LEVERAGING DECEPTION TECHNIQUES FOR STRONG DETECTION

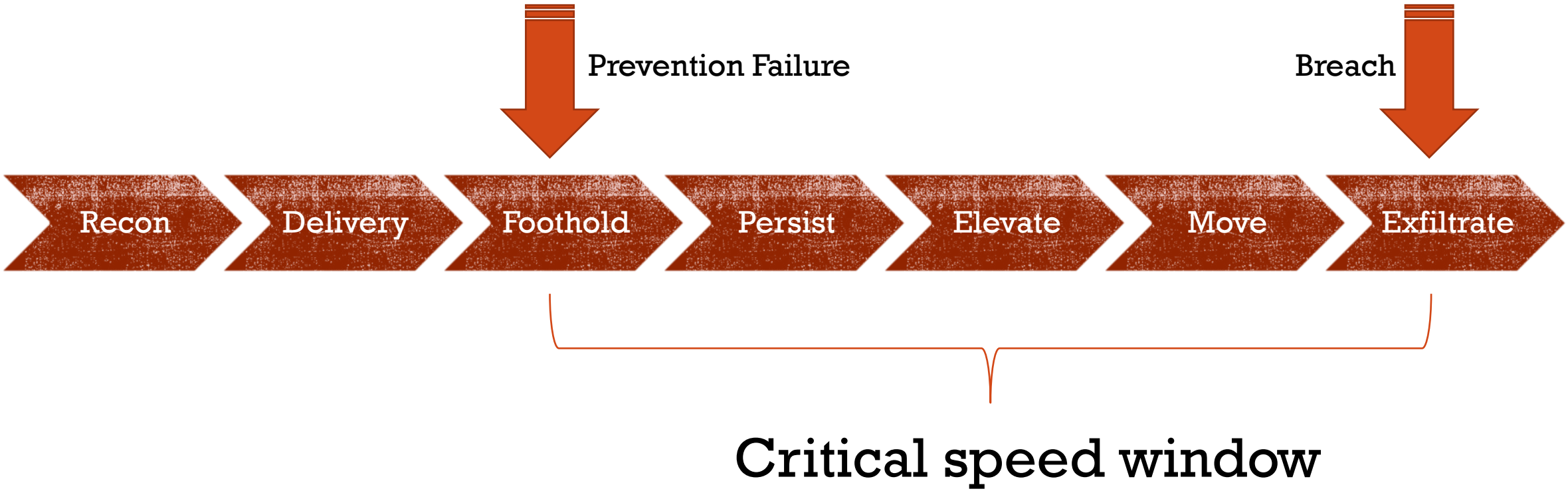
Tim Crothers

THANKS!

DEFENDERS DILEMMA

"Breaches are inevitable because the defenders have to be right 100% of the time whereas the attackers only have to be right once."

