完整的攻击框架。框架中的所有代码均在 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量 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：

1. 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、管理接口等。
2. 安装后门
    - ELF版的 winnti 后门
    - PHP 后门
3. 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`ac290ca4b5d9bab434594b08e0883fc5`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《Decade of the RATs》中提到的后门 **PWNLNX tool**，以及 IntezerLabs 于 2020 年 9 月 23 日推文提及的样本几乎完全一致。目前，大多数安全厂商已将该样本识别为 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，尚未有其他黑客团体使用的相关报道。此次活动投入的C2 156.251.163[.]120在其存活时间内，能够正确响应样本的网络请求，与后门建立交

互。从`样本的专属性和C2的有效性`来说，基本能够排除其他黑产团伙利用失活样本干扰归属研判的可能性。

1. **样本的专属性**：winnti 后门是 Winnti 组织的标志性工具，未见流传于其他黑产团伙的证据。
2. **C2 的有效性**：C2 地址能够正常交互，进一步表明此次活动是真实的攻击行为。

`基于 winnti 后门的真实性 和 Glutton的投递行为`，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，网络通信以及基础设施存在隐匿能力不足的问题，有点失水准，具体包括：

1. C2 网络通信缺乏加密：协议过于基础非常容易被逆向。
2. Downloader 网络通信未启用 HTTPS：HTTP通信非常容易被拦截或监控。
3. PHP 样本缺乏加密或混淆：样本以源码形式存在，可直接阅读了解功能。
4. 基础设施的欺骗性不强：活动投入的域名thinkphp1[.]com的伪装程度太低。

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术实现的简单为判断带来了不确定。在归属分析中要充分考虑网络黑产的复杂性和防御方情报的滞后性，为避免因单一线索而形成误导性结论，我们采用保守研判方法，以 **中等信心将 Glutton 定性为 Winnti 组织的新武器**。

黑白受害者

以请求C2 `cc.thinkphp1[.]com`做为被感染的标识，从我们的数据来看，受害者主要分布在中美俩地，涉及信息传输，商务服务，社会保障等行业。

在我们的溯源过程中，还发现了一个有意思的现象，Glutton的作者专门针对黑灰产的生产系统投毒，意图进行黑吃黑。时间回到2024年7月，我们以"b11st=0;"特

征在VirusTotal进行狩猎，先后发现了5个被感染的文件，由不同的国家上传到VT。

INDEX	MD5	DETECTION	FIRST SEEN	COUNTRY
1	3f8273575d4c75053110a3d237fda32c	2/65	2024.08.11	China
2	c1f6b7282408d4dfdc46e22bbdb3050f	0/59	2024.09.17	Germary
3	96fef42b234920f3eacfe718728b08a1	0/63	2024.10.14	SINGAPORE
4	ad150541a0a3e83b42da4752eb7e269b	1/62	2024.11.02	UNITED STATES
5	ad0d88982c7b297bb91bb9b4759ce0ab	4/41	2024.11.27	UNITED STATES

其中编号1，2，3是单个PHP文件；编号4，5为压缩包，包含一套完整的业务系统。它们之中最特别的是编号4，它是一套**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。

VT中显示它的Compressed Parents是`shuadan109.timibbs.cc_20241026_175636.tar.gz`，通过这个线索，我们发现了它的下载页面，售价高达980USDT，约等于人民币7000。

这个所谓的天美论坛上有大量博彩，棋牌，刷单等网络黑灰产的源码，售价不菲。

虽然我们没有花钱去验证VT上的那套代码是否为此论坛的原始代码（7000块，太贵啦！），但Glutton背后的黑客与这个论坛的关系无非存在以下几种可能性

1. 黑客是论坛用户，购买资源后投毒
2. 黑客入侵论坛，向其网络资源投毒
3. 黑客属于论坛，共同研发带毒的网络资源
4. 黑客与论坛无关，研发了带毒的网络资源被论坛收录

无论哪种情况是，都暴露了作者收割黑灰产的意图，我们小小的揣摩一下他的心思。

> *“大把大把的钱让搞博彩，棋牌，刷单的菜鸡黑产给赚了，真是造孽啊。怎么样抢他们的钱呢？嗯，向市场中投入大量的带有后门的黑灰业务系统，让菜鸡们给自己“打工”，时机成熟时就收割一波，搞一波快钱。嘿嘿，即使他们发现了，还不敢报警，只能哑巴吃黄连，完美！”*

