

One XSS To Rule The Enterprise



Kurt Grutzmacher - ToorCon X
grutz @ jingojango.net

Who am I?



<http://grutztopia.jingojango.net/>

Corporate Penetration Tester for nearly a decade for the Federal Reserve System (economy not my fault) and now at Pacific Gas & Electric (Enron not my fault).

Dabbler in Metasploit development, getting Free MacWorld passes, half-price LinuxWorld passes, security research, etc.

HACKERS ON COLD HARD DIRT!

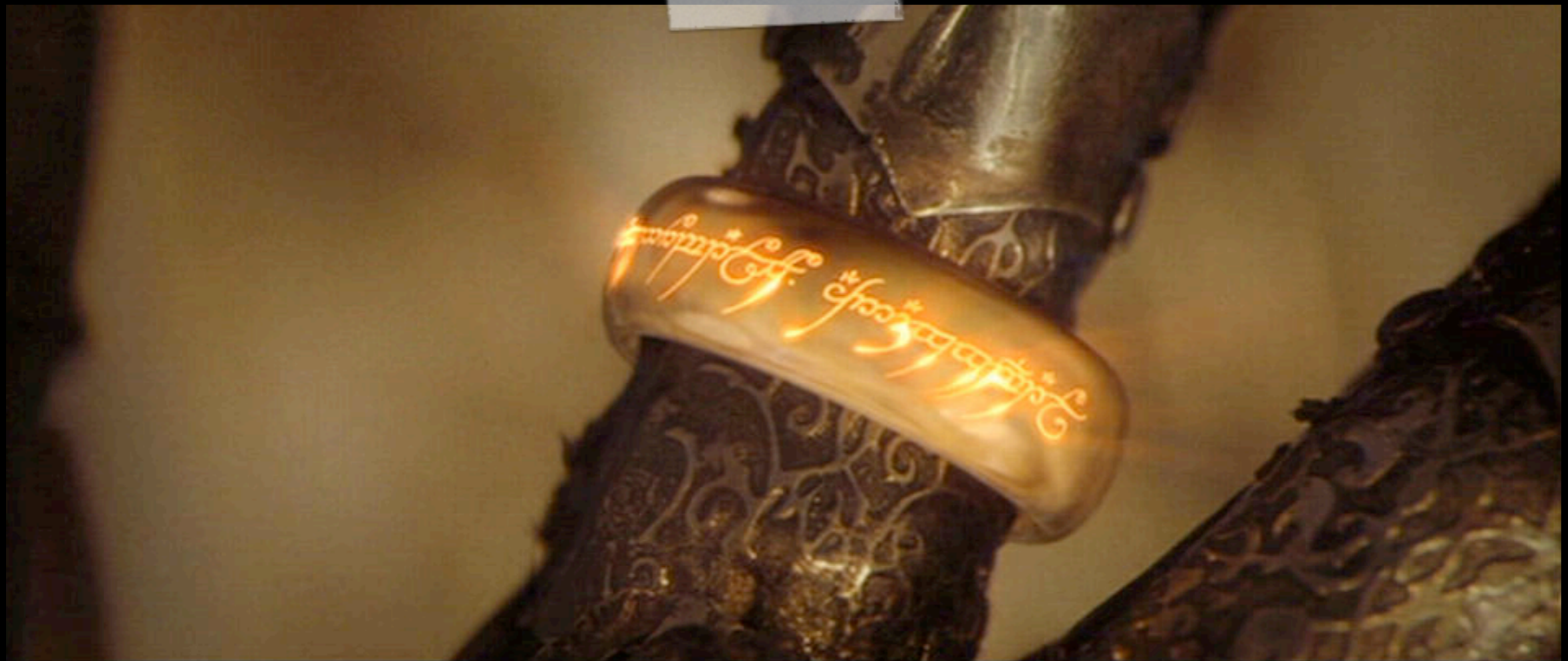


Why this matters

As enterprises started to secure their internal networks we kept finding less and less when it came to “remotely exploitable” systems.

After shifting to more Web Security tests the question came up: “What can you do with an XSS inside the corporate network?”

Answer: How about owning the Domain? (yes, really)



Defining The Precious

Quick Definitions

LM - *LAN Manager*

Really old and really tired, never use this again

Finally disabled by default in Vista and Server 2008

NTLM - *NT LAN Manager*

Replaced “LAN Manager” (for a good reason)

A “suite” of protocols for authentication and security: “NTLM Security Support Provider (NTLMSSP)”

Also known as “ntlm 0.12”

Describes an authentication protocol *and* the hash result

Kerberos - *Kerberos*

But not *just* Kerberos, Microsoft’s **extended** Kerberos!

Scrabble take two

Nonce - *Number Used Once*

Used to defeat replay attacks

SSPI - *Security Support Provider Interface*

Microsoft API to several security routines

SPNEGO - *Simple and Protected GSSAPI Negotiation Mechanism*

I don't know what to speak to you, so lets negotiate!

