SEC Consult

Spooky authentication at a distance

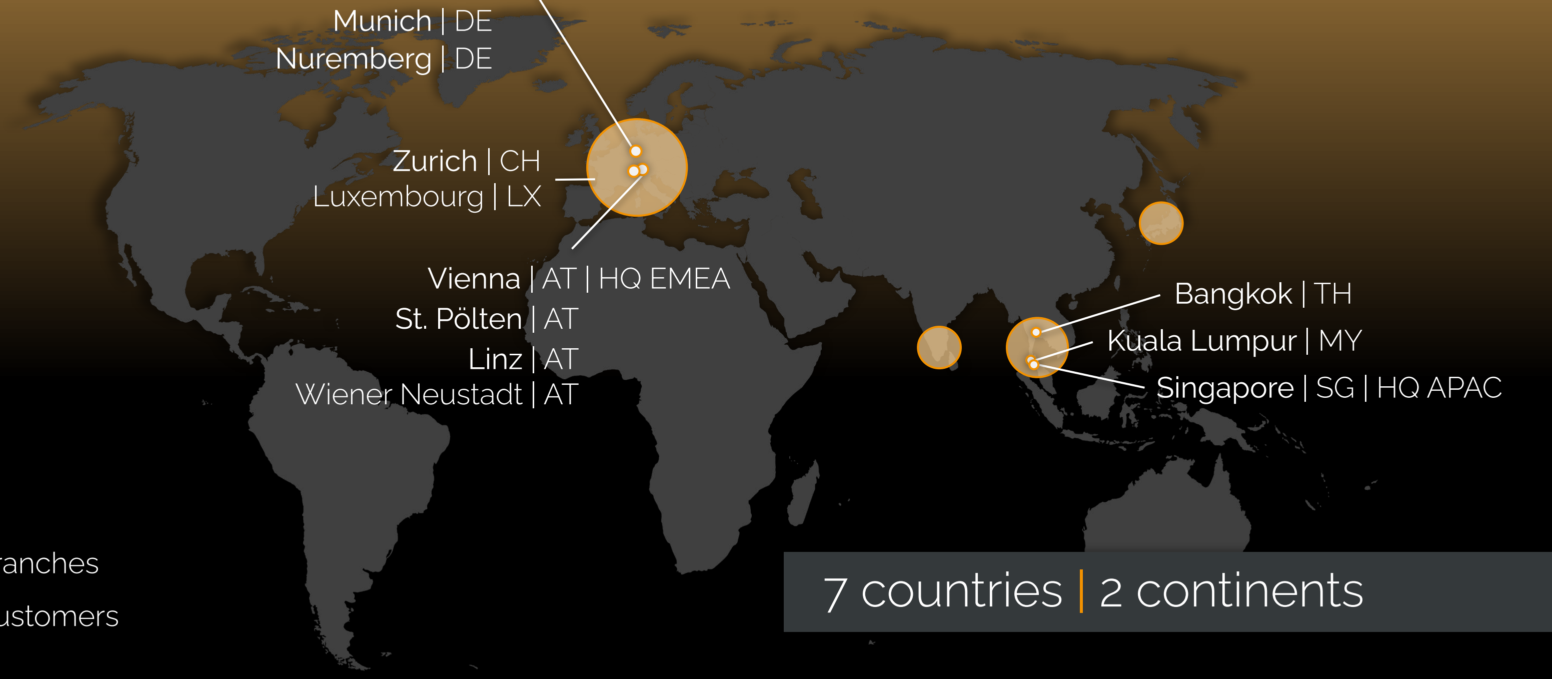
SEC Consult

an atos company

Title: Spooky authentication at a distance | Responsible: T. Jos | Version / Date: V1.0/2023-06 | Confidentiality Class: Public

SEC Consult

an atos company



Berlin | DE
Bochum | DE
Munich | DE
Nuremberg | DE

Zurich | CH
Luxembourg | LX

Vienna | AT | HQ EMEA
St. Pölten | AT
Linz | AT
Wiener Neustadt | AT

Bangkok | TH
Kuala Lumpur | MY
Singapore | SG | HQ APAC

SEC Consult Branches

SEC Consult Customers

7 countries | 2 continents

SEC Consult Vulnerability Lab

AVAYA
TREND MICRO
HÖRMANN
ZTE
konzept-ix.com
PEPPERL+FUCHS
sage
ZYXEL
SevOne
PLATINUM
riverbed
MOXA
Webtrekk
EQDKPlus
FORTINET
genua.
navetti
PHOENIX CONTACT
php fusion
ManageEngine
SYBASE
GO
Fronius
UBIQUITI NETWORKS
InfiRay
IrfanView
altus
SALTO inspiredaccess
eBrigade
skype
adb
WAGO
RSA
Governikus KG
LimeSurvey
gemalto a Thales company
BROADCOM
ORACLE
PIMCORE
Progress
Zeta Producer
SIGLENT
VDG
SecurEnvoy
SONY
WSO2
FileRun
Namirial Information Technology
XM 雄迈
sprecher automation
ptc
VGATEOBD
citrix
MyBiz
SOFICO
aruba
Vibratissimo
Info-ZIP
SilverStripe
phpBB
KERIO
AGFEO
ATLASSIAN
TYPO3
Solar-Log
JUNG
NORTEL
portrait displays
MyBB
sonatype
LINE
ASUS
ubee
devolo
AMX HARMAN
SKYBOX SECURITY
Miele
KATHREIN
MICRO FOCUS
SIEMENS
LINKSYS
Novell
openEMR
OpenProject
COMPAREX
Wonder CMS
umbraco
.js
tao
CoDeSys
GPartEd
f5
SAWMILL
VERGE
WEB.DE
Alcatel-Lucent
RED LION
MySQL
KONICA MINOLTA
vizensoft
wipanda
RefinedWiki
APACHE SOFTWARE FOUNDATION
digitalSTROM
ALLNET
nacromedia
McAfee
KCodes
tp-link
LOXONE
snom
HiKAM
GUIDANCE SOFTWARE
GENTICS
Alfresco
OPENVPN
Bitdefender
Symantec.
shopizer
SCHRACK TECHNIK
NICE
Red Hat Ansible
documentum by EMC
arkeia SOFTWARE
Pixel Tonic
MICROSENS
T Mobile
sas
Schneider Electric
TitanHQ
ZABBIX
vodafone
KODI
ctera
PLEX
NetApp
NetIQ
JFORUM
nitro
censornet.
SOPHOS
soSci der onlineFragebogen
GROUNDWORK
Barracuda
NET MODULE
Apache OpenOffice
IceWarp
FUJITSU SIEMENS
AppliCure Technologies
modsecurity Open Source Web Application Firewall
HUAWEI
websense
cisco
horde
UNIRGY
Voxtronic.
SecCommerce
PHOTOGRAPHY on the Net
Check Point SOFTWARE TECHNOLOGIES LTD
WORDPRESS
ergon
Plone
SOURCEFORGE
SITECORE
SiS
ELBA electronic banking
SONICWALL
READSOFT
SCALIX
NOKIA
shopmetrics
korenix
GMX
vtiger
Toenda Software
YAHOO!
sugarcrm
motorola
LINBIT
W-OGORO
AXIS COMMUNICATIONS
WD
BMD
CODESYS
flatCore
SAP
php
level one

SEC Consult

an atos company

Spooky authentication at a distance

Why I don’t care about your passwords

About me

Name: Tamas Jos

Aliases:

@skelsec

github.com/skelsec

Night Job:

Open-source developer

Creator of many security projects

Member of Porchetta Industries

Bigger Projects-o-mine:

pypyKatz

MsLDAP

aioSMB

Jackdaw

minikerberos

Aardwolf

antlmrelay

Amurex

Other: Likes to go on tangents and rants about unrelated topics

The promise

By the end of this presentation you’ll learn about a new technique that allows you to:

- Use the authentication context of a domain user, of a remote machine, from your own computer

Requirements:

- Small agent running on target machine
- No dumping of credentials, in fact we won’t be seeing any plaintext credentials
- No administrator privileges required
- No special permissions required (e.g. special tokens etc.)
- No exploits, everything that will be shown here is “by-design”

