no_proxy for subnet or range of ips

Asked 3 years, 10 months ago   Modified 1 year ago   Viewed 13k times

2

I have machine connected to 2 Nets, some for my local network I want bypass (not go through) proxy:

How define no_proxy for range of local network for example 192.168.100.0/24

This syntax 192.168.100.0/24 works in browser definition however it seems int is not working in /etc/environment or no_proxy setting!

networking   server   proxy

Share   Improve this question   Follow

edited Apr 29, 2019 at 20:30

asked Apr 29, 2019 at 20:02
Rastin
784 ●2 ●11 ●25

1 Answer

Sorted by: Highest score (default)

1

Unfortunately there is no syntax available for no_proxy which would cover a subnet. The only solution I am aware of is to generate the whole list with something like

```
export no_proxy=${no_proxy},$(echo 192.168.100.{1..255} | sed 's/ /,/g')
```

Share   Improve this answer   Follow

answered Apr 30, 2019 at 6:00
marosg
1,313 ●10 ●16

Sources for this? It really seems to depend on which binary makes use of these binaries. And interpretations appear to differ. – Frank N Nov 26, 2019 at 11:16

Which part of that link contradicts my answer? It also states no_proxy covers only hosts, not subnets – marosg Nov 27, 2019 at 13:26

What is the proper way to deal with the HTTPProxyCIDR WARNING and to setup no_proxy in an environment that sits behind a proxy

Asked 3 years, 11 months ago Modified 3 years, 3 months ago Viewed 7k times

1

I'm relatively new to k8s. I have now set up a cluster several times to ensure that I understand the process. I have struggled with networking a bit. I am currently initializing as follows:

```
kubeadm init --apiserver-advertise-address=10.93.98.204 --pod-network-
cidr=10.244.0.0/16
```

In response to this I see the following warning:

```
[WARNING HTTPProxyCIDR]: connection to "10.96.0.0/12" uses proxy
"http://proxy.corp.sensis.
com:3128". This may lead to malfunctional cluster setup. Make sure that Pod and
Services IP ranges
specified correctly as exceptions in proxy configuration
```

Amongst other things, I am trying to ensure that I configure the cluster correctly, and the overlay network (flannel).

I've attempted to establish the no_proxy environment variable (centos 7).

The way that i tried this was as follows was to update /etc/profile.d/proxy.sh as follows:

```
printf -v lan '%s,'
"10.93.98.204","10.93.98.23","10.93.98.36","10.93.103.236","10.93.97.123","10.93.97.
printf -v service '%s,' 10.244.{1..255}.{1..255}
export no_proxy="${lan%,},${service%,},127.0.0.1";
#export no_proxy="${lan%,},10.244.0.0/16,127.0.0.1";
export NO_PROXY=$no_proxy
```

However, this approach results in a massive string ($no_proxy) that far exceeds the maximum length within the Linux environment.

I've also tried using the pod-network-cidr in the no_proxy (`10.244.0.0/16` - commented out in the above)

Two questions: - What is the proper way to deal with this warning (WARNING HTTPProxyCIDR)? - How can I set no_proxy so that my flannel network overlay works and my cluster works

kubernetes kubeadm

Share Improve this question edