Secure Systems Design and Technology Development. Veliko Tarnovo, Bulgaria
Proceedings of the International Scientific Conference, 2026. Volume III, 219-222

DCSync Attack from an Adversary Point of View

Dimitar Nikolov
IT Department
Nikola Vaptsarov Naval Academy
Varna, Bulgaria
d.nikolov@naval-acad.bg

Abstract – This paper examines the DCSync attack from an adversarial perspective, analysing its operational logic, execution variants, and the detection surface each variant presents within monitored Active Directory environments. The study was conducted in a controlled laboratory environment comprising a Windows Server 2019 domain and a monitoring stack including Wireshark, Suricata, and Windows Event Log auditing, enabling systematic comparison of five distinct execution approaches ranging from PowerShell in-memory execution and Mimikatz invocation to Beacon Object File deployment and Impacket-based execution from Linux hosts. Experimental results demonstrate that DCSync's widely assumed low-noise characteristic is method-dependent rather than inherent. Network traffic analysis identifies non-domain-controller source IP addresses, TTL values indicative of Linux origin, and the absence of Kerberos authentication as reliable detection indicators, while Windows Event ID 4662 provides complementary log-based coverage across all tested variants. The findings confirm that combined network, log, and protocol-level monitoring provides effective detection across the full range of execution paths, and position DCSync as a technique whose stealth is contingent on tooling selection, authentication protocol, and operational timing rather than on any intrinsic property of the attack itself.

Keywords – Active Directory, credential access, DCSync, detection, domain replication, operational security, privilege escalation, Windows security.

I. INTRODUCTION

DCSync is a stealthy attack geared towards gaining access to domain account NTLM password hashes via abuse of legitimate functionality used for data replication and synchronisation between domain controllers in an Active Directory environment [1]. The attack itself usually happens in the latest stages of a kill chain, when sufficient control and presence have been established in the environment. The attack leverages the Directory Replication Service Remote Protocol (MS-DRSR) [2] to extract credentials without running code on the target domain controller memory or directly compromising the NTDS.dit file. It requires a domain account with replication permissions, specifically Replicating Directory Changes and Replicating Directory Changes All. These permissions are granted by default to the Domain Admins, Enterprise Admins, Administrators, and Domain Controllers groups, and the System account. Read-only domain controllers (RODCs) can also be targeted. The synchronisation is performed via a replication request utilising the DRSUAPI interface, specifically the GetNCChanges function - the same API call and mechanism used in legitimate domain controller to domain controller replication - which makes the attack fairly stealthy by design.

Legitimate replication calls between domain controllers happen frequently and can be triggered by changes to the Active Directory database, scheduled intervals, or manual administrative actions. Event-based triggers include urgent replication events such as account lockouts, changes to the LSA Secret, and PDC Emulator master changes [3], all of which bypass typical schedules and initiate immediate replication. Update notifications represent a further event-based trigger, whereby when a change is made the source domain controller notifies its replication partners after a short delay. If the Knowledge Consistency Checker creates a new connection object due to topology changes, replication is triggered as well. Scheduled triggers govern intra-site replication, which occurs automatically by default at five-minute intervals, and inter-site replication, which is scheduled by an administrator via site links and designed to manage bandwidth usage. Administratively initiated replication can be triggered voluntarily by any user with access to an account holding the required permissions. System event triggers include domain controller boot and promotion events, both of which initiate synchronisation via Server Manager or PowerShell to ensure the controller is up to date before accepting client requests [4].

All the above demonstrates that legitimate replication happens fairly often - at least once every five minutes - but still adheres to a rhythmic pattern to a certain extent. This is operationally significant since this rhythm affects offensive operational security choices and simultaneously creates the baseline for identifying anomalous behaviour during threat hunting [6].

Online ISSN 3044-7771
https://doi.org/10.68302/std2026.vol3.213

II. DCSYNC ATTACK CHAIN

The DCSync attack chain consists of four sequential steps, each carrying distinct operational security considerations. The first step requires taking over or creating an account capable of triggering replication. Most of the default accounts holding these permissions reside on the domain controller, which is a highly monitored area and a position an offensive operator should seek to avoid. The preferred strategy is hunting for domain users or weak service accounts that have been additionally granted replication permissions. BloodHound, SharpHound, and ACL manipulation [5] are the paths most commonly taken at this stage, typically in combination with other lateral movement techniques [7].

