Contents

[Download 1](#_Toc192239160)

[Option 1: Download via git 1](#_Toc192239161)

[Option 2: Download manually from git 1](#_Toc192239162)

[PowerShell Execution Policy and File Security 1](#_Toc192239163)

[Import 1](#_Toc192239164)

[Option 1: Import the module to the current session only 1](#_Toc192239165)

[Option 2: Import the module at PowerShell startup 1](#_Toc192239166)

[Option 1: Edit the PowerShell profile via a scipt 1](#_Toc192239167)

[Option 2: Edit the PowerShell profile manually 1](#_Toc192239168)

[Connect to API 1](#_Toc192239169)

[Examples 1](#_Toc192239170)

[Example 1: Start new session and save the url/token pair 1](#_Toc192239171)

[Example 2: Start new session and save the url/token pair 1](#_Toc192239172)

[Example 3: Start new session using on of the the saved url/token pairs 1](#_Toc192239173)

[Example 3: Start new session by entering the exact url 1](#_Toc192239174)

[Example 4: Start a new session automatically with a new powershell session 1](#_Toc192239175)

[List All Zabbix Module Commands 1](#_Toc192239176)

[Get-ZXHost 1](#_Toc192239177)

[Examples 1](#_Toc192239178)

[Example 1: Search for a single host based on an exact Name match. 1](#_Toc192239179)

[Example 2: Search for hosts based on a Name pattern. 1](#_Toc192239180)

[Example 3: Fetch hosts and interfaces 1](#_Toc192239181)

[Example 4: Search for hosts based on an Alias pattern. 1](#_Toc192239182)

[Example 5: Fetch host “items”. 1](#_Toc192239183)

[Example 6: Fetch host “items”, fetch only name and type of each item. 1](#_Toc192239184)

[Example 7: Fetch host “items” with all properties. 1](#_Toc192239185)

[Example 8: Fetch host triggers. 1](#_Toc192239186)

[Get-ZXMaintenance 1](#_Toc192239187)

[Example 1: Get maintenance mode information 1](#_Toc192239188)

[New-ZXTagFilter 1](#_Toc192239189)

[Example 1: Create a list of tags with operators and use it to find hosts that match all the conditions. 1](#_Toc192239190)

[Get-ZXTrigger 1](#_Toc192239191)

[Example 1: Get a list of all triggers on a single host wit all trigger properties 1](#_Toc192239192)

[Example 2: Get a list of triggers on a single host with only specific trigger properties 1](#_Toc192239193)

[Example 3: Get a list of trigger properties 1](#_Toc192239194)

[Get-ZXAction 1](#_Toc192239195)

[Common Parameters 1](#_Toc192239196)

[Output 1](#_Toc192239197)

[Example 1: Select specific properties of an object 1](#_Toc192239198)

[WhatIf 1](#_Toc192239199)

[Example 1: Shows a JSON request without actually making the API call 1](#_Toc192239200)

[CountOutput 1](#_Toc192239201)

[Example 1: Don’t fetch any data, just show count the output. 1](#_Toc192239202)

[ShowJsonRequest 1](#_Toc192239203)

[Example 1: Make the API call and also show the json that was used to make the call. 1](#_Toc192239204)

[Limit 1](#_Toc192239205)

[Example 1: Limit the number of returned result 1](#_Toc192239206)

# Download

## Option 1: Download via git

git clone https://github.com/JanTkacSk/PSZabbix7.git

## Option 2: Download manually from git

[https://github.com/JanTkacSk/PSZabbix7.git](https://github.com/JanTkacSk/PSZabbix6.0.git) > **Code** > **Download.zip**

Extract the downloaded file and rename if needed from **PSZabbix7-main** to **PSZAbix7** and put it in a convenient location.

It is strongly recommended to use one of the standard locations which you can get by running the following command,

Ideally the one in the Users folder. If the directory does not exist, you can create it.