Summary

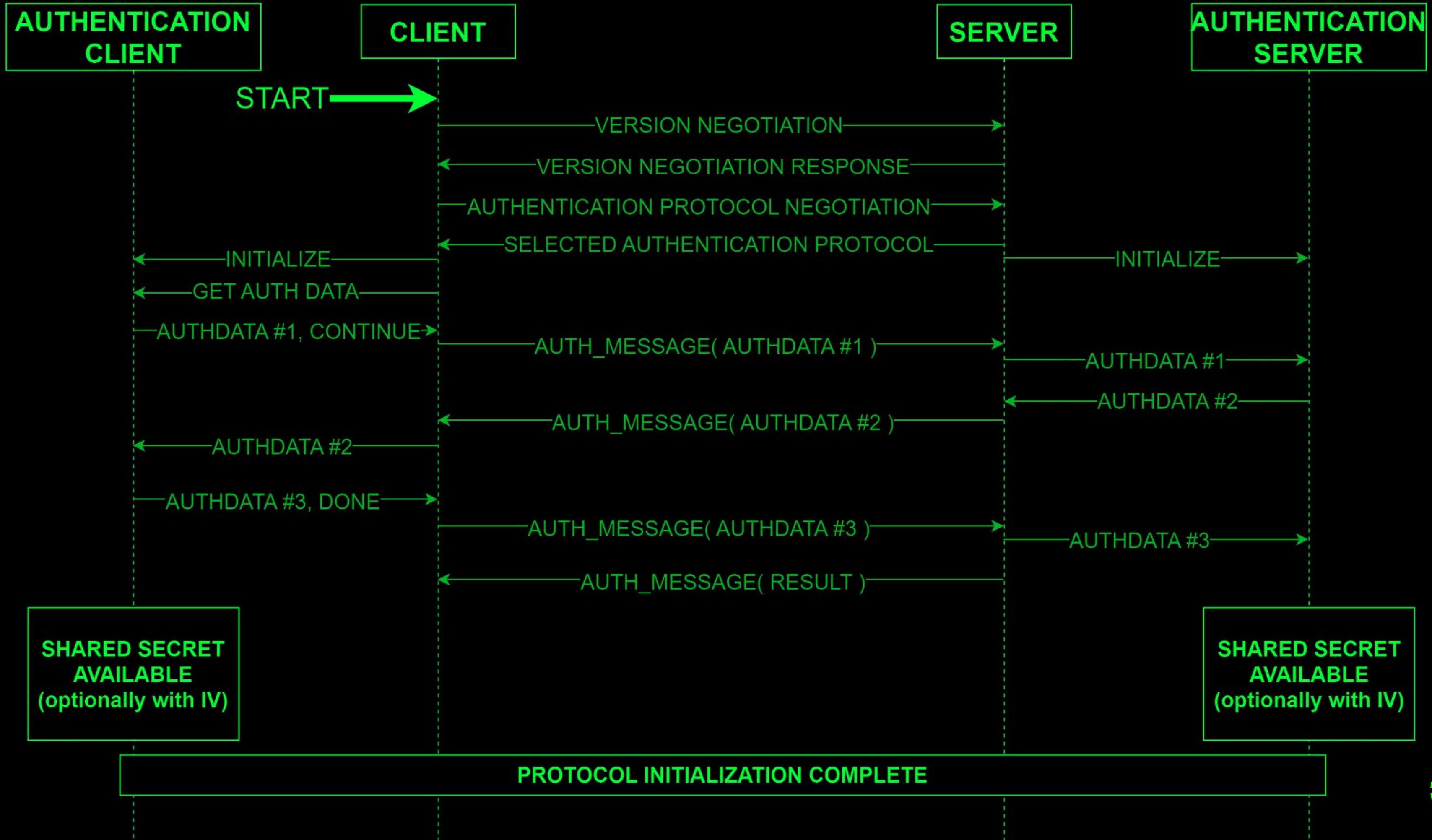
- Windows Authentication Protocols
  - General description
  - Authentication methods in LDAP (and a bit of SMB)
  - LADDERS SO MANY LADDERS
- What is SSPI
  - General description
  - How to use it
- How to authenticate locally, but remotely, but also locally…

Windows Authentication Protocols

- In a nutshell -

Authentication protocols - where are they?

Authentication protocols - Generalization



AUTHENTICATION CLIENT

CLIENT

SERVER

AUTHENTICATION SERVER

START

VERSION NEGOTIATION

VERSION NEGOTIATION RESPONSE

AUTHENTICATION PROTOCOL NEGOTIATION

SELECTED AUTHENTICATION PROTOCOL

INITIALIZE

INITIALIZE

GET AUTH DATA

AUTHDATA #1, CONTINUE

AUTH_MESSAGE(AUTHDATA #1)

AUTHDATA #1

AUTHDATA #2

AUTH_MESSAGE(AUTHDATA #2)

AUTHDATA #2

AUTHDATA #3, DONE

AUTH_MESSAGE(AUTHDATA #3)

AUTHDATA #3

AUTH_MESSAGE(RESULT)

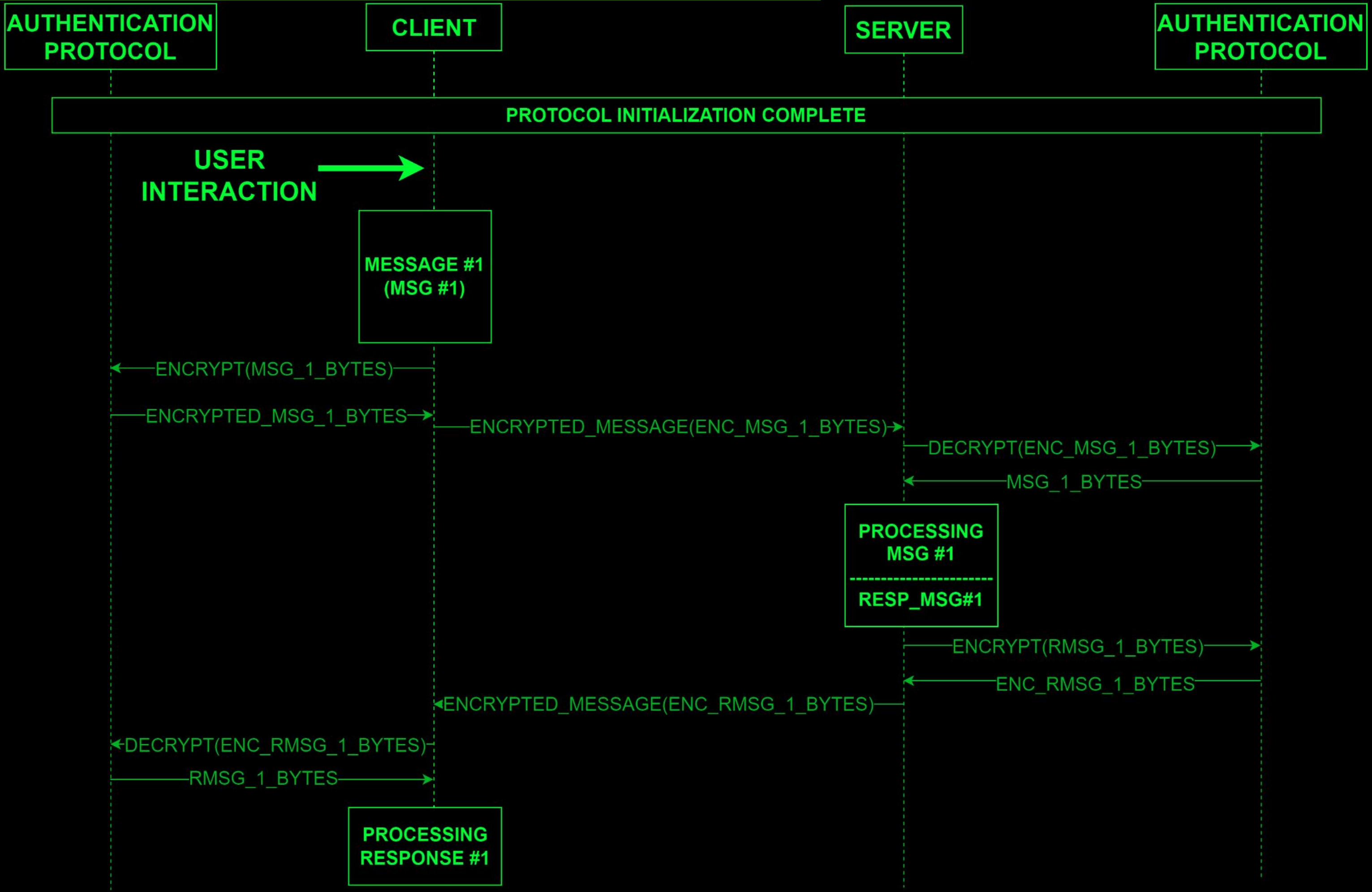
SHARED SECRET AVAILABLE (optionally with IV)

SHARED SECRET AVAILABLE (optionally with IV)