The second step involves locating the target domain controller. One of the less operationally secure approaches is skipping proper host enumeration and performing a scan. Regardless of how well configured, a scan is an action that is well signatured and carries a risk of disclosing the operator's current position. Since all domain-joined machines communicate almost constantly with the domain controller, it is quite often sufficient to monitor network traffic, check logs and caches, or perform a simple DNS resolve or network configuration query to locate the domain controller without generating anomalous activity. The third step is sending the replication request. Numerous tools are capable of sending a replication request and receiving the resulting hashes, including Mimikatz in all of its forms and variants, Impacket, PowerShell, and built-in utilities in most popular C2 framework implants and beacons. All of these require authentication material and an account with replication permissions. The fourth and final step is utilising the retrieved hashes to move laterally, vertically, or to act on operational objectives. This step is also one of the determining factors in what constitutes the best tooling choice in the previous step, since different downstream applications impose different requirements on the format and completeness of the extracted credential material.

The techniques and tools most commonly used at step three each carry distinct trade-offs. PowerShell-based implementations, such as the Invoke-DCSync scripts available from pentestfactory and S3cur3Th1sSh1t, pull Mimikatz, PowerView, and ADRecon from GitHub into memory and perform DCSync without writing primary tooling to disk. The S3cur3Th1sSh1t variant additionally extracts the krbtgt password, builds a Golden Ticket, imports it, and adds a specified account to the Enterprise Admins group in a single execution chain, though it does output multiple files to disk [8]. To reduce detection risk, scripts should be staged on a payload server within offensive infrastructure, protected where possible by redirectors, and executed on a softened legitimate Active Directory machine with detection and logging disabled - ideally a new or compromised domain controller, which aids in blending with legitimate replication traffic.

Mimikatz remains one of the most widely used tools for DCSync execution. Synchronisation of all domain accounts should be avoided, as the /all flag produces a measurable peak in network traffic and is operationally unnecessary given that sufficient access has already been established by the point DCSync is employed. Targeted extraction of specific high-value accounts is the preferred approach. SafetyKatz, a combination of a slightly modified Mimikatz build and a .NET PE Loader created by Will Schroeder of SpecterOps, provides equivalent capability with improved evasion characteristics.

A Beacon Object File represents the most operationally secure execution path for DCSync where a C2 implant is already established on the target machine. A BOF is a compiled C program written to a convention that allows it to execute within a beacon or implant process and use internal beacon APIs, extending the agent with new post-exploitation features without spawning new processes visible to monitoring tools. Initially an exclusively CobaltStrike feature, BOF execution has been widely adopted across modern C2 frameworks including PoshC2, Sliver, AdaptixC2, and Havoc. Since most current C2 implants operate as fileless malware running entirely in memory, and since BOF execution inherits the implant's existing unhooking and alternative API call mechanisms, this approach minimises both disk and process-level detection vectors. Mimikatz, SafetyKatz, Kiwi, and Invoke-DCSync all exist as assemblies or are implemented directly within C2 agent frameworks [9].

Impacket's secretsdump.py provides DCSync capability from a Linux host and is not recommended in well-monitored environments, as the detection section of this paper demonstrates it presents multiple observable indicators. If used, Kerberos ticket authentication is strongly recommended. In DCSync attacks, different authentication mechanisms and protocols can be employed, including Kerberos, MS-RPC, and SMBv3. Kerberos ticket authentication represents the common secure baseline, and any deviations - including plaintext passwords, pass-the-hash using NTLMv1 hashes, or raw SMB authentication - are outliers that should be considered operationally unsafe.

III. DETECTION OF THE DCSYNC ATTACK

Detection of DCSync [10] activity is most effective when network monitoring, Windows event log correlation, and protocol-level analysis are deployed in combination, since no single indicator provides reliable coverage across all execution methods. Each layer of detection addresses a different subset of the attack surface, and their convergence produces a high-confidence detection posture that is resistant to the operational security measures an adversary is likely to employ.

