Attacking Kerberos Deployments

Breaking the Intranet

Rachel Engel, Brad Hill and Scott Stender

Black Hat USA 2010

iSEC PARTNERS

https://www.isecpartners.com

About Us

- Who are you?
  - Security Consultants at iSEC Partners
  - Work in our application security consulting practice
  - Based in Seattle

- What is this talk about?
  - Performing practical attacks against common Kerberos deployment patterns

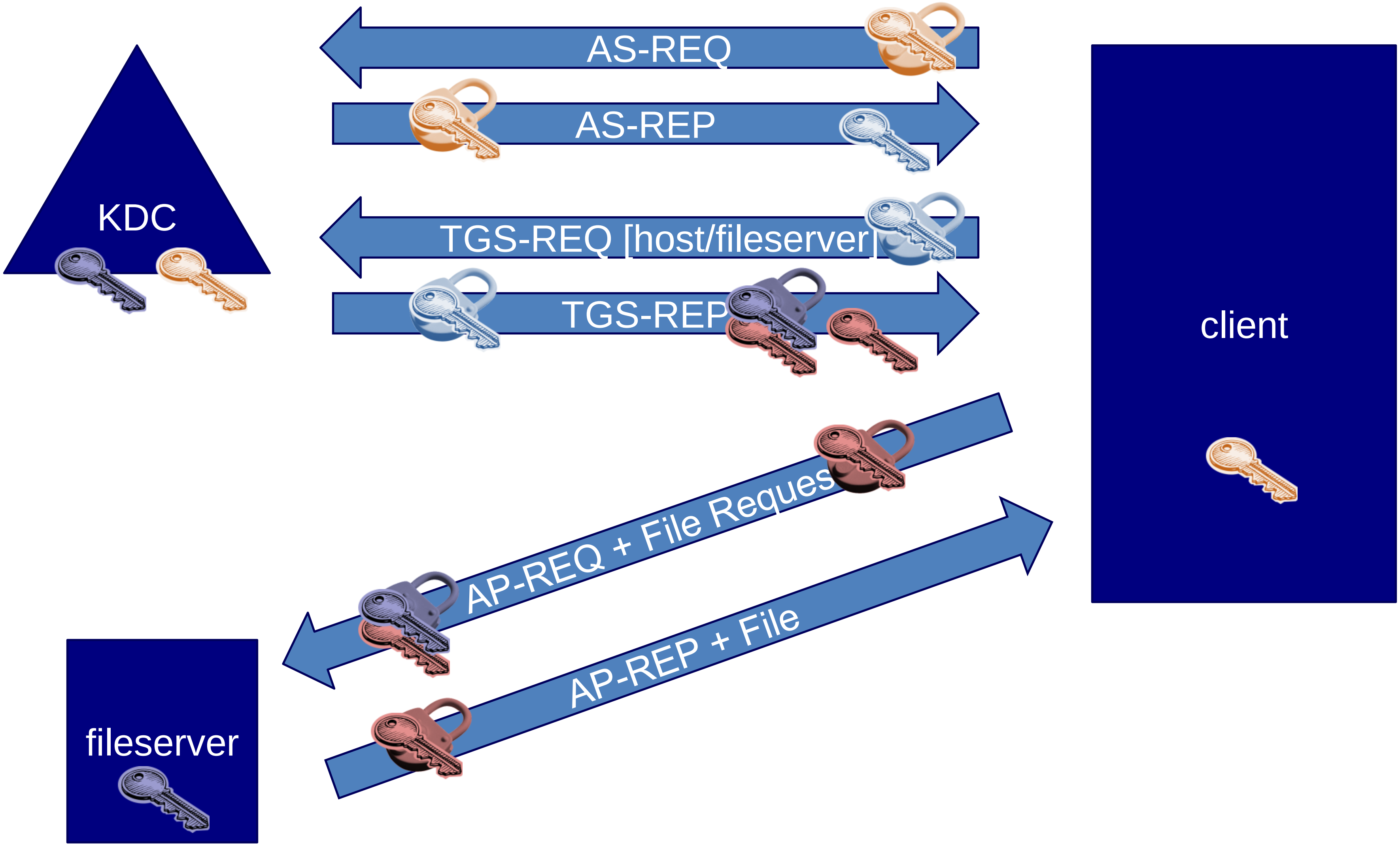
- Why should I care?
  - If you have authenticated to another machine at work, you have probably used Kerberos

Agenda

- Protocol Overview
- Initial Authentication and Etype Downgrades
- PKINIT: Kerberos and Smart Cards
  - Hijacking Active Directory Workstations with Smart Card login: Own one box, own the Enterprise
- Hijacking Kerberized Services
  - AP-REQ replay attack and defense
  - Mutual authentication and SPNs

A quick introduction to Kerberos

Kerberos: The Basic Protocol



Kerberos rules the Intranet

- Interoperable and standardized
- Most widely utilized and preferred protocol for authentication in large, centrally managed environments
  - Windows Active Directory Networks
  - Large educational networks on Unix/Linux
- Still being adopted in new places
  - Hadoop
  - Web Services
  - InfoCard