IWA - *Integrated Windows Authentication*

The act of negotiating authentication type using SPNEGO

Windows (NTLM) Authentication

NTLM Authentication Protocol is a challenge-response scheme that can be broken into three “Types”:

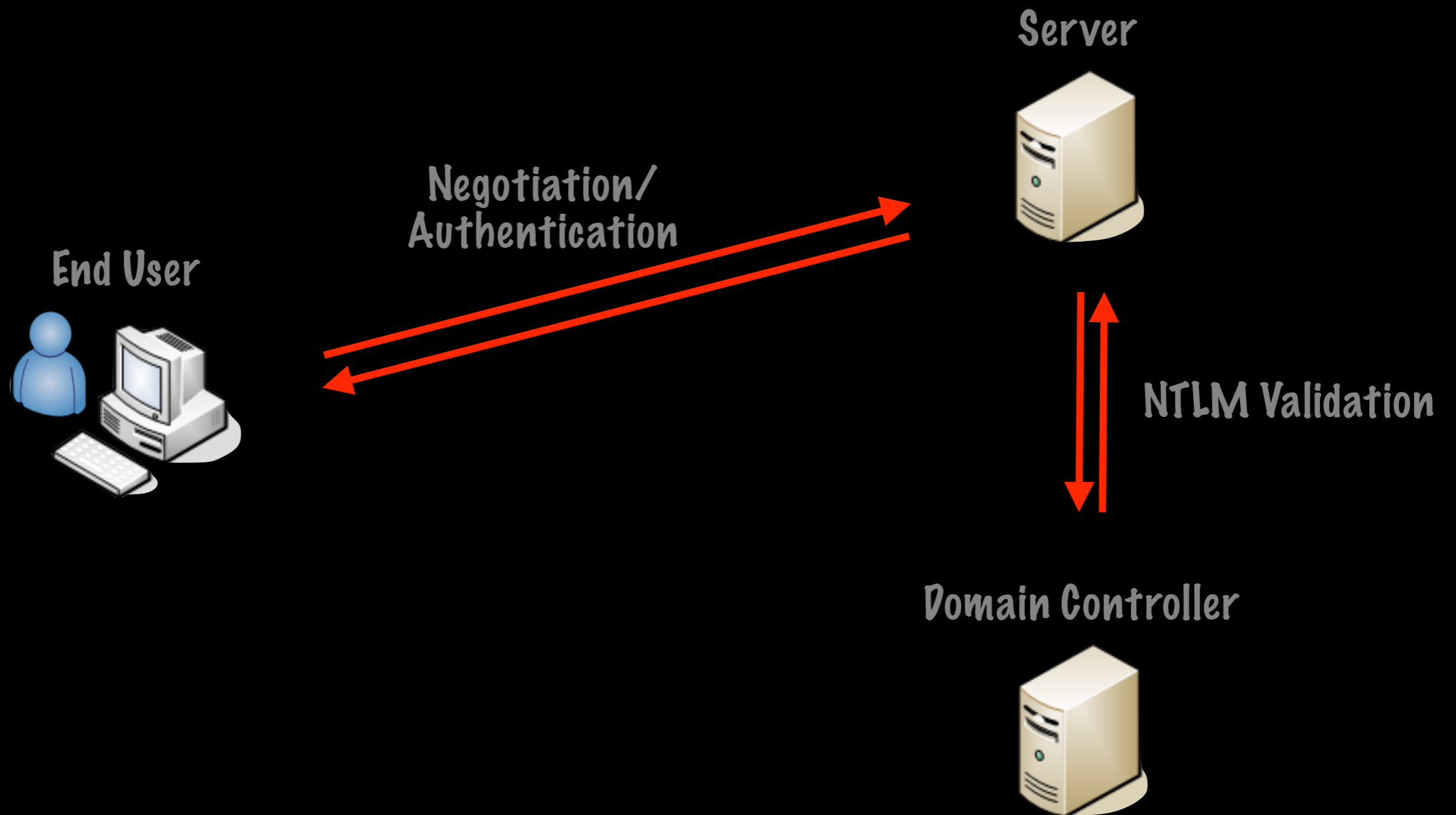
Type 1: Client sends “Hi, I want to talk to you”

Type 2: Server sends “Ok, here are the various features and protocols I support including a *nonce* for you to encrypt your hashes with so nobody can replay it later in case they capture it. Oh and the domain you should authenticate to.”

