in2al5d p3in4er is Almost Completely Undetectable

blog.morphisec.com/in2al5d-p3in4er

Arnold Osipov and Michael Dereviashkin

What Makes Invalid Printer Loader so Stealthy?

Posted by Arnold Osipov and Michael Dereviashkin on April 18, 2023

- Tweet
-

The Aurora stealer is a notorious Golang-based information stealer that's been gaining popularity from the end of 2022 through the first quarter of 2023. The Morphisec Threat Labs team has been tracing its activities using our prevention telemetry along with dark-web activities.

In this blog post however, we aren't going to cover Aurora, but a sometimes overlooked, extremely critical component of the attack delivery chain. The component that makes Aurora's delivery stealthy and dangerous is a highly evasive loader we named "in2al5d p3in4er."

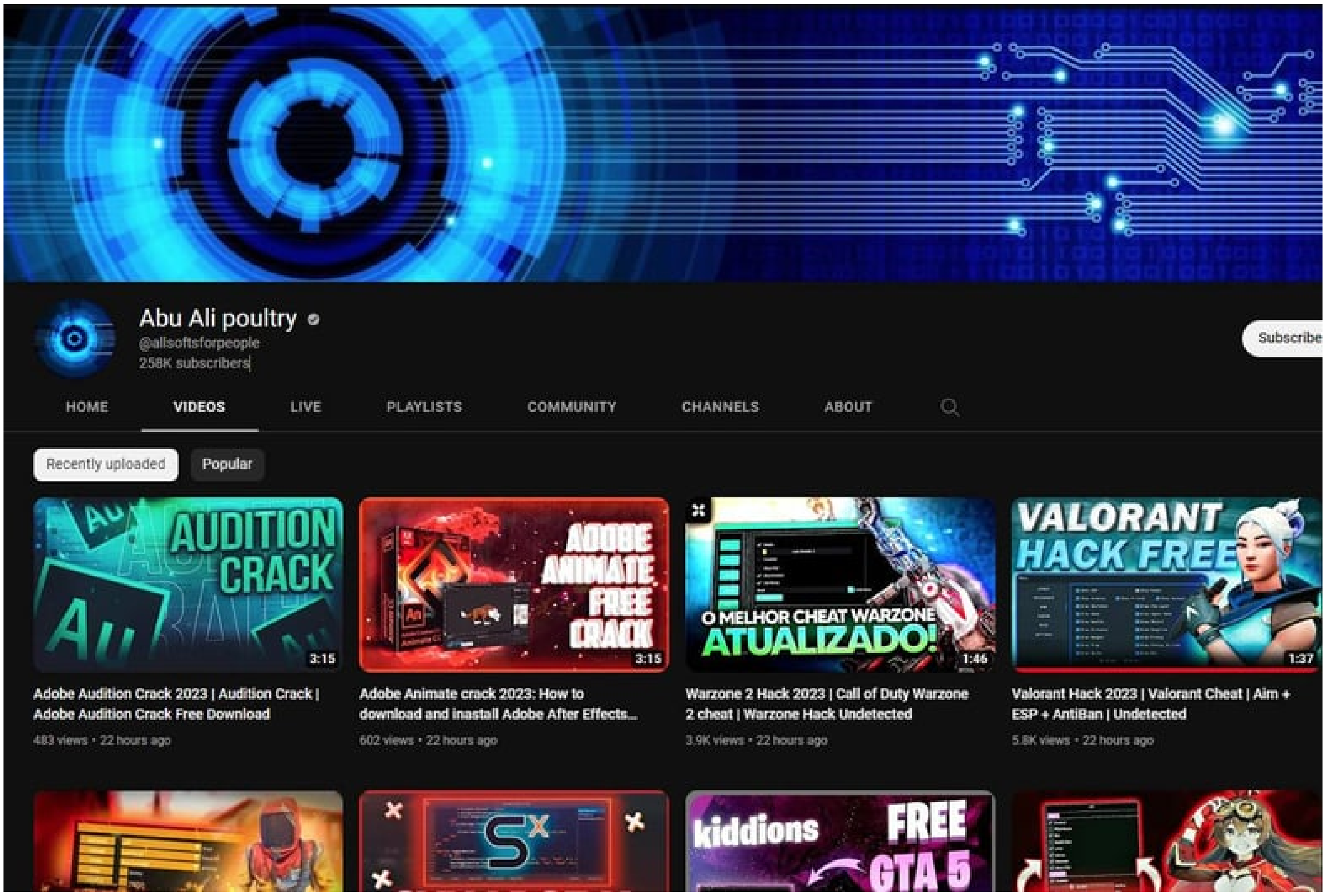
The in2al5d p3in4er loader is compiled with Embarcadero RAD Studio and targets endpoint workstations using advanced anti-VM (virtual machine) technique we describe in detail in this post.

We also cover new techniques with significant negative outcomes that represent a concerning change in the landscape, especially in the new era of ChatGPT.

Delivery

Threat actors always find innovative ways to spread malware and access sensitive information. One growing trend is using YouTube as a malware distribution channel. Hackers take over popular YouTube accounts and post videos with links to malicious websites or downloads. To increase video visibility, they use search engine optimization (SEO) tags to make a video rank higher in search results.

Stealing YouTube accounts is a lucrative business for cybercriminals. Many underground forums and marketplaces offer these services for a fee. The threat actor in this post appears to be using this method as a service, including the websites the videos redirect to.



Adobe Audition Crack 2023 | Audition Crack | Adobe Audition Crack Free Download
483 views • 22 hours ago

