

Defending the

Hypervisor

Using Offensive Tooling to Validate vSphere Security

Darryl G. Baker

Blue Team Con 2026

TOOL RELEASE

VEXED v2.0

vSphere EXploitation &
Extraction Device

WHOAMI?

Active Directory Security Guy... and I break Windows

Darryl G. Baker
@DFIRDEFERRED

Security Researcher, Netwrix

Ham Extra

Creator of AD Hacking Village

Tinkerer of Toys

Builder of Things



SCOPE & TEST ENVIRONMENT

Every finding proven in a controlled lab

System	Version / Build	Role
ESXi 7	6.5.0 build-4564106	Standalone hypervisor
ESXi 8	8.0.3 build-24677879	Standalone hypervisor
vCenter 7	7.0.3.00300 (Photon OS 3)	VCSA appliance
vCenter 8	8.0.3.00400 (Photon OS 4)	VCSA appliance
Domain Controller	Windows Server 2016 DC	WYVERN.LOCAL

Domain: WYVERN.LOCAL (2016 functional level)

Two-way transitive trust with DRAGON.CORP

Enterprise Root CA (wyvern-CA, 5-yr validity)

[+] Confirmed by packet capture, gcore heap dumps, klist, and functional attack chains

// AGENDA

01

ESXi AD Admin Group

CVE-2024-37085, nested groups, hidden root access

02

Built-in Accounts

Every account is root. Meet vpxuser.

03

Credential Exposure in ESXi

Kerberos proxy, keytabs, ccache, strace

04

vCenter Vulnerabilities

Plaintext LDAP, private keys, JVM heap dumps

05

Best Practices

IWA, LDAPS, Kerberos hardening

06

VEXED Demo

VEXED in action

01

THE ESXi ADMIN GROUP

The AD group that controls root access to your hypervisors

CVE-2024-37085

ESXi Active Directory Integration Authentication Bypass

How It Works

Two ESXi host settings control AD admin access:

```
Config.HostAgent.plugins.hostsvc.  
  esxAdminsGroup  
  Default: "ESX Admins"
```

```
Config.HostAgent.plugins.hostsvc.  
  esxAdminsGroupAutoAdd  
  Default: false
```

Members of the configured group get
FULL ROOT ACCESS to every ESXi host.

Exploited In The Wild

Storm-0506 (Black Basta)

Ransomware operators targeting VMware

Octo Tempest (Scattered Spider)

Social engineering + ESXi admin group

No ESXi-specific exploit needed.

Just AD group membership modification.

AD group mgmt = ESXi root access mgmt

INFINITE NESTING, INFINITE RISK

The group can be nested ANYWHERE

Tested: 20 levels deep = ROOT ACCESS

Tested: 20 nested groups = ROOT ACCESS

Tested: Different OUs = ROOT ACCESS

Tested: Different containers = ROOT ACCESS

No nesting limit.

No OU restriction.

No boundary whatsoever.

Who can grant themselves root?

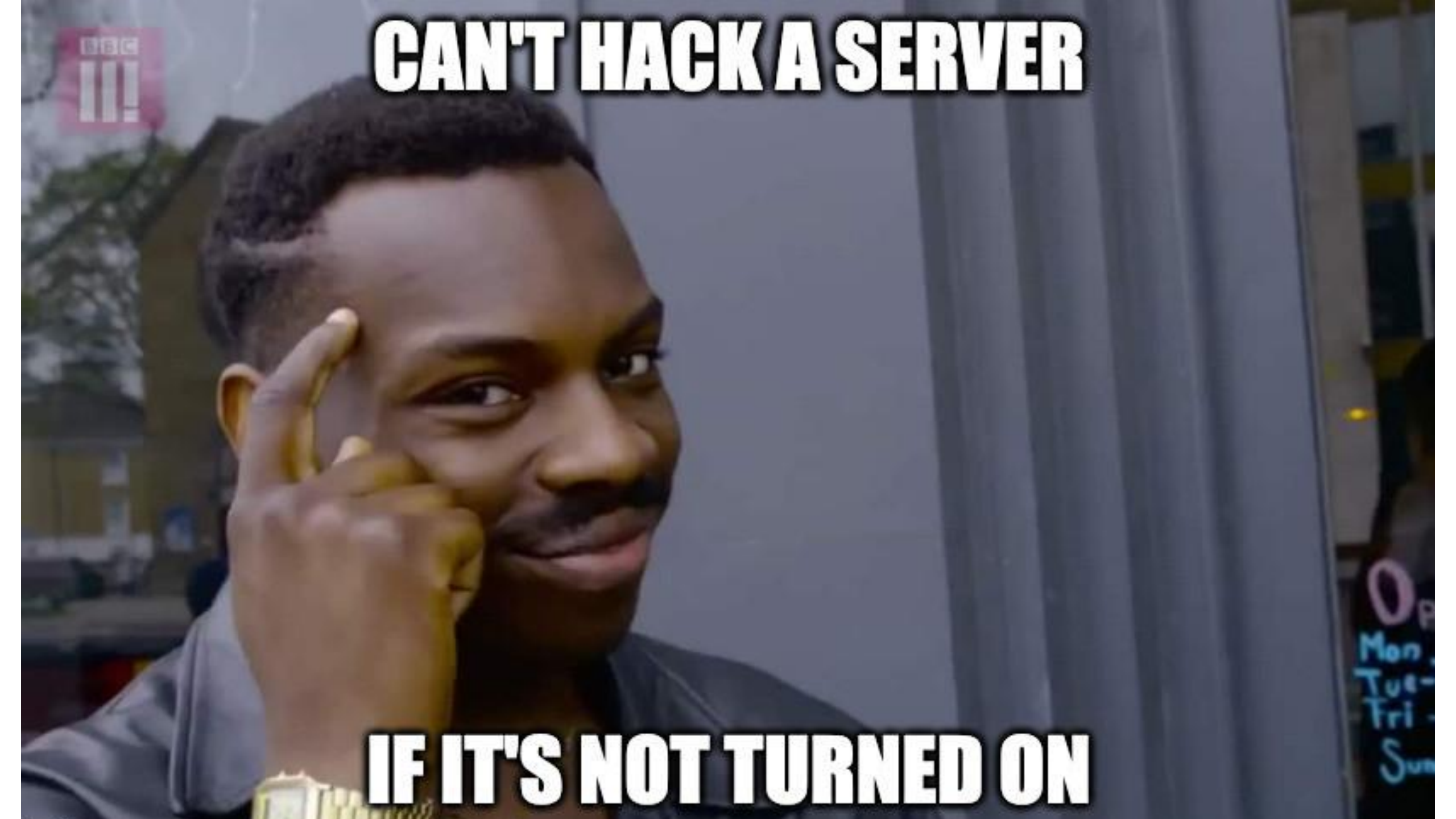
Delegated OU admins

Help Desk with group management rights

Any account with GenericWrite

on the group object

```
ESX Admins (default group)
|
+-- OU=ServerAdmins
|   |
|   +-- OU=Tier2Support
|       |
|       +-- OU=RegionalIT
|           |
|           +-- ... (20 levels deep)
|               |
|               +-- attacker
|                   = ROOT on ALL ESXi
|
+-- CN=RandomGroup,OU=Finance
|   +-- helpdesk_user
|       = ROOT on ALL ESXi
|
+-- CN=Nested,OU=Marketing,OU=East
|   +-- compromised_account
|       = ROOT on ALL ESXi
```



CAN'T HACK A SERVER

IF IT'S NOT TURNED ON

THE AUTOADD ESCALATION

If `esxAdminsGroupAutoAdd = true`:

Anyone with ESXi API access can change the group name to:

```
# Change admin group to "Domain Users"
esxcli system account set --id root \
    --esxAdminsGroup "Domain Users"

# Result: EVERY domain user = ESXi root
# Also works: "Authenticated Users"
```

[!] Treat ESX Admins group with SAME sensitivity as Domain Admins

02

BUILT-IN ACCOUNTS

Every account is root. There are no privilege levels.

ONE DOES NOT SIMPLY

SSH AS ROOT

THE ROOT EQUIVALENCE PROBLEM

All SSH sessions = UID 0 (root)

Regardless of which domain user logs in.
No intermediate privilege levels exist.

Role	Access Level	Practical Impact	Useful?
Admin	Full root access	All files, all VMs, all processes, VMDK access	Yes (too much)
ReadOnly	3 read privileges	Can see, can't touch	Barely
NoAccess	Zero privileges	Nothing	No
NoCryptoAdmin	Admin minus crypto	Non-functional on standalone ESXi	No

[!] There is no 'VM power user', no 'backup admin', no 'network manager' role on standalone ESXi

```
$ id => uid=0(root) gid=0(root)
```

MEET vpxuser

What is vpxuser?

The proxy account vCenter uses to manage ESXi hosts.