PROTOCOL INITIALIZATION COMPLETE

Authentication protocols - Encryption / Decryption

SEC Consult



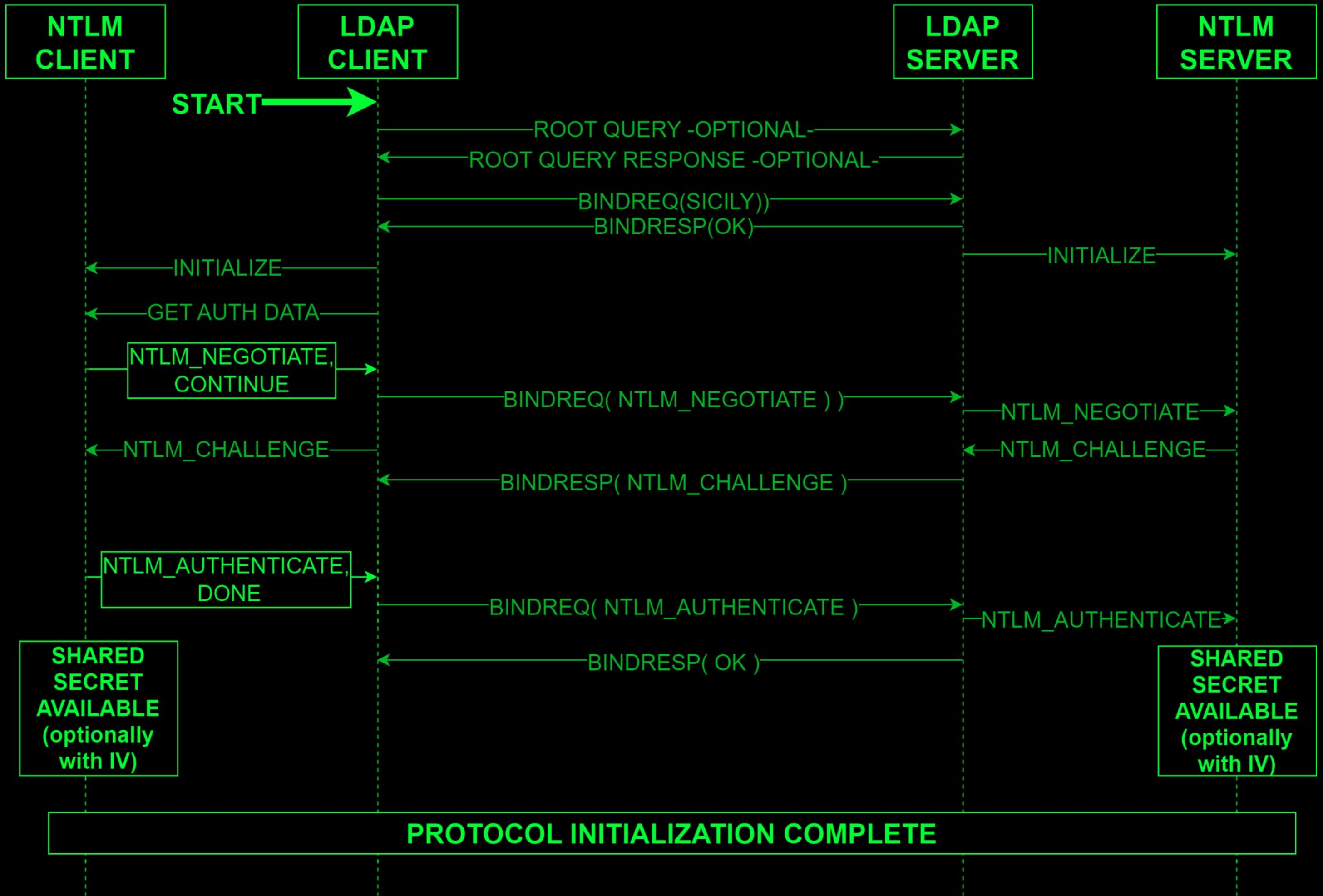
A word from the author

In the following part we are going to use LDAP as baseline to discuss authentication mechanisms relevant for this presentation

LDAP is widely used in Windows domains, and supports all major authentication types we need to touch

It also perfectly represents all evolutionary dead ends

LDAP - SICILY Authentication (NTLM)



NTLM CLIENT

LDAP CLIENT

LDAP SERVER

NTLM SERVER

START

ROOT QUERY -OPTIONAL-

ROOT QUERY RESPONSE -OPTIONAL-

BINDREQ(SICILY))

BINDRESP(OK)

INITIALIZE

INITIALIZE

GET AUTH DATA

NTLM_NEGOTIATE, CONTINUE

BINDREQ(NTLM_NEGOTIATE))

NTLM_NEGOTIATE

NTLM_CHALLENGE

NTLM_CHALLENGE

BINDRESP(NTLM_CHALLENGE)

NTLM_AUTHENTICATE, DONE

BINDREQ(NTLM_AUTHENTICATE)

NTLM_AUTHENTICATE

BINDRESP(OK)

SHARED SECRET AVAILABLE (optionally with IV)

SHARED SECRET AVAILABLE (optionally with IV)

PROTOCOL INITIALIZATION COMPLETE

SPNEGO - The authentication negotiator

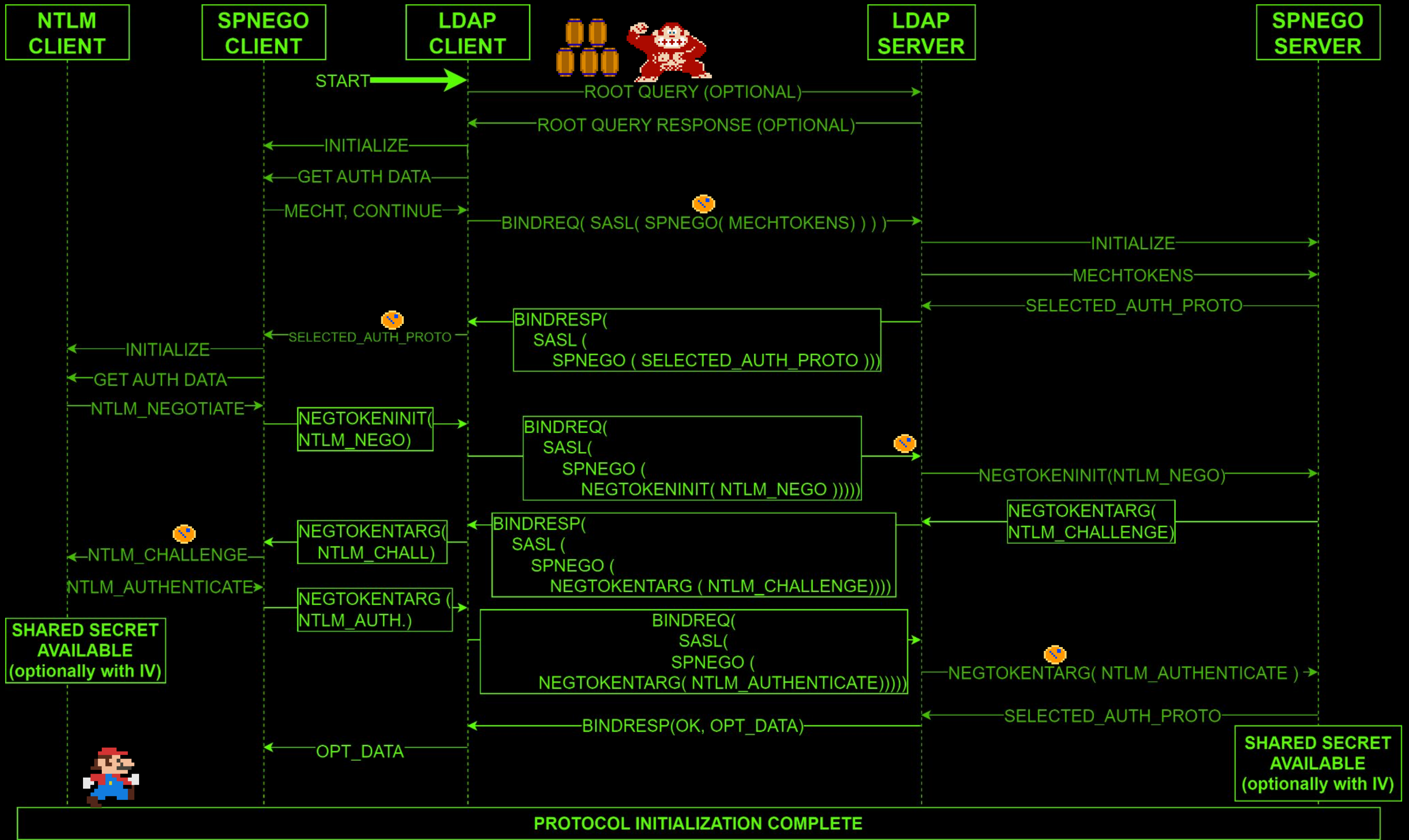
- SPNEGO stands for Simple and Protected GSSAPI Negotiation Mechanism
- Defined in RFC 2478
- Provides a way to negotiate the type of authentication used between a client and server
- This is usually (but not limited to) Kerberos or NTLM
- SPNEGO itself doesn't provide authentication; it simply selects the mechanism for the authentication
- SPNEGO is beneficial in environments where multiple authentication mechanisms are in use, as it allows for the selection of a common mechanism between client and server

SPNEGO - In case of LDAP

- LDAP supports the following authentication solutions
  - SIMPLE:
    - NONE: No authentication (anonymous BIND)
    - PLAIN: User + password sent over the wire
  - SICILY: I don’t even… This is raw NTLM
  - SASL:
    - EXTERNAL: Uses client TLS/SSL certificate as authentication
    - SPNEGO: What we’re discussing now
    - GSSAPI: This again allows different authentication mechanisms to be used
    - DIGEST-MD5: Because it’s 1999
    - … (many others)

LDAP - SASL - SPNEGO - NTLM Authentication

SEC Consult



NTLM CLIENT

SPNEGO CLIENT

LDAP CLIENT

LDAP SERVER

SPNEGO SERVER

START

ROOT QUERY (OPTIONAL)

ROOT QUERY RESPONSE (OPTIONAL)

INITIALIZE

GET AUTH DATA

MECHT, CONTINUE

BINDREQ(SASL(SPNEGO(MECHTOKENS))))

INITIALIZE

MECHTOKENS

SELECTED_AUTH_PROTO

BINDRESP(
SASL (
SPNEGO (SELECTED_AUTH_PROTO)))

SELECTED_AUTH_PROTO

INITIALIZE

GET AUTH DATA

NTLM_NEGOTIATE

NEGTOKENINIT(
NTLM_NEGO)

BINDREQ(
SASL(
SPNEGO (
NEGTOKENINIT(NTLM_NEGO)))))

NEGTOKENINIT(NTLM_NEGO)

NEGTOKENTARG(
NTLM_CHALLENGE)

BINDRESP(
SASL (
SPNEGO (
NEGTOKENTARG (NTLM_CHALLENGE))))

NEGTOKENTARG(
NTLM_CHALL)

NTLM_CHALLENGE

NTLM_AUTHENTICATE

NEGTOKENTARG (
NTLM_AUTH.)