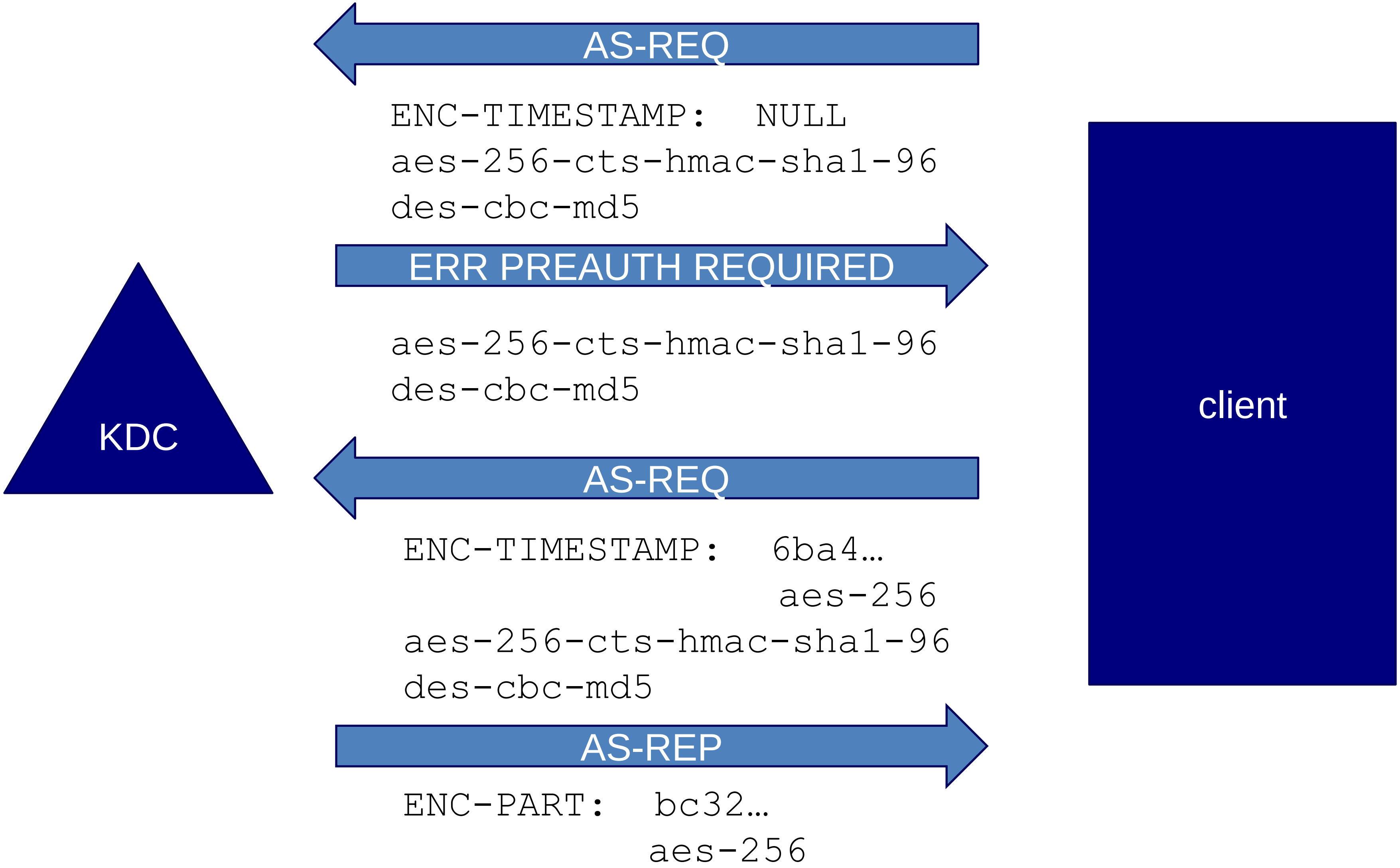
Initial Authentication and Etypes

Cryptographic Primitives

- Cryptographic Agility was a big driver for Kerberos v5
- Etypes define the set of primitives to be used for cryptographic operations
- Examples include:

```
aes256-cts-hmac-sha1-96
aes128-cts-hmac-sha1-96
rc4-hmac
des-cbc-md5
rc4-hmac-exp
```

Etype Negotiation



KDC

client

AS-REQ

```
ENC-TIMESTAMP:  NULL
aes-256-cts-hmac-sha1-96
des-cbc-md5
```

ERR PREAUTH REQUIRED

```
aes-256-cts-hmac-sha1-96
des-cbc-md5
```

AS-REQ

```
ENC-TIMESTAMP:  6ba4…
                 aes-256
aes-256-cts-hmac-sha1-96
des-cbc-md5
```

AS-REP

```
ENC-PART:  bc32…
            aes-256
```

Attacking Etype Negotiation

- How can an active attacker influence etype negotiation to his or her advantage?