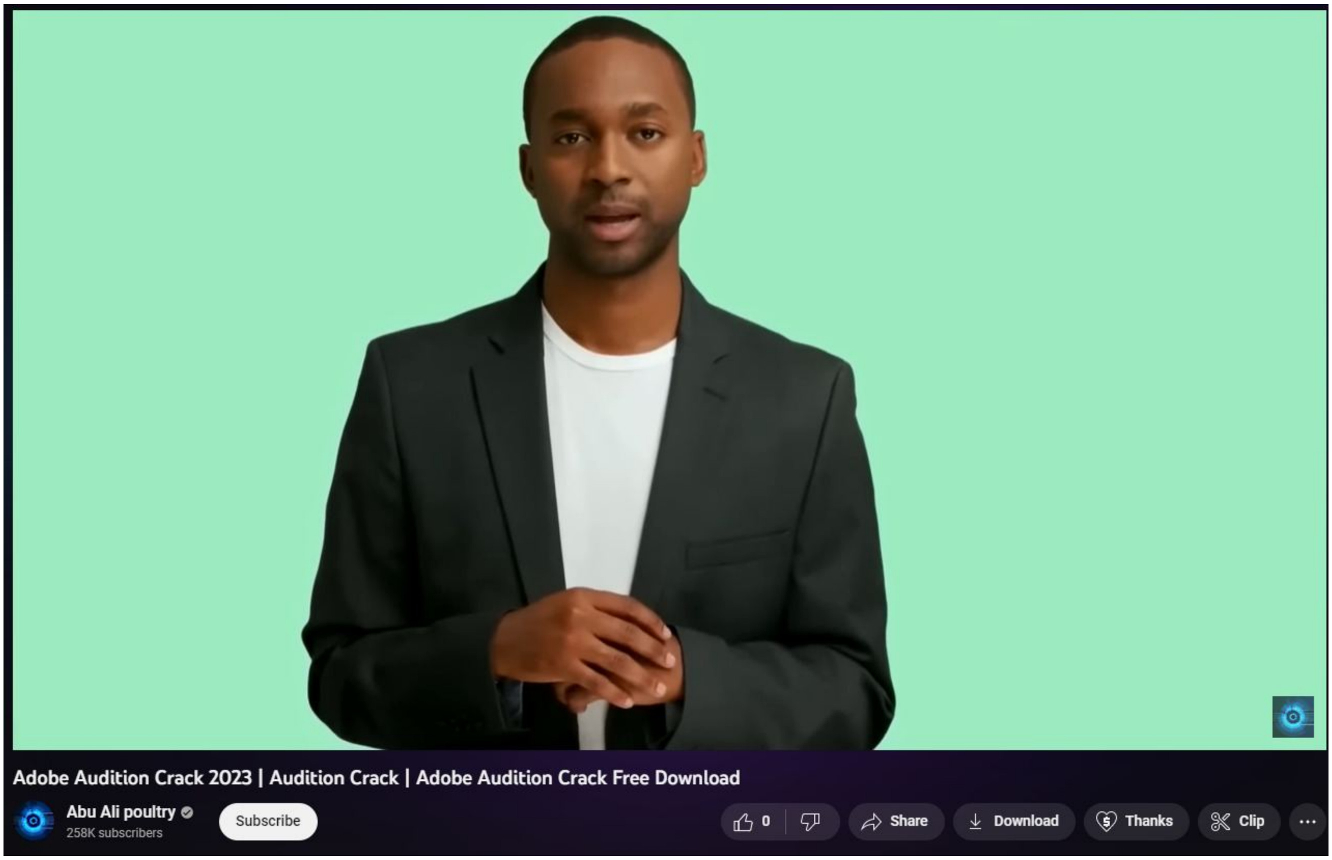
Adobe Animate crack 2023: How to download and inastall Adobe After Effects...
602 views • 22 hours ago

Warzone 2 Hack 2023 | Call of Duty Warzone 2 cheat | Warzone Hack Undetected
3.9K views • 22 hours ago

Valorant Hack 2023 | Valorant Cheat | Aim + ESP + AntiBan | Undetected
5.8K views • 22 hours ago

Compromised YouTube channel

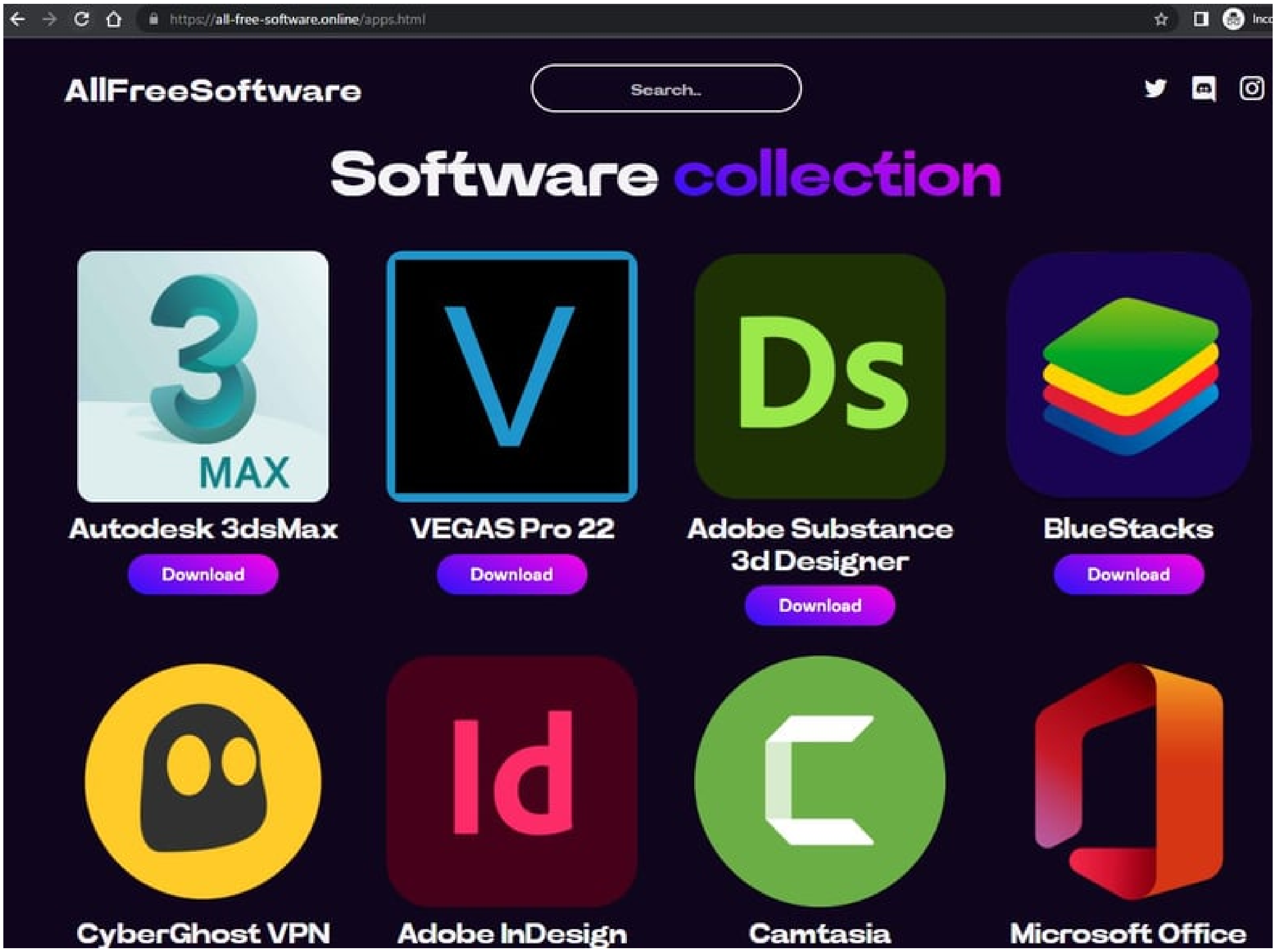
The service uses artificial intelligence (AI) to generate videos. This streamlines the process of creating convincing budget backed content. And it allows threat actors to automate diverse lure creation, such as fake software that tricks users into clicking on links.



Adobe Audition Crack 2023 | Audition Crack | Adobe Audition Crack Free Download

Example video with download instructions

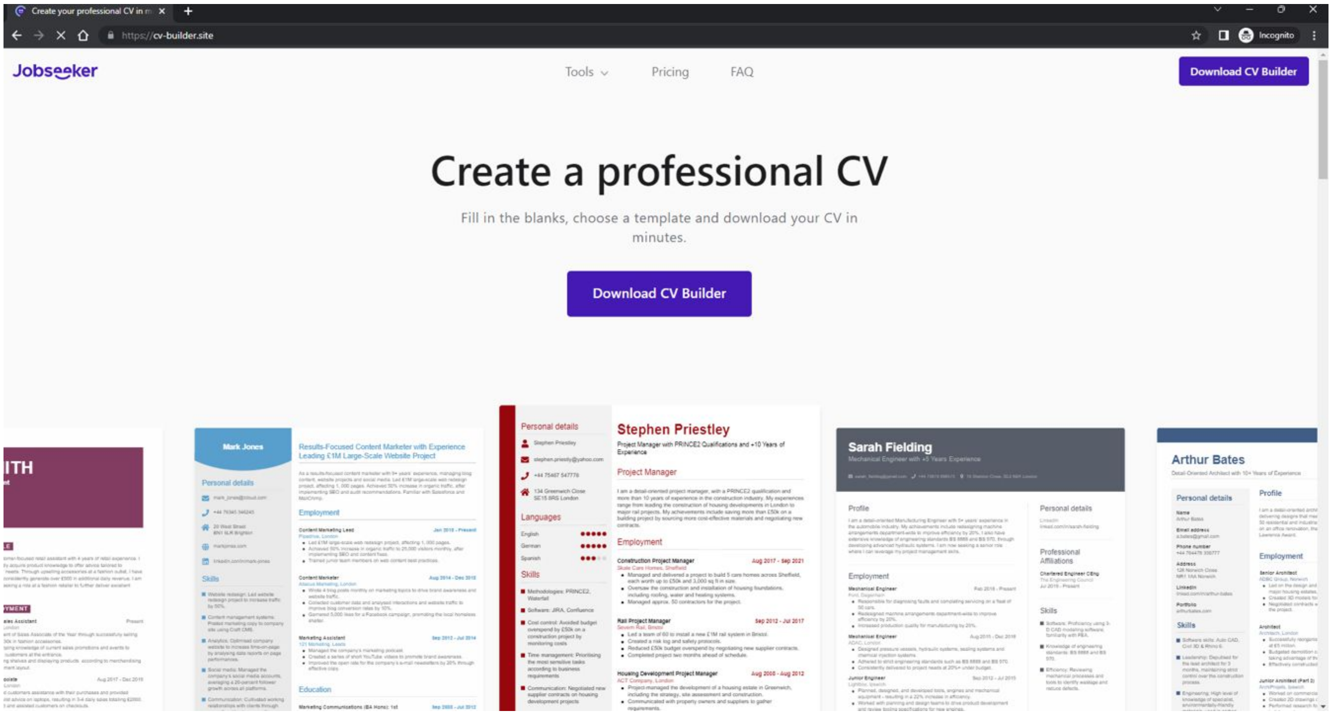
The above video redirects to different websites that look legitimate.



Fake websites redirected by the YouTube video

The service the threat actors are using enables the creation of decoy websites that look identical to the original websites. These fake websites use similar URLs, logos, and branding to convincingly appear legitimate. Once a user visits one of these sites, they're enticed to download an application containing malware or lured to enter sensitive/personal information into the decoy generated website. These websites use geographical targeting to deliver content based on the visitor's geo-location.

The website below is a clone of jobseeker.com



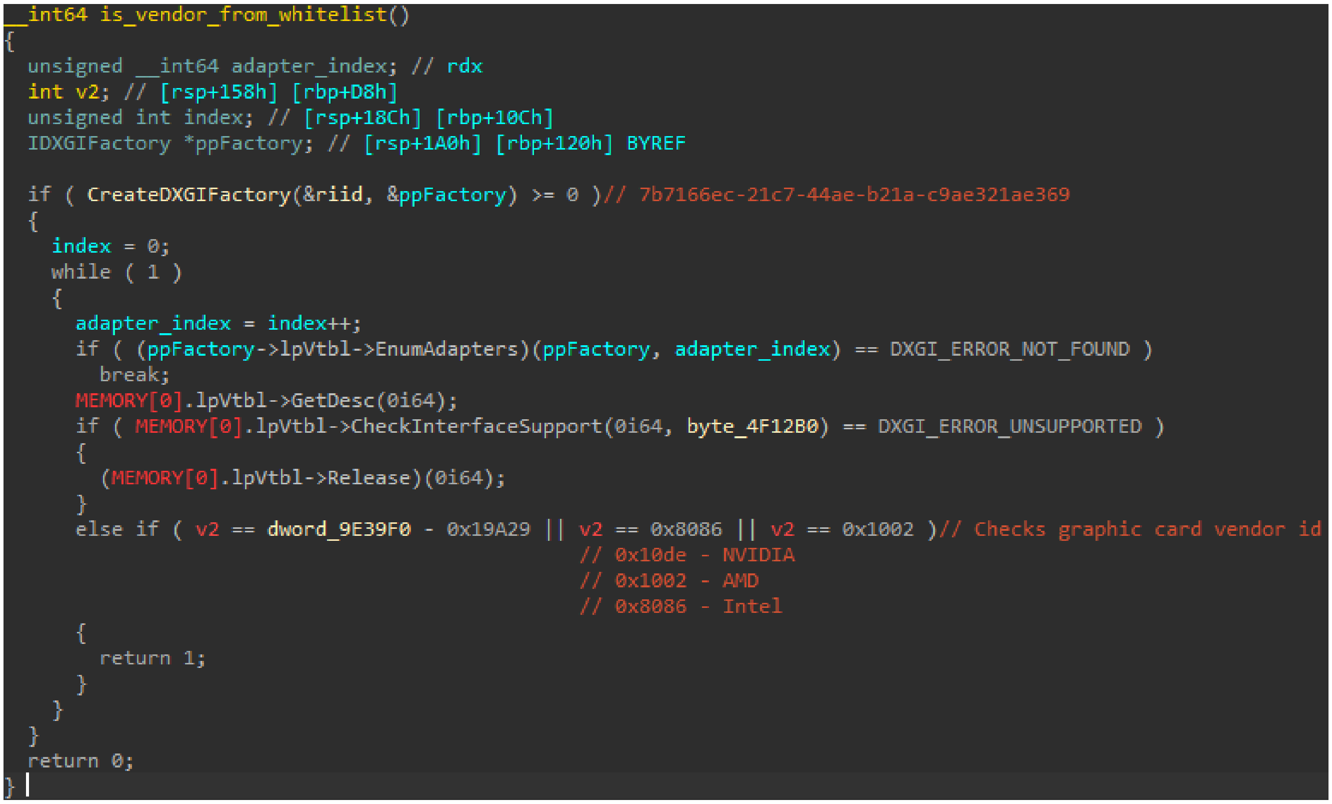
Create a professional CV

Fill in the blanks, choose a template and download your CV in minutes.

Decoy website for jobseeker.com

Technical Analysis

This loader uses a surprisingly simple yet highly effective evasion technique. It leverages the usage of CreateDXGIFactory function of dxgi.dll library to query the vendor ID of the graphics card installed on a system. The ID is then compared against a whitelist of vendor IDs; specifically, the 0x10de (NVIDIA), 0x1002 (AMD), and 0x8086 (Intel) graphics cards. If the vendor ID doesn't match the whitelisted values, the loader makes it appear as a benign application by terminating itself.

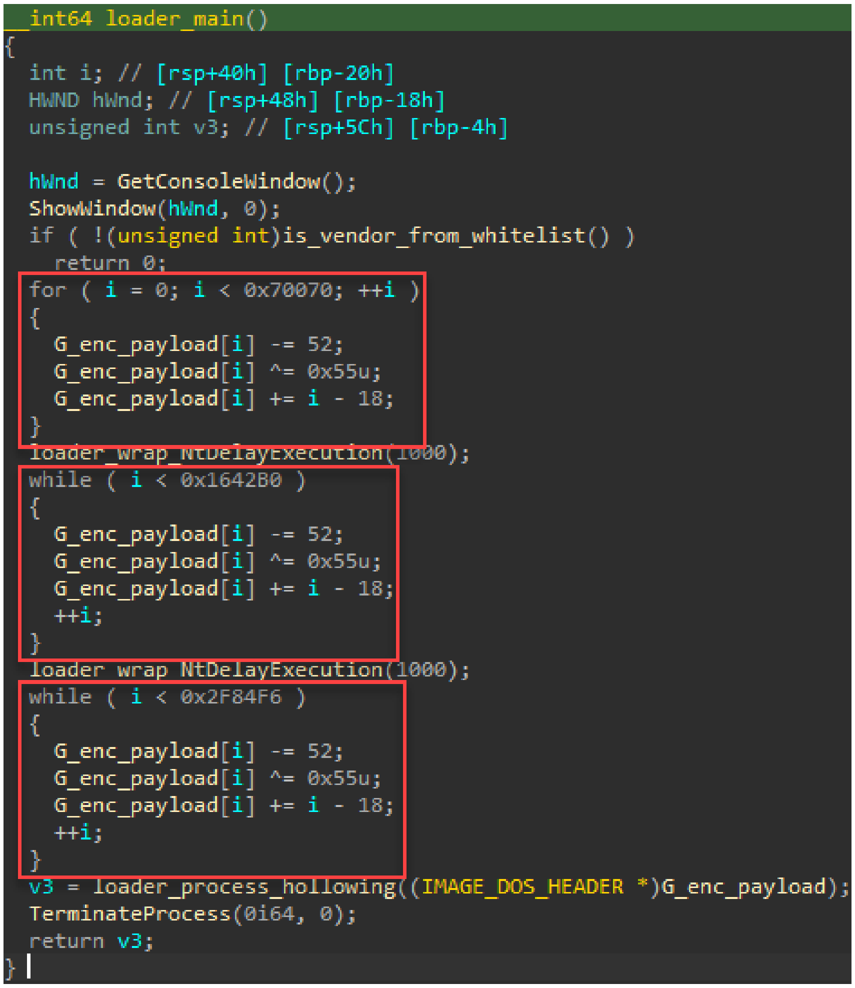


```
__int64 is_vendor_from_whitelist()
{
  unsigned __int64 adapter_index; // rdx
  int v2; // [rsp+158h] [rbp+D8h]
  unsigned int index; // [rsp+18Ch] [rbp+10Ch]
  IDXGIFactory *ppFactory; // [rsp+1A0h] [rbp+120h] BYREF

  if ( CreateDXGIFactory(&riid, &ppFactory) >= 0 )// 7b7166ec-21c7-44ae-b21a-c9ae321ae369
  {
    index = 0;
    while ( 1 )
    {
      adapter_index = index++;
      if ( (ppFactory->lpVtbl->EnumAdapters)(ppFactory, adapter_index) == DXGI_ERROR_NOT_FOUND )
        break;
      MEMORY[0].lpVtbl->GetDesc(0i64);
      if ( MEMORY[0].lpVtbl->CheckInterfaceSupport(0i64, byte_4F12B0) == DXGI_ERROR_UNSUPPORTED )
      {
        (MEMORY[0].lpVtbl->Release)(0i64);
      }
      else if ( v2 == dword_9E39F0 - 0x19A29 || v2 == 0x8086 || v2 == 0x1002 )// Checks graphic card vendor id
                                                // 0x10de - NVIDIA
                                                // 0x1002 - AMD
                                                // 0x8086 - Intel
      {
        return 1;
      }
    }
  }
  return 0;
}
```

The anti-VM function checks the graphics vendor ID

After checking the vendor IDs, the loader decrypts the final payload in separate chunks and injects it into `sihost.exe` using a process hollowing technique.

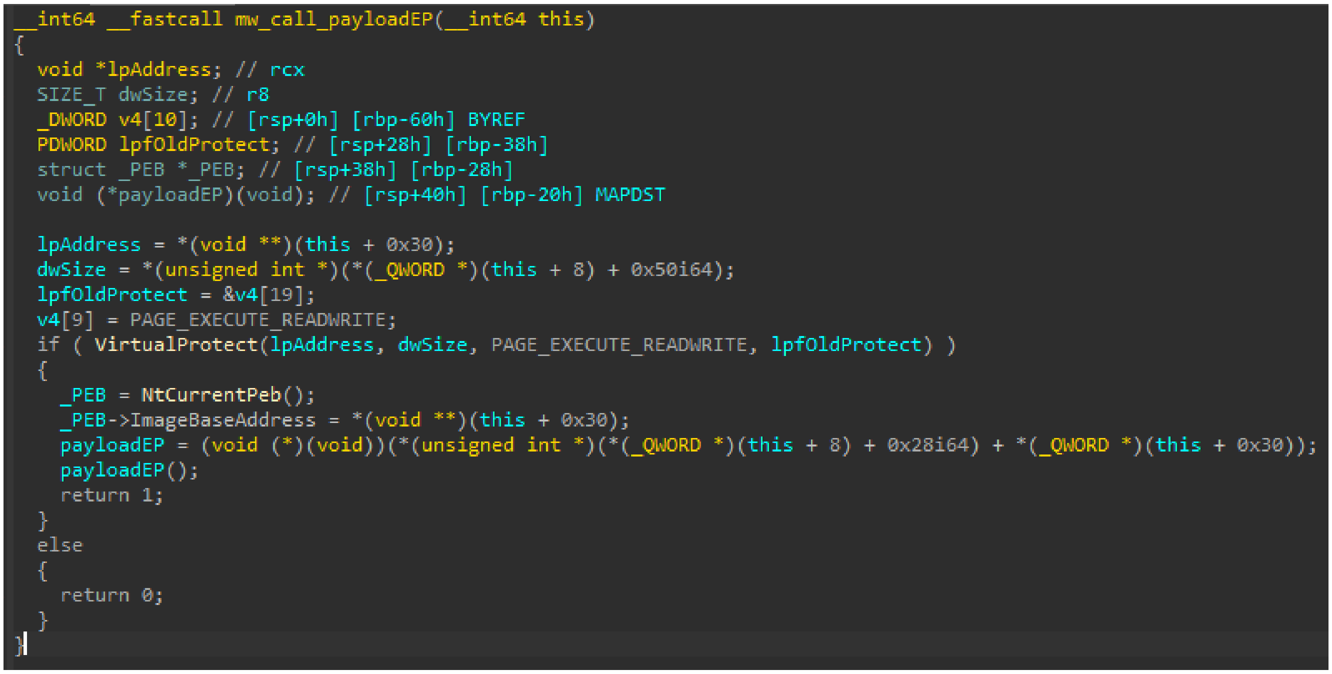


```
__int64 loader_main()
{
  int i; // [rsp+40h] [rbp-20h]
  HWND hWnd; // [rsp+48h] [rbp-18h]
  unsigned int v3; // [rsp+5Ch] [rbp-4h]

  hWnd = GetConsoleWindow();
  ShowWindow(hWnd, 0);
  if ( !(unsigned int)is_vendor_from_whitelist() )
    return 0;
  for ( i = 0; i < 0x70070; ++i )
  {
    G_enc_payload[i] -= 52;
    G_enc_payload[i] ^= 0x55u;
    G_enc_payload[i] += i - 18;
  }
  loader_wrap_NtDelayExecution(1000);
  while ( i < 0x1642B0 )
  {
    G_enc_payload[i] -= 52;
    G_enc_payload[i] ^= 0x55u;
    G_enc_payload[i] += i - 18;
    ++i;
  }
  loader_wrap_NtDelayExecution(1000);
  while ( i < 0x2F84F6 )
  {
    G_enc_payload[i] -= 52;
    G_enc_payload[i] ^= 0x55u;
    G_enc_payload[i] += i - 18;
    ++i;
  }
  v3 = loader_process_hollowing((IMAGE_DOS_HEADER *)G_enc_payload);
  TerminateProcess(0i64, 0);
  return v3;
}
```

Payload decryption routine

Some samples of the loader do not use the process hollowing technique. Instead, they allocate memory to write the decrypted payload into the allocated memory, and then make a call to the payload entry point (EP).

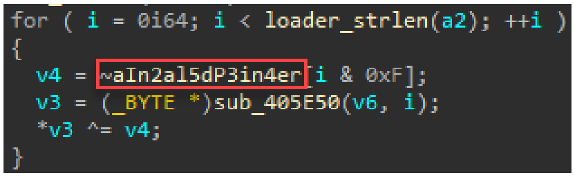


```
__int64 __fastcall mw_call_payloadEP(__int64 this)
{
  void *lpAddress; // rcx
  SIZE_T dwSize; // r8
  _DWORD v4[10]; // [rsp+0h] [rbp-60h] BYREF
  PDWORD lpfOldProtect; // [rsp+28h] [rbp-38h]
  struct _PEB *_PEB; // [rsp+38h] [rbp-28h]
  void (*payloadEP)(void); // [rsp+40h] [rbp-20h] MAPDST

  lpAddress = *(void **)(this + 0x30);
  dwSize = *(unsigned int *)(*(_QWORD *)(this + 8) + 0x50i64);
  lpfOldProtect = &v4[19];
  v4[9] = PAGE_EXECUTE_READWRITE;
  if ( VirtualProtect(lpAddress, dwSize, PAGE_EXECUTE_READWRITE, lpfOldProtect) )
  {
    _PEB = NtCurrentPeb();
    _PEB->ImageBaseAddress = *(void **)(this + 0x30);
    payloadEP = (void (*)(void))(*(unsigned int *)(*(_QWORD *)(this + 8) + 0x28i64) + *(_QWORD *)(this + 0x30));
    payloadEP();
    return 1;
  }
  else
  {
    return 0;
  }
}
```

Payload injection

During the injection process, all loader samples resolve the necessary Win APIs dynamically and decrypt these names using a XOR key: "**in2al5d p3in4er**" (invalid printer), as illustrated in the figure below.

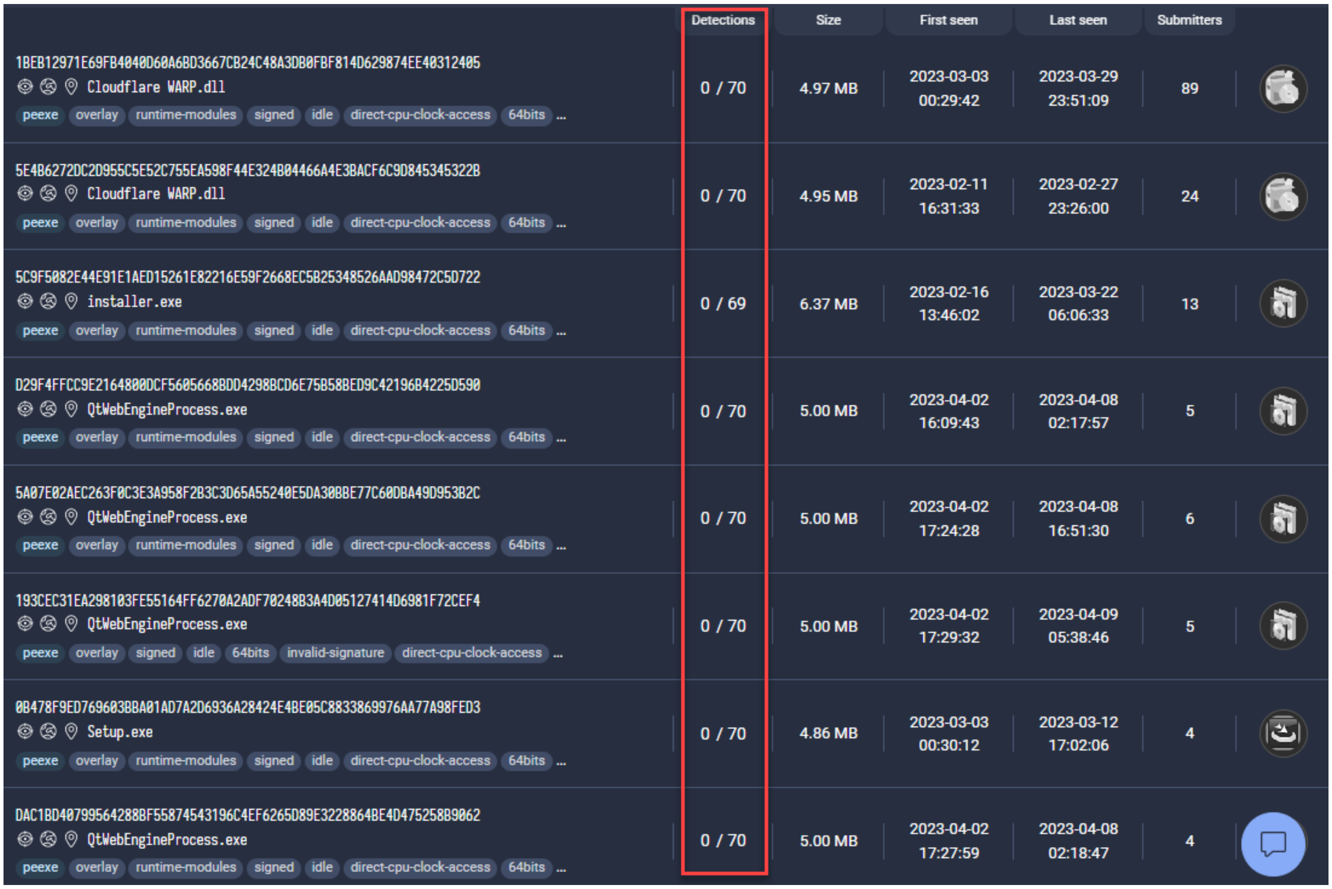


```
for ( i = 0i64; i < loader_strlen(a2); ++i )
{
  v4 = ~aIn2al5dP3in4er[i & 0xF];
  v3 = (_BYTE *)sub_405E50(v6, i);
  *v3 ^= v4;
}
```

String decryption routine using the "invalid printer" key

Another element helping the low detection rate on VirusTotal is the threat actor using Embarcadero RAD Studio to generate executables. Embarcadero RAD Studio is an integrated development environment (IDE) for building software applications that runs on various platforms and operating systems.

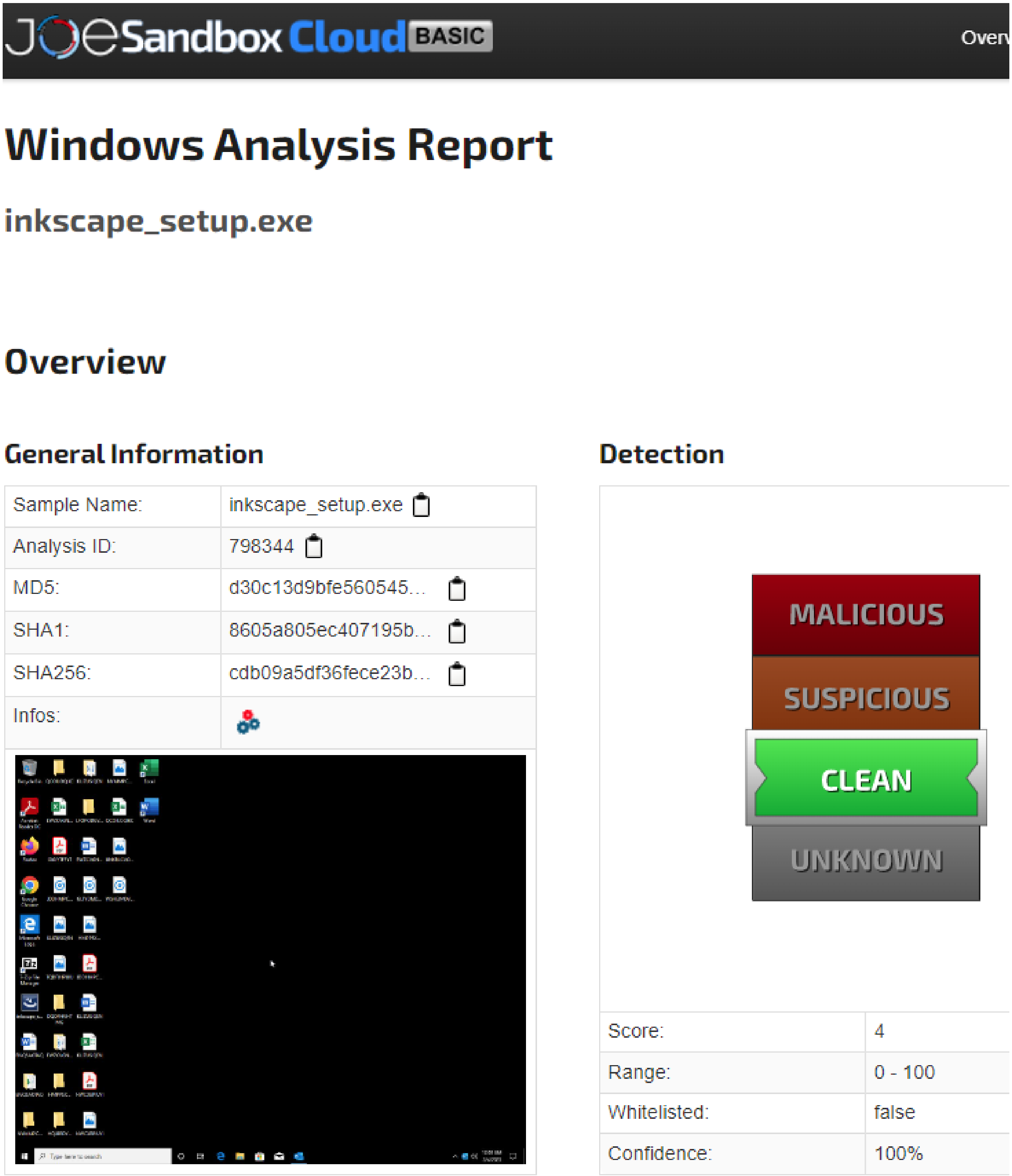
Examining samples from our investigation, we found the threat actor is compiling executables using several options from the Embarcadero IDE. Those with the lowest detection rate on VirusTotal are compiled using “BCC64.exe”, a new Clang based C++ compiler from Embarcadero. This compiler is based on a different code base than a “Standard Library” (Dinkumware) or “Runtime Library” (compiler-rt) and generates optimized code which changes the entry point and execution flow. This breaks security vendors’ indicators, such as signatures composed from “malicious/suspicious code block.”