Key Facts:

Auto-created when ESXi is added to vCenter inventory

Password auto-rotated by vCenter (encrypted with symkey.dat AES-256-CBC)

Has FULL Admin privileges on ESXi

vCenter decrypts vpxuser passwords from the VPX PostgreSQL database

User identity does NOT propagate

ESXi sees only vpxuser, not the actual administrator

Attack Chain: VPX Decrypt

1. Root on vCenter

Read /etc/vmware-vpx/ssl/symkey.dat

2. Query VPX database

Encrypted vpxuser passwords for every managed ESXi host

3. AES-256-CBC decrypt

symkey.dat + IV from DB = cleartext

4. SSH to every ESXi host

ssh vpxuser@<esxi_ip>
Full root access

CVE-2022-22948: symkey.dat was world-readable. Fixed, but root still owns it.

ESXi ADMIN = DOMAIN ADMIN

When a Domain Controller VM runs on the host

5 Confirmed Attack Chains

1. Offline NTDS extraction (CRITICAL)
Browse datastore -> download DC VMDK
Mount offline -> secretsdump ntds.dit
= every domain hash. No exploit.
2. AD state rollback (revert past IR:
krbtgt reset + cert revocation)
3. Domain DoS (power off DC VMs)
4. MITM (inject NIC into DC VM)
5. Credential theft from the ESXi host

Why It Works

The legitimate Admin role is enough:

- Full datastore browse + copy
- Snapshot / revert / power
- Direct VMDK manipulation

No CVE. No memory corruption.

The Admin role IS the exploit.

Fix: never host DC VMs without vCenter RBAC. Exclude snapshot/power/datastore on DC folders. JIT+MFA.

[!] ESXi Admin on a host running a DC VM = full domain compromise

03

CREDENTIAL EXPOSURE ON ESXi

Following the password: network, disk, and memory

Primary Path

Port 22

1. SSH Password Auth

Password transmitted via SSH tunnel. ESXi constructs Kerberos AS-REQ pre-auth to DC. Protected in transit on wire, but creates high risk on host disk.

Exposure: Writes 10-hr TGT cache to `/tmp/krb5cc_*` readable by any session (all run as root).

API Access

Port 443

2. vSphere API (Basic Auth)

Uses HTTP Basic Auth header over TLS 1.2. Backend hostd service validates via Kerberos to DC. Does not offer SPNEGO/Negotiate.

Note: Does NOT persist TGT files to disk, reducing pass-the-ticket risk compared to SSH.

Asymmetric

Local Only

3. SSH Public Key Auth

Works **only for local accounts** (root/esxid) via non-standard key file paths. Domain user SSH keys fail because Likewise/PAM forces password auth.

Persistence Risk: Backdoor local accounts with public keys survive AD password rotations.

Cert Auth

Kerberos

4. PKINIT (kinit)

Allows passwordless TGT acquisition via user certificates. Functional on ESXi 7/8 only after editing `krb5.conf` to remove smartcard enforcement rules.

Status: Works with custom config; defaults enforce physical hardware smart cards.

ESXi: THE KERBEROS PROXY

It's not full Kerberos. ESXi proxies your password.

AUTHENTICATION FLOW

User	ESXi Host	Domain Controller
------	-----------	-------------------

```
| --- SSH/TLS ----->
```

```
(password in
encrypted
tunnel)
```

```
| --- Kerberos AS-REQ --> |
```

```
(pre-auth data, |
encrypted timestamp)
```

```
|<-- Kerberos AS-REP ---|
```

(TGT returned)

```
|<-- Session -----
```

```
(root shell)
```

Good: Password never crosses network in cleartext

Bad: Every auth = password required. No password-free path exists.

THE LOCKOUT BYPASS

Unlimited brute force against domain accounts

The PAM Exemption

ESXi's system-auth-tally PAM stack skips pam_tally2 for any UID > 10000.

Every domain user has UID > 10000.

**Result: domain accounts have NO
lockout via ESXi SSH.**

VMware's intent: dodge a bug where large UIDs grew the tally file to 'tens of GB.' The workaround disabled lockout for every domain account.

Attack Impact

Spray / brute force, no lockout:

```
hydra -l superman -P rockyou.txt \\ssh://10.0.5.21
```

- No local lockout on ESXi
- DC lockout can be masked: ESXi proxies the AS-REQ, so the source looks like ESXi, not the attacker

**Detection: DC 4771/4776 volume
sourced from ESXi host IPs**

KERBEROS KEYTABS

What is a keytab?

A file containing long-term Kerberos keys for service principal names (SPNs).