- Lie to the server about client capabilities
  - Downgrade initial anonymous AS-REQ
  - Downgrade the authenticated AS-REQ

- Lie to the client about server capabilities
  - Downgrade ERR PREAUTH REQUIRED and several others

Benefits of Downgrade

- The key used to encrypt the authenticator is derived directly from the user's password.
- Try:
  - Active downgrade
  - Capture authenticator

  - Use the key to make your own authenticator later

Benefits of Downgrade

- Frank O’Dwyer demonstrated the feasibility of password grinding on RC4 – other etypes are similarly vulnerable
- Newer etypes have been designed to resist such attacks
- Even when exhaustive key search is unavailable, downgrade can make password grinding feasible

Does this Affect Me?

- Windows 2008 / Windows Vista and previous enable DES for both outbound and inbound

- Rather recent open-source distributions of Kerberos do the same, but your mileage will vary on your distribution and configuration steps.

- Windows 7 emits, but does not accept, export-grade RC4

- Enabling DES etypes is still surprisingly common for interoperability

Protecting Against Downgrade

- In a word, disable “weak” etypes
  - DES, Export-Grade
  - If possible, everything but the latest and greatest AES

- Disabling etypes
  - Always configurable in MIT and similar distributions
  - Windows 2008 R2 / Windows 7 introduced a new security policy for this

- These are increasingly disabled by default
  - Windows 2008 R2, MIT Kerb 1.8

Public Key Kerberos and Smart Cards

Basics of PKINIT

Client Preauthenticator

- AuthPack
  - KdcName
  - KdcRealm
  - Cusec
  - Ctime
  - Nonce
- Client Certificate
- RSA SHA1 Signature

KDC Reply

- EncKeyPack
  - RecipientInfo
    - IssuerAndSerialNumber
    - Encrypted Key
  - EncryptedContentInfo
    - ReplyKeyPack
      - ReplyKey
      - AS Checksum
    - ReplyKeyPack Signature
    - KDC Certificate

Mutual authentication?

- In traditional Kerberos, the user and KDC shared a secret.
- Now we have PKI involved. As HTTPS has repeatedly shown us, PKI is tricky.

Who are the trust roots for Public Key Kerberos?

- Luckily, the usual suspects from the Web are not involved.

- Certificates must be issued by a specific root CA(s)
  - Config file for Unix/Linux clients
  - Registry and Active Directory for Windows clients

- Client certs must be issued by this authority and have the Smart Card Authentication EKU
- How is the KDC authenticated by the client?

PKINIT KDC Authentication

- Certificate must be issued by the designated authority.
- Must have the subject indicated in a correct format.
  - Usually a UPN (email address)

- MIT & Heimdal look for the KDC Key Purpose ID EKU.

- What do Windows clients verify? Not documented.

New group policy in Vista SP1: “Require strict KDC validation”

“This policy setting controls the Kerberos client's behavior in validating the KDC certificate.”

“If you enable this policy setting, the Kerberos client requires that the KDC's X.509 certificate contains the KDC key purpose object identifier in the Extended Key Usage (EKU) extensions, and that the KDC's X.509 certificate contains a dNSName subjectAltName (SAN) extension that matches the DNS name of the domain. If the computer is joined to a domain, the Kerberos client requires that the KDC's X.509 certificate must be signed by a Certificate Authority (CA) in the NTAUTH store. If the computer is not joined to a domain, the Kerberos client allows the root CA certificate on the smart card to be used in the path validation of the KDC's X.509 certificate.”

Yadda…yadda…yadda…

“If you disable or do not configure this policy setting, the Kerberos client will require only that the KDC certificate contain the Server Authentication purpose object identifier in the EKU extensions.”

