Get Windows Logon Password in Memory Dump

Deok9

DDeok9@gmail.com

Contents of Table

Introduction

기존 윈도우 로그온 패스워드 획득 방법

- **Mimikatz가 나오기 전**
  - **레지스트리**
  - **윈도우 로그온 세션**

F-Insight 6월 The Stealing Windows Password 참고

사용 파일	패스워드의 NTLM 해쉬 획득 방법
SAM	SAM 하이브 파일의 값 복호화
NTDS.DIT	NTDS.DIT의 데이터베이스 테이블 추출 후 복호화
NTDS.DIT /SAM	NTDS.DIT/SAM 하이브 파일의 Password History 정보를 이용
SECURITY	SECURITY 하이브 파일의 LSA Secret 복호화
SECURITY	SECURITY 하이브 파일의 Cached Domain Logon 정보를 이용
MSV1.0	윈도우 로그온 세션의 Credential 정보를 이용

문제점 & Mimikatz

- 모두 NTLM 해시 값을 획득하는 방법들임
  - 패스워드 크랙 도구를 통해 크랙해야 함
    - ✓ John the Ripper, Ophcrack, Cain & Abel ETC
  - 패스워드가 매우 길다면?
    - ✓ 평문 얻기 위해 매우 많은 시간 소요

- 문제점 해결을 위해 Mimikatz 등장(2012)
  - 라이브 상태에서 DLL Injection을 통해 윈도우 로그온 패스워드 평문 획득
  - 윈도우 인증 패키지를 사용

WDigest.dll

윈도우 인증 패키지

- 윈도우 보안을 구현하는 주요 구성 요소 중 하나
  - LSASS 프로세스와 클라이언트 프로세스 내에서 실행되는 DLL을 포함
  - 인증 패키지에서 사용하는 DLL : 주어진 사용자 이름과 패스워드가 일치하는지 여부 검사
    - ✓ 일치하는 경우 : LSASS에 좀 더 상세한 사용자 정보 반환 -> LSASS가 토큰 생성
  - Challenge-Response 방식을 통해 특정 필요 데이터를 메모리에 항상 가지고 있는 특징

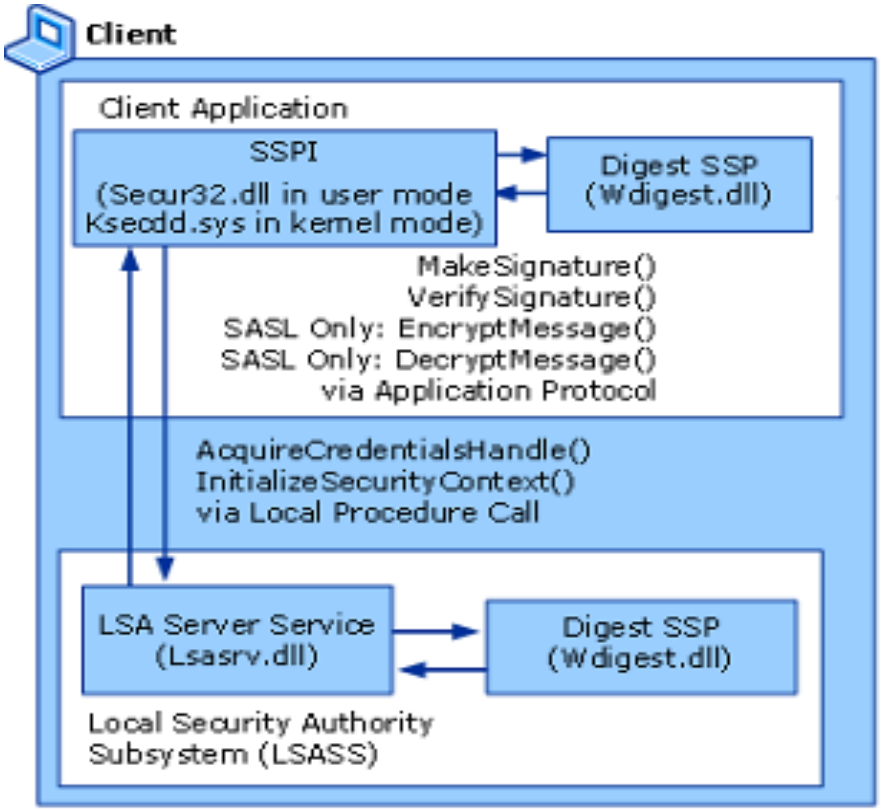
- 대표적인 패키지
  - MSV1_0, TsPkg, WDigest, LiveSSP, Kerberos, SSP 등
  - Remote RDP, 웹 서비스 등 다양한 용도에 의해 구현

WDigest.dll 소개

- 윈도우 XP 에서 처음 소개 됨
- HTTP Digest 인증 및 SASL(Simple Authentication Security Layer) 교환에서 사용자 인증을 위해 개발
- NTLM 프로토콜과 같이 Challenge-Response 방식을 사용
- 인증을 위해서는 사용자의 평문 패스워드가 필요한 특징이 있어 이를 악용 가능