Equivalent to the machine account password in cryptographic form.

Location: /etc/krb5.keytab

Permissions: 644 (WORLD-READABLE!)

Should be: 600 (root only)

Attack Impact

Forge silver tickets for any SPN registered to the ESXi host

Impersonate any user to the service

66 keytab entries found in testing

(AES-256, AES-128, RC4-HMAC keys)

```
# Extract keytab from ESXi
$ scp root@esxi:/etc/krb5.keytab ./stolen.keytab

# List all principals and encryption types
$ klist -kte stolen.keytab
```

KVNO	Principal	Enctype
3	host/esxi.lan@WYVERN.LOCAL	aes256-cts-hmac-sha1-96
3	host/esxi.lan@WYVERN.LOCAL	arcfour-hmac (RC4!)
3	ESXI\$@WYVERN.LOCAL	aes256-cts-hmac-sha1-96

KERBEROS CREDENTIAL CACHE (.ccache)

```
# TGTs persist on disk after SSH logout. No cleanup.
$ ls -la /tmp/krb5cc_*
-rw----- root  root  2057  /tmp/krb5cc_0
-rw----- root  root  3844  /tmp/krb5cc_257950799  # superman
-rw----- root  root  4135  /tmp/krb5cc_257952828  # miles.morales

# Every SSH session = UID 0 = can read ALL caches
$ klist /tmp/krb5cc_257950799
Default principal: superman@WYVERN.LOCAL
  krbtgt/WYVERN.LOCAL@WYVERN.LOCAL (valid 10 hours!)
  host/esxi.lan@WYVERN.LOCAL (RC4-HMAC!)
```

Linux: Direct Pass-the-Ticket

```
$ export KRB5CCNAME=/tmp/stolen_tgt
$ klist # Confirms: superman TGT
$ kvno cifs/dc.wyvern.local
$ kvno ldap/dc.wyvern.local
$ smbclient //dc/SYSVOL -k
$ ldapsearch -Y GSSAPI ...
```

Windows: Rubeus Import

```
# Convert ccache to kirbi
$ ticketConverter.py tgt superman.kirbi

# Import into current session
Rubeus.exe ptt /ticket:superman.kirbi

# Now operating as superman everywhere
```

WHY TGT THEFT IS DEVASTATING

Factor	Impact
Victims are Domain Admins	Users who SSH to ESXi = highest-priv accounts
No session isolation	Any UID 0 session can read any user's TGT
Theft is invisible	ESXi does NOT log reads to /tmp/krb5cc_*
TGTs accumulate	Maintenance window = 5-10 DA TGTs simultaneously
TGTs persist after logout	No cleanup on SSH disconnect
Local users can steal too	Backdoor accounts (UID 0) = domain escalation
Machine cache = bonus	LDAP service tickets for AD recon as ESXi\$ account

[!] Attack window: 10 hours (TGT lifetime) + 12 hours (renewable). Zero audit trail.

PROCESS MEMORY: strace ON ESXi

ESXi runs hostd, lwsmd, and vpxa as target processes
Attaching strace captures live authentication traffic
including Kerberos exchanges and credential material

```
# Attach to hostd process to capture live auth
$ strace -f -e trace=read,write -p $(pgrep hostd) \
  -s 4096 -o /tmp/hostd_capture.log &

# Wait for authentication events...
# Then search the capture for credential material:

$ grep -a 'password\|NTLM\|krb5\|AS-REQ' /tmp/hostd_capture.log

# Also target lwsmd (Likewise) and vpxa (vCenter agent)
$ strace -f -e trace=read,write -p $(pgrep lwsmd) ...
$ strace -f -e trace=read,write -p $(pgrep vpxa) ...
```

BONUS: Machine password in PLAINTEXT at Likewise registry Pstore\PasswordInfo

04

vCENTER VULNERABILITIES

Plaintext LDAP, private key extraction, JVM heap dumps

DEFAULT CONFIG

Port 389

1. AD-over-LDAP

The standard setup wizard default. Executes 4 to 9 LDAP simple binds per authentication event to the DC.

Network Risk: Transmits AD Bind account & user passwords in plain ASCII on the wire (DC Event 2889).

RECOMMENDED

Port 636

2. AD-over-LDAPS

Wraps LDAP simple binds inside a TLS encrypted channel. Requires adding DC CA root certificate to vCenter.

Impact: Eliminates network cleartext sniffing entirely. Does not fix disk or STS JVM memory retention.

