


# HTB Sherlock: Campfire-2

 [htb-sherlock](#) [forensics](#) [sherlock-campfire-2](#) [ctf](#) [hackthebox](#) [dfir](#) [eventlogs](#) [evtx-dump](#) [win-event-4769](#) [win-event-4768](#) [win-event-5140](#) [as-rep-roast](#)

Jul 26, 2024

HTB Sherlock: Campfire-2

- Challenge Info
- Background
- Analysis
- Results

The second in the Campfire Sherlock series about active directory attacks is about AS-REP-Roasting, an attack against users configured to not require preauthentication when interaction with Kerberos. I'll examine the event logs to show which user account was compromised in the attack, as well as the workstation that was compromised to perform the attack.

## Challenge Info

Name	<div>Campfire-2</div> <div>Play on HackTheBox</div>
Release Date	27 June 2024
Retire Date	27 June 2024
Difficulty	Very Easy
Category	DFIR
Creator	<div> CyberJunkie Moderator 2 613 hackthebox.com</div>

## Background

### Scenario

Forela's Network is constantly under attack. The security system raised an alert about an old admin account requesting a ticket from KDC on a domain controller. Inventory shows that this user account is not used as of now so you are tasked to take a look at this. This may be an AsREP roasting attack as anyone can request any user's ticket which has preauthentication disabled.

Notes from the scenario:

- Old admin-level account.
- Potential AS-REP-Roasting attack.

### Questions

To solve this challenge, I'll need to answer the following 5 questions:

- When did the ASREP Roasting attack occur, and when did the attacker request the Kerberos ticket for the vulnerable user?
- Please confirm the User Account that was targeted by the attacker.
- What was the SID of the account?
- It is crucial to identify the compromised user account and the workstation responsible for this attack. Please list the internal IP address of the compromised asset to assist our threat-hunting team.
- We do not have any artifacts from the source machine yet. Using the same DC Security logs, can you confirm the user account used to perform the ASREP Roasting attack so we can contain the

compromised account/s?

## Data

The download contains a single file, `Security.evtx`:

```
0xdf@hacky$ unzip -l campfire-2.zip
Archive:  campfire-2.zip
  Length      Date    Time    Name
-----
 1118208  2024-05-29 11:41   Security.evtx
-----
 1118208                          1 file
```

After decompressing, it is a Windows event log:

```
0xdf@hacky$ file Security.evtx
Security.evtx: MS Windows Vista Event Log, 3 chunks (no. 2 in use), next record no.
153
```

## Artifact Background

Event logs are becoming relatively routine with Sherlocks investigating Windows systems. This file represents a single event log, the Security log. Security has things like logon / logoff, authentication, account management, etc.

## Tools

[EvtxECmd](#) is a really nice Windows tool for parsing event logs into JSON. For this time, I'll use `evtx_dump` (from [omerbenamram's evtx repo](#), download the binary from the [Releases page](#)), which works on Linux as well:

```
0xdf@hacky$ evtx_dump -o jsonl -t 1 -f Security.json Security.evtx
0xdf@hacky$ wc -l Security.json
152 Security.json
```

There's 152 entries (one per line), which matches up with where `file` showed the next record would be 153.

## AS-REP-Roasting

In a typical Kerberos authentication flow, the user first authenticates to the DC, and then sends the user a authentication server response (AS-REP) for the requested service. Part of that ticket is encrypted with the NTLM hash of the service account, so that the service can read it. Kerberoasting is trying to brute force that decryption to get the service's password.

However, there is an option that user accounts can have set called "Do not require Kerberos preauthentication" (`UF_DONT_REQUIRE_PREAUTH`). If this is set, the user doesn't have to authenticate before requesting the ticket. Part of the AS-REP contains information encrypted with the user's hash (so they still need the user's password to use it). This encrypted section presents an opportunity for the attacker, as I can try to brute force passwords against this to recover a weak password. That's known as AS-REP-Roasting. [This blog](#) from Blumira shows more details.