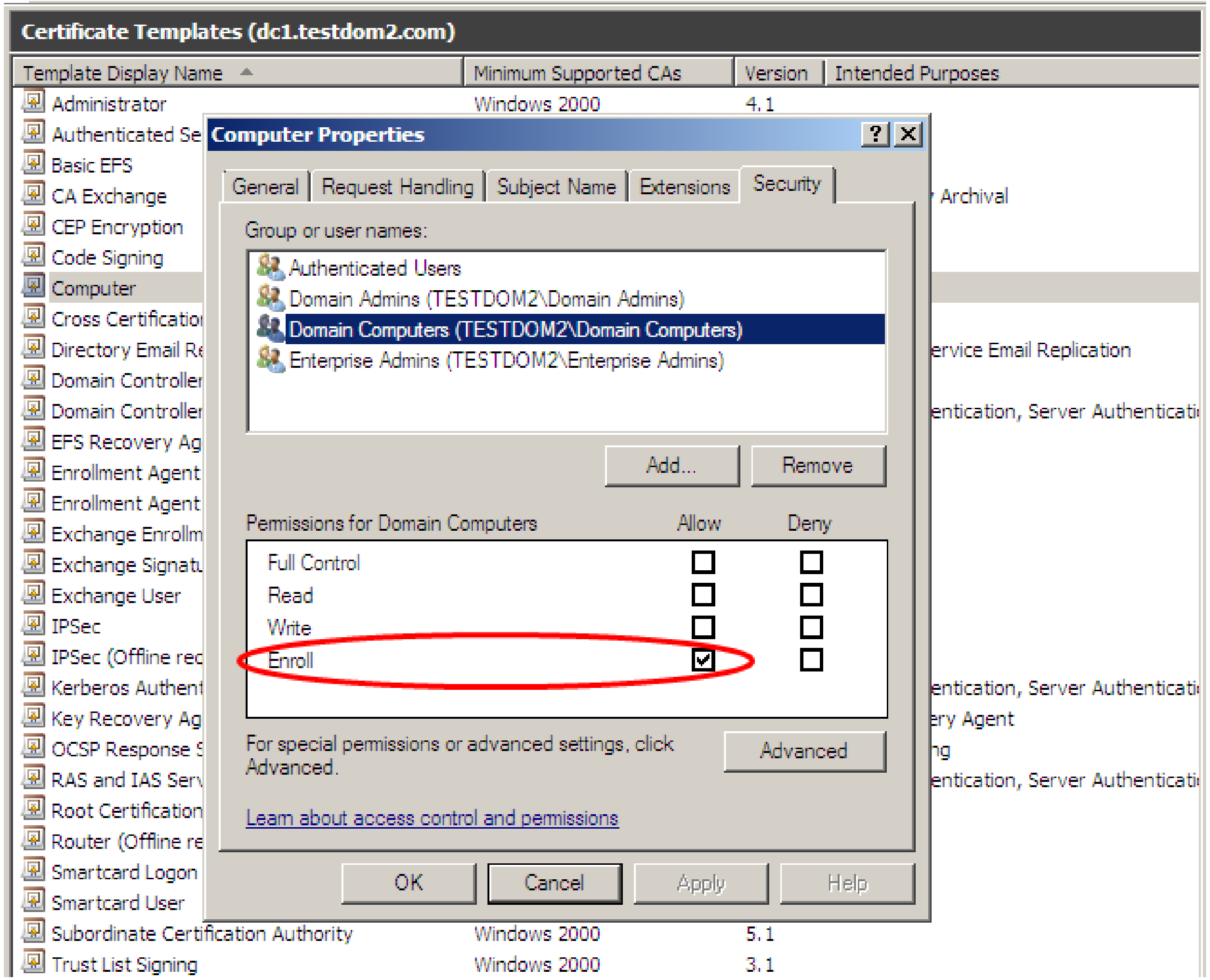
- *What other certificates have the Server Auth EKU?*

Web Server template

- Have you been following security advice to use HTTPS on your intranet? How many internal web servers do you have with certificates issued by the Enterprise CA?
- How much to you trust these systems and those with administrative access to them?
- Even if you use NAP/NAC, at least one of these is accessible to non-compliant clients. (remediation server)

Computer Template

- Default AD Cert Services Enterprise Authority configuration:



Certificate Templates (dc1.testdom2.com)

Template Display Name	Minimum Supported CAs	Version	Intended Purposes
Administrator	Windows 2000	4.1	
Authenticated Se			
Basic EFS			
CA Exchange			Archival
CEP Encryption			
Code Signing			
Computer			
Cross Certificatio			
Directory Email Re			ervice Email Replication
Domain Controller			
Domain Controller			entication, Server Authenticati
EFS Recovery Ag			
Enrollment Agent			
Enrollment Agent			
Exchange Enrollm			
Exchange Signatu			
Exchange User			
IPSec			
IPSec (Offline req			
Kerberos Authent			entication, Server Authenticati
Key Recovery Ag			ery Agent
OCSP Response S			ng
RAS and IAS Serv			entication, Server Authenticati
Root Certification			
Router (Offline re			
Smartcard Logon			
Smartcard User			
Subordinate Certification Authority	Windows 2000	5.1	
Trust List Signing	Windows 2000	3.1	

Computer Properties

Group or user names:

- Authenticated Users
- Domain Admins (TESTDOM2\Domain Admins)
- Domain Computers (TESTDOM2\Domain Computers)
- Enterprise Admins (TESTDOM2\Enterprise Admins)

Permissions for Domain Computers	Allow	Deny
Full Control	□	□
Read	□	□
Write	□	□
Enroll	☑	□

For special permissions or advanced settings, click Advanced.