ZERO-PASSWORD

LoginBySSPI

3. SPNEGO / IWA

Authenticates via Kerberos service tickets (`Connect-VIServer` without credentials). Supported on SOAP/PowerCLI.

vCenter 8 Note: Requires connecting strictly via the PNID hostname to avoid endpoint rejection.

INTERNAL

Localhost

4. Internal SSO (vsphere.local)

Authenticates local admin accounts against internal vmdir LDAP service using SASL/SRP over localhost.

Password is never sent over external network, but copies persist in STS JVM heap after login.

MANAGEMENT LAYER

mTLS Solution Users

5. vCenter-to-ESXi Proxy Authentication

vCenter manages ESXi hosts using Solution User Certificates (mutual TLS). User credentials/identity do **NOT** propagate to ESXi. All actions executed by vCenter appear in ESXi logs as the local `vpxuser` account.

DEFAULT AUTH: PLAINTEXT LDAP BIND

vCenter validates AD credentials via LDAP simple bind on port 389.

Every authentication = 4-9 LDAP simple binds = passwords in PLAINTEXT on the wire.

```
LDAP Bind Request (captured via tcpdump)
```

```
-----  
Offset 0x0040:
```

```
7375 7065 726d 616e 4077  superman@w  
7976 6572 6e2e 6c6f 6361  yvern.local  
6c                                     l  
80 0b                               Simple Auth (0x80)  
7665 7279 5365 6375 7265  verySecure  
31                                     1  
-----
```

PASSWORD IS RIGHT THERE. Nothing to decrypt.

DC logs Event 2889 (Binding Type 1) for each insecure bind

SSO admin password (administrator@vsphere.local) is SAFE - uses SASL/SRP on localhost

Confirmed on both vCenter 7 and vCenter 8

LDAP RESPONSE MODIFICATION

4 bytes turns any user into an ESXi admin

Zero Integrity Protection

vCenter's default LDAP (389) - and ESXi's Likewise (LdapSignAndSeal=0) - do UNSIGNED, unsealed LDAP lookups.

A MITM attacker rewrites the DC's reply and neither client can detect the substitution.

~200 lines of Python. +4 bytes.

The Injection

1. MITM vCenter <-> DC (ARP, rogue switch, or iptables on vCenter)
2. Intercept 268-byte SearchResultEntry
3. Inject CN=ESX Admins into memberOf
268 bytes -> 272 bytes
4. Forward the modified response
5. STS issues an elevated SAML token
-> the user now appears as Admin

VEXED module: `ldap_mod` (CVE-2024-37085)

[!] LDAPS fixes this. Plain LDAP has no way to detect the tamper.

VMCA: THE CROWN JEWEL

vCenter Certificate Authority Root CA Private Key

What is VMCA?

Built-in Certificate Authority managing:

- Machine SSL certificates
- Solution user certificates
- ESXi host certificates

Private key location:

`/var/lib/vmware/vmca/privatekey.pem`

Readable by root/sudo on both
vCenter 7 and vCenter 8

No CRL/OCSP infrastructure.

Cannot revoke forged certificates.

Only fix: Replace ENTIRE VMCA root CA

Forge Anything

1. Extract VMCA root CA key
2. Forge certificate for any user
administrator@vsphere.local
with Smart Card Logon ECU
Valid for 10 YEARS
3. Impersonate solution users
4. PKINIT cert forgery
-> Domain compromise

Certificate survives:

- Password changes
- Account lockouts
- Credential resets
- EVERYTHING**

JVM HEAP: PASSWORDS IN MEMORY

vCenter Version	AD Bind Password	SSO Admin Password	Total Cleartext Copies
vCenter 7 (7.0.3)	17 copies	4 copies	21 copies
vCenter 8 (8.0.3)	12 copies	10 copies	22 copies

WHY? Not a bug. Architectural consequence.

STS must hold bind password for LDAP simple binds.

Java immutable Strings create copies during concatenation + SAML generation.