## Analysis

### AS-REP-Roasting

#### Find Authentication Events

To perform an AS-REP-Roast attack, the actor will request a TGT for a service from the DC. This will generate a [4768 event log](#), "A Kerberos authentication ticket (TGT) was requested". Taking a quick look at the first event, I'll see that it has the id in `.Event.System.EventID`:

```

0xdf@hacky$ cat Security.json | jq '.[0]' -s
{
  "Event": {
    "#attributes": {
      "xmlns": "http://schemas.microsoft.com/win/2004/08/events/event"
    },
    "EventData": {
      "CommandLine": "",
      "MandatoryLabel": "S-1-16-16384",
      "NewProcessId": "0xac",
      "NewProcessName": "Registry",
      "ParentProcessName": "",
      "ProcessId": "0x4",
      "SubjectDomainName": "-",
      "SubjectLogonId": "0x3e7",
      "SubjectUserName": "-",
      "SubjectUserSid": "S-1-5-18",
      "TargetDomainName": "-",
      "TargetLogonId": "0x0",
      "TargetUserName": "-",
      "TargetUserSid": "S-1-0-0",
      "TokenElevationType": "%1936"
    },
    "System": {
      "Channel": "Security",
      "Computer": "DC01.forela.local",
      "Correlation": null,
      "EventID": 4688,
      "EventRecordID": 6168,
      "Execution": {
        "#attributes": {
          "ProcessID": 4,
          "ThreadID": 176
        }
      },
      "Keywords": "0x8020000000000000",
      "Level": 0,
      "Opcode": 0,
      "Provider": {
        "#attributes": {
          "Guid": "54849625-5478-4994-A5BA-3E3B0328C30D",
          "Name": "Microsoft-Windows-Security-Auditing"
        }
      },
      "Security": null,
      "Task": 13312,
      "TimeCreated": {
        "#attributes": {
          "SystemTime": "2024-05-29T06:33:53.454933Z"
        }
      },
      "Version": 2
    }
  }
}

```

The `-s` is because the log file is not an array of events, but just events one per line. `-s` will “slurp” them into an array to loop over. Then I can get the first event with `.[0]`.

To select only the 4768 events, I’ll use `select`:

```

0xdf@hacky$ cat Security.json | jq '.Event | select(.System.EventID == 4768)' -c | wc
-1
11

```

`-c` outputs one event per line, so there are 11 events left after this filter.

Find AS-REP-Roast Event

There are likely many legit authentication events in this data. Another thing to look at is if the authentication used pre-authentication (AS-REP-Roasting does not by definition).

There's another field, `PreAuthType`. For all but one of these, it's set to 2:

```
oxdf@hacky$ cat Security.json | jq '.Event | select(.System.EventID == 4768) | .EventData.PreAuthType' -r
2
2
2
2
2
2
0
2
2
2
2
2
```

[This post](#) shows that 2 is `PA-ENC-TIMESTAMP`, which is the standard method for preauthentication. 0 is for "Logon without Pre-Authentication". There's one event like that! I'll filter on it:

```
oxdf@hacky$ cat Security.json | jq '.Event | select(.System.EventID == 4768 and
.EventData.PreAuthType == "0")'
{
  "#attributes": {
    "xmlns": "http://schemas.microsoft.com/win/2004/08/events/event"
  },
  "EventData": {
    "CertIssuerName": "",
    "CertSerialNumber": "",
    "CertThumbprint": "",
    "IpAddress": "::ffff:172.17.79.129",
    "IpPort": "61965",
    "PreAuthType": "0",
    "ServiceName": "krbtgt",
    "ServiceSid": "S-1-5-21-3239415629-1862073780-2394361899-502",
    "Status": "0x0",
    "TargetDomainName": "forela.local",
    "TargetSid": "S-1-5-21-3239415629-1862073780-2394361899-1601",
    "TargetUserName": "arthur.kyle",
    "TicketEncryptionType": "0x17",
    "TicketOptions": "0x40800010"
  },
  "System": {
    "Channel": "Security",
    "Computer": "DC01.forela.local",
    "Correlation": null,
    "EventID": 4768,
    "EventRecordID": 6241,
    "Execution": {
      "#attributes": {
        "ProcessID": 752,
        "ThreadID": 3188
      }
    },
    "Keywords": "0x8020000000000000",
    "Level": 0,
    "Opcode": 0,
    "Provider": {
      "#attributes": {
        "Guid": "54849625-5478-4994-A5BA-3E3B0328C30D",
        "Name": "Microsoft-Windows-Security-Auditing"
      }
    },
    "Security": null,
    "Task": 14339,
    "TimeCreated": {
      "#attributes": {
        "SystemTime": "2024-05-29T06:36:40.246362Z"
      }
    },
    "Version": 0
  }
}
```