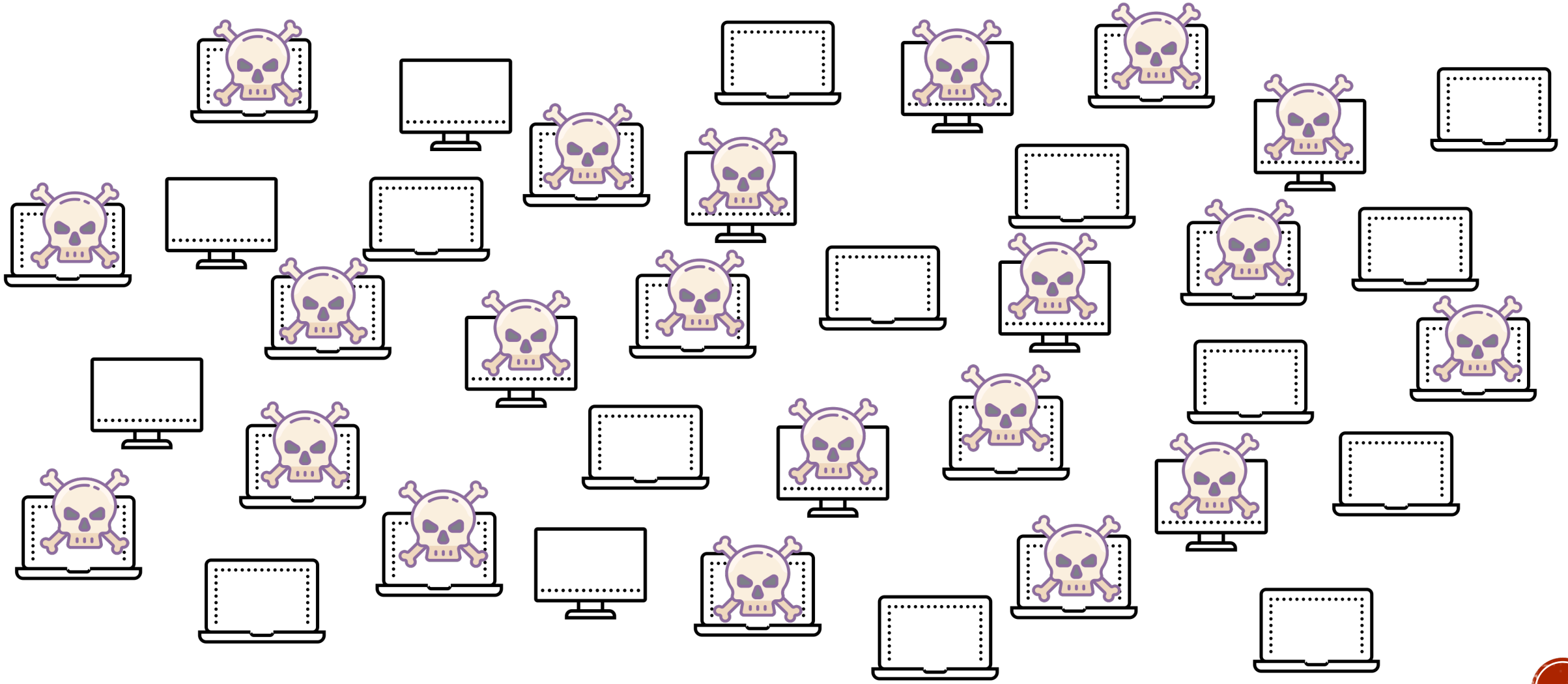
WHEN DOES A BREACH OCCUR?



ATTACKERS DILEMMA

How do they get to their goal and out without tripping a single one of our detection ‘landmines’?

OBJECTIVE?



HOW?