$env:psmodulepath -split ";"

C:\Users\testuser\Documents\WindowsPowerShell\Modules

C:\Program Files\WindowsPowerShell\Modules

C:\Windows\system32\WindowsPowerShell\v1.0\Modules

C:\Program Files\Citrix\PowerShellModules\

# PowerShell Execution Policy and File Security

Set the execution policy to RemoteSigned to be able to run local scripts.

Set-ExecutionPolicy RemoteSigned -Scope CurrentUser -Force

If you downloaded the file from the internet, you may have to unblock the files, because PowerShell will not allow you to run a script downloaded from the interned before you unblock it.

Get-ChildItem C:\YourPath\PSZabbix7 -Recurse | Unblock-File

# Import

## Option 1: Import the module to the current session only

The module will have to be re-imported every time you open PowerShell. Adjust the path based on the location of your package.

Import the module

Import-Module C:\YourPath\PSZabbix7\PSZabbix7.psm1

## Option 2: Import the module at PowerShell startup

### Option 1: Edit the PowerShell profile via a scipt

Open PowerShell or PowerShell\_ise or a PowerShell terminal in VSCode and run the following command. Adjust the path based on the location of your package.

. C:\Users\YourUser\PSZabbix7\ZX.UpdatePSProfile.ps1

### Option 2: Edit the PowerShell profile manually

Open PowerShell or PowerShell\_ise or Terminal in VSCode

You may have to repeat this process for different PowerShell versions, classic PowerShell console, PowerShell ISE , VS code….

Open the profile script by running the following command. If it does not work, check the next line.

notepad $Profile

If the profile does not exist and if the notepad does not create it for you automatically, run this command:

New-Item -ItemType File $PROFILE -Force

Add the following line into the file and save it. Adjust the path based on the location of your package.

Import-Module C:\Users\YourUser\PSZabbix7\PSZabbix7.psm1

# Connect to API

Use New-ZXTokenSession to set your API token and URL. The token and the url can be saved in a secure string in your appdata directory. It can be loaded automatically when you open PowerShell by editing your PowerShell profile.

Use New-ZXLogonSession to load your credentials in a similar way. Use Stop-ZXSession to terminate your session when you are finished.

The session id is automatically stored in your profile, If you don not log off, the session will remain in your profile and the script will attempt to use the same session next time.

New-ZXTokenSession

## Examples

### Example 1: Start new session and save the url/token pair

New-ZXTokenSession -Save

### Example 2: Start new session and save the url/token pair

You will be prompted to fill in the password. The token will be encrypted and saved in your profile folder.

New-ZXTokenSession -Url https://YourZabbixAPIURL -Save

### Example 3: Start new session using on of the the saved url/token pairs

New-ZXTokenSession -Load

[0] https://zabbix-prod.local/api\_jsonrpc.php

[1] https://zabbix-test.local/api\_jsonrpc.php

Select the number and press enter: 0

userid : 146

username : john.doe@local

name : Jan

surname : Tkac

url :

autologin : 1

autologout : 0

lang : en\_US

refresh : 30s

theme : dark-theme

attempt\_failed : 0

attempt\_ip :

attempt\_clock : 0

rows\_per\_page : 500

timezone : Europe/Berlin

roleid : 3

type : 3

userip : 195.168.0.4

debug\_mode : 0

gui\_access : 0

### Example 3: Start new session by entering the exact url

This command can be added to PowerShell profile and loaded automatically at PowerShell start.

New-ZXTokenSession -Url https://zabbix-prod.local/api\_jsonrpc.php -Load

### Example 4: Start a new session automatically with a new powershell session

Open the PowerShell profile. PowerShell\_ise and standard powershell console have separate profiles, so you may have to repeat this process.

notepad $Profile

Add the following line into the profile:

New-ZXTokenSession -Url https://YourZabbixAPIURL -Load

# List All Zabbix Module Commands

Get-Command -Module PsZabbix7

#or

Get-Command \*ZX\*

CommandType Name Version Source

----------- ---- ------- ------

Function Add-ZXHostGroup 0.0 PSZabbix7

Function Add-ZXHostNameSuffix 0.0 PSZabbix7

Function Add-ZXHostNameSuffixX 0.0 PSZabbix7

Function Get-ZXAction 0.0 PSZabbix7

Function Get-ZXAlert 0.0 PSZabbix7

Function Get-ZXApiVersion 0.0 PSZabbix7

Function Get-ZXAuditLog 0.0 PSZabbix7

Function Get-ZXDiscoveryRule 0.0 PSZabbix7

Function Get-ZXEvent 0.0 PSZabbix7

Function Get-ZXHistory 0.0 PSZabbix7

Function Get-ZXHost 0.0 PSZabbix7

Function Get-ZXHostGroup 0.0 PSZabbix7

Function Get-ZXHostInterface 0.0 PSZabbix7

Function Get-ZXItem 0.0 PSZabbix7

Function Get-ZXItemPrototype 0.0 PSZabbix7

Function Get-ZXMaintenance 0.0 PSZabbix7

Function Get-ZXProblem 0.0 PSZabbix7

Function Get-ZXProxy 0.0 PSZabbix7

Function Get-ZXService 0.0 PSZabbix7

Function GEt-ZXSession 0.0 PSZabbix7

Function Get-ZXTemplate 0.0 PSZabbix7

Function Get-ZXTrigger 0.0 PSZabbix7

Function New-ZXHost 0.0 PSZabbix7

Function New-ZXLogonSession 0.0 PSZabbix7

Function New-ZXProblemTagList 0.0 PSZabbix7

Function New-ZXService 0.0 PSZabbix7

Function New-ZXTagFilter 0.0 PSZabbix7

Function New-ZXTagList 0.0 PSZabbix7

Function New-ZXTokenSession 0.0 PSZabbix7

Function Remove-ZXHost 0.0 PSZabbix7

Function Remove-ZXHostGroup 0.0 PSZabbix7

Function Remove-ZXHostNameSuffixX 0.0 PSZabbix7

Function Set-ZXHostName 0.0 PSZabbix7

Function Set-ZXHostStatus 0.0 PSZabbix7

Function Set-ZXHostStatusX 0.0 PSZabbix7

Function Stop-ZXSession 0.0 PSZabbix7

Function Update-ZXHostTagList 0.0 PSZabbix7

Function Update-ZXHostTagListX 0.0 PSZabbix7

Function Update-ZXHostTemplateList 0.0 PSZabbix7

Function Update-ZXService 0.0 PSZabbix7

# Get-ZXHost

## Examples

### Example 1: Search for a single host based on an exact Name match.

Return only those results that exactly match the given host name(s). There is no result for “zx\_test\_host\_50” because it does not exist. In Zabbix API, host “**name”** is called “**host**” and “**visible name”** is called “**name**”. In PowerShell module “**name**” is “**Name**” and the “**visible name**” is “**Alias**”

Get-ZXHost -Name zx\_test\_host\_1,zx\_test\_host\_2,zx\_test\_host\_50

hostid : 492558

host : zx\_test\_host\_1

name : zx\_alias\_1

status : 1

proxy\_hostid : 36710

hostid : 494040

host : zx\_test\_host\_2

name : zx\_alias\_2

status : 1

proxy\_hostid : 36710

### Example 2: Search for hosts based on a Name pattern.

Return results that match the given pattern (case-insensitive). Accepts a host name to search for. If no additional options are given, this will perform a LIKE "%…%" search.