Analysis Results

If I assume that this ticket is an attack, I can learn a lot about it from this event. The timestamp is a bit buried in the data structure (Task 1):

```
oxdf@hacky$ cat Security.json | jq '.Event | select(.System.EventID == 4768 and
.EventData.PreAuthType == "0") | .System.TimeCreated["#attributes"].SystemTime' -r
2024-05-29T06:36:40.246362Z
```

The target user and their security identifier (SID) is in EventData (Task 2, 3):

```
0xdf@hacky$ cat Security.json | jq '.Event | select(.System.EventID == 4768 and
.EventData.PreAuthType == "0") | .EventData.TargetUserName' -r
arthur.kyle
0xdf@hacky$ cat Security.json | jq '.Event | select(.System.EventID == 4768 and
.EventData.PreAuthType == "0") | .EventData.TargetSid' -r
S-1-5-21-3239415629-1862073780-2394361899-1601
```

The IP address is also there (Task 4):

```
0xdf@hacky$ cat Security.json | jq '.Event | select(.System.EventID == 4768 and
.EventData.PreAuthType == "0") | .EventData.IpAddress' -r
::ffff:172.17.79.129
```

The IPv4 is just 172.17.79.129.

## Source PC

### Strategy

The last question is asking about the user account used to perform the AS-REP-Roast. This is a bit of an odd question, as this attack doesn’t require authentication to perform. Still, getting access to a DC to make requests often means compromising a user’s workstation. I’ll look for what other activity comes from that IP.

### Log Overview

The logs are already one per line, so I’ll use `grep` to get any log that contains the IP, and then pipe that into `jq` to get the event ids:

```
0xdf@hacky$ cat Security.json | grep 172.17.79.129 | jq -r .Event.System.EventID |
sort | uniq -c | sort -nr
4 5140
1 4769
1 4768
```

4768 is the log I was looking at above. [4769](#) is a “A Kerberos service ticket was requested”, and [5140](#) is “A network share object was accessed”.

### User Activity

The 4769 log is requesting a service ticket:

```
0xdf@hacky$ cat Security.json | grep 172.17.79.129 | jq '. |
select(.Event.System.EventID == 4769) | .Event.EventData'
{
  "IpAddress": "::ffff:172.17.79.129",
  "IpPort": "61975",
  "LogonGuid": "543ACECF-87DD-45D9-CF0D-6C1F28070DC3",
  "ServiceName": "DC01$",
  "ServiceSid": "S-1-5-21-3239415629-1862073780-2394361899-1000",
  "Status": "0x0",
  "TargetDomainName": "FORELA.LOCAL",
  "TargetUserName": "happy.grunwald@FORELA.LOCAL",
  "TicketEncryptionType": "0x12",
  "TicketOptions": "0x40810000",
  "TransmittedServices": "-"
}
```

The user requesting it is happy.grunwald, who must be the user on this PC (Task 5). That same user is involved in three of the four 5140 events trying to access the network share:

```
0xdf@hacky$ cat Security.json | grep 172.17.79.129 | jq '. |
select(.Event.System.EventID == 5140) | .Event.EventData.SubjectUserName' -r
happy.grunwald
happy.grunwald
happy.grunwald
ANONYMOUS LOGON
```

## Results

1. When did the ASREP Roasting attack occur, and when did the attacker request the Kerberos ticket for the vulnerable user?

2024-05-29 06:36:40

2. Please confirm the User Account that was targeted by the attacker.

arthur.kyle

3. What was the SID of the account?

S-1-5-21-3239415629-1862073780-2394361899-1601

4. It is crucial to identify the compromised user account and the workstation responsible for this attack. Please list the internal IP address of the compromised asset to assist our threat-hunting team.

172.17.79.129


5. We do not have any artifacts from the source machine yet. Using the same DC Security logs, can you confirm the user account used to perform the ASREP Roasting attack so we can contain the compromised account/s?


happy.grunwald


0xdf hacks stuff


0xdf hacks stuff  
[0xdf.223@gmail.com](mailto:0xdf.223@gmail.com)

 [0xdf](#)

 [0xdf](#)

 [feed](#)

 [0xdf](#)

 [@0xdf@infosec.exchange](#)

CTF solutions, malware analysis, home lab development