BINDREQ(
SASL(
SPNEGO (
NEGTOKENTARG(NTLM_AUTHENTICATE)))))

NEGTOKENTARG(NTLM_AUTHENTICATE)

SHARED SECRET AVAILABLE (optionally with IV)

SELECTED_AUTH_PROTO

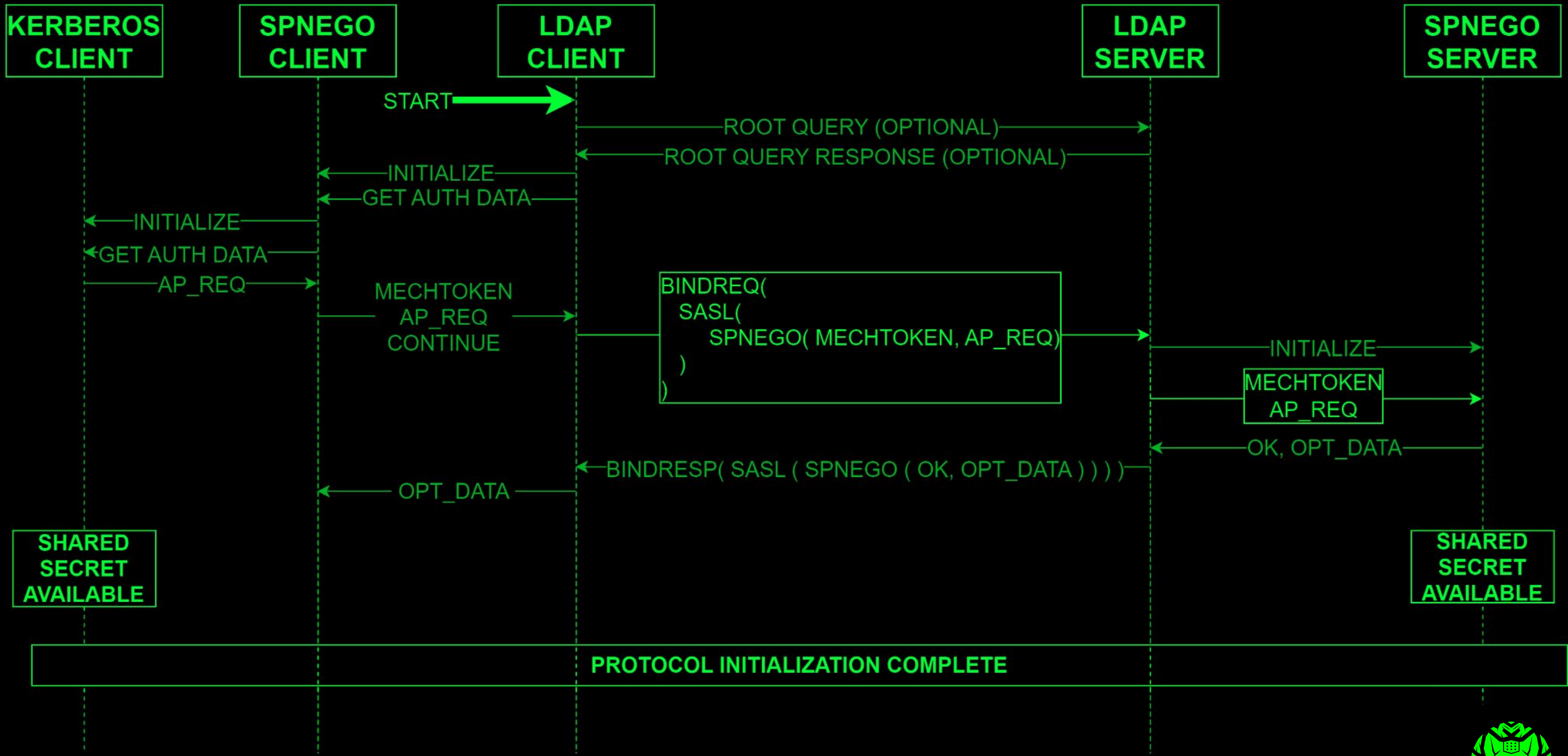
BINDRESP(OK, OPT_DATA)

OPT_DATA

SHARED SECRET AVAILABLE (optionally with IV)

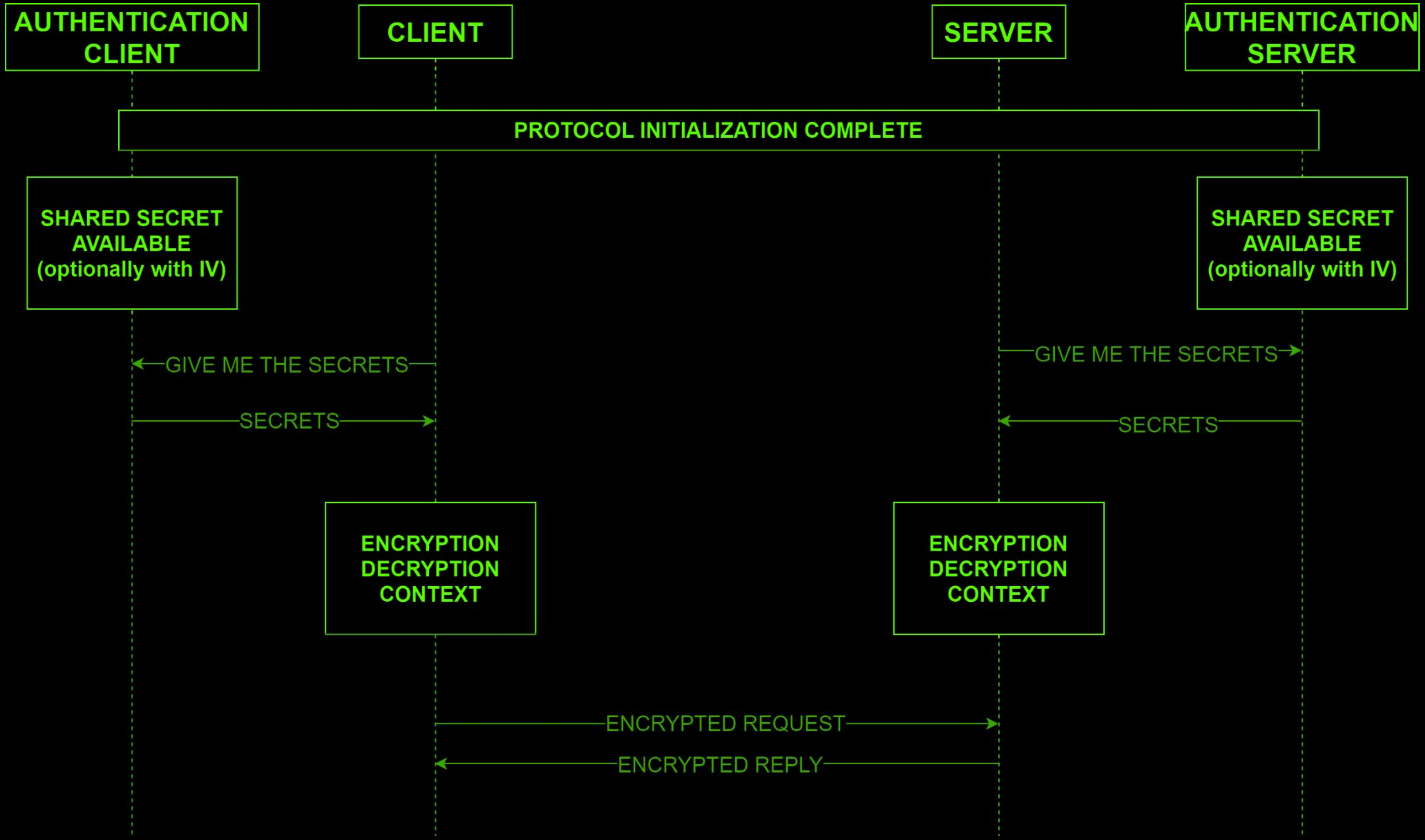
PROTOCOL INITIALIZATION COMPLETE

LDAP - SASL - SPNEGO - Kerberos Authentication



SMB ENCRYPTION/DECRYPTION

SEC Consult



AUTHENTICATION CLIENT

CLIENT

SERVER

AUTHENTICATION SERVER

PROTOCOL INITIALIZATION COMPLETE

SHARED SECRET AVAILABLE (optionally with IV)

SHARED SECRET AVAILABLE (optionally with IV)

ENCRYPTION DECRYPTION CONTEXT

ENCRYPTION DECRYPTION CONTEXT

Security Support Provider Interface (SSPI)

SSPI - Overview

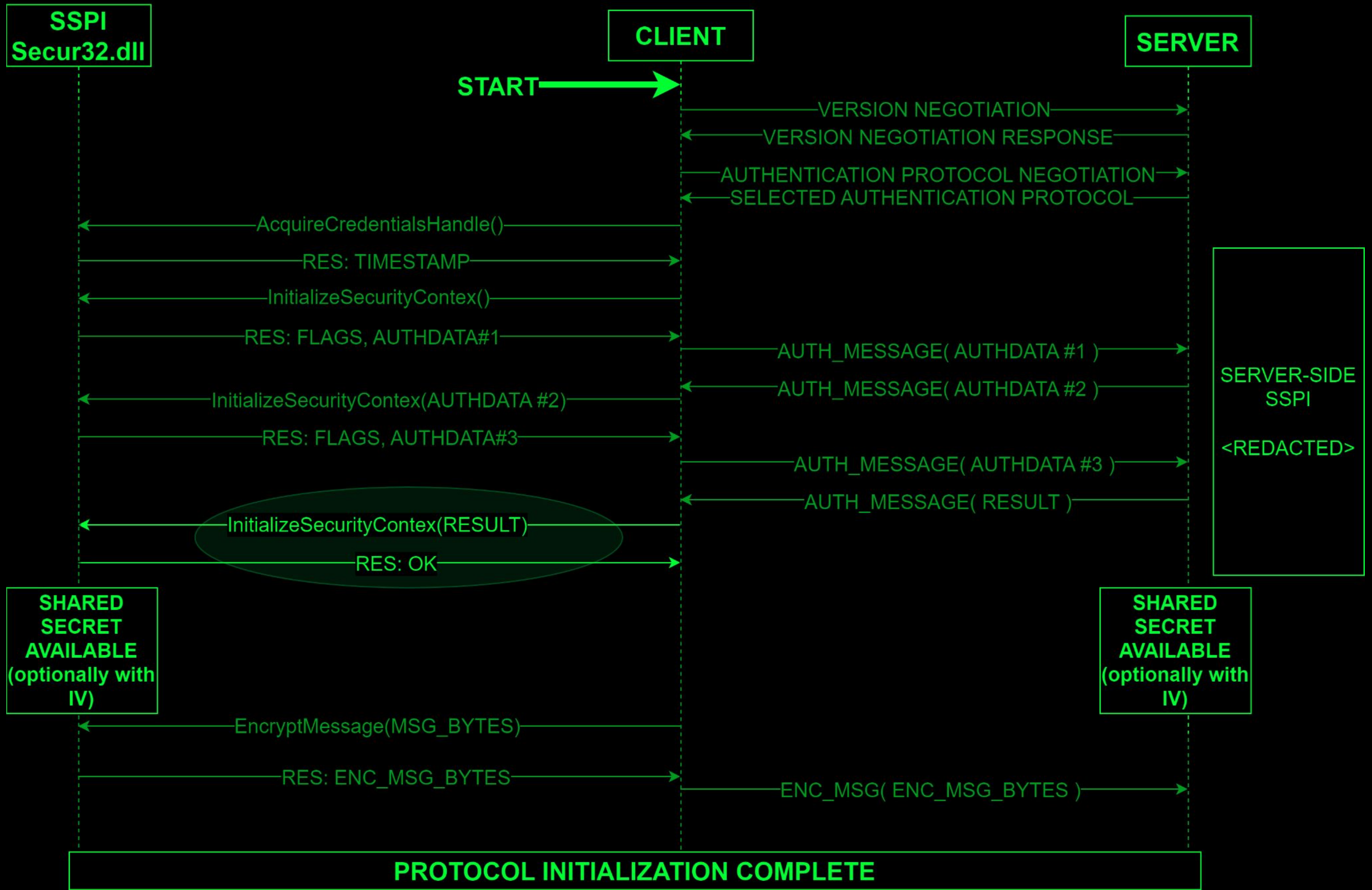
- Security Support Provider Interface
- API by Microsoft used in Windows systems
- SSPI allows applications to use various security models available on a computer or network without changing the interface to the security system
- The key function of SSPI is to provide a framework that abstracts the specifics of individual security models
- It supports various security models like Kerberos, NTLM, Schannel (for SSL/TLS) and others
- SSPI provides a mechanism to connect, sign on or off, encrypt or decrypt messages, and validate authority
- It works with security support providers (SSPs), which are dynamic-link libraries (DLLs), that make one or more security packages available to applications

SSPI - Methods

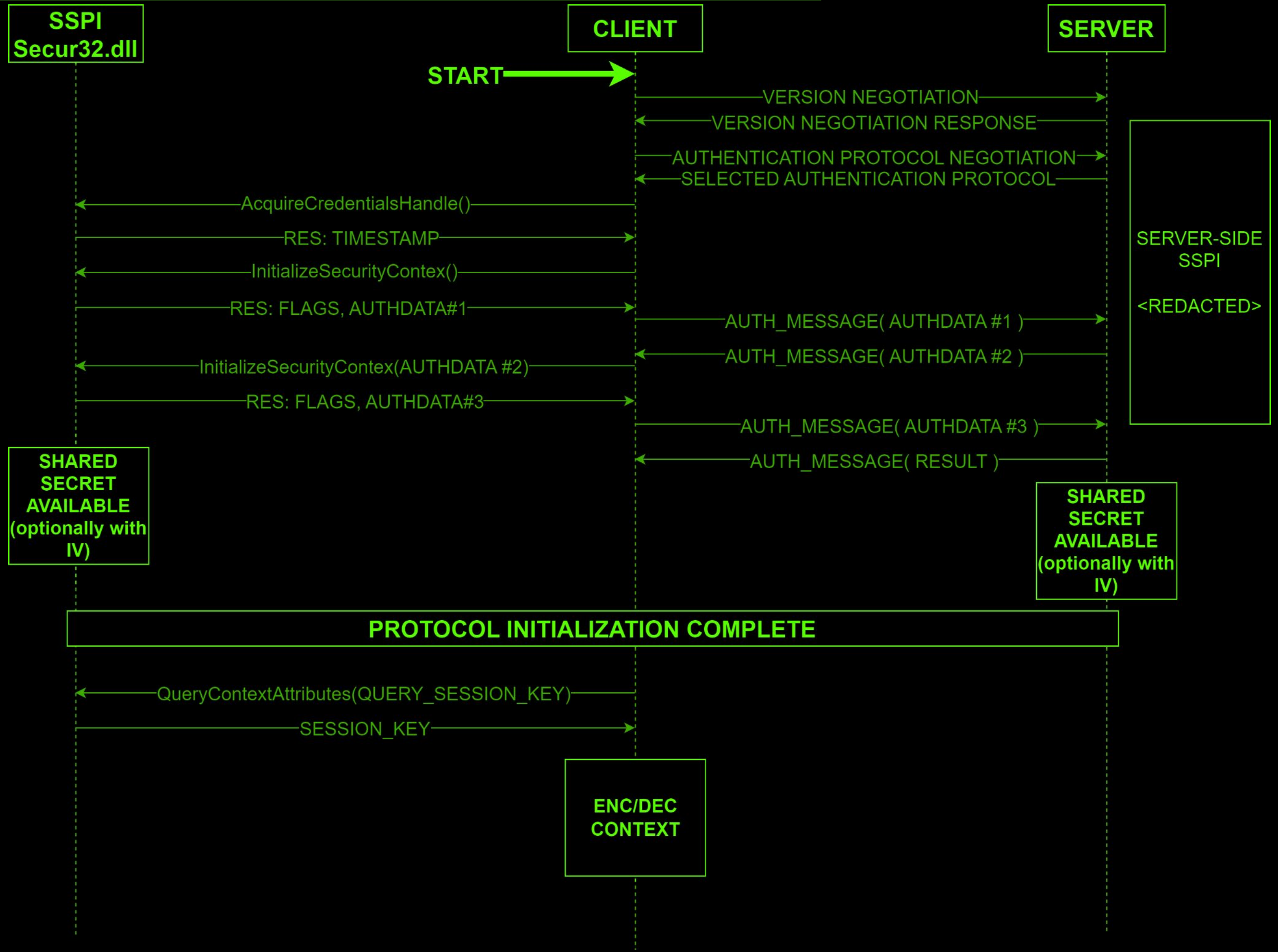
For a basic client authentication via SSPI, the following methods can be used:

- AcquireCredentialsHandle
- InitializeSecurityContext
- QueryContextAttributes
- EncryptMessage
- DecryptMessage

SSPI - Generalization


SSPI
Secur32.dll
CLIENT
SERVER
START
VERSION NEGOTIATION
VERSION NEGOTIATION RESPONSE
AUTHENTICATION PROTOCOL NEGOTIATION
SELECTED AUTHENTICATION PROTOCOL
AcquireCredentialsHandle()
RES: TIMESTAMP
InitializeSecurityContex()
RES: FLAGS, AUTHDATA#1
AUTH_MESSAGE(AUTHDATA #1)
AUTH_MESSAGE(AUTHDATA #2)
InitializeSecurityContex(AUTHDATA #2)
RES: FLAGS, AUTHDATA#3
AUTH_MESSAGE(AUTHDATA #3)
AUTH_MESSAGE(RESULT)
InitializeSecurityContex(RESULT)
RES: OK
SERVER-SIDE SSPI
<REDACTED>
SHARED SECRET AVAILABLE (optionally with IV)
SHARED SECRET AVAILABLE (optionally with IV)
EncryptMessage(MSG_BYTES)
RES: ENC_MSG_BYTES
ENC_MSG(ENC_MSG_BYTES)
PROTOCOL INITIALIZATION COMPLETE

SSPI - How to use it SMB



SSPI
Secur32.dll

CLIENT

SERVER

START

VERSION NEGOTIATION

VERSION NEGOTIATION RESPONSE

AUTHENTICATION PROTOCOL NEGOTIATION

SELECTED AUTHENTICATION PROTOCOL

AcquireCredentialsHandle()

RES: TIMESTAMP

InitializeSecurityContex()

RES: FLAGS, AUTHDATA#1

AUTH_MESSAGE(AUTHDATA #1)

AUTH_MESSAGE(AUTHDATA #2)

InitializeSecurityContex(AUTHDATA #2)

RES: FLAGS, AUTHDATA#3

AUTH_MESSAGE(AUTHDATA #3)

AUTH_MESSAGE(RESULT)

SERVER-SIDE
SSPI

<REDACTED>

SHARED SECRET AVAILABLE (optionally with IV)

SHARED SECRET AVAILABLE (optionally with IV)