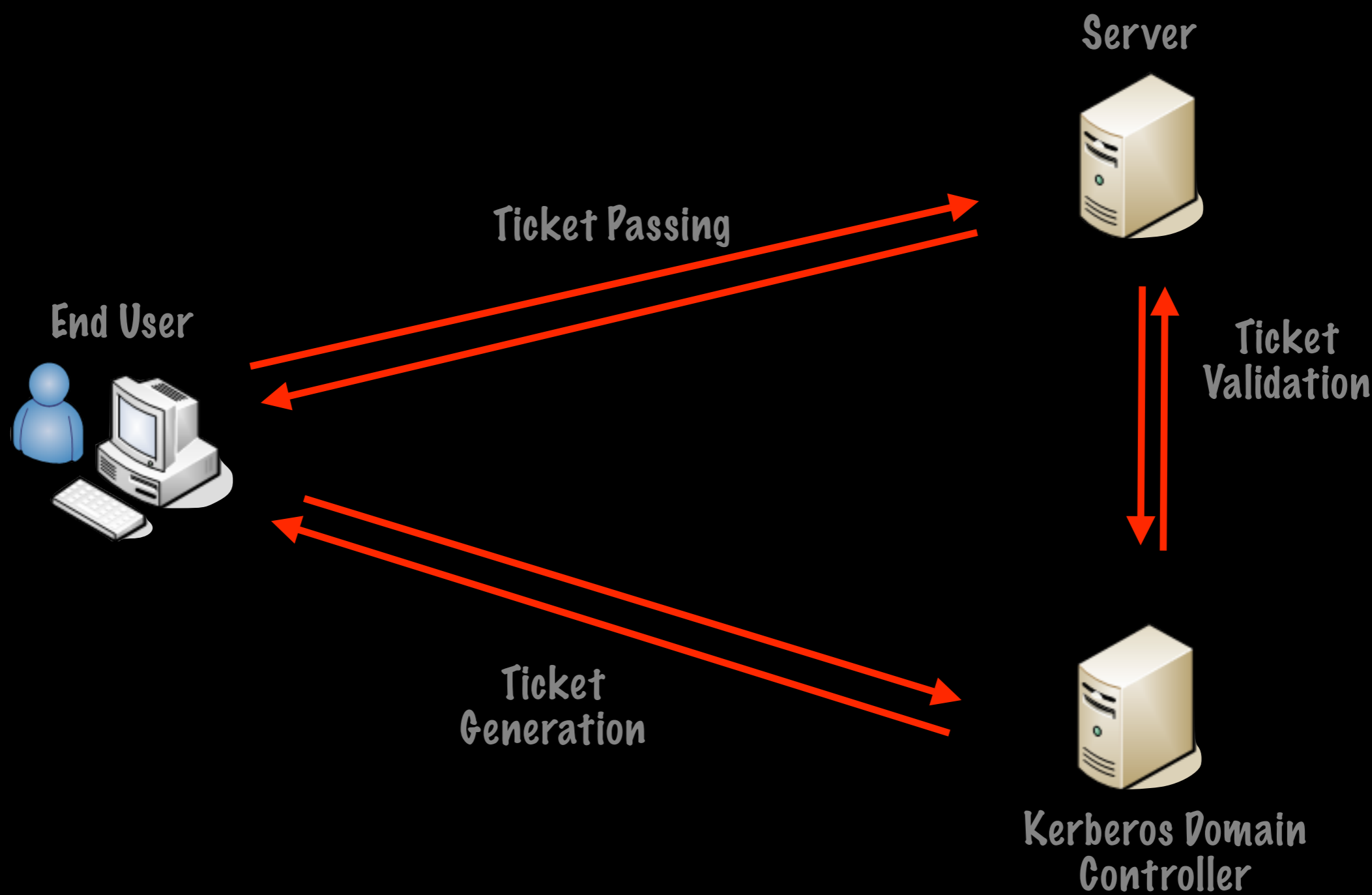
Type 3: Client response “Sweet, I agree on the features you desire and support them in my daily life. Here’s the username, domain again, workstation name, and the encrypted LM and NTLM hashes.”

The server recovers the LM/NTLM hashes and compares them to its internal table and grants / denies accordingly in the response to a Type 3 message.

NTLM Flow



KERBEROS Flow



NTLM is supported...

in Microsoft products (IIS, IAS, Exchange, Internet Explorer)

in Samba and Apache and PAM

in other browsers (Mozilla Firefox and Safari)

in proxy servers to support browsers who don't do NTLM

in your iPhone (really!) for Enterprises

in OSX to connect to Windows shares

in WinCE to connect to Windows shares

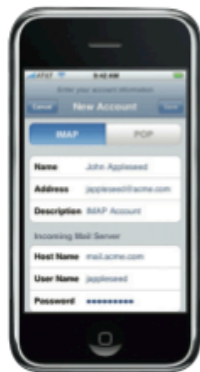
in ToasterBrandConsumerDevice to connect to Windows shares

in * to connect to Windows shares



iPhone and IMAP

iPhone also supports strong authentication methods, including industry-standard MD5 Challenge-Response and **NTLMv2**.



IMAP or POP-enabled mail solutions
 iPhone supports industry-standard IMAP4- and POP3-enabled mail solutions on a range of server platforms, including Windows, UNIX, Linux, and Mac OS X.

Additional information regarding the IMAP4rev1 standard can be found at www.imap.org.

With support for the IMAP mail protocol, iPhone can integrate with just about any mail server environment. If the server supports IMAP and is configured to require user authentication and SSL, iPhone provides a highly secure, standards-based approach to email deployment. In a typical deployment, iPhone establishes direct access to an IMAP-enabled server over port 993 and access to SMTP servers over port 587. These servers can be located within a DMZ subnetwork, behind a corporate firewall, or both. With SSL, iPhone supports 128-bit encryption and X.509 root certificates issued by the major certificate authorities. iPhone also supports strong authentication methods, including industry-standard MD5 Challenge-Response and NTLMv2.

IMAP Network Setup

The IT or network administrator will need to complete these key steps to enable direct access from iPhone to an IMAP-enabled mail solution:

- Open port 993 to allow email to be received through the firewall. The proxy server must be set to IMAP over SSL. SSL ensures that mail is securely encrypted during wireless transmission.
- As a best practice and for additional security protection, install a digital certificate on the server from a trusted certificate authority (CA) such as VeriSign. Installing a certificate from a CA is an important step in ensuring that your proxy server is a trusted entity within your corporate infrastructure.

NTLM seems strong...

It is pretty strong:

1. Cleartext is NOT converted to upper case
2. Passwords are NOT broken into blocks of 7 bytes
3. DES not so good but it's the last step to generate results and client/server nonces protect from pre-computed attacks
4. Server nonces do *not* protect pre-computed attacks however.

In the grand scheme of things the LM and NTLM hashes should be considered the same as cleartext passwords. When obtained an attacker does not need to find the cleartext in order for them to be used.

I needs my precious!