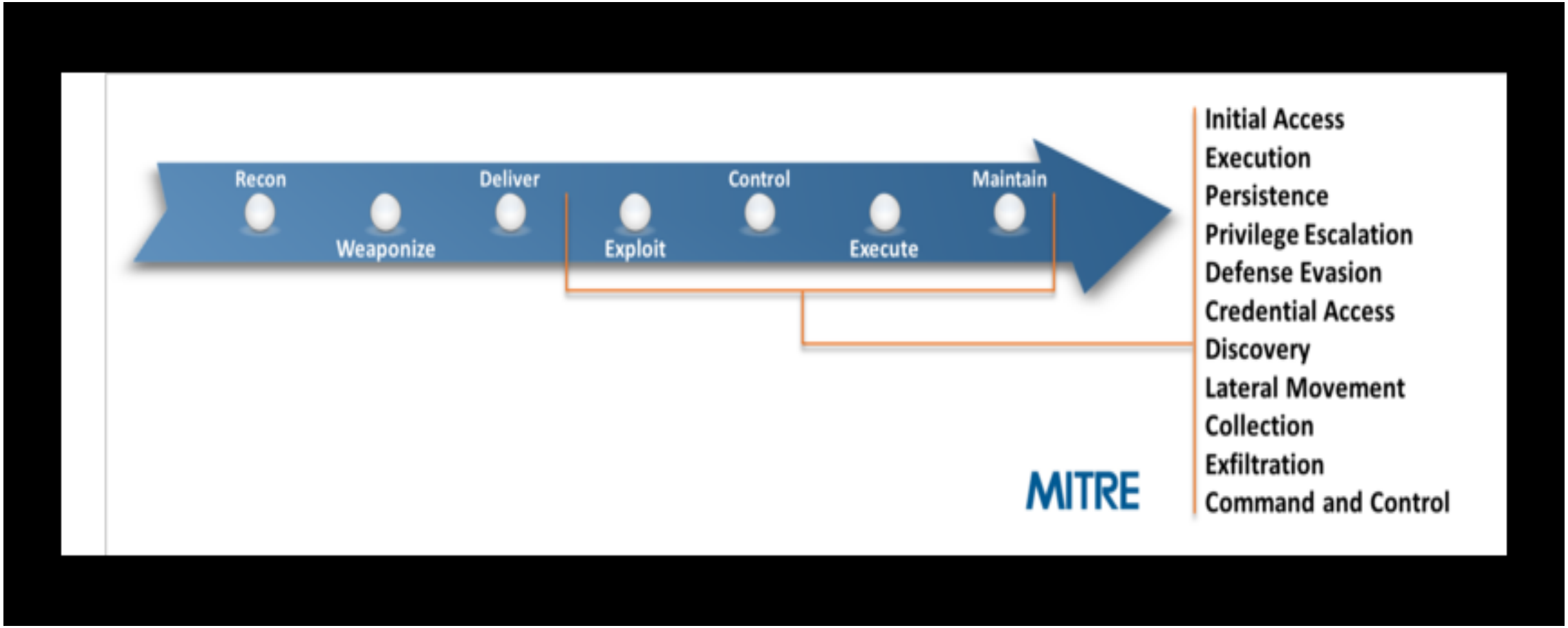


Image credit: https://attack.mitre.org/wiki/File:MITRE_attack_tactics.png

WHAT?

Credential Dumping	**Credential Access**	**Cached Credentials** The DCC2 (Domain Cached Credentials version 2) hash, used by Windows Vista and newer caches credentials when the domain controller is unavailable. The number of default cached credentials per system. This hash does not allow pass-the-hash style attacks. A number of tools can be used to retrieve the SAM file through in-memory techniques. • pwdumpx.exe • gsecdump • Mimikatz Alternatively, reg.exe can be used to extract from the Registry and Creddump7 used to gather the credentials. Notes: Cached credentials for Windows Vista are derived using PBKDF2. **Local Security Authority (LSA) Secrets** With SYSTEM access to a host, the LSA secrets often allows trivial access from a local account to domain-based account credentials. The Registry is used to store the LSA secrets. When service domain users, their passwords are stored in the Registry. If auto-logon is enabled, this information will be stored in the Registry as well. A number of tools can be used to retrieve the SAM file throu • pwdumpx.exe • gsecdump • Mimikatz • secretsdump.py Alternatively, reg.exe can be used to extract from the Registry and Creddump7 used to gather the credentials. Notes: The passwords extracted by his mechanism are UTF-16 encoded, which means that they are returned in plaintext. Windows 10 adds protections for LSA Secrets described in Mitigation. **NTDS from Domain Controller** Active Directory stores information about members of the domain including devices and users to verify credentials and define access rights. The Active Directory domain database is stored in the NT be located in %SystemRoot%\NTDS\Ntds.dit of a domain controller.[6] The following tools and techniques can be used to enumerate the NTDS file and the contents of the entire Active Directory hashes.

Reference: https://attack.mitre.org/wiki/Credential_Access

LURES?

- Popular targets of credential dumping include
  - SAM
  - Cached credentials
  - LSA secrets
  - NTDS from Domain Controller
  - Group Policy Preference (GPP) files
  - Plaintext credentials

WEAPONIZING OUR ENDPOINTS

1. Plan out your lures for maximum realness
2. Create fake lure domain accounts
3. Log in locally on end points with lure domains to cache them locally
4. Add local cache credentials to strengthen lures
5. Change lure domain account passwords
6. Implement alerting on attempted use of our lure accounts

WINDOWS AUTHENTICATION

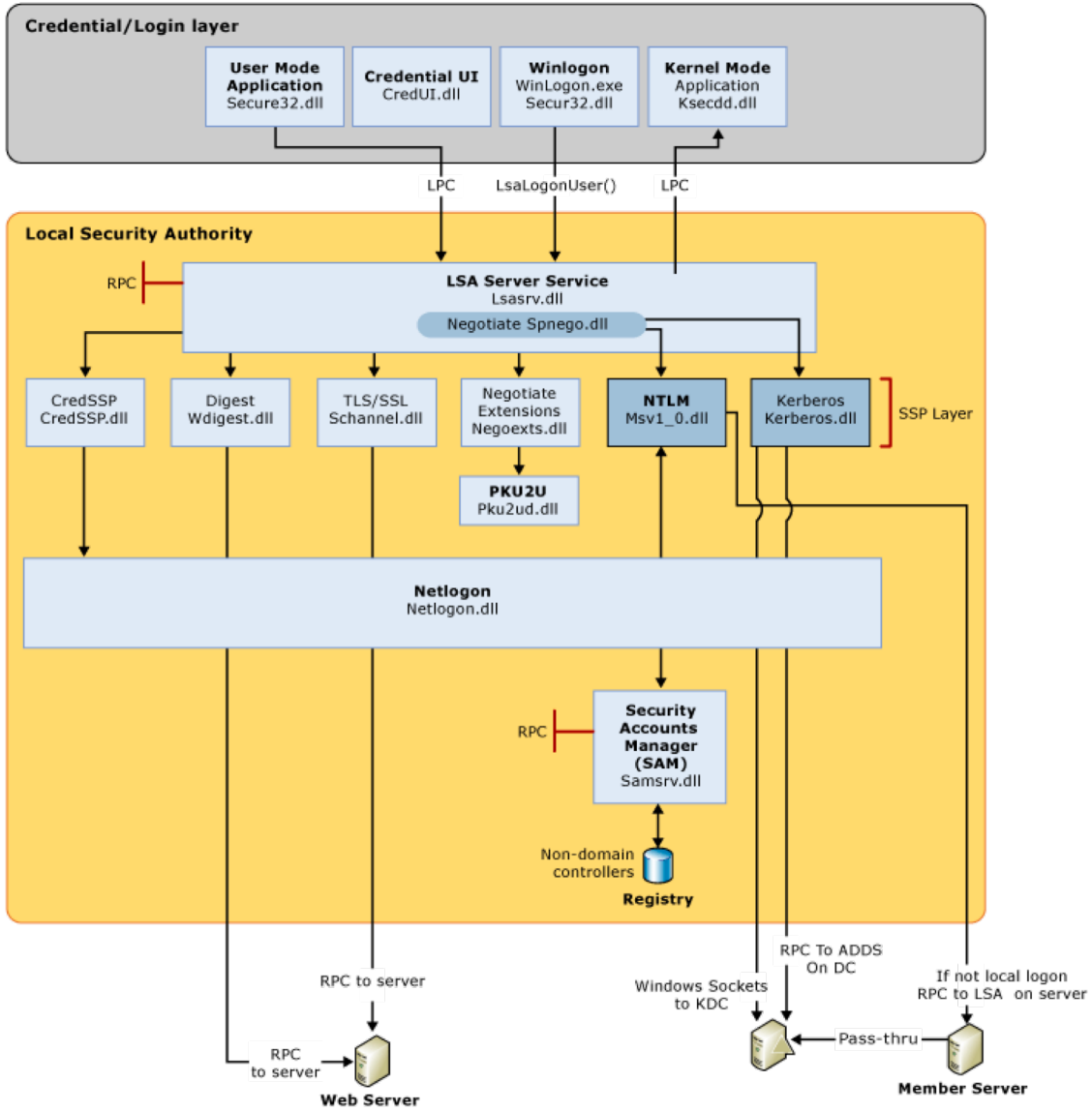


Image from https://docs.microsoft.com/en-us/windows-server/security/windows-authentication/credentials-processes-in-windows-authentication

LOCAL CACHING

The stored credentials are directly associated with the Local Security Authority Subsystem Service (LSASS) logon sessions that have been started after the last restart and have not been closed. For example, LSA sessions with stored LSA credentials are

- Logs on to a local session or Remote Desktop Protocol (RDP) session on the computer
- Runs a task by using the **RunAs** option
- Runs an active Windows service on the computer
- Runs a scheduled task or batch job
- Runs a task on the local computer by using a remote administration tool

ccount processes, are stored on the hard disk drive. Some of these secrets are credentials that must persist after reboot, and they are stored in

- Account password for the computer's Active Directory Domain Services (AD DS) account
- Account passwords for Windows services that are configured on the computer
- Account passwords for configured scheduled tasks
- Account passwords for IIS application pools and websites
- Passwords for Microsoft accounts

Image from https://docs.microsoft.com/en-us/windows-server/security/windows-authentication/credentials-processes-in-windows-authentication

DEMO

WEAPONIZING OUR ENDPOINTS

1. Plan out your lures for maximum realness
    1. ID's should follow company standards
    2. Consider which endpoints to add lure caches to and which not to
    3. Consider whether just some endpoints (and which) should have local secrets cached
2. Create fake lure domain accounts
3. Log in locally on end points with lure domains to cache them locally
4. Add local secrets credentials to strengthen lures
5. Change lure domain account passwords
6. Implement alerting on attempted use of our lure accounts

CHALLENGES

- Actually disabling the lure accounts makes detection difficult
- Because the failure occurs against the AD DC's you won't detect until they attempt to use
- Making it realistic is problematic

NEXT STEPS

- Weaponizing your data sources works well too!
  - Wiki's with lure accounts/passwords
  - Github repositories with lure accounts/passwords
  - File shares…
  - Sharepoint…
- Lots of other ways to approach this problem
  - Scheduled tasks to create local cached lures
  - Make it part of your system build process?

THANK YOU!


badsecurity@gmail.com


@soinull


linkedin.com/in/tim-crothers-5458738/


https://github.com/Soinull/Weaponizing_Endpoints