The primary log-based detection signal is Windows Event ID 4662 [11], generated when a principal exercises an extended right against a directory object. Specifically, the appearance of the GUID 1131f6ad-9c07-11d1-f79f-00c04fc2dcd2 - representing the control access right for Replicating Directory Changes All - in a 4662 event indicates that a replication operation has been performed against the directory. The analytical value of this indicator depends on the identity of the account triggering the event. Any account outside the Domain Controllers, Domain Admins, and Enterprise Admins groups generating this event represents an anomaly that warrants immediate

investigation [12]. In environments where replication permission grants are properly controlled and audited, this indicator carries a high signal-to-noise ratio and a low false positive rate.

Network monitoring and traffic analysis represent the most reliable and method-independent detection layer, capable of identifying DCSync activity regardless of the tooling used by the attacker. The first and most actionable network indicator is source IP anomaly: DsGetNCChanges replication requests originating from IP addresses outside the designated domain controller subnet are inherently anomalous, since in a correctly configured environment only domain controllers initiate replication requests. A detection rule implemented in an intrusion detection or prevention system such as Snort or Suricata, configured to alert on DsGetNCChanges [13] requests from non-domain-controller IP addresses, provides a high-fidelity detection signal with minimal tuning overhead. Fig. 1 illustrates a captured DsGetNCChanges request from a non-domain-controller host, showing the source IP and DRSR protocol identifier that together provide the basis for this detection rule.

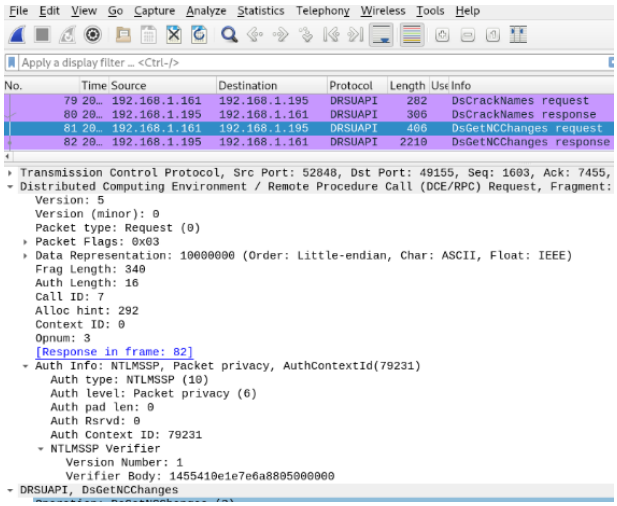

Fig.1. DsGetNCChanges Request.

The second network indicator is TTL-based operating system fingerprinting. Examination of the IP TTL value in replication request packets provides a reliable method for identifying the origin operating system of the requesting host. A TTL value of 64, as illustrated in Fig. 2, is characteristic of Linux default IP stack behaviour and is distinctly different from the TTL of 128 produced by Windows hosts. When a DsGetNCChanges request originates from an IP address outside the domain controller subnet and carries a TTL of 64, the combination provides a highly specific detection signature for Impacket-from-Linux executions without requiring deep packet inspection or protocol decryption.

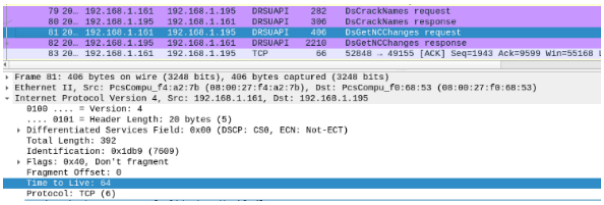

Fig. 2. DsGetNCChanges Request TTL 64

The third network indicator is protocol composition anomaly. In a correctly functioning Active Directory environment, all domain controller to domain controller replication sessions include Kerberos authentication as part of the session establishment. Protocol analysis of traffic associated with Impacket NTLM-authenticated DCSync executions reveals a session composed entirely of SMB2 [14] traffic with no corresponding Kerberos exchanges, as illustrated in Fig. 3. The complete absence of Kerberos traffic in a replication session is a high-confidence anomaly that is straightforwardly detectable through protocol distribution analysis of captured traffic. It should be noted that Windows-based executions using Mimikatz or BOF methods authenticate via Kerberos by default, producing session profiles that are consistent with legitimate replication and substantially reducing the value of this indicator for those specific methods [15].