Glutton分析

我们目前捕获了多个Glutton组件，包括 `task_loader`、`init_task`、`client_loader`、`client_task`、`fetch_task` 和 `l0ader_shell`（注：client_loader、client_task 和 fetch_task 等名称为我们根据其功能命名）。这些文件每个约 3000 行代码，均未进行任何加密或混淆，功能实现一目了然。本文将聚焦分析其核心功能代码，感兴趣的读者可自行查阅完整源码以获取更多细节。

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，构建一个无落地载荷的攻击框架。该攻击框架的核心功能包括感染目标设备中的 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的适配性，还使其在安全防御中更加难以检测和溯源。

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，还可能通过黑产源码论坛发布预植入 l0ader_shell 的业务系统，以进一步扩大感染范围，对黑产进行定向攻击。

设备感染Glutton后有以下迹象

1. 文件层面：PHP文件中被植入l0ader_shell
2. 进程层面：监听UDP端口6006的winnti backdoor进程`php-fpm`；udp通信的php backdoor进程`[kworker/0:0HC ]`

Part1: task_loader

task_loader核心功能是检查执行环境，根据环境的不同，分别使用run_task_by_system，run&get_php_code，run_task_by_fpm，run_task_direct函数下载执行下一阶段的Payload。

其中run_task_by_fpm没有实现，run_task_by_system，run&get_php_code下载的Payload为init_task，而run_task_direct下载的Payload则是modify_php，事实上modify_php是init_task的子集。

FUNCTION	PATH	EXECUTION ENV
run_task_by_system	/v11/init_task.gz	new php process
run&get_php_code	/v11/init_task.gz	fastcgi
run_task_direct	/v11/modify_php_v11.gz	oringinal php process

Part2: init_task

init_task的主要任务有3个：elf_install则是下载执行winnti后门，bt_modify用于感染宝塔，收集敏感信息，php_modify感染php页面，植入下一阶段Payload的启动代码。

0x01:elf_install 任务

elf_install任务，它向服务器请求winnti后门，将其伪装成/lib/php-fpm文件，并通过向/etc/init.d/network中插入`export OLD=$PATH; export PATH=/usr/lib/; php-fpm; export PATH=$OLD;`实现持久化。目前一共检测到3个不同的下载URL，它们提供了同一个winnti后门。

URL	MD5
172.247.127[.]210/v10/php-fpm	ac290ca4b5d9bab434594b08e0883fc5
v6.thinkphp1[.]com/v11/php-fpm	ac290ca4b5d9bab434594b08e0883fc5
v20.thinkphp1[.]com/static/v20/php-fpm	ac290ca4b5d9bab434594b08e0883fc5

ac290ca4b5d9bab434594b08e0883fc5与BlackBerry曝光样本高度相似，只是新增了`C2更新，样本更新`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[.]120。

目前该IP已失活，但历史上可以看出它能正常响应样本的网络请求，是一个真实的winnti c2。

0x02: bt_modify任务

在bt_modify任务中，find_all函数用于采集宝塔的敏感信息；do_modify函数用于修改宝塔的系统文件。

find_all收集用户名，密码，手机号，SSH凭证等23项敏感信息，压缩后上报给C2。

ADMIN_PATH	BT_APASS	BASIC_AUTH	BASIC_PASS	BASIC_USER
bt_clients	crontabs	databases	bt_dir	bt_domain
bt_ftps	bt_https	bt_mobile	mysql_root	bt_pass_md5
bt_passwd	phpmyadmin	bt_port	bt_sites	bt_sites_path
bt_ssh	bt_user_md5	bt_username		

实际产生的流量如下所示，将body部分内容在cyberchef中使用url decode + raw inflate即可还原。

do_modify

do_modify对宝塔框架中的init.py，public.py，ssh_terminal.py，files.py，config.py，panelSSL.py，userlogin.py进行修改，实现窃取登录凭证，token，暴露资产等目地。

- 窃取凭证，token等
- 暴露资产

0x03: php_modify任务

php_modify任务通过以下代码片段对thinkphp，yii，larvavel，dedecms等PHP框架进行修改。

修改逻辑是在PHP框架代码中出现$ref_line代码的位置插入v11_code，如果所有的ref_lines都没有命中，则在文件尾部加入v11_code。

这些篡改的页面被调用时，v11_code也就得到了执行的机会。v11_code由 `v11_begin + PHPCODE_MAIN + v11_end` 3部分组成，其中v11_begin与v11_end的值分别 `b11st=0；b11end=0;` 而PHPCODE_MAIN则是init_task中一个const变量，保存了一个名为l0ader的函数，它正是l0ader_shell。

l0ader函数的功能有俩个：

1. 使用UDP协议上报被主机信息以及被触发页面的访问参数，上报地址为 **v6.thinkphp1[.]com:9988**
   实际产生的流量如下所示：
2. 构建HTTP请求，下载执行下一阶段的client_loader。
   实际产生的流量如下所示：

Part3: client_loader

client_loader实际上是init_task的重构版本，它支持init_task所有的功能，只是在代码的组织方式上也所不同。
第一个大变化是php_modify，它的10ader函数的代码开始使用混淆。

loader的功能依旧，只不过使用的网络基础设施有所变化。

FILE	REPORTER	DOWNLOADER
init_task	udp://v6.thinkphp1[.]com:9988	v6.thinkphp1[.]com/php?
client_laoder	udp://v20.thinkphp1[.]com:9988	v20.thinkphp1[.]com/init?

第二个大变化是新增了一个新功能，下载执行具有后门功能的client。

读者或许会问已经有了winnti后门，为什么还要实现一个client呢？事实上本次活动，针对的不仅是Linux系统，Windows/Mac也是目标。通过PHP来实现后门功能具有一些明显的优势：

- 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备通用的查杀方法

Part4: client_task

client_task的主要任务有俩个：1是启动PHP后门木马，2是定期执行fetch_task任务。

0x1 PHP后门

client_socket类实现了php后门的功能框架，它硬编码了一个C2 **cc.thinkphp1.com:9501**，支持TCP,UDP俩种通信方式，默认采用UDP通信。

client_v1类继承client_socket，通过process_std_cmd_v1类处理C2下发的指令。

这个php后门支持22个不同的指令，以下为指令号以及对应的功能。

ID	FUNCTION
1	ping(udp only)
2	pong(udp only)
10	login
31	keepalive
148	set connection config
149	switch connection to tcp
150	switch connection to udp
151	shell
152	upload/download file via tcp
189	get_temp_dir
190	scandir
191	get dir info
192	mkdir
193	write file
194	read file
195	create file
196	rm
197	copy file

ID	FUNCTION
198	rename file
199	chmod
200	chown
201	eval php code

UDP和TCP的通信过程几乎一样，除了UDP一个前置的验活过程，即client 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个字节为magic，第2个字节为指令码。magic的值暗示了传输的数据被压缩与否，其中0xf0表示没有压缩，0xf1表示启动压缩。当数据大于32节字节时，启用压缩。

login指令中的数据该如何解析呢？0xf1表示压缩，0x89表示data的长度，其余部分是data，使用rawInflate即可解压，内容包括host_user, host_os, host_name, host_cwd等信息。

0x2 Fetch_task

Fetch_task每小时执行一次，向远程服务器 `http://v20.thinkphp1.com/v20/fetch` 请求PHP代码，解压执行。

从我们的跟踪系统来看，目前Fetch_task拉取的payload是MD5值为69ed3ec3262a0d9cc4fd60cebfef2a17的**client_loader**。

彩蛋

jklwang.com

Glutton通过do_tp5_request函数清理旧版本对Request.php文件感染。通过$ref_lines的中的代码，我们至少知道VT 0检测的域名`jklwang.com`也是Glutton的资产。

HackBrowserData

6月14 日，`v20.thinkphp1.com` 曾传播 Mac 版 **HackBrowserData** 工具。

该工具可解密并导出浏览器中的数据，包括密码、浏览历史、Cookie、书签、信用卡信息、下载历史、LocalStorage 及扩展程序。

我们推测其使用场景为：当黑灰产在本地调试或二次开发已被植入后门的业务系统时，攻击者会下发 **HackBrowserData** 工具，窃取黑灰产自身的高价值敏感信息，形成“黑吃黑”的攻击链。

总结

根据`init_task`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一年。除传统的针对“白方”的网络犯罪活动外，**Glutton** 还展现了对“黑方”浓厚的兴趣，其作者明显具有“赢三次”的野心，具体体现在以下三方面：

1. **窃取黑灰产发起者的高价值敏感信息**；
2. **收割黑灰产业务本身带来的巨额经济利益**；
3. **收集黑灰产参与者的敏感数据，为后续钓鱼或社工活动奠定基础**。

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，以及时采取应对措施，包括：

1. 清理PHP中的l0ader_shell
2. 清理进程中的winnti 后门进程，以及php后门进程

3. 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，其初始访问路径（Init Access）仍不清晰。我们欢迎有相关情报的友商及读者提供更多线索，帮助完善对 **Glutton** 的技战术矩阵及归属分析，共同维护网络安全。

如果您对我们的研究感兴趣，欢迎通过 X 平台 与我们联系！

IOC

MD5

```
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
```

C2

```
cc.thinkphp1[.]com
156.251.163[.]120
```

Downloader

```
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
```