Learn about access control and permissions

Impersonate the KDC in PKINIT

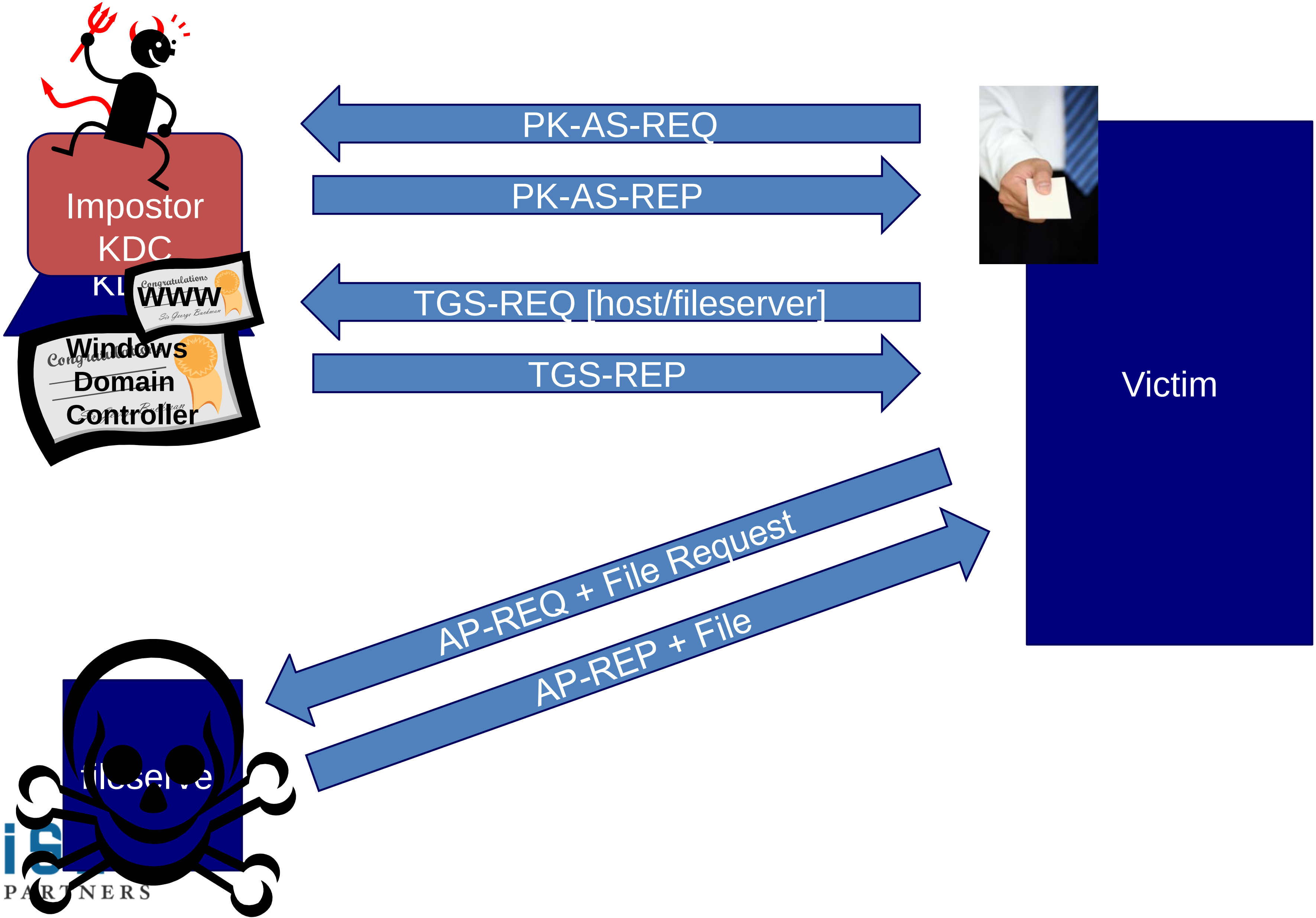
- In a default install, any workstation in the domain has access to credentials that allow impersonation of the KDC.

- In a default install, works for all clients through and including Win 7.
- And for MIT and Heimdal clients configured for interop with a Windows Server KDC.
  - (pkinit_require_eku = false)