NTLM has shown its survivability by hanging on to “backwards compatibility” and ubiquitous deployment. If it’s everywhere what is the incentive to get rid of it?

You’ve got:

- Replay protection

- Mixed case support from cleartext to ciphertext

- Server nonces in v1

- Client and Server nonces in v2

- Message digests

- Timestamps

..sounds good so far!

Lets not get ahead of ourselves just yet.

In an ENTERPRISE we have the joyful tune of “Single Sign-On”. When a workstation becomes a member of the domain any user that logs on can access their resources with only having to type their password once during the log on process

This means that the cleartext or LM/NTLM ciphertext may be stored within the memory of the workstation throughout the session or beyond!

It also means that authentication can happen at the request of an application and not by a user.



The Past . . . precious. . .

Our Threat Model

The NTLM protocols pre-suppose an Enterprise authentication system using Windows Domains or Active Directory.

Evildoers must fit within this environment in order to take advantage of it so they usually have to have inside access.

Doesn't mean this isn't an external threat, just that at this time I can't think of or have seen an attack from the outside in.

Protocol Downgrade

During SPNEGO the client gets the first word on protocol support:

Signing, Sealing, use NTLM, Always Sign, send Target block, etc.

The server responds with their own list of support:

NTLM2 key, Target block included, 128-bit encryption, etc

If both sides agree the client sends all the requisite data for an authentication attempt and waits for a response.

Using MITM tools such as Cain & Able or Ettercap an attacker can force either side to negotiate LOWER than they would have otherwise.

Replay Attacks

Comes in two forms:

- Network capture and replay if no nonce

- Obtaining the LM/NTLM hashes and using them during auth

“Pass The Hash” is the term and it is pure awesome:

- Obtain privileges on a server or workstation

- Dump a copy of stored hashes (SAM, LSASS, running processes)

- Skip the part of “converting to LM/NTLM” during the Network authentication routines

- Who needs to crack hashes anymore?

Tools for Replay

Obtain hashes:

FGDump

PWDumpX

Cain & Able

Pass The Hash Toolkit

Metasploit, Canvas, CORE Impact

Passing The Hash

Hydra

Patches for Samba from JoMoKun

Pass The Hash Toolkit

NTLMAPS P-t-H

Metasploit, Canvas, CORE Impact

SMB Relay (original)

<http://www.xfocus.net/articles/200305/smbrelay.html>

First released in March, 2001 at @tantaCon by Sir Dystic of cDc

Listens for NBT requests and collects LM/NTLM hashes for cracking

Version 1:

- Connects back to the requester using their credentials

- Emulates an SMB server for the attacker to connect to

- TCP/IP Addresses only

- Generally great for one-off attacks

Version 2:

- Supported NetBIOS names

- Relay to a third-party host

SMB Relay (Metasploit)

Re-engineering of SMB Relay script as a Metasploit attack module

Metasploit already had LM/NTLM hash capture support since 2.7

Can connect back to original host or forward to a single host

Works great if:

- Users are local administrators

- Server service has been started on their workstations

- or the users have rights to your destination host

See last year's "Tactical Exploitation" presentation for other cool ideas.



Give Me My Precious!

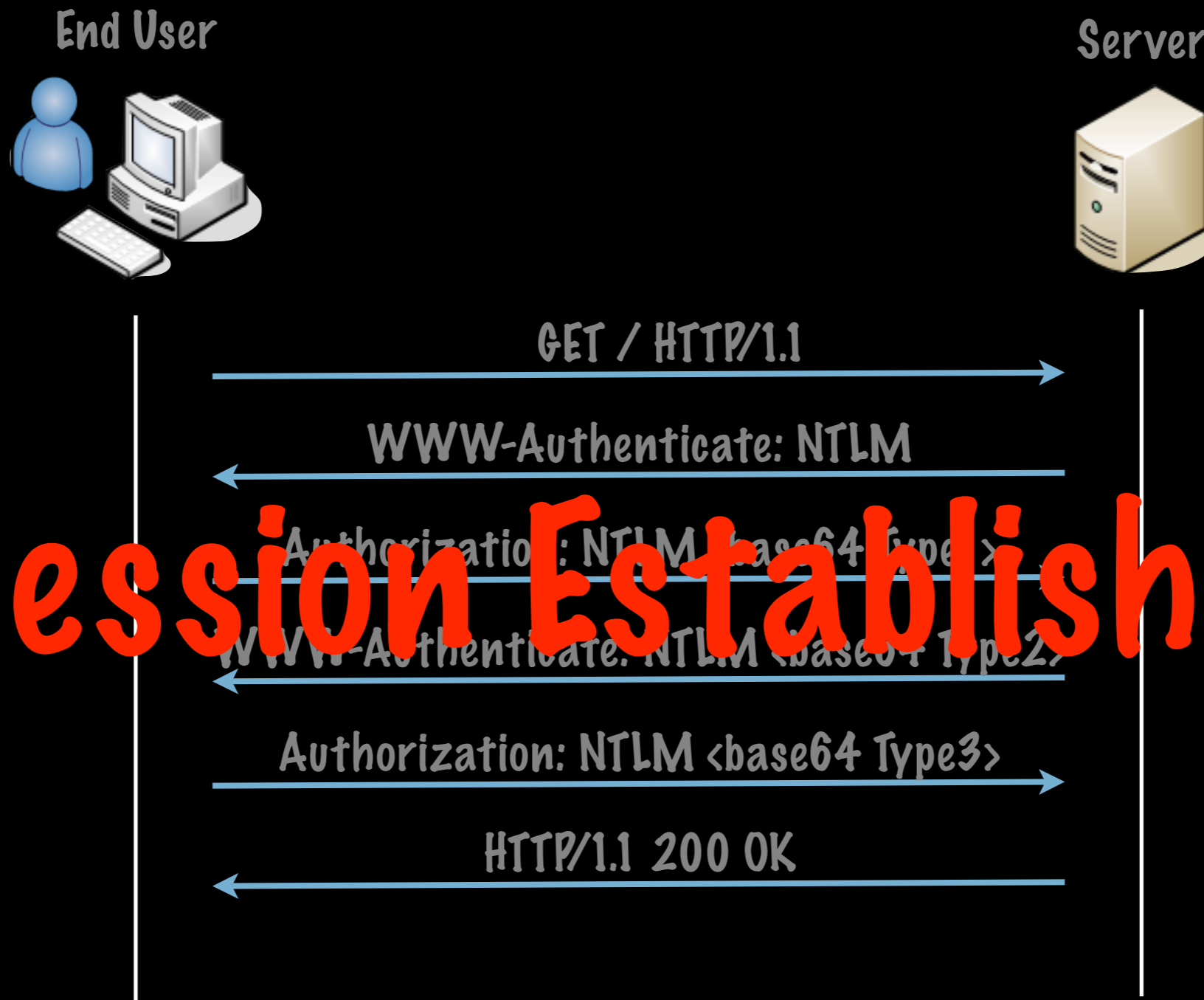
NTLM over ...

HTTP, IMAP, POP3, SMTP, NNTP, etc. . .

While NTLM is a Microsoft protocol, in order to fully support SSO it has to support standard protocols. NTLM “Type Messages” are Base64 encodings of the NTLM protocol to transmit over 7-bit protocols.

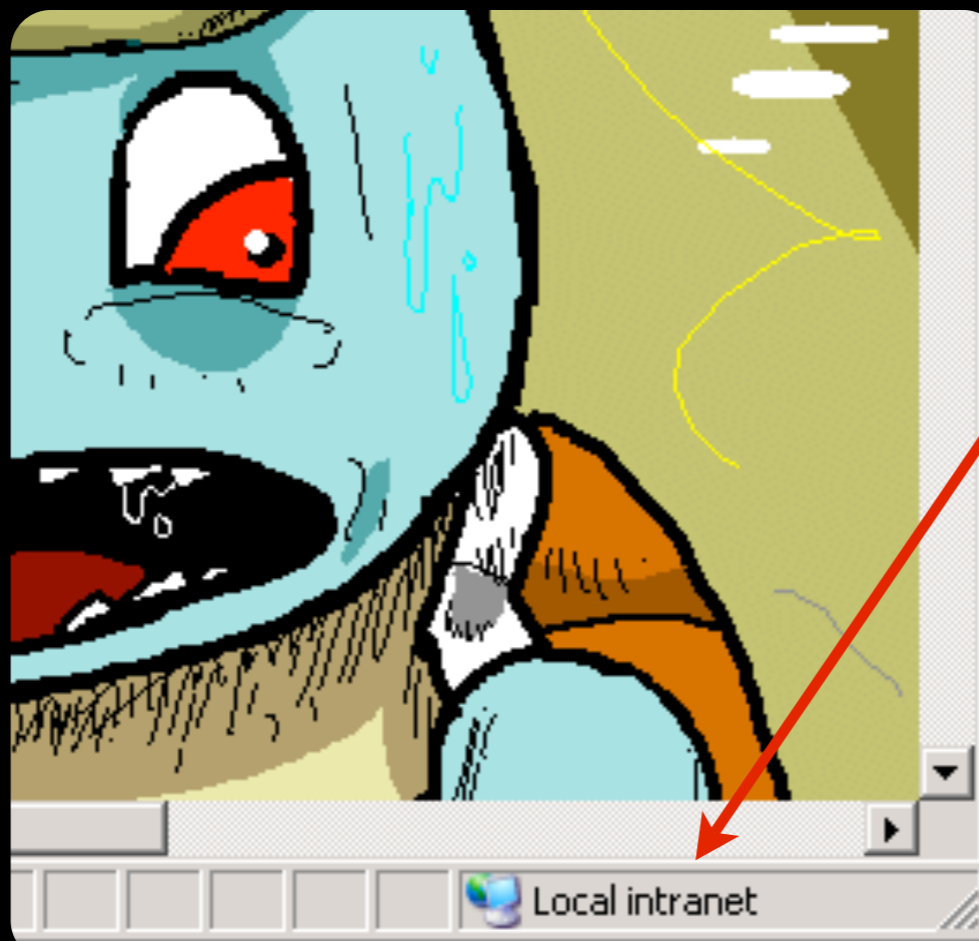
Part of the Integrated Windows Authentication suite.

HTTP NTLM Auth



IE Trust Zones

In order for Internet Explorer to perform Integrated Windows Authentication the browser must be in the “Local Intranet” or a customized zone.