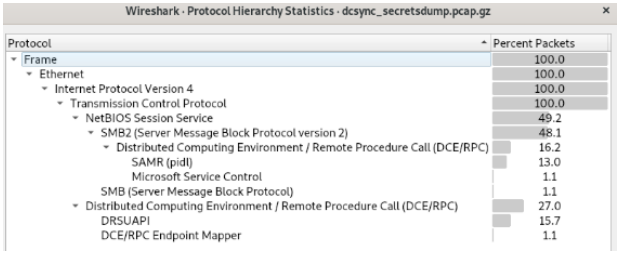

Fig. 3. SMB2 DCSync.

The three indicators together - non-domain-controller source IP, TTL value of 64, and SMB-only protocol composition absent Kerberos - form a highly specific composite signature for the use of Impacket from a Linux host to execute DCSync following lateral access to the local network segment. The convergence of all three in a single captured session, as observed in the laboratory environment, reduces the probability of false positive classification to negligible levels. The fourth network indicator is traffic volume anomaly. Full domain synchronisation executions, extracting all account hashes simultaneously, produce a measurable spike in DRSR traffic volume relative to the established replication baseline, as illustrated in Fig. 4. This peak is both unnecessary from an operational perspective - since targeted extraction of high-value accounts provides equivalent leverage - and avoidable, representing an operational security failure that significantly increases detection probability.

Protocol decryption capabilities substantially enhance detection coverage across all execution methods, including Windows-based approaches that otherwise produce network profiles consistent with legitimate replication. The DCSync attack relies on a number of Microsoft protocols including Kerberos, MS-RPC, and SMBv3, all of which

employ encryption in default enterprise configurations. Security monitoring tools equipped with decryption capabilities for these protocols can perform early detection of abnormal behaviour within encrypted sessions, including the identification of forged Kerberos tickets and anomalous MS-RPC call patterns that would otherwise be invisible to signature-based network detection. Protocol decryption should therefore be considered an essential capability for any security monitoring architecture deployed in environments where Active Directory credential theft represents a credible threat scenario.

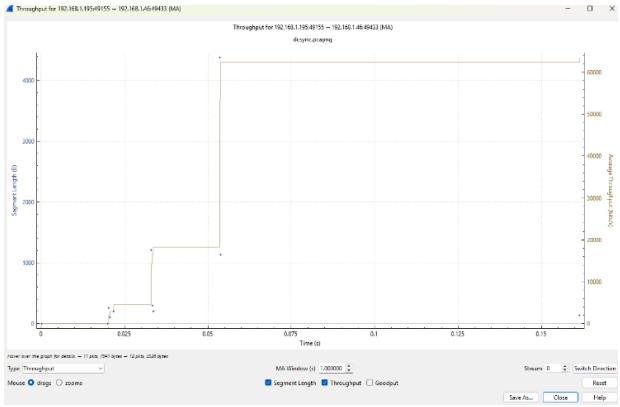

Fig.4. DCSync all request traffic peak.

IV. Conclusion

DCSync represents a high-value technique in the final stages of Active Directory compromise, offering an adversary the ability to extract domain credential material without direct interaction with domain controller memory or the NTDS.dit database. Its operational value, however, is contingent on the disciplined application of sound operational security practices at every stage of its execution chain - from privilege acquisition and domain controller location through tooling selection, authentication protocol choice, and the scope of the replication request itself. The experimental findings presented in this paper confirm that DCSync's reputation as an inherently low-noise technique is method-dependent rather than absolute, and that operationally careless execution - particularly through Impacket from a Linux host using NTLM authentication, or through full-domain synchronisation - produces a distinctive and reliably detectable forensic signature across network, log, and protocol monitoring layers simultaneously.

DCSync is most correctly understood not as a terminal objective but as an enabling technique whose primary value lies in what it makes possible downstream. Its strongest application is as a component of Kerberos-based persistence strategies, where the credential material it yields - most critically the krbtgt hash and targeted service account hashes - enables the fabrication of Golden, Silver, Diamond, and Platinum tickets, providing an adversary with durable, difficult-to-remediate Enterprise Admin-level persistence within the domain. It is within this broader strategic context that DCSync's risk profile must be evaluated: the operational security investment required to execute it stealthily is justified precisely by the long-term persistence and privilege it unlocks, and any execution approach that compromises stealth for convenience undermines the very objective the technique is intended to serve.