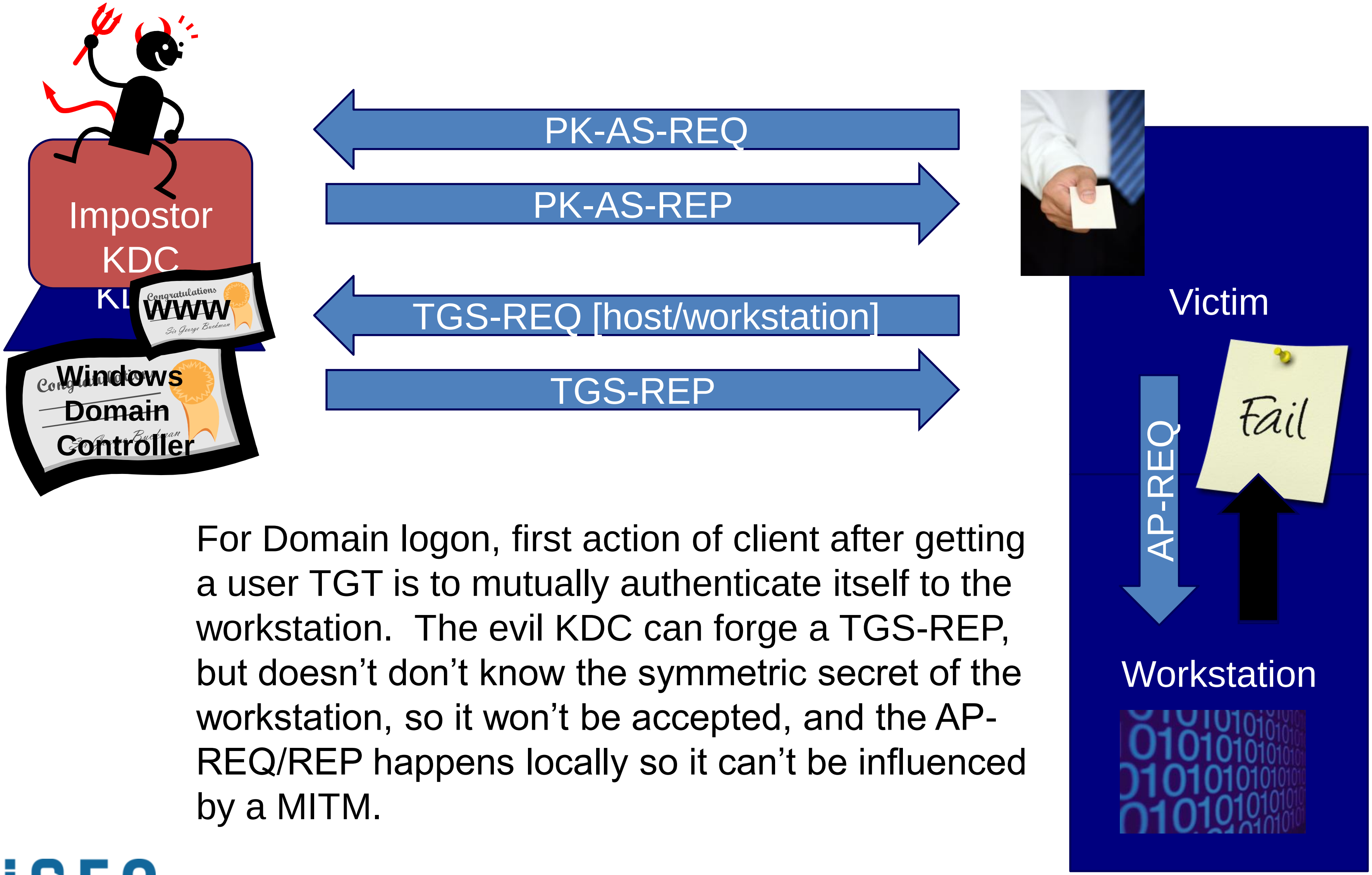
Can’t the Kerberos client match the X.509 Subject in the KDC certificate?

- MIT & Heimdal could check that the name is in the list of KDCs for the realm in /etc/krb5.conf, but don’t
- Windows doesn’t know who the DC / KDC is. It asks the network via a combination of insecure protocols:
  - DNS SRV records
  - NetBIOS
  - Unauthenticated CLDAP
- Doesn’t bother do to DNS to CNAME match, anyway
  - DNSSEC won’t save you
  - And Kerberos traffic is usually exempt from IPSEC policy

Elevation: MIT/Heimdal kinit+NFS



Elevation: Windows Smart Card Logon



For Domain logon, first action of client after getting a user TGT is to mutually authenticate itself to the workstation. The evil KDC can forge a TGS-REP, but doesn’t don’t know the symmetric secret of the workstation, so it won’t be accepted, and the AP-REQ/REP happens locally so it can’t be influenced by a MITM.

How to get around this?

- Find a scenario where the computer account verification isn't needed or can't happen.
- Domain join
  - Based entirely on user credentials
  - If we have an account that is privileged to join machines to the domain: Act as silent MITM, learn system account password.
    - *Assuming control of such an account may already be "game over" in many deployments.*
  - Or join to impostor domain, supply policy that provides persistent control, then re-join to real domain once a user with appropriate privilege logs in again.