Copies persist until garbage collection reclaims them.

```
# Dump STS process memory
$ gcore -o /tmp/sts_dump $(pgrep -f vmware-std)
# STS heap = 2.18 GB on vCenter 8

# Search for known passwords
$ strings /tmp/sts_dump.* | grep -c 'verySecure1'
17  # <-- 17 cleartext copies of the bind password
$ strings /tmp/sts_dump.* | grep -c 'ssoAdminPass'
4   # <-- 4 cleartext copies of the SSO admin password
```

THE vCENTER 8 PARADOX

More secure at the machine layer, quietly broken for admins

v8 Improvements (machine layer)

- + Kerberos AES-256 machine auth
- + SASL/SRP for vmdir (no simple bind)
- + 256-bit encryption keys
- + On-demand DC connections

Real reductions in the machine-level attack surface.

v8 Regressions (user layer)

- AD GROUP permissions silently non-functional (RetrieveUserGroups = 0)
- 6-9 LDAP simple binds per auth (vs 4-6)
= MORE cleartext exposure per login
- SPNEGO breaks unless you connect via the exact PNID hostname

[!] Upgrading v7 -> v8 can silently drop your group-based access control

AUTHENTICATION METHOD MATRIX

What works where - and what it costs you

Method	Network	Disk	Memory	SSO	Available On
ESXi SSH (Kerberos)	Protected	Keytab 644 / cleartext pwd	User TGT in /tmp	No	ESXi
ESXi API (TLS)	Protected	Same as above	Same as above	No	ESXi
vCenter LDAP 389	CLEARTEXT	AES-ECB (co-located key)	Cleartext heap	No	vCenter (default)
vCenter LDAPS 636	Protected	AES-ECB (co-located key)	Cleartext heap	No	vCenter
vCenter SPNEGO/IWA	None (ticket)	Machine keytab	Ticket only	YES	vC SOAP/PowerCLI
vCenter SSO	None (internal)	SASL/SRP hashed	Transient	YES	vCenter
PKINIT (ESXi)	Cert only	Cert file	Ticket only	YES	ESXi (mod cfg)

[i] Only SPNEGO and SSO avoid password exposure at every layer. No ESXi method does.

ESXi Hypervisor

Model: Kerberos Client Proxy



- ✓ **No Wire Cleartext:** Kerberos pre-authentication protects passwords over the network.
- ✗ **No Session Token Isolation:** Grants shell/API session token directly; caches full 10-hour Kerberos TGTs on disk.
- ✗ **Root Equivalence:** All SSH sessions run as UID 0 (root). No privilege separation.
- ✗ **Account Lockout Bypass:** PAM explicitly skips lockout (`pam_tally2`) for domain users.
- ✗ **Plaintext Machine Password:** Stored unencrypted in Likewise registry (`Pstore\PasswordInfo`).

vCenter Server (VCSA)

Model: LDAP Client + SAML Token Vault



- ✗ **Default Wire Cleartext:** Default LDAP identity source (Port 389) sends bind & user credentials in cleartext ASCII.
- ✓ **SAML Token Blast Radius:** Successful authentication issues signed 30-min SAML tokens; passwords are not retained in tokens.
- ✓ **Granular RBAC:** 35 (v7) to 48 (v8) distinct roles with fine-grained privilege inheritance.
- ✓ **Passwordless SPNEGO:** Native support via `LoginBySSPI` SOAP method (zero password transmission).
- ✗ **Memory Retention:** STS JVM process heap retains 21-22 cleartext password copies in memory.

 Comprehensive Authentication Feature Matrix

FEATURE / CAPABILITY	STANDALONE ESXI	VCENTER SERVER (VCSA)	SECURITY IMPACT SUMMARY
Primary Identity Protocol	Kerberos (Port 88)	LDAP / LDAPS + SAML 2.0	ESXi protects wire auth; vCenter isolates post-auth sessions via SAML.
Passwordless Single Sign-On (SPNEGO)	NOT SUPPORTED (GSSAPI compiled out of sshd)	SUPPORTED (LoginBySSPI via SOAP/PowerCLI)	vCenter enables complete password elimination for management scripts; ESXi forces passwords.
Network Wire Exposure (Default)	Encrypted (Kerberos pre-auth)	CLEARTEXT (Port 389 Simple Bind)	vCenter default identity source exposes bind & user passwords to network sniffing.
Post-Auth Session Tokens	Kerberos TGT Caches (/tmp/krb5cc_*)	Signed SAML 2.0 Bearer Tokens	ESXi TGTs persist 10h on disk, readable by root. vCenter SAML tokens expire in 30m without passwords.
Disk Credential Storage	Plaintext machine pwd in registry; 644 keytab	AES-128/256 ECB bind pwd in vmdir DB	ESXi machine credentials are trivial to extract locally. vCenter keys are co-located with DB.
Process Memory Exposure	Transient Kerberos process state	21 - 22 Cleartext Passwords in STS JVM Heap	Any process with root/gcore access on VCSA can dump bind & SSO admin passwords from RAM.
RBAC Granularity	4 Roles (Admin = Full Root)	35 (v7) to 48 (v8) Granular Roles	ESXi offers no intermediate roles. vCenter enables strict least-privilege scoping.
SSH Public Key Support	Local Accounts Only (Fails for AD Users)	Fails for All Users (pam_mgmt_cli.so)	Domain users cannot use SSH keys on either platform. ESXi local key auth survives pwd resets.
Account Lockout Policy	BYPASSED for Domain Accounts	Enforced via SSO Policy	ESXi PAM explicitly skips lockout for UID > 10000, allowing unlimited SSH brute-force attempts.