	Detections	Size	First seen	Last seen	Submitters
1BEB12971E69FB4040D60A6BD3667CB24C48A3DB0FBF814D629874EE40312405 Cloudflare WARP.dll peexe overlay runtime-modules signed idle direct-cpu-clock-access 64bits ...	0 / 70	4.97 MB	2023-03-03 00:29:42	2023-03-29 23:51:09	89
5E4B6272DC2D955C5E52C755EA598F44E324B04466A4E3BACF6C9D845345322B Cloudflare WARP.dll peexe overlay runtime-modules signed idle direct-cpu-clock-access 64bits ...	0 / 70	4.95 MB	2023-02-11 16:31:33	2023-02-27 23:26:00	24
5C9F5082E44E91E1AED15261E82216E59F2668EC5B25348526AAD98472C5D722 installer.exe peexe overlay runtime-modules signed idle direct-cpu-clock-access 64bits ...	0 / 69	6.37 MB	2023-02-16 13:46:02	2023-03-22 06:06:33	13
D29F4FFCC9E2164800DCF5605668BDD4298BCD6E75B58BED9C42196B4225D590 QtWebEngineProcess.exe peexe overlay runtime-modules signed idle direct-cpu-clock-access 64bits ...	0 / 70	5.00 MB	2023-04-02 16:09:43	2023-04-08 02:17:57	5
5A07E02AEC263F0C3E3A958F2B3C3D65A55240E5DA30BBE77C60DBA49D953B2C QtWebEngineProcess.exe peexe overlay runtime-modules signed idle direct-cpu-clock-access 64bits ...	0 / 70	5.00 MB	2023-04-02 17:24:28	2023-04-08 16:51:30	6
193CEC31EA298103FE55164FF6270A2ADF70248B3A4D05127414D6981F72CEF4 QtWebEngineProcess.exe peexe overlay signed idle 64bits invalid-signature direct-cpu-clock-access ...	0 / 70	5.00 MB	2023-04-02 17:29:32	2023-04-09 05:38:46	5
0B478F9ED769603BBA01AD7A2D6936A28424E4BE05C8833869976AA77A98FED3 Setup.exe peexe overlay runtime-modules signed idle direct-cpu-clock-access 64bits ...	0 / 70	4.86 MB	2023-03-03 00:30:12	2023-03-12 17:02:06	4
DAC1BD40799564288BF55874543196C4EF6265D89E3228864BE4D475258B9062 QtWebEngineProcess.exe peexe overlay runtime-modules signed idle direct-cpu-clock-access 64bits ...	0 / 70	5.00 MB	2023-04-02 17:27:59	2023-04-08 02:18:47	4

A zero detection rate on VirusTotal

The loader is particularly successful in evading sandboxes and virtual machines. As seen in the image below, the loader has evaded execution by multiple sandboxes. This is a testament to the effectiveness of the loader's simplicity, which leverages a specific aspect of system configuration to achieve exceptional stealth.



Windows Analysis Report

inkscape_setup.exe

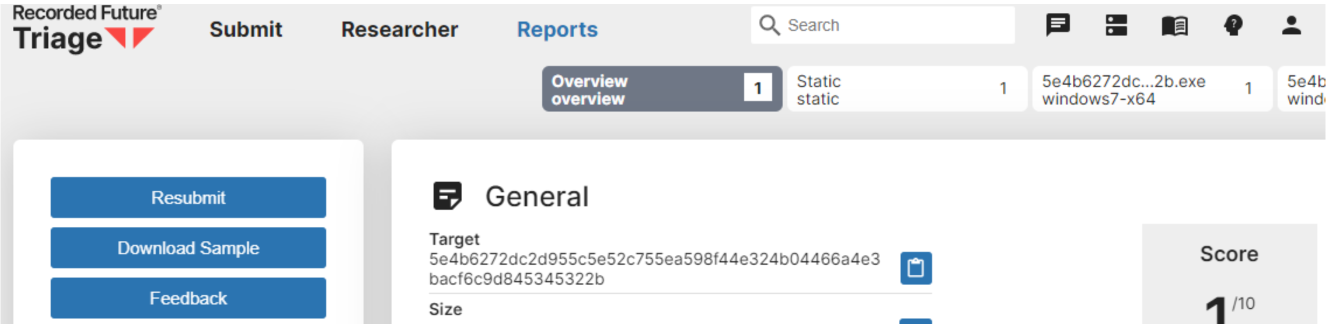
Overview

General Information

Sample Name:	inkscape_setup.exe
Analysis ID:	798344
MD5:	d30c13d9bfe560545...
SHA1:	8605a805ec407195b...
SHA256:	cdb09a5df36fece23b...
Infos:	

Detection

Score:	4
Range:	0 - 100
Whitelisted:	false
Confidence:	100%

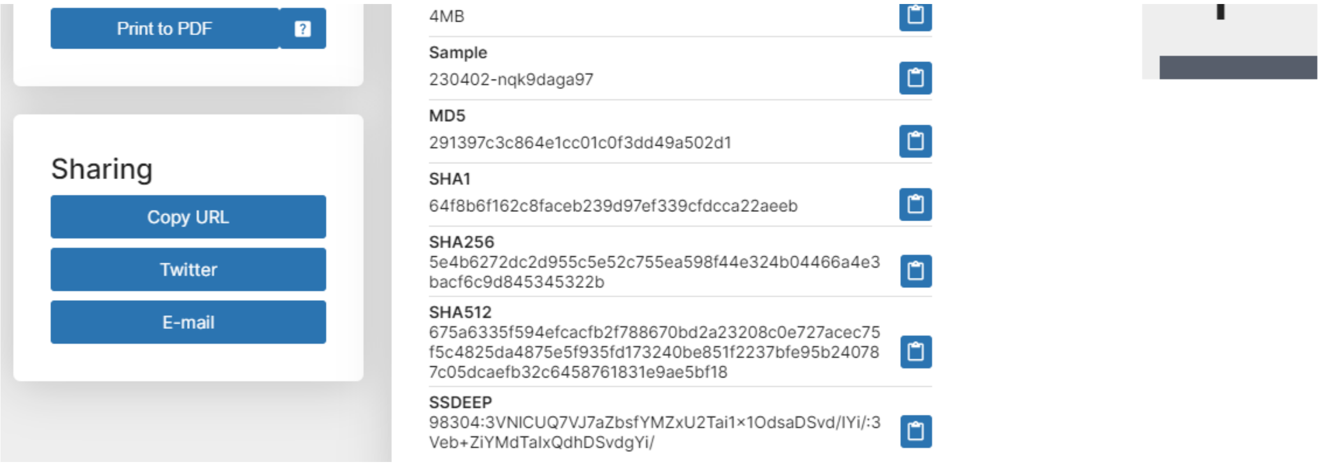


General

Target
5e4b6272dc2d955c5e52c755ea598f44e324b04466a4e3bacf6c9d845345322b

Size

Score
1 /10



4MB

Sample
230402-nqk9daga97

MD5
291397c3c864e1cc01c0f3dd49a502d1

SHA1
64f8b6f162c8faceb239d97ef339cfdcca22aeeb

SHA256
5e4b6272dc2d955c5e52c755ea598f44e324b04466a4e3bacf6c9d845345322b

SHA512
675a6335f594efcacfb2f788670bd2a23208c0e727acec75f5c4825da4875e5f935fd173240be851f2237bfe95b240787c05dcaefb32c6458761831e9ae5bf18

SSDEEP
98304:3VNICUQ7VJ7aZbsfYMZxU2Tai1×1OdsaDSvd/IYi/:3Veb+ZiYMdTalxQdhDSvdgYi/

Example of sandboxes that fail to execute

Note: A similar technique for evading virtual machines was covered in 2016. It checked the graphics' card vendor ID using the DXGI.dll. Although the anti-VM techniques observed in the wild are generally based on blacklisting, this approach uses an inverse check which only searches for graphic cards the target victim should own. This helps to evade sandbox analysis that searches for blacklists of graphic cards.