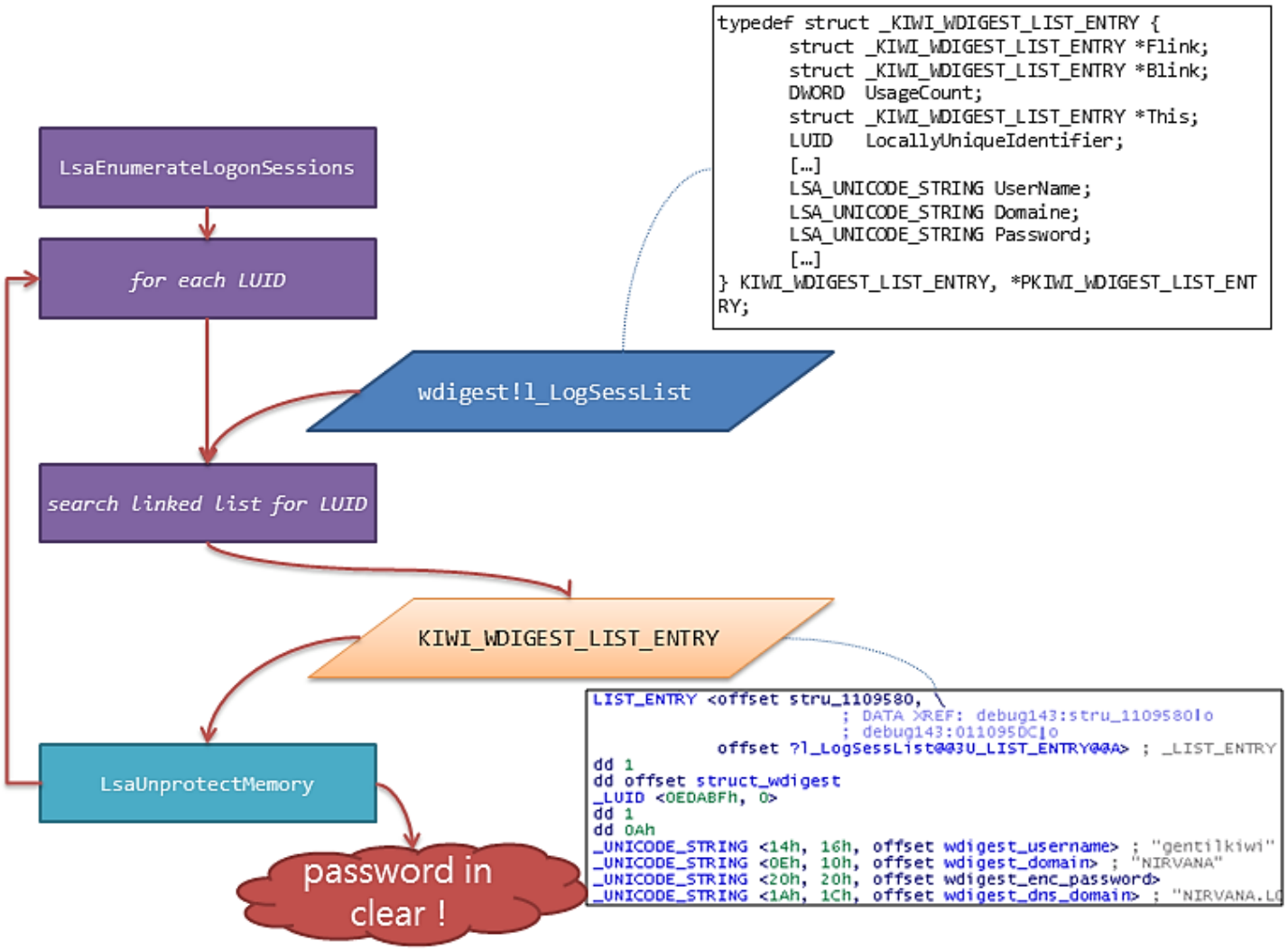
WDigest.dll Digest 인증 아키텍처

- **WDigest.dll**
  - 인증을 위해 사용되는 SSP 구현
- **Lsasrv.dll**
  - 보안 서비스 관리 및 동작에 관여
- **Secure32.dll**
  - 유저 모드 응용프로그램 SSPI 구현
- **Ksecdd.sys**
  - 커널 보안 장치 드라이버가 유저 모드에서 LSASS와 통신하는데 사용