PROTOCOL INITIALIZATION COMPLETE

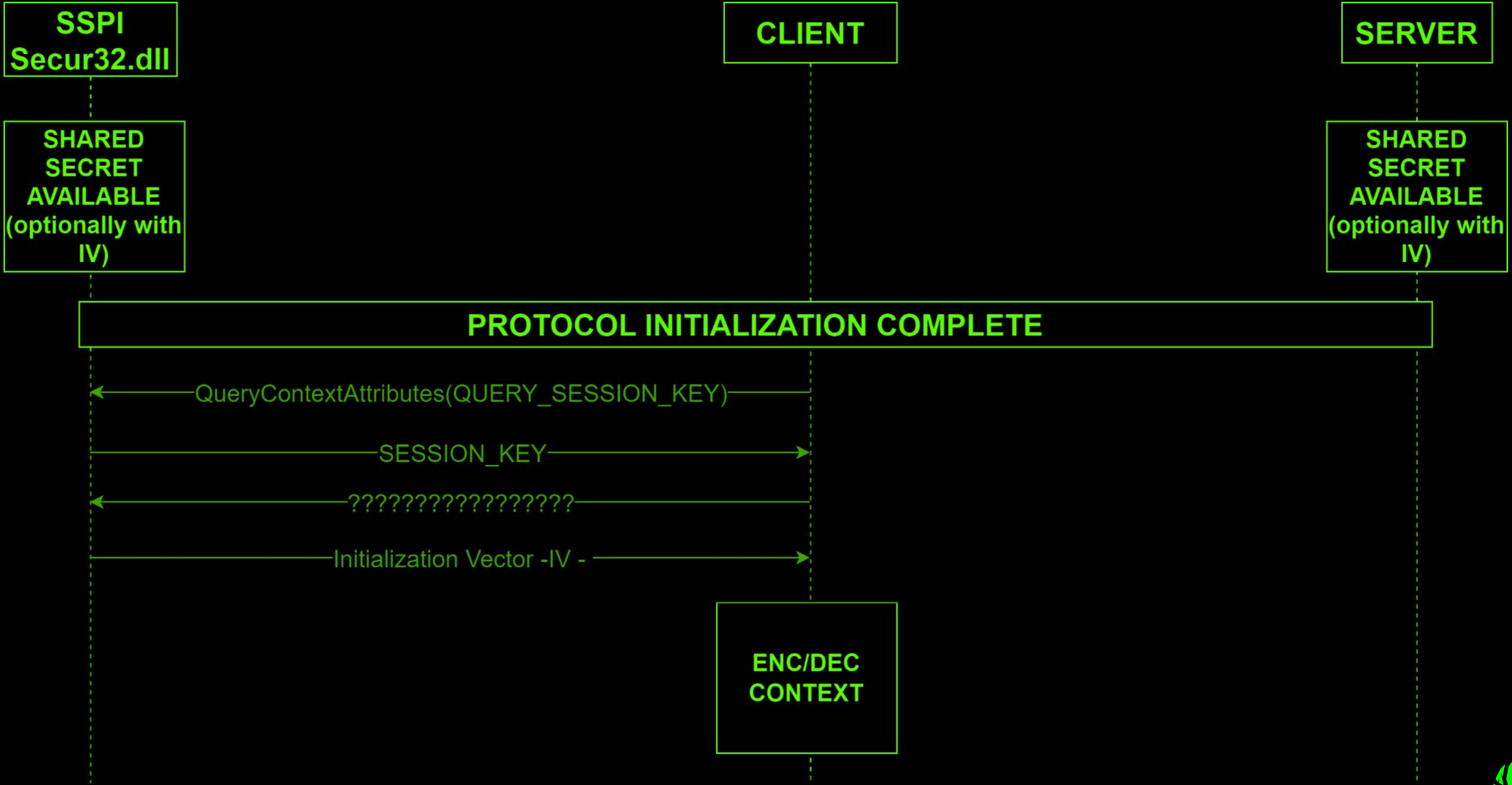
QueryContextAttributes(QUERY_SESSION_KEY)

SESSION_KEY

ENC/DEC
CONTEXT

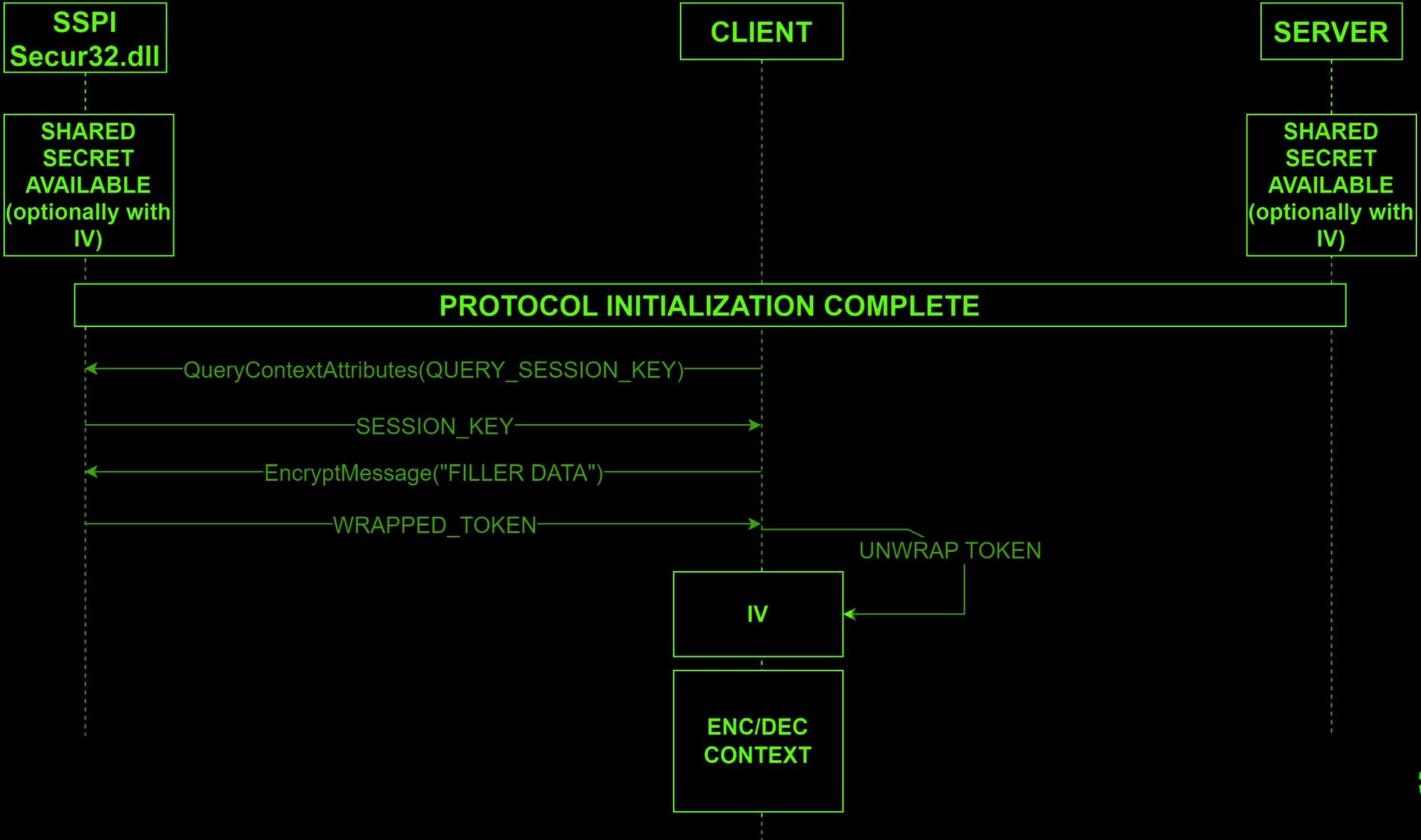
SSPI - But there is a problem with mutual authentication

SEC Consult


SSPI
Secur32.dll
CLIENT
SERVER
SHARED SECRET AVAILABLE (optionally with IV)
SHARED SECRET AVAILABLE (optionally with IV)
PROTOCOL INITIALIZATION COMPLETE
QueryContextAttributes(QUERY_SESSION_KEY)
SESSION_KEY
?????????????????
Initialization Vector -IV -
ENC/DEC CONTEXT

SSPI - Solution for mutual authentication

SEC Consult


SSPI
Secur32.dll
CLIENT
SERVER
SHARED SECRET AVAILABLE (optionally with IV)
SHARED SECRET AVAILABLE (optionally with IV)
PROTOCOL INITIALIZATION COMPLETE
QueryContextAttributes(QUERY_SESSION_KEY)
SESSION_KEY
EncryptMessage("FILLER DATA")
WRAPPED_TOKEN
UNWRAP TOKEN
IV
ENC/DEC CONTEXT

Your LSASS, but on **MY** machine

SSPIProxy - The goal

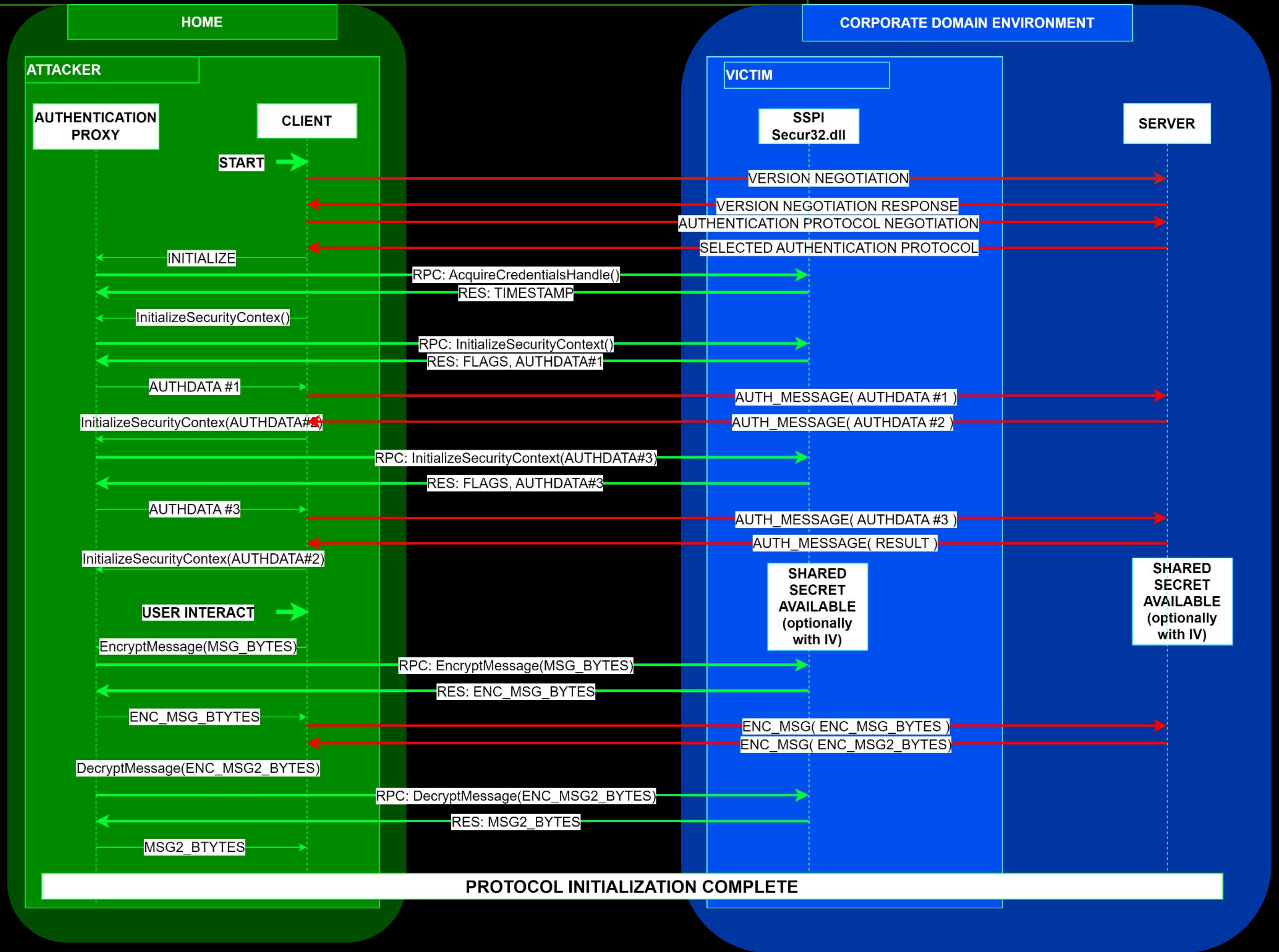
SEC Consult

Imagine a system where one would be able to use SSPI function calls remotely:

- Authentication on behalf of current user context
- SSPI does not require admin privilege to perform authentication
- No need to touch LSASS directly (e.g. no dumping)
- The proxy process doesn’t do anything suspicious
- After authentication finishes, the authentication context on the target machine is not needed to continue the communication on higher-layers

If implemented correctly, the authentication proxy would allow authentication from a 3rd party machine (e.g. teamserver) to a server on the internal network of the target environment, **WITHOUT** additional code to be pushed on the victim machine besides the initial agent

SSPIProxy - Generic logic



HOME

ATTACKER

AUTHENTICATION PROXY

CLIENT

CORPORATE DOMAIN ENVIRONMENT

VICTIM

SSPI Secur32.dll

SERVER

START

VERSION NEGOTIATION

VERSION NEGOTIATION RESPONSE

AUTHENTICATION PROTOCOL NEGOTIATION

SELECTED AUTHENTICATION PROTOCOL

INITIALIZE

RPC: AcquireCredentialsHandle()

RES: TIMESTAMP

InitializeSecurityContex()

RPC: InitializeSecurityContext()

RES: FLAGS, AUTHDATA#1

AUTHDATA #1

AUTH_MESSAGE(AUTHDATA #1)

InitializeSecurityContex(AUTHDATA#2)

AUTH_MESSAGE(AUTHDATA #2)

RPC: InitializeSecurityContext(AUTHDATA#3)

RES: FLAGS, AUTHDATA#3

AUTHDATA #3

AUTH_MESSAGE(AUTHDATA #3)

AUTH_MESSAGE(RESULT)

InitializeSecurityContex(AUTHDATA#2)

SHARED SECRET AVAILABLE (optionally with IV)

SHARED SECRET AVAILABLE (optionally with IV)

USER INTERACT

EncryptMessage(MSG_BYTES)

RPC: EncryptMessage(MSG_BYTES)

RES: ENC_MSG_BYTES

ENC_MSG_BTYTES

ENC_MSG(ENC_MSG_BYTES)

ENC_MSG(ENC_MSG2_BYTES)

DecryptMessage(ENC_MSG2_BYTES)

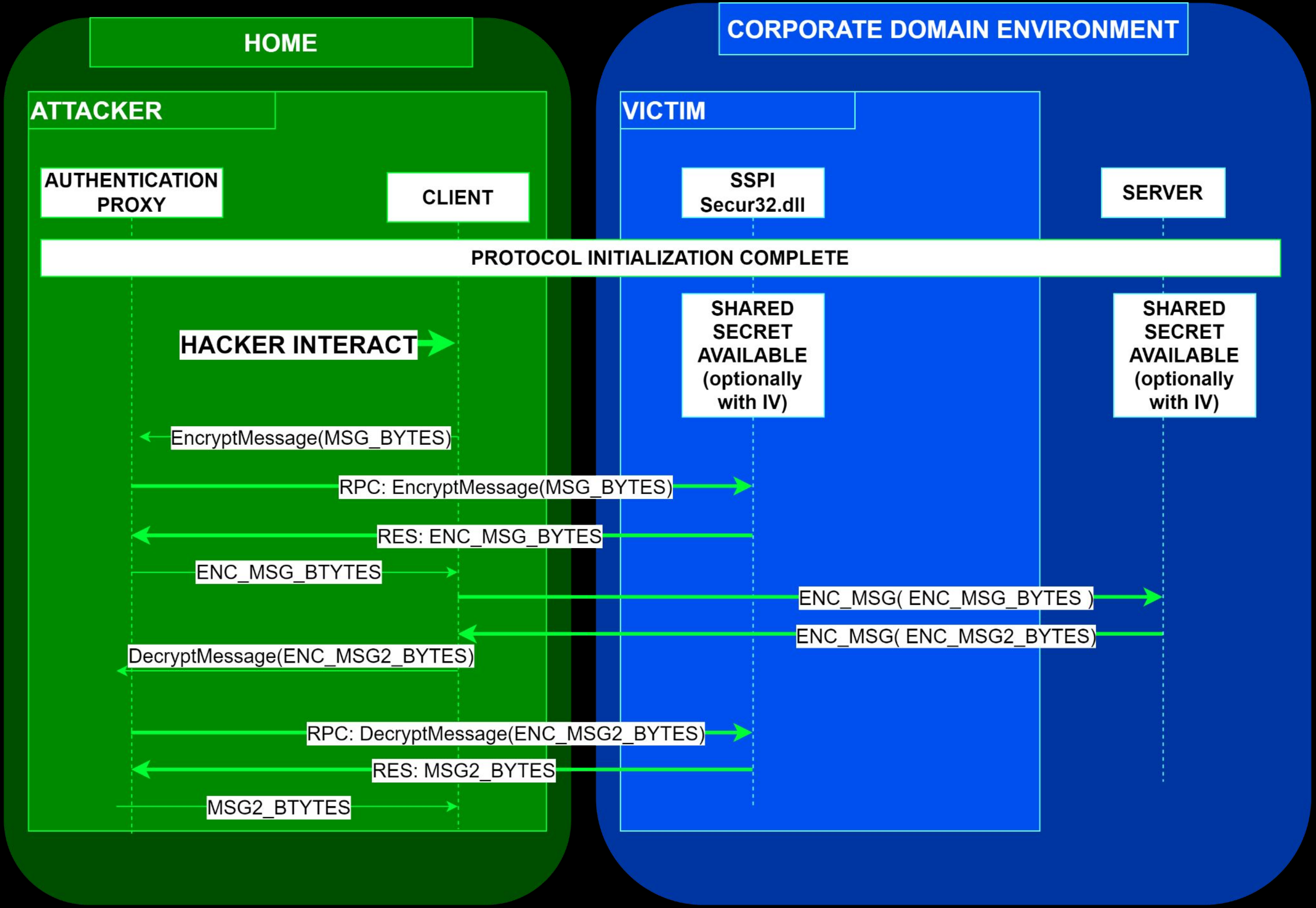
RPC: DecryptMessage(ENC_MSG2_BYTES)