Automatic Authentication

<http://support.microsoft.com/kb/258063>

The following conditions must be met for Internet Explorer to automatically authenticate a user's logon and password and maintain security:

- Windows Integrated authentication, also known as Windows NT Challenge / Response, must be enabled in the Web site properties in IIS. Anonymous authentication is attempted first, followed by Windows Integrated authentication, Digest authentication (if applicable), and finally Basic (clear text) authentication.
- Both the client and the Web server must be either in the same Microsoft Windows NT-based or Microsoft Windows 2000-based domain or in trusted Windows NT-based or Windows 2000-based domains in which the user's account can be granted permissions to resources on the IIS-based computer.
- The user's browser must be Internet Explorer. Internet Explorer is the only browser that supports Windows Integrated authentication (NTCR).
- Internet Explorer must consider the requested URL to be on the intranet (local). If the computer name portion of the requested URL contains periods (such as <http://www.microsoft.com> and <http://10.0.0.1>), Internet Explorer assumes that the requested address exists on the Internet and does not pass any credentials automatically. Addresses without periods (such as <http://webserver>) are considered to be on the intranet (local); Internet Explorer passes credentials automatically. The only exception is addresses included in the Intranet zone in Internet Explorer.

Mozilla Auth Setup

Firefox supports NTLM! In about:config

Enable IWA:

```
network.negotiate-auth.trusted-uris: list,of,uris  
network.negotiate-auth.using-native-gsslib: true
```

Or just NTLM:

```
network.automatic-ntlm-auth.trusted-uris: list,of,uris  
network.ntlm.send-lm-response: false
```

Firefox 3.0 does honor IE's Trust Zones but not for IWA.

NTLM in <browser>

Opera does not support NTLM authentication directly, you must go through a proxy server.

Safari for Windows will do NTLM but does not do Integrated Windows Auth. Typically a proxy server is used for OS X or stored credentials in the keychain.

Wget/CURL both support NTLM on the command line.

Links/Lynx ... why? maybe, not something I checked - usually use a proxy server like NTLMAPS.

Monkey In The Middle?



What does this mean?

As a **Rogue Server** we're able to bridge authentication requests using Type Messages by directing HTTP requests to us ()

NTLMv2 messages pass through without modification

This was previously possible via SMBRelay but very limited target scope using WPAD+SOCKS or forcing file:// or smb:// connections

Jesse Burns @ iSEC Partners described the HTTP->SMB link in 2004 but never released source code

In late 2007 I implemented a hash collector and HTTP->POP3 bridge

Earlier this year Eric Rachner released "scurvy" @ ToorCon Seattle



Hi Squirtle!

Squirtle? What the...

Squirtle is a **Rogue Server with Controlling Desires!** It does not require:

Man In The Middle techniques such as:

ARP Poisoning

DNS Redirection

GRE Tunnels

Squirtle does require:

The browser be in a “trusted zone” for IWA to work

Support for WWW-Authenticate: NTLM

You somehow direct the browsers to it (XSS, proxy, , etc)

Implements “*Pass The Dutchie*” attacks:

Dutchie == Server requirements and client response

Why it all matters. . .

The screenshot shows the OSVDB website interface. At the top, there is a navigation bar with the OSVDB logo and links for 'Search OSVDB', 'Browse', 'Vendors', 'Project Info', 'Help OSVDB!', and 'Spons'. Below the navigation bar, there is a 'DONATE NOW!' button and a 'User Status' section with a link to '>Account'. A 'Quick Searches' section contains three search boxes: 'General Search', 'OSVDB ID Lookup', and 'Vendor Search', each with a 'Go' button. Below this is an 'Advertisements' section. The main content area displays search results for the query 'xss'. It shows 'Results: 6961' and links for 'Hide Narrow Search' and 'Show Descriptions'. The search query is 'xss'. A table lists the number of vulnerabilities for each year from 2001 to 2006, with a link to 'Remove XSS'.

| Vulnerability Type | Disclosure Year |
|----------------------------|---|
| Remove XSS | 2006 2004 |
| | 2005 1355 |
| | 2007 1348 |
| | 2008 1009 |
| | 2004 619 |
| | 2003 363 |
| | 2002 218 |
| | 2001 38 |

Passing The ~~Dutchie~~ Precious!

Clients are given a pre-computed session key during their first connection to Squirtle in a browser cookie. All requests reference this key.

keepalives are sent from the Client to the Squirtle controller for action requests. The timeout is specified within the config file.

Evil Agents can request specific clients authenticate with a given nonce or specific variables.

Evil Agent requests are “protected” via basic-auth.

Data and Session State is logged and maintained in a SQL database (sqlite, mysql or postgres)

Passing The Dutchie

Evil Agent



Squirtle



Controlled Clients



Session 13 Response with this Nonce



NTLMv2 challenge



SQUIRTLE!
SQUIRTLE!

Authentication Nonce



NTLMv2 challenge
Session 13
Authentication
Completed
Awaiting response!



Enterprise Server

| Session | Username | Req Type | Type2 |
|---------|----------|----------|---------------|
| 1 | A | Nonce | Type2 message |
| 2 | B | Nonce | Hashes |
| 3 | C | Nonce | Hashes |

XSS on page file



Corporate Homepage

What does this mean?

Past attacks against Windows authentication have been either directed at a single server or back towards the client.

By corralling clients and exposing an API to externally written tools, Squirtle allows proxy servers to be written in any language that the attacker desires. They don't need to worry about grabbing clients and holding on to them, let Squirtle do that.

Existing frameworks such as Metasploit, Canvas and CORE Impact can use Squirtle to perform attacks against resources that require authentication without having to obtain cleartext or LM/NTLM hashes!

Thinking about it...

Squirtle was created because after finding a ton of internal XSS vulnerabilities the client's response always seemed to be: "Great, you can run a port scanner or send print jobs. What else ya got?"