Get-ZXHost -NameSearch zx\_test

hostid : 492558

host : zx\_test\_host\_1

name : zx\_alias\_1

status : 1

proxy\_hostid : 36710

hostid : 494040

host : zx\_test\_host\_2

name : zx\_alias\_2

status : 1

proxy\_hostid : 36710

### Example 3: Fetch hosts and interfaces

Get-ZXHost -NameSearch test -IncludeInterfaces -Output host | Select-Object hostid,host,@{n="IPs";e={$\_.interfaces.ip}}

hostid host IPs

------ ---- ----------

476172 vm-win-test-1.test.local

481649 vm-win-test-2.test.local 10.61.185.9

28948 vm-win-test-3.test.local 10.30.0.71

491371 vm-win-test-4.test.local

25244 vm-win-test-5.test.local {192.168.160.28,192.168.160.28}

Get-ZXHost -NameSearch test -IncludeInterfaces -Output host

hostid host interfaces

------ ---- ----------

476172 vm-win-test-1.test.local {}

481649 vm-win-test-2.test.local {@{ip=10.61.185.9; port=161}}

28948 vm-win-test-3.test.local {@{ip=10.30.0.71; port=10050}}

491371 vm-win-test-4.test.local {@{ip=; port=10050}}

25244 vm-win-test-5.test.local {@{ip=192.168.160.28; port=10050}, @{ip=192.168.160.28; port=9005}}

### Example 4: Search for hosts based on an Alias pattern.

Return results that match the given pattern (case-insensitive). Accepts a host name to search for. If no additional options are given, this will perform a LIKE "%…%" search.

Get-ZXHost -AliasSearch some-alias

### Example 5: Fetch host “items”.

By using -IncludeItems you will get a basic set of properties for each item.

Get-ZXHost -NameSearch vm-win -IncludeItems

hostid : 499271

host : vm-win-test-1

name : vm-win-test-1

status : 0

proxy\_hostid : 0

items : {@{itemid=53119623; name=C:: Access; type=0; delay=3m; master\_itemid=0; lastvalue=0}, @{itemid=53119533;

name=caCertificate; type=0; delay=1h; master\_itemid=0; lastvalue=}, @{itemid=53119534; name=caPendingUpdates;

type=7; delay=4h; master\_itemid=0; lastvalue=}, @{itemid=53119625; name=C:: Free space; type=7; delay=3m;

master\_itemid=0; lastvalue=0}…}

### Example 6: Fetch host “items”, fetch only name and type of each item.

Get-ZXHost -NameSearch vm-win -IncludeItems -ItemProperties name,type -Output host

hostid host items

------ ---- -----

473312 test-webserver-123456789.local {@{name=Zabbix agent running - Host name; type=0}, @{name=Zabbix agent running - Status; type=7}, @{name=Zabbix agent running - Version; type=7}, @{name=Disk IO sum (a…

### Example 7: Fetch host “items” with all properties.

Get-ZXHost -NameSearch vm-win -IncludeItems -ItemProperties extend

### Example 8: Fetch host triggers.

Get-ZXHost -Name test -IncludeTriggers -TriggerProperties description | Select-Object -ExpandProperty triggers

triggerid description

--------- -----------

5718296 System not reachable by ICMP

5718297 Zabbix agent not available (or nodata for {$AGENT.NODATA\_TIMEOUT})

5718298 High swap space usage (less than {$SWAP.PFREE.MIN.CRIT}% free)

5718299 High swap space usage (less than {$SWAP.PFREE.MIN.WARN}% free)

5718300 High memory utilization ( >{$MEMORY.UTIL.CRIT}% for 5m)

5718301 Load average too high (per CPU load over {$LOAD\_AVG\_PER\_CPU.MAX.WARN} for 60m)

5718302 High CPU utilization (over {$CPU.UTIL.CRIT}% for 30m)

5718309 Zombie processes (over {$PROCS.ZOMBIES.WARN} for 60m)

5718310 Zombie processes (over {$PROCS.ZOMBIES.CRIT} for 120m)

5718311 SCSI Device Timeout < 180 seconds

5718312 NTP Time Offset +- 150 seconds

# Get-ZXMaintenance

### Example 1: Get maintenance mode information

Maintenanceid can be used to search the auditlog.