RES: MSG2_BYTES

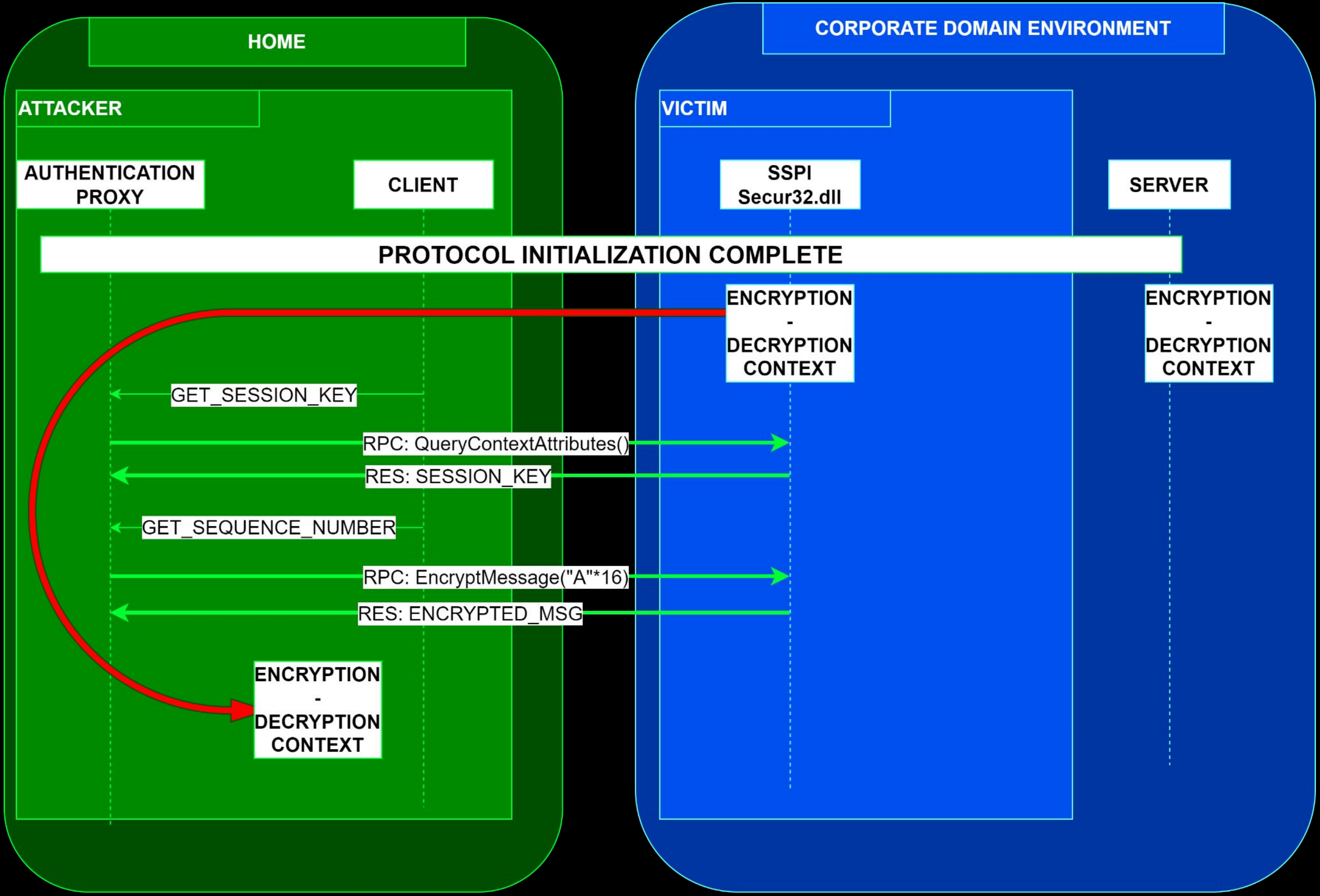
MSG2_BTYTES

PROTOCOL INITIALIZATION COMPLETE

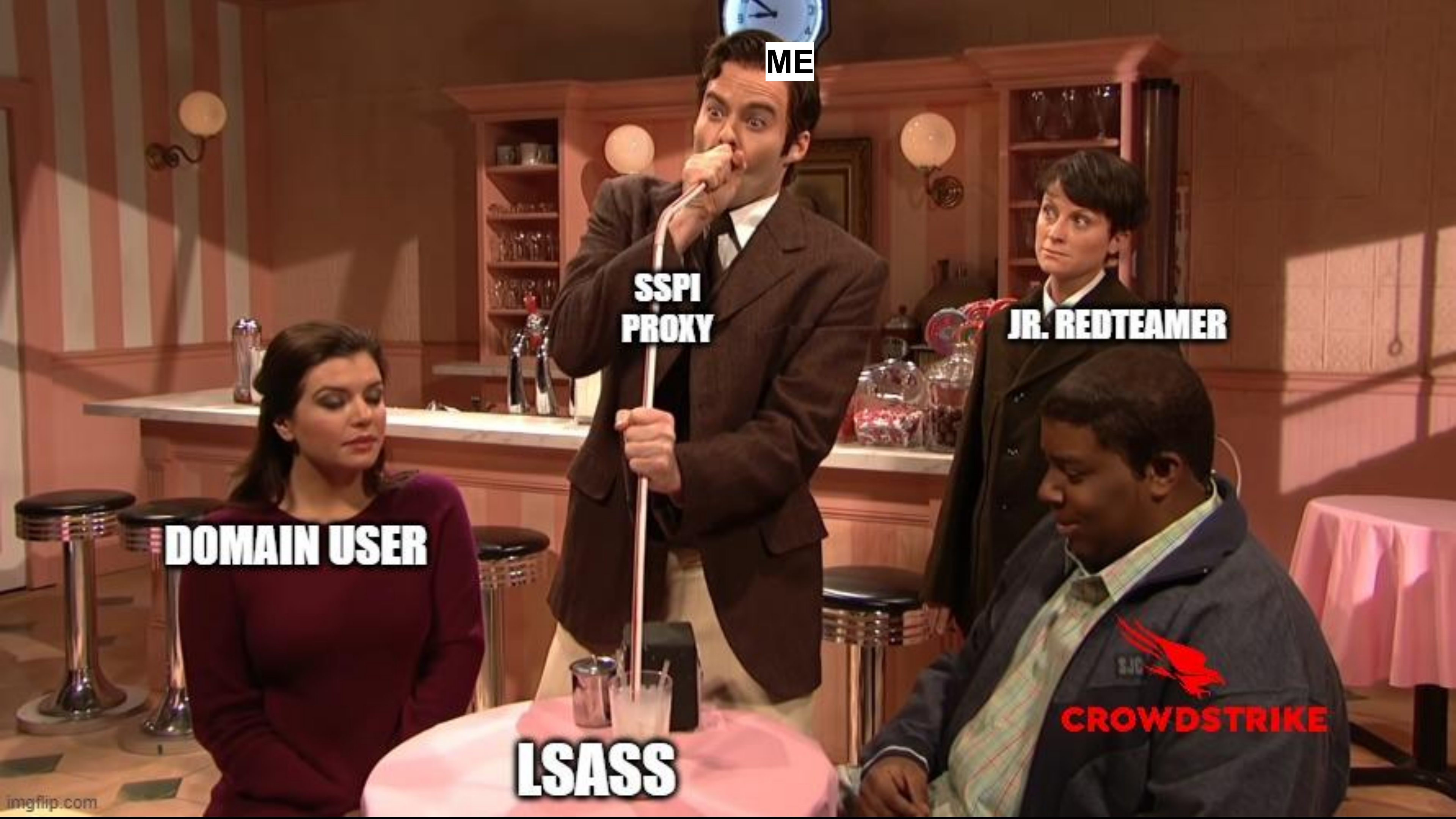
SSPIProxy - Encryption/Decryption the wrong way


HOME
CORPORATE DOMAIN ENVIRONMENT
ATTACKER
VICTIM
AUTHENTICATION PROXY
CLIENT
SSPI Secur32.dll
SERVER
PROTOCOL INITIALIZATION COMPLETE
SHARED SECRET AVAILABLE (optionally with IV)
SHARED SECRET AVAILABLE (optionally with IV)
HACKER INTERACT
EncryptMessage(MSG_BYTES)
RPC: EncryptMessage(MSG_BYTES)
RES: ENC_MSG_BYTES
ENC_MSG_BTYTES
ENC_MSG(ENC_MSG_BYTES)
ENC_MSG(ENC_MSG2_BYTES)
DecryptMessage(ENC_MSG2_BYTES)
RPC: DecryptMessage(ENC_MSG2_BYTES)
RES: MSG2_BYTES
MSG2_BTYTES

SSPIProxy - Encryption/Decryption the right way



HOME
ATTACKER
AUTHENTICATION PROXY
CLIENT
CORPORATE DOMAIN ENVIRONMENT
VICTIM
SSPI Secur32.dll
SERVER
PROTOCOL INITIALIZATION COMPLETE
ENCRYPTION - DECRYPTION CONTEXT
ENCRYPTION - DECRYPTION CONTEXT
GET_SESSION_KEY
RPC: QueryContextAttributes()
RES: SESSION_KEY
GET_SEQUENCE_NUMBER
RPC: EncryptMessage("A"*16)
RES: ENCRYPTED_MSG
ENCRYPTION - DECRYPTION CONTEXT


ME
SSPI PROXY
JR. REDTEAMER
DOMAIN USER
LSASS
CROWDSTRIKE
imgflip.com

DEMO

SSPIProxy - Current status

Requires an agent + libraries to use custom auth protocol implementation on server side.

SSPIProxy agent demo is implemented in a Python library called “WSNET[1]”. It is published under MIT license on Github.

Authentication protocol library called “asyauth[2]” is published under MIT license on Github.

All my major protocol implementation libraries use “asyauth” under the hood, thus they already have a way to support and use this technique

1 - https://github.com/skelsec/wsnet

2 - https://github.com/skelsec/asyauth

SSPIProxy - Comparison

SEC Consult

Benefits of this solution:

- Can authenticate to network services without any exploits on the initial foothold
- No need to push additional code on the initial foothold
- Work from home
- AV/EDR detection is minimal
- The main complexity is in the code of the attacker machine -> Agent can be (re)implemented quite easily in other ways to avoid detection
- Drive the SoC insane - with multiple compromised hosts, you can trigger authentication from different workstations, but use a network connection from a workstation where the user can’t even log in

Let’s discuss some drawbacks:

- This technique (as of now) requires custom protocol implementations that allow interfacing with the authentication proxy. (Unless you are @_EthicalChaos_)
- Compared to traditional C2 solutions, this solution will generate much more network traffic, as it needs to invoke the RPC calls for the authentication proxy for each connection created to a target server

SSPIProxy - FUTURE

The SSPIProxy technique will keep on living in all my projects, I have big plans for it. See you at DEF CON 31 for more :)

- ALSO -

There is this person on Twitter by the handle of @_EthicalChaos_

- A few years ago, in unrelated research, he published a project called “lsarelayx[3]”
- It hooks LSASS itself to override/redirect NTLM authentication mechanism for ALL local windows applications
- This could be extended to support Kerberos/SPNEGO.
- It would allow any windows application (on the attackers machine) to leverage SSPIProxy functionality

[3] - https://github.com/CCob/lsarelayxet

SSPIProxy - FUTURE

There is at least one more authentication method which could be implemented using SSPIProxy logic:

- Kerberos - PKINIT

By implementing 3 more RPC calls, we can proxy certificate based kerberos authentication using the target user’s certificate store

This is on my ROADmap to make @_dirkjan happy

Thank you

SEC Consult

Shoutout to the following people who helped me - in no specific order -

@_EthicalChaos_ - LSARELAYX

@BoreanJordan - smbprotocol / python-gssapi

@awakecoding - sspi-rs

@agsolino - Impacket

Links to the projects discussed in this presentation:

- https://github.com/skelsec/aiosmb
- https://github.com/skelsec/msldap
- https://github.com/skelsec/asyauth
- https://github.com/skelsec/wsnet

Q & A

THANK YOU

Please sponsor us on Porchetta Industries!

https://porchetta.industries