So here's what we got:

- Internal servers with web programming errors (XSS, SQL, etc)

- Open SMB c:\inetpub\www shares with write access

- An internal PHPNuke or Blog or Other "Open Source" Scripts

- Sending an E-mail with a link inside of it

- The evil act of opening Microsoft Office documents

They will all be controlled by the mighty Squirtle!

Evil Agent Functions

URI

Description

/controller/listsessions/

List all session states

/controller/allhashes/

List all hashes

/controller/allusers/

List all users

/controller/listuser/

List a specific user

/controller/session/

List a specific session

/controller/redirect/

Force a redirect (lose the client)

/controller/static/

Request static auth

/controller/type2/

Request response to Type 2 msg

/controller/clearsession/

Clear a session state

Integrating Squirtle

Easy fit into proxy / attack tools that already support NTLM:

Step 1: Generate a Type 1 message to the server

Step 2: Take the raw Type 2 response and Base64 encode it

Step 3: Send it to Squirtle for processing

/controller/type2?key=<keyid>&type2=<base64 message>

Step 4: Base64 decode Squirtle's result

['result'] = <base64 result>

Step 5: Enter the packet details into the response.

In some cases where Base64 Type Messages are already used (HTTP, IMAP, etc) skip Step 2 and use the Squirtle result as it is.

Python Example

```
def process_squirtle(self, SQURL, SQUSER, SQPASS, SQKEY, msg2):
    """
    msg2 = urllib.quote(msg2)
    auth_handler = urllib2.HTTPBasicAuthHandler()
    auth_handler.add_password(realm='Squirtle Realm', uri=SQURL, user=SQUSER, passwd=SQPASS)
    urloper = urllib2.build_opener(auth_handler)
    urllib2.install_opener(urloper)

    dutchieurl = "%scontroller/type2?key=%s&type2=%s" % (SQURL, SQKEY, msg2)
    try:
        res = urllib2.urlopen(dutchieurl)
    except urllib2.URLError, e:
        print '*** Error talking to Squirtle.' + str(e.code) + ': ' + e.reason + '\n'
        return ''

    response = res.read()
    try:
        response = simplejson.loads(response)
    except Exception, e:
        print '*** Error receiving response from Squirtle: ' + response + '\n'
        return ''

    if response['status'] == 'ok':
        return response['result']
    else:
        print '*** Response from Squirtle: ' + response['status'] + '\n'
        return ''
```

Rrrrrrruby

```
def process_squirtle(sqdata = '')
  require 'json'
  type2pkt = self.pktdata[self.pktdata.index("NTLMSSP")..self.pktdata.length]
  type2msg = Rex::Text.encode_base64(type2pkt)
  sqclient = Rex::Proto::Http::Client.new(sqdata['SQHost'], sqdata['SQPort'])
  begin
    req = sqclient.request_cgi(
      'method'      => 'GET',
      'uri'         => '/controller/type2',
      'vars_get'    => { 'key' => sqdata['SQKey'], 'type2' => type2msg },
      'basic_auth' => sqdata['SQAuth']
    )
    resp = sqclient.send_recv(req, 500)
    parsedresp = JSON.parse(resp.body)
    if (parsedresp['status'] == 'ok')
      (domain, user, host, lm, nt) = Rex::Proto::SMB::Utils.process_type3_message(parsedresp['result'])
    else
      (domain, user, lm, nt) = '', '', '', ''
    end
  end
  ensure
    sqclient.close
  end
  return domain.to_s, user.to_s, lm.to_s, nt.to_s
end
```

Changes . . .

Proxy integration:

NTLMAPS (done, don't need no management interface!)

Metasploit Integration (almost there. . . is it Sunday yet?)

SMBClient (from IMPACKET -> 81% complete)

up-imaproxy (73.9492% complete)

Proxies that keep TCP connections open are awesome!

Administrative web or Flex / Air interface (TODO still - curl works tho :)

~~Additional integration with MySQL, Postgress, etc~~ **COMPLETE!**

Sample API code for Python, Ruby, C (libcurl)

<http://code.google.com/p/squirtle>

bidoof
bidoof
bidoof is on fire



we dont need no
water let that
poke-y-man burn



im dying squirtle



Demonstrations

Squirtle → Web Proxy

1. Modified “ntlmmaps” proxy to talk to Squirtle when web server requests NTLM authentication.
2. Sends Squirtle the Type 2 message for processing using an attacker-specified user.
3. Squirtle builds the request, waits for the keepalive timeout, then requests the controlled browser authenticate.
4. Resulting Type 3 message is delivered back to Squirtle and then passed along to the server to complete authentication.
5. NTLMv2 APPROVED!

DEMO!

Squirtle → smbclient

1. Modified Metasploit to talk to Squirtle for NTLM authentication.
2. Sends Squirtle the negotiation packet in Base64 and the session key.
3. Squirtle builds the request, waits for the keepalive timeout, then requests the controlled browser authenticate.
4. Resulting Type 3 message is delivered back to smbclient where it's unpacked and placed into their respective SMB fields.
5. SMB packet delivered to server to complete authentication.

(still not there, but the theory is sound!!)



Saving The Precious

Kerberos!

Kerberos-based Windows Authentication will halt this attack but not every enterprise can support it yet.

You can not force a Windows client to not allow NTLM Auth.