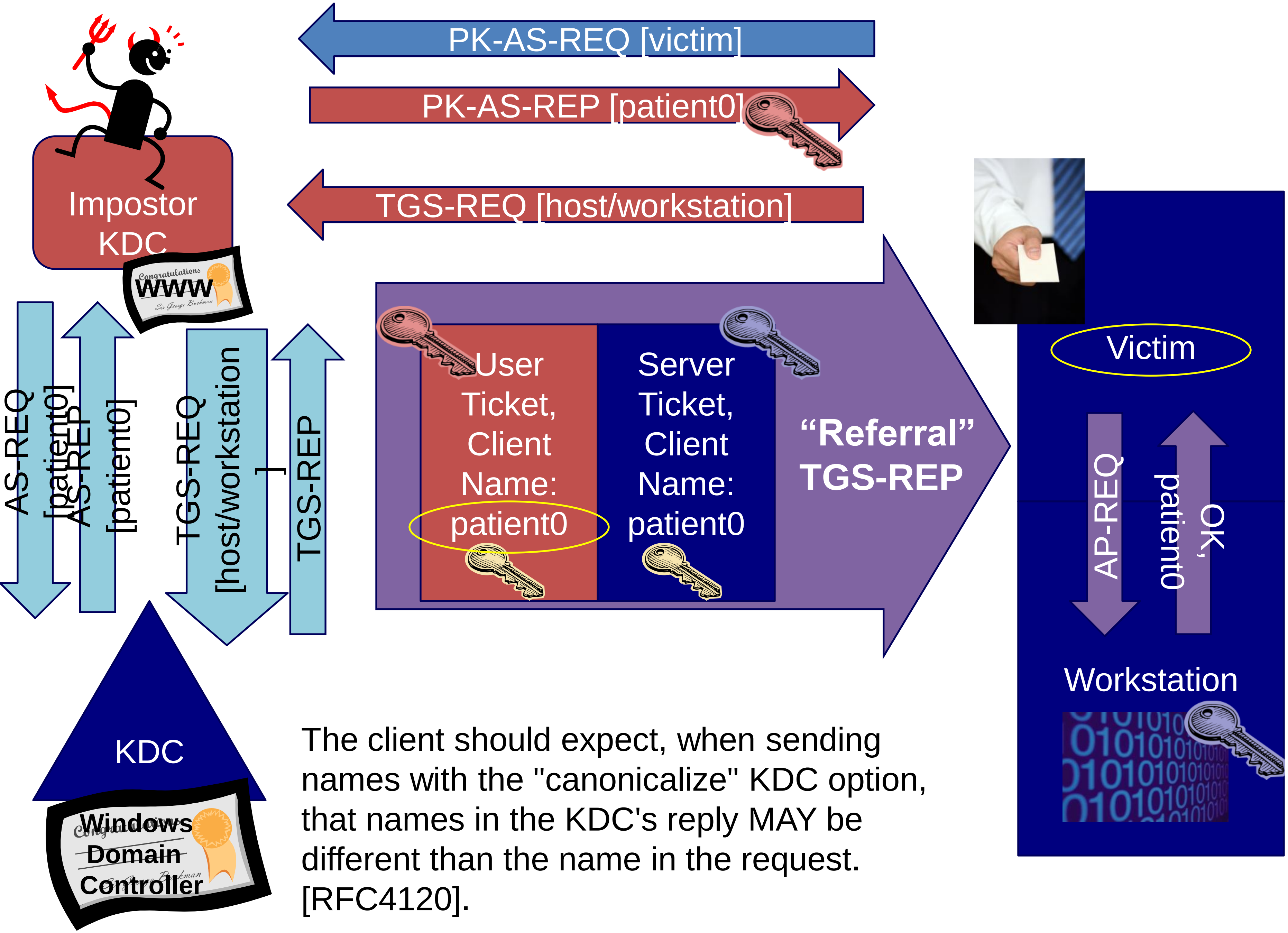
We can do better…

What if we have a conspiring user?

- Usually, all users allowed to logon to all workstations.

- User Principal Name Canonicalization:

“If the "canonicalize" KDC option is set, then the KDC MAY change the client and server principal names and types in the AS response and ticket returned from the name type of the client name in the request. In a TGS exchange, the server principal name and type may be changed.” [draft-ietf-krb-wg-kerberos-referrals-11]



PK-AS-REQ [victim]

PK-AS-REP [patient0]

TGS-REQ [host/workstation]

Impostor KDC

Congratulations WWW Sir George Buckman

AS-REQ [patient0]

AS-REP [patient0]

TGS-REQ [host/workstation]

TGS-REP

User Ticket, Client Name: patient0

Server Ticket, Client Name: patient0

“Referral” TGS-REP

Victim

AP-REQ

OK, patient0

Workstation

KDC

Windows Domain Controller

The client should expect, when sending names with the "canonicalize" KDC option, that names in the KDC's reply MAY be different than the name in the request. [RFC4120].

What now?

- User is at trusted, healthy workstation. Has just logged on with smart card. But we control the interactive user session and can impersonate any other server, push policy as the DC, etc.
- Run a trojan:
  - Put up the “Installing updates…” screen.
  - After a suitable delay, put up the “Insert a smart card to unlock” screen. If user forgets to Ctl-Alt-Delete and enters their PIN.
  - Unlock smart card, make AS-REQ to real KDC.
    - Get NTOWF from PAC supplemental credential buffer
    - Get TGT renewable for 7 days
  - Remove trojan, reboot.

Own One Machine, Own The Enterprise

- Complex and difficult elevation path, but reliable and quiet.

- Few, if any, network forensics traces.
  - Normal protocols on normal ports.
  - Must be allowed through firewalls.
  - Differences between normal and attack payloads are in the encrypted portion of authentication protocols.

- IPSec and DNSSEC won’t stop it.

Smart Card Kerberos Recommendations

Turn on Strict KDC Validation Policy

- Available on Vista SP1 and above
- Retire XP or don't use Smart Cards

- Enroll all Domain Controllers for the "Kerberos Authentication" template first.
  - Included in AD Certificate Services on Windows Server 2008 and later
  - Includes KDC Authentication EKU and Domain DNS and NetBIOS names as Subject Alt Names
  - Still not default for DCs on Windows Server 2008 domains

Domain Join

- Defaults are not secure – and it is hard to apply policy to make it secure before a machine is joined to the domain.
  - Strengthen local policy on default images
- Reduce the number of users privileged to join machines to the domain
  - Audit domain joins (event ID 645)
  - Compare “Caller User Name” to expected
- Use an account with a strong password
- Use offline domain join or join only on an isolated, trusted network

Fixing it for MIT & Heimdal

- Linux KDC = OK *(with good issuance policies!)*

- For Windows KDC, don't turn off *pkinit_require_eku*. Re-issue server certificates as described.