05

BEST PRACTICES

Configuring vCenter + ESXi to minimize credential exposure

AUTHENTICATION HARDENING

Method	Network	Disk	Memory	Verdict
LDAP (port 389)	CLEARTEXT	Encrypted*	Cleartext in heap	NEVER USE
LDAPS (port 636)	Encrypted	Encrypted*	Cleartext in heap	Minimum
SPNEGO/IWA	Kerberos ticket	No password stored	No password in heap	RECOMMENDED

Switch to LDAPS (Minimum)

1. Ensure DC has valid TLS cert
2. vSphere Client > Administration > SSO > Configuration > Identity Sources
3. Edit: change ldap:// to ldaps://
Port 389 -> Port 636
4. Test authentication

Result: Zero cleartext on network

Result: Zero DC Event 2889

Still: Passwords in STS heap

Enable SPNEGO/IWA (Best)

PowerCLI with zero passwords:

```
Connect-VIServer -Server  
vcenter.wyvern.local  
# No -User, no -Password!
```

Requirements:

DNS A + PTR records for PNID
HTTP/ and HOST/ SPNs registered
vCenter PNID = AD domain FQDN
(Critical for vCenter 8!)

Result: ZERO password exposure at ALL layers

SPNEGO: THE PNID TRAP

The #1 reason password-free auth 'doesn't work'

vCenter SDK binds to the PNID ONLY

The SDK endpoint accepts the PNID hostname and NOTHING else - even with correct DNS A/PTR and registered SPNs.

SUCCEEDS (via PNID):

```
Connect-VIServer vcenter8.vsphere.local
```

FAILS (via AD FQDN):

```
Connect-VIServer vcenter8.wyvern.local  
'No endpoint listening ...'
```

Get It Right

vCenter 7: PNID usually = AD FQDN, so SPNEGO 'just works'.

vCenter 8: PNID defaults to

<host>.vsphere.local

New deployments: set the PNID to an AD-domain FQDN at install time.

SPNEGO requirements:

- DNS A + PTR for the PNID
- HTTP/ and HOST/ SPNs registered
- connect via the PNID hostname

ADDITIONAL HARDENING

Action	Command / Setting	Impact
Enforce AES-only Kerberos	Set msDS-SupportedEncryptionTypes=24 on all ESXi/vCenter computer accounts	Eliminates RC4-HMAC kerberoasting surface
Fix keytab permissions	chmod 600 /etc/krb5.keytab	Blocks silver ticket attacks from non-root processes
Disable SSH on ESXi	esxcli system settings advanced set -o /UserVars/ESShellEnabled -i 0	Eliminates TGT cache creation entirely
Replace bind account	Use dedicated service account with read-only AD permissions	Limits blast radius of credential exposure
Monitor VMCA key	FIM on /var/lib/vmware/vmca/privatekey.pem	Detects cert forgery attempts
Protect ESX Admins group	Monitor + restrict delegation Audit nested membership	Prevents silent privilege escalation

WHAT DEFENDERS SHOULD MONITOR

Most of this is detectable at the DC

Signal	What it catches
DC Event 2889 (Binding Type 1)	Cleartext LDAP binds / LDAPS reverted to LDAP
DC Event 4768 pre-auth type 16	PKINIT - forged VMCA / smart-card cert use
DC Event 4771 / 4776 from ESXi IPs	Brute force via the ESXi lockout bypass
FIM: vmca/privatekey.pem	VMCA root key theft -> certificate forgery
FIM: /etc/krb5.keytab (644)	Silver-ticket keytab theft
Audit NESTED ESX Admins membership	Silent privilege escalation to ESXi root

[!] ESXi logs no reads of /tmp/krb5cc_* - your TGT-theft detection has to live at the DC

⚠ Credential Exposure Pipeline: Layer-by-Layer Risk

LAYER 1: NETWORK WIRE

Cleartext LDAP Sniffing

vCenter Default: 4–9 LDAP simple binds per login expose `superman@wyvern.local` & user passwords in plain ASCII (Tag 0x80) on Port 389.