The NTLM protocol will be supported for a long time because:

- Kerberos requires the client and the server can talk to the TGT

- NTLM only requires the server be able to talk to an authority

Will Kerberos hold? Only time will tell but it's been pretty good so far.

Stop the pain!

For Web Servers:

Turn off support for “WWW-Authenticate: NTLM”

Enforce Kerberos authentication

For SMTP, IMAP, NNTP, everything else etc.

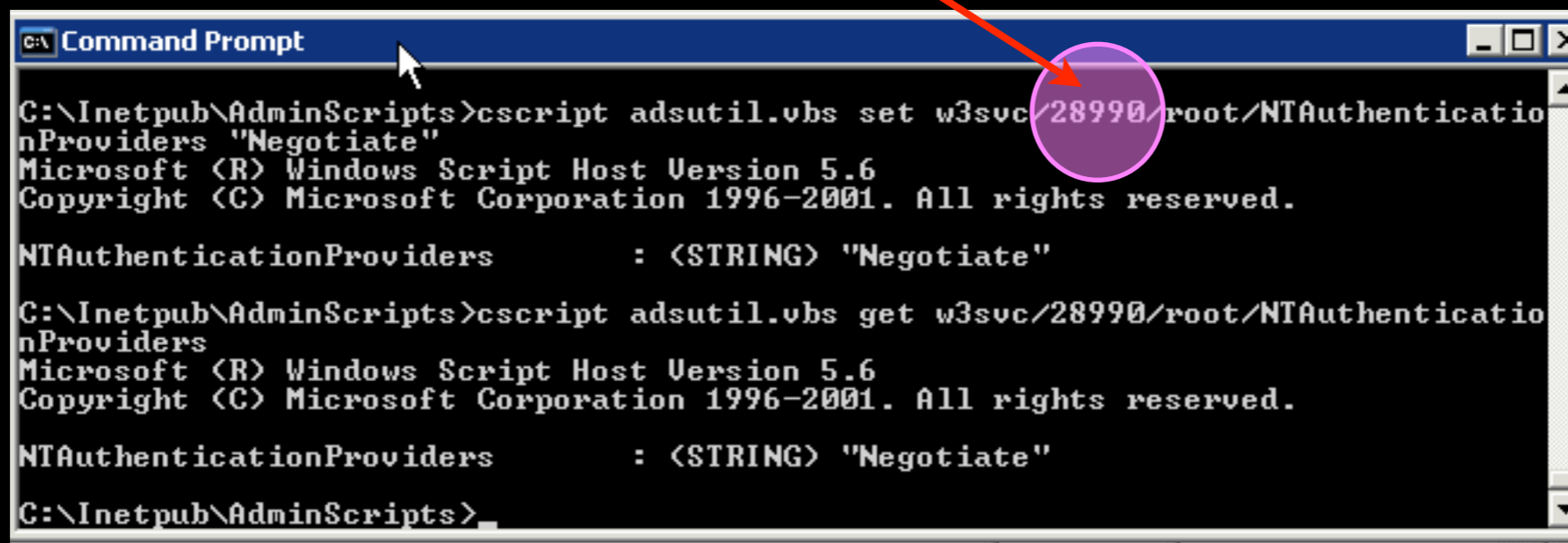
Enable Kerberos authentication

For all:

Require message signing for all communication at the very least.

Protecting IIS

For each instance, follow <http://support.microsoft.com/kb/215383>
`cscript adsutil.vbs set w3svc/instance#/root/NTAuthenticationProviders "Negotiate"`



```
C:\Inetpub\AdminScripts>cscript adsutil.vbs set w3svc/28990/root/NTAuthenticationProviders "Negotiate"
Microsoft (R) Windows Script Host Version 5.6
Copyright (C) Microsoft Corporation 1996-2001. All rights reserved.

NTAuthenticationProviders      : <STRING> "Negotiate"

C:\Inetpub\AdminScripts>cscript adsutil.vbs get w3svc/28990/root/NTAuthenticationProviders
Microsoft (R) Windows Script Host Version 5.6
Copyright (C) Microsoft Corporation 1996-2001. All rights reserved.

NTAuthenticationProviders      : <STRING> "Negotiate"

C:\Inetpub\AdminScripts>
```

This will break NTLM-only supported systems like NTLM proxies so do your testing before hand.

Signing/Sealing?

Signing maintains the integrity of the message

Sealing provides message confidentiality

NTLMv1 uses RC4 and a 40-bit key

NTLMv2 adds a lot more complexity with bigger key spaces

I've been told Signing/Sealing would halt this attack but I'm not convinced.

This is supposed to stop the modification of the packet in-transit however Squirtle does not do ANY packet modification

Testing further. . .

Forcing the client

Not possible. If a browser is in the Local Intranet zone and sees the “WWW-Authenticate: NTLM” header they will attempt to authorize with it.

Best bet at the moment is to enable NTLMv2-only, packet signing/sealing and get rid of all your Windows NT servers (you have gotten rid of them, right? RIGHT?)

At least with NTLMv2 decryption will take a long time.

(definition of “long” may change over time)

Thanks!

Squirtle: <http://code.google.com/p/squirtle>

Pass The Hash Toolkit: <http://oss.coresecurity.com/projects/pshtoolkit.htm>

FGDump: <http://www.foofus.net/fizzgig/fgdump/>

Cain & Abel: <http://www.oxid.it/cain.html>

Special thanks to natron and parity, two guys who kept things interesting!

<http://grutztopia.jingojango.net/> <-- Slides available and other useless crap