$HostID = Get-ZXHost -Name vm-win-test-1.test.local -Output hostid

Get-ZXMaintenance -HostId $HostID.hostid

maintenanceid : 1280564

name : vm-win-test-1.test.local

maintenance\_type : 0

description : “SomeText”

active\_since : 1737010800

active\_till : 1768604400

tags\_evaltype : 0

CreationTime :

# New-ZXTagFilter

### Example 1: Create a list of tags with operators and use it to find hosts that match all the conditions.

Using Get-Host with -WhatIf parameter to show what the json request looks like without actually making the API call.

$TagFilter = New-ZXTagFilter

$TagFilter.AddTag("#tag\_1","exists").AddTag("#tag\_2","notexists").AddTag("tag\_3","equals","value")

Get-ZXHost -Tag $TagFilter.Tags -WhatIf

JSON REQUEST

{

"jsonrpc": "2.0",

"method": "host.get",

"params": {

"output": [

"hostid",

"host",

"name",

"status",

"proxy\_hostid"

],

"tags": [

{

"tag": "#tag\_1",

"operator": "4",

"value": ""

},

{

"tag": "#tag\_2",

"operator": "5",

"value": ""

},

{

"tag": "tag\_3",

"operator": "1",

"value": "value"

}

]

},

"auth": "\*\*\*\*\*",

"id": "1"

}

# Get-ZXTrigger

### Example 1: Get a list of all triggers on a single host wit all trigger properties

Use “extend” to get all trigger properties.

If you want to export the data co .csv and paste it to Excel, do not use Format-Table, Use ConvertTo-CSV

$ZXHost = Get-ZXHost -Name Test-Host -Output host,id

Get-ZXTrigger -HostID $ZXHost.hostid -Output extend | Format-Table

triggerid expression description url status value priority lastchange comments

--------- ---------- ----------- --- ------ ----- -------- ---------- --------

9618884 {22087821}>{$ICMP\_LOSS\_WARN} and {22087821}<100 High ICMP ping loss 0 0 3 0

9618885 {22087822}>{$ICMP\_RESPONSE\_TIME\_WARN} High ICMP ping response time 0 0 3 0

9618886 {22087823}=0… Unavailable by ICMP ping 0 0 3 0 The last …

9618887 {22087825}=0… Unavailable by ICMP ping Critical 0 0 5 0 The last …

9618888 {22087827}=0… Unavailable by ICMP ping High 0 0 4 0 The last …

9618889 {22087829}=0 The host is not available by SNMP 0 0 3 1737720758 No SNMP d…

9618890 {22087830}<10m Device has been restarted or reinitialized 0 0 2 0 The recor…

9618891 {22087831}>{$CPU.UTIL.CRIT} High CPU utilization (over {$CPU.UTIL.CRIT}% for 5m) 0 0 2 0 CPU utili…

10001260 {22748086}<>5 Fan number 1 is in critical state. Current state is {ITEM.LASTVALUE1} 0 0 3 0 Please ch…

10001261 {22748087}<>5 Fan number 2 is in critical state. Current state is {ITEM.LASTVALUE1} 0 0 3 0 Please ch…