Get Windows Logon Password in Live

동작 과정


LsaEnumerateLogonSessions
for each LUID
wdigest!l_LogSessList
search linked list for LUID
KIWI_WDIGEST_LIST_ENTRY
LsaUnprotectMemory
password in clear !
typedef struct _KIWI_WDIGEST_LIST_ENTRY {
struct _KIWI_WDIGEST_LIST_ENTRY *Flink;
struct _KIWI_WDIGEST_LIST_ENTRY *Blink;
DWORD UsageCount;
struct _KIWI_WDIGEST_LIST_ENTRY *This;
LUID LocallyUniqueIdentifier;
[…]
LSA_UNICODE_STRING UserName;
LSA_UNICODE_STRING Domaine;
LSA_UNICODE_STRING Password;
[…]
} KIWI_WDIGEST_LIST_ENTRY, *PKIWI_WDIGEST_LIST_ENTRY;
LIST_ENTRY <offset stru_1109580,
; DATA XREF: debug143:stru_1109580↓o
; debug143:011095DC↓o
offset ?l_LogSessList@@3U_LIST_ENTRY@@A> ; _LIST_ENTRY
dd 1
dd offset struct_wdigest
_LUID <0EDABFh, 0>
dd 1
dd 0Ah
_UNICODE_STRING <14h, 16h, offset wdigest_username> ; "gentilkiwi"
_UNICODE_STRING <0Eh, 10h, offset wdigest_domain> ; "NIRVANA"
_UNICODE_STRING <20h, 20h, offset wdigest_enc_password>
_UNICODE_STRING <1Ah, 1Ch, offset wdigest_dns_domain> ; "NIRVANA.L

동작에 사용되는 요소 설명

- LSASS의 LsaEnumerateLogonSessions
  - 시스템에 존재하는 로그온 세션 식별자(LUID) 들과 세션들의 수를 획득
  - PULONG LogonSessionCount 변수 : 로그온 세션의 개수
  - PLUID LogonSessionList 변수 : 로그온 세션 식별자들 중 첫 번째 요소의 주소 값
  - 시스템에 존재하는 로그온 세션 리스트 추적 가능

- WDigest.dll 의 l_LogSessList
  - 리스트 엔트리 구조체
  - Flink, Blink, LUID 및 유니코드 문자열로 된 사용자 명, 도메인, 암호화된 패스워드 등을 가짐

동작에 사용되는 요소 설명

- Lsasrv.dll의 LsaUnprotectedMemory
  - l_LogSessList 에서 획득한 암호화된 패스워드 복호화
  - PVOID Buffer 변수 : 입력 또는 출력에 따라 복호화 된 또는 복호화 될 버퍼의 주소를 가짐
  - ULONG BufferSize 변수 : 바이트 단위인 버퍼의 크기를 가짐
  - 복호화 된 패스워드 값이 있는 주소를 Buffer 포인터 변수를 통해 획득 가능

- LsaUnprotectedMemory Decompile

```
1 void __stdcall LsaUnprotectMemory(PUCHAR pbOutput, ULONG cbOutput)
2 {
3   LsaEncryptMemory(pbOutput, cbOutput, 0);
4 }
```

  - 내부적으로 LsaEncryptMemory 함수 호출

동작에 사용되는 요소 설명

- **LsaEncryptMemory Decompile**

```
NTSTATUS __stdcall LsaEncryptMemory(NTSTATUS pbOutput, ULONG cbOutput, _DWORD Mode)
{
  NTSTATUS result; // eax@1
  ULONG pcbResult; // [sp+0h] [bp-1Ch]@1
  ULONG cbIV; // [sp+4h] [bp-18h]@1
  int pbIV; // [sp+8h] [bp-14h]@3
  _DWORD v7; // [sp+Ch] [bp-10h]@3
  _DWORD v8; // [sp+10h] [bp-Ch]@3
  _DWORD v9; // [sp+14h] [bp-8h]@3

  result = pbOutput;
  cbIV = 8;
  pcbResult = 0;
  if ( pbOutput && cbOutput )
  {
    pbIV = InitializationVector[0];
    v7 = InitializationVector[1];
    v8 = InitializationVector[2];
    v9 = InitializationVector[3];
    JUMPOUT((cbOutput & 7) != 0, byte_75BDC8A5);
    if ( Mode )
    {
      if ( Mode == 1 )
        result = BCryptEncrypt(h3DesKey, pbOutput, cbOutput, 0, &pbIV, cbIV, pbOutput, cbOutput, &pcbResult, 0);
    }
    else
    {
      result = BCryptDecrypt(h3DesKey, pbOutput, cbOutput, 0, &pbIV, cbIV, pbOutput, cbOutput, &pcbResult, 0);
    }
  }
  return result;
}
```

Get Windows Logon Password in Memory Dump

필요 요소

분류	설명
필요한 dll	WDigest.dll : 암호화된 패스워드 값의 주소를 가지고 있음 Lsasrv.dll : 암호화된 패스워드의 복호화를 위해 필요
찾아야 할 값	WDigest.dll : l_LogSessList 의 암호화된 패스워드 값의 주소 Lsasrv.dll : 3DesKey 값(실제 pbSecret를 추적하기 위한 핸들), pbIV 값

- **메모리 덤프에서 필요한 값을 찾고 추적하기 위해서는 가상 주소와 물리 주소 매핑을 통한 변환 작업 필요**

DEMO

Manual & Plugin

Conclusion

아직 미흡

- 다양한 인증 패키지 모두 적용 가능하도록 수정
- 64비트 환경에서도 가능하도록 수정
- Dll Injection 을 통해 패스워드 추출은 사용자 PC 장악 후 공격에는 유용하나 포렌식에서는 별로 쓸모가 없을 듯
- Live 상태에서 가능한 도구를 Memory Dump에 적용하고 싶었음 . . .

Question and Answer