References

[1] Invoke-DCSync, "Invoke-DCSync Linux Implementation." [Online]. Available: https://github.com/pentestfactory/Invoke-DCSync/tree/main/linux [Accessed: Mar. 26, 2026].

[2] ViperOne, "Credential Dumping: NTDS." [Online]. Available: https://viperone.gitbook.io/pentest-everything/everything/everything-active-directory/credential-access/credential-dumping/ntds [Accessed: Mar. 25, 2026].

[3] ViperOne, "DCSync Attack." [Online]. Available: https://viperone.gitbook.io/pentest-everything/everything/everything-active-directory/credential-access/credential-dumping/dcsync/dcsync-attack [Accessed: Mar. 26, 2026].

[4] 0xss0rz, "DCSync Attack (Internal Pentest)." [Online]. Available: https://0xss0rz.gitbook.io/0xss0rz/pentest/internal-pentest/dcsync [Accessed: Mar. 26, 2026].

[5] W. Uentin, "InvokeRogueDC Tool." [Online]. Available: https://github.com/Wuentin/InvokeRogueDC [Accessed: Mar. 26, 2026].

[6] D. Dimitrov et al., "The Risk of Personal Smart Devices in the Operational Security for Military Bases, Personnel and Missions," Environment. Technology. Resources. Proceedings of the International Scientific and Practical Conference, vol. 2, pp. 99–106, Jun. 2025, https://doi.org/10.17770/etr2025vol2.8616 [Accessed: Mar. 26, 2026].

[7] U. Ravindran, "AD Series: DC Sync Attacks," Medium, 2023. [Online]. Available: https://medium.com/@urshilaravindran/ad-series-dc-sync-attacks-e76bb54308f5 [Accessed: Mar. 28, 2026].

[8] Active Directory Security, "Mimikatz DCSync Detection," Oct. 2018. [Online]. Available: https://www.active-directory-security.com/2018/10/mimikatz-dcsync-detection.html [Accessed: Mar. 28, 2026].

[9] N. Vladimirova and D. Dimitrova, "Simulation of a LoRa-Based Wearable System for Maritime Emergency Tracking Using MATLAB," in Proc. 19th Int. Conf. on Communications, Electromagnetics and Medical Applications (CEMA'25), Athens, Greece, 2025, pp. 16-20. ISSN: 1314-2100. [Online]. Available: http://rcvt.tu-sofia.bg/CEMA/proceedings/CEMA_2025_proc.pdf [Accessed: Mar. 28, 2026].

[10] Netwrix, "What is DCSync: An Introduction." [Online]. Available: https://netwrix.com/en/resources/blog/what-is-dcsync-an-introduction/ [Accessed: Mar. 15, 2026].

[11] ExtraHop, "DCSync Attack Overview." [Online]. Available: https://www.extrahop.com/resources/attacks/dcsync [Accessed: Mar. 29, 2026].

[12] D. Dimitrov and E. Andreev, "China's Strategic Competition in Cyberspace: Volt Typhoon and Salt Typhoon as a Projection of Power, a More Aggressive Posture and a Future Beyond Espionage," Environment. Technology. Resources. Proceedings of the International Scientific and Practical Conference, vol. 2, pp. 115–122, 2025. Available: https://doi.org/10.17770/etr2025vol2.8618 [Accessed: March 28, 2026].

[13] Altered Security, "A Primer on DCSync Attack and Detection." [Online]. Available: https://www.alteredsecurity.com/post/a-primer-on-dcsync-attack-and-detection [Accessed: Apr. 05, 2026].

[14] W. Schroeder, "SafetyKatz Credential Extraction Tool," GhostPack, 2018. [Online]. Available: https://github.com/GhostPack/SafetyKatz [Accessed: Apr. 05, 2026].

[15] P0142, "DCSync Beacon Object File." [Online]. Available: https://github.com/P0142/DCSync-Bof [Accessed: Apr. 05, 2026].