Defending Against in2al5d p3in4er

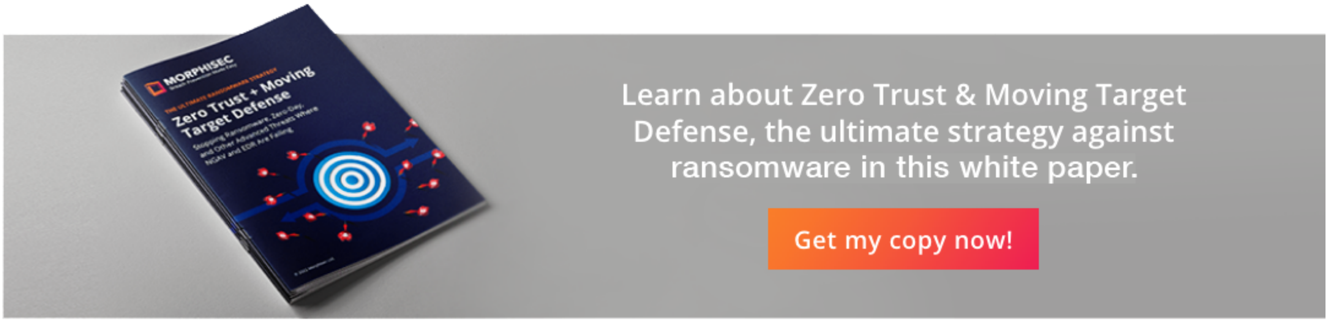
Malware loaders like in2al5d p3in4er are essentially basic delivery. They establish a foothold between an attacker and their compromised target and are typically the first stage of an attack.

The attackers behind in2al5d p3in4er are combining it with widely accessible social engineering tools for a high impact campaign that takes over popular YouTube accounts and directs viewers to convincing looking fake websites. They're then enticed to download malware from a fake website.

One of the first steps to avoid compromise by in2al5d p3in4er is training organization employees how to detect social engineering campaigns: ensuring URLs are legitimate, and not downloading cracked versions of software.

But as the detection ratio of zero on VirusTotal makes clear, reactive, detection-based cybersecurity technologies like those used in NGAV, EPP, and EDR/XDR are barely capable of detecting and stopping in2al5d p3in4er. Stopping this loader requires different technology.

Morphisec's Automated Moving Target Defense (AMTD) technology takes a fundamentally different approach to cybersecurity that isn't detection-based. Instead of waiting for malware to breach a system before detecting it, Morphisec proactively secures runtime memory to prevent any unauthorized code—like in2al5d p3in4er—from executing, regardless of whether a recognizable signature or behavior pattern exists for it. To learn more about Moving Target Defense, read the free white paper: *Zero Trust + Moving Target Defense—The Ultimate Ransomware Strategy*.


MORPHISEC
THE ULTIMATE RANSOMWARE STRATEGY
Zero Trust + Moving Target Defense
Stopping Ransomware, Zero-Day, and Other Advanced Threats Where NGAV and EDR Are Failing
Learn about Zero Trust & Moving Target Defense, the ultimate strategy against ransomware in this white paper.
Get my copy now!

IOCs (Indicators of Compromise)

Malicious/Compromised Websites

- cv-builder[.]site
- siamaster.com[.]mx
- chatgptex[.]us
- allfreesoftware[.]online
- all-free-software[.]online

Hash - Loader	C2 - Aurora

380978251b2c661ff15b2610763770dfa14fb360ad0ca64243e0d5d5893952cb 66383d931f13bcdd07ca6aa50030968e44d8607cf19bdaf70ed4f9ac704ac4d1 cdb09a5df36fece23bc3c9df101fe65724327b827ec43aa9ce0b3b76bdcc3101 adb6808f97191d961687b5f30f35c843686699d70f482f4d7d8d4f41e84faba6 2c540f5220b7ba3cd6efcd2fe8091fc24f8da11be4b1782c4e502261ef48da82 0b478f9ed769603bba01ad7a2d6936a28424e4be05c8833869976aa77a98fed3 bb9a16632ae94bbfae713f56c51dc5d2ff6199ccd051e2285ea90c6dfeb5d4b1 b299898055262e065afd5b0479b2e3a190e314cab3ead3722b2cdbe3534a2681 11cc3e4b8413a1b8c0c0d7193c2f26670d4765f96e797140c809d4a0655f9cb4 c1dafead343c67d203d2fa9050967fea868fc517f8d66b23cc166642d8b7985b e18455804b8a6a008a2b357265802ad35f6441bbcc359ab5bd5df994f201ab36	45.15.156.182:8081
095b9e90e1c9d7d95f362fe381512da60bb31727c068c6dfabe055ab387aed82 737f6c8e1a8ea4a9064a0cfaa4ceed495481d9bb133ed5d6bfeda3a83351af9d a4cab01d61d8c18876d4b53d52de365fb9b512430371fd4217359159f3c507f6 e8cccc9f9b124826c0e43897f0e21124b4b0cd7991f434b0dd7838bed7e361b3 3c3622ab5f449166ad804ee73fde6aedf6934aa960701f7edefca9b5aedfffd8	45.15.156.70:8081

5c9f5082e44e91e1aed15261e82216e59f2668ec5b25348526aad98472c5d722 fa2cfb4b76c38ab4cef4592ed703a8866ffffec67adbe9ea057882b47a4bd7b7 780285087fb35911db189ba92c6c8d251a1f640b3a473e0ce7ea27f59cde492b	45.132.106.77:8081
Df99a29a0fd7f62e3798260b2068c711a5356346b7b0c0477e30643138345fba ffd1f682649507fd850a8faf76b4c3c498dbcbef70bd0202126b91f24d5c1408	199.127.62.3:8081
7ed926820973cd3c14b783109094604369e37cc06cb08a338b856f4e5cf2684c cc79ef40d93c939c43624504461ec5ddf8279624ae88a739e7382c181e18104b	94.142.138.73:8081
1beb12971e69fb4040d60a6bd3667cb24c48a3db0fbf814d629874ee40312405	94.142.138.84:8081
4accbeea02170cb5215fbe937fabae030986f84ee7acc983cc15ce120e073861	94.142.138.218:8081
e827cde3646048c9e09a61eeb45cc37f1d8a20190762c513ddf1e9dc13e4b897	199.247.24.79:8081
5e4b6272dc2d955c5e52c755ea598f44e324b04466a4e3bacf6c9d845345322b	5.34.180.208:8081