Are Smart Cards better than passwords?

- Yes, but…
- Default configuration for workstations and KDCs in an Active Directory is vulnerable
- Be careful with Enterprise CA management
- Be careful with domain join
- Windows XP crypto and policy options are past their sell-by date for both smart card and standard Kerberos

Hijacking Kerberized Applications

The Punchline

- Replay attacks are often effective against poorly Kerberized services.
  - You want a tie between authentication and protocol, but have to build it yourself.
- Authentication to a Kerberized service is accomplished with the AP-REQ message.
- This message can be replayed (Kasslin, Tikkanen, Virtanen. *Kerberos V Security: Replay Attacks.* AUSTRALIAN INFORMATION WARFARE & IT SECURITY 2004)

AP-REQ

Message Field	Manner of Protection
Version, Message Type, and Options	Unprotected
Target Principal & Realm	Unprotected
Ticket (includes session key, client principal name & address)	Encrypted using service key
Authenticator (ctime, cutime, cksum)	Encrypted using session key

Authenticator

"The authenticator is used to prevent invalid replay of tickets by proving to the server that the client knows the session key of the ticket and thus is entitled to use the ticket." – rfc 4120

Authenticator

- Contains material used to detect replays
- cksum: checksum field
  - Sometimes blank (useless)
  - Service binding (containing a magic number)
  - Can’t protect bidirectional protocol
- cutime, ctime: used to verify AP-REQ freshness
- cname(ticket): contains client network address. Network address is spoofable.
- Cached on service to detect replays (an authenticator can only be sent once)

Why This Doesn’t Work

- A single checksum in adequate for bidirectional protocols
- An attacker actively intercepting traffic is going to send spoofed tickets immediately, will pass cutime/ctime freshness check.
- The attacker will appear to be coming from the client’s ip, checking that does no good.
- The attacker can intercept and send a copy before the client, caching does no good.
- Don’t rely on the authenticator alone to detect replays.

The Right Way To Do It

- “The integrity of the messages exchanged between principals can also be guaranteed by using the session key (passed in the ticket and contained in the credentials). This approach provides detection of both replay attacks and message stream modification attacks. It is accomplished by generating and transmitting a collision-proof checksum (elsewhere called a hash or digest function) of the client's message, keyed with the session key.” – rfc 4120

What can be done

- "You will perish in flames!"

  – Louis Tully

- Developers
  - Ensure session has integrity protection that uses the kerberos established session key
- Administrators
  - Evaluate services for poor kerberos integration. Ensure integrity and encryption are provided between kerberos endpoints (possibly stunnel or ipsec)

What can be done - Windows

- Each major authenticated protocol in Windows provides some mechanism for binding
- If you use something off the beaten path, you may need to call EncryptMessage and SignMessage yourself
- However, the “some mechanism” is not always up to the developer or the system administrator…

Binding Protocols

Protocol Name	Dev Binding	Admin Binding
LDAP	Use LDAP API to require Signing/Sealing	Specify “Require Signing” security policy for client and server.
RPC	Set binding on client to require Packet Integrity or better. Check the same in the security callback on the server.	N/A
DCOM	Set proxy blanket on client to require Packet Integrity or better. Configure service to require the same authorization level.	Set machine-wide default and per-App DCOM authorization levels using the Component Services MMC plug-in.

Binding Protocols

Protocol Name	Dev Binding	Admin Binding
SMB / Named Pipe	N/A	Specify “Digitally Sign Communications Always” for network clients and servers.
HTTPS	Transparent in most applications.	Enable Extended Protection registry keys and on web applications

Mutual Authentication?

- One major benefit of Kerberos is true mutual authentication.
- “Mutual authentication” means, at best, “I have a session key with my friend designated by this SPN.”
- How do self-organizing background services get that SPN? Here’s a hint:

Mutual Authentication?

You do not get mutual authentication if you:

- Ask the attacker what their SPN is
- Call an API that asks the attacker what their SPN is
- Ask DNS what the attacker’s SPN is
- Pull the attacker’s SPN out of a service definition served by…the attacker
- Fail to set a proper, fully-qualified, SPN
- Fail to set any SPN whatsoever

Configuring SPNs

- This is a surprisingly widespread and difficult problem

- The "easy" way is human configuration.
  - Error-prone
  - Fails to scale

- A well-organized set of services can provide secure, automated service resolution
  - Often carries custom requirements and limitations

Thank you for coming!

rachel@isecpartners.com
brad@isecpartners.com
scott@isecpartners.com

What we didn't have time to tell you in one hour can be found at: https://www.isecpartners.com/