### Example 2: Get a list of triggers on a single host with only specific trigger properties

$ZXHost = Get-ZXHost -Name Test-Host -Output host,id

Get-ZXTrigger -HostID $ZXHost.hostid -Output description,value,expression

triggerid description expression

--------- ----------- ----------

9618884 High ICMP ping loss {22087821}>{$ICMP\_LOSS\_WARN} and {22087821}<100

9618885 High ICMP ping response time {22087822}>{$ICMP\_RESPONSE\_TIME\_WARN}

9618886 Unavailable by ICMP ping {22087823}=0…

9618887 Unavailable by ICMP ping Critical {22087825}=0…

9618888 Unavailable by ICMP ping High {22087827}=0…

9618889 The host is not available by SNMP {22087829}=0

9618890 Device has been restarted or reinitialized {22087830}<10m

9618891 High CPU utilization (over {$CPU.UTIL.CRIT}% for 5m) {22087831}>{$CPU.UTIL.CRIT}

10001260 Fan number 1 is in critical state. Current state is {ITEM.LASTVALUE1} {22748086}<>5

10001261 Fan number 2 is in critical state. Current state is {ITEM.LASTVALUE1} {22748087}<>5

10001262 Fan number 3 is in critical state. Current state is {ITEM.LASTVALUE1} {22748088}<>5

10001263 PSU number 1 is in critical state. Current state is {ITEM.LASTVALUE1} {22748089}=4…

10001264 Memory utilization is ( >{$MEMORY.UTIL.MAX}% for 5m) {22748090}>={$MEMORY.UTIL.MAX}

10001265 Memory utilization is ( >{$MEMORY.UTIL.WARN}% for 5m) {22748091}>={$MEMORY.UTIL.WARN}

### Example 3: Get a list of trigger properties

#Get any host with at least one trigger

$ZXHost = Get-ZXHost -Name Test-Host -Output name,id

#Get the list of all possible properties

Get-ZXTrigger -HostID $ZXHost.hostid -Output Extend | Select-Object -First 1 | Get-Member -MemberType NoteProperty | Select-Object Name

Name

----

comments

correlation\_mode

correlation\_tag

description

error

event\_name

expression

flags

lastchange

manual\_close

opdata

priority

recovery\_expression

recovery\_mode

state

status

templateid

triggerid

type

url

url\_name

uuid

value

# Get-ZXAction

$Alert = Get-ZXAlert -EventID 0123456789

Get-ZXAction -ActionID $alert.actionid -Output name

actionid name

-------- ----

123 Alert issues to HelpDesk (critical) (Default)

# Common Parameters

## Output

### Example 1: Select specific properties of an object

If you don’t use this parameter, only a few selected properties are fetched depending on the object type, those are hardcoded.

You have an option to use -**Extend** to show all properties or select a specific properties as in this case.

Get-ZXHost -NameSearch test -IncludeTriggers -IncludeInterfaces -InterfaceProperties ip -Output name,status

hostid status host

------ ------ ----

481427 1 Test\_host1

500178 0 Test\_host2

494848 0 Test\_host3

481779 0 Test\_host4

506002 0 Test\_host5

Get-ZXHost -NameSearch test -IncludeTriggers -IncludeInterfaces -InterfaceProperties ip -Output extend

## WhatIf

### Example 1: Shows a JSON request without actually making the API call

Get-ZXHost -NameSearch test -IncludeTriggers -IncludeInterfaces -InterfaceProperties ip -Output name,status -WhatIf

JSON REQUEST

{

"jsonrpc": "2.0",

"method": "host.get",

"params": {

"output": [

"name",

"status"

],

"search": {

"host": "test"

},

"selectInterfaces": [

"ip"

],

"selectTriggers": [

"name",

"value"

]

},

"auth": "\*\*\*\*\*",

"id": "1"

}

## CountOutput

### Example 1: Don’t fetch any data, just show count the output.

Get-ZXHost -Status Disabled -CountOutput

3724

## ShowJsonRequest

### Example 1: Make the API call and also show the json that was used to make the call.

Get-ZXHost -Status Disabled -CountOutput -ShowJsonRequest

JSON REQUEST

{

"jsonrpc": "2.0",

"method": "host.get",

"params": {

"output": [

"hostid",

"host",

"name",

"status",

"proxy\_hostid"

],

"filter": {

"status": "1"

},

"countOutput": "true"

},

"auth": "\*\*\*\*\*",

"id": "1"

}

3724

## Limit

### Example 1: Limit the number of returned result

Good especially for getting samples to avoid fetching too much data

Get-ZXHost -NameSearch test -Limit 5 -ShowJsonRequest

JSON REQUEST

{

"jsonrpc": "2.0",

"method": "host.get",

"params": {

"output": [

"host"

],

"search": {

"host": "test"

},

"limit": 5

},

"auth": "\*\*\*\*\*",

"id": "1"

}

hostid host

------ ----

494848 Test\_host1

481779 Test\_host2

506002 Test\_host3

500178 Test\_host4

481427 Test\_host3