ESXi Difference: Uses Kerberos pre-auth (Port 88). Passwords never cross the wire in cleartext.

LAYER 2: DISK ARTIFACTS

TGTs, Registries & Keytabs

ESXi Host Storage:

- Machine pwd in plaintext in Likewise registry.
- World-readable keytab (`644` permissions).
- `/tmp/krb5cc_*` TGTs persist post-logout.

vCenter Storage: Bind pwd encrypted in vmdir LMDB (AES-ECB) with key co-located on disk.

LAYER 3: PROCESS RAM

JVM Heap Credential Accumulation

vCenter STS Heap: Core dumps of `vmware-stsd` reveal **21 to 22 cleartext passwords** (12x AD bind pwd, 10x SSO admin pwd) due to immutable Java String creation.

ESXi RAM: Retains active Kerberos tickets and session structures in kernel/lsass memory.

✅ Required Hardening Measures

1. Enforce LDAPS (Port 636)

Change vCenter AD identity source from `ldap://` to `ldaps://` to immediately eliminate wire cleartext exposure.

2. Restrict ESXi Keytab Permissions

Execute `chmod 600 /etc/krb5.keytab` on all ESXi hosts to prevent non-root processes from stealing service keys.

3. Enforce AES-Only Kerberos

Set `msDS-SupportedEncryptionTypes = 24` on all ESXi/vCenter AD computer accounts to disable vulnerable RC4-HMAC service tickets.

4. Disable ESXi SSH by Default

Keep SSH disabled on ESXi hosts to prevent TGT cache creation (`/tmp/krb5cc_*`) and pass-the-ticket lateral movement.

06

LIVE DEMO: VEXED v2.0

vSphere EXploitation & Extraction Device

VEXED v2.0

Automated vSphere Post-Exploitation Assessment

#	Module	Description	Target
1	detect	Fingerprint ESXi vs vCenter	All
2	enumerate	Users, groups, permissions	All
3	credentials	Keytabs, ccache, registry, shadow	All
4	vmca	VMCA/SSL key extraction	All
5	vpv_decrypt	VPX database credential decryption	vCenter
6	proc_memory	ESXi process memory (strace)	ESXi
7	jvm_heap	JVM heap scanning (gcore)	vCenter
8	saml_forge	SAML token forgery via STS cert	vCenter
9	hardening	Configuration audit (14+ checks)	All
10	pivot	Auto-pivot to managed ESXi hosts	vCenter
11	attack_graph	Interactive Cytoscape.js graph	All

DEMO TIME

```
# ESXi Assessment
```

```
$ vexed -t 10.0.5.21 -u superman@wyvern.local \  
    --auto --dumpcreds --json
```

```
# vCenter Full Chain Assessment
```

```
$ vexed -t 10.0.5.23 -u administrator@vsphere.local \  
    --sso-password 'sso_pass' --auto --dumpcreds
```

```
vCenter Full Attack Chain
```

```
=====
```

```
SSH (root) --> vCenter
```

```
|-> Keytab -----> Domain Compromise
```

```
|-> vmdir -> STS Cert
```

```
|          |-> Forged SAML -> Admin API
```

```
|-> VMCA Key -> Forged Certs
```

```
|          |-> PKINIT ----> Domain Compromise
```

```
|-> symkey.dat -> VPX Decrypt
```

```
|          |-> SSH to all ESXi hosts
```

```
'-> TGTs (ccache) -----> Domain Compromise
```





THREE DECISIONS DECIDE YOUR POSTURE

Everything reduces to these

01 Standalone ESXi vs vCenter

Standalone + direct AD join cannot be secured: no SSO, all UID 0, world-readable keytab, plaintext machine password, lockout bypass. Manage through vCenter.

02 Which identity-source protocol

LDAP/389 = cleartext at every layer. LDAPS/636 = the minimum. SPNEGO / IWA = the only zero-exposure path.

03 How the ESXi admin group is governed

AD group management IS ESXi root management.
Treat esxAdminsGroup like Domain Admins.

*"The passwords are out there and
tickets are on disk.
The certificates survive everything.*

*The question is whether you
know where they are."*

Darryl G. Baker

VEXED: github.com/dfirdeferred/vexed
Blog Site: <https://blog.menrva-tech.com>

QUESTIONS?

Research Full paper: <https://blog.menrva-tech.com/blog/2026/03/27/vSphere-AD-Integration-Security-Implications/>