

OSINT Reporting Regarding DPRK and TA505 Overlap – One Night in Norfolk

Published: 2019-04-10 · Archived: 2026-04-05 19:59:12 UTC

Yesterday, at SAS2019, BAE Systems [presented findings](#) related to DPRK SWIFT heist activity that took place in 2018. As part of this research (a leaked video of the presentation is available online), BAE included two key points not previously disclosed in the public domain:

- The existence of a PowerShell backdoor attributable to DPRK, which the researchers dubbed PowerBrace
- A possible overlap between TA505 intrusions and DPRK intrusions, suggesting a possible hand-off between the two groups.

This blog will leave a full analysis of those two points and the supporting context to the people that found them, as it's theirs to share; however, data that may support such conclusions have been available in open source for quite some time.

In early January, VNCert [issued an alert](#) regarding attacks targeting financial institutions, containing a mix of DPRK IOCs (including a keylogger referred to as PSLogger [previously analyzed by this blog](#)), TA505 IOCs (previously [published by 360 TIC](#)), and a handful of PowerShell scripts that are generally identical aside from a handful of configuration changes. Furthermore, the aforementioned keylogger was first uploaded by a submitter (fabd7a52) in Pakistan in December 2018. That same submitter acted as the first uploader for one of the PowerShell samples identified below (b88d4d72fdabfc040ac7fb768bf72dcd), further corroborating a possible link.

Given the multi-sourced reporting overlaps and the additional Pakistan findings mentioned above, this blog assesses that the PowerShell scripts in question likely belong to the same family of DPRK-attributable malware reported by BAE systems.

A listing of selected IOCs is below the fold, alongside a few brief notes (and a script) for *how* to analyze the PowerShell malware.

IOCs from VNCert

TA505:

These contain infrastructure overlaps with reporting from the same month found here:

<https://ti.360.net/blog/articles/excel-4.0-macro-utilized-by-ta505-to-target-financial-institutions-recently-en/>

MD5: 5B7244C47104F169B0840440CDEDE788

MD5: cc29adb5b78300b0f17e566ad461b2c7

MD5: E00499E21F9DCF77FC990400B8B3C2B5

MD5: 53F7BE945D5755BB628DEECB71CDCBF2

MD5: 9c35e9aa9255aa2214d704668b039ef6

- 3) Rename those other functions
- 4) Repeat

An example of decoded data (with variables and functions renamed manually) is below:

```
Function beacon_function
{
    ${19cc06b5f0e7488a95899ad2ad05f73b} = $env:COMPUTERNAME
    ${user_variable} = "Unknown"
    ${version_var} = "Unknown"
    ${os_arc_var_2} = "Unknown"
    ${reg_proxyserver_var} = "NotExist"
    ${reg_proxyenable_var} = ''
    ${becomes_ip} = "Unknown"
    ${c2_str_var} = ''
    ${private:2ad6060be48c4e999e2dd45f4750ae12} = $env:COMPUTERNAME;
    ${unused_var} = @{};
    ${private:wmi_info_var} = $null;
    ${compname_var} = $env:COMPUTERNAME;

    ${mac_ip_pair} = gwm Win32_NetworkAdapterConfiguration | select macaddress,ipaddress
    foreach(${array_item} in ${mac_ip_pair})
    {
        if((${array_item}.macaddress -ne $null) -and (${array_item}.macaddress.Length -ne 0))
        {
            ${becomes_ip} = ${array_item}.ipaddress[0]
            break;
        }
    }
    ${compname_var} = ${compname_var} + -join (({48 .. 57} + {65 .. 90} | Get-Random -Count 8 | % { [char]$_ })
    ${2f301a081eb74ec1b82252b83a7a80d6} = New-Object System.Security.Cryptography.SHA1Managed
    ${be4707b654e44e994115fe275bf0c05} = [System.Text.Encoding]::UTF8.GetBytes(${compname_var});
    ${0ec6f241bc38414e9b2d8a8f0c44401a} = ${2f301a081eb74ec1b82252b83a7a80d6}.ComputeHash(${be4707b654e44e994115fe275bf0c05});
}
```

A portion of the partially decoded and deobfuscated PowerShell backdoor

A script to assist with this is here:

```
import base64
import re

c = open("c:\\users\\[username]\\desktop\\[filename]").readlines()

line_list = []

for line in c:
    #print(line)
    try:
        enc = re.search("(?<=\\$(\\[Text.Encoding\\]::Unicode.GetString\\(\\[Convert\\]::FromBase64String\\(\\).*?(?=\\))",
            print(line)
            print(enc)
            d = ('"' + base64.b64decode(enc) + '"')
            e = (re.sub("\\$(\\[Text.Encoding\\]::Unicode.GetString\\(\\[Convert\\]::FromBase64String\\(\\).*?\\)\\)",
            f = re.sub("\\0", "", e)
            line_list.append(f)

    except:
        line_list.append(line)

with open("c:\\users\\[username]\\desktop\\laz_decoded.ps1", "wt") as t:
    for unit in line_list:
        t.write(unit)
```

Post navigation

Source: <https://norfolkinfosec.com/osint-reporting-on-dprk-and-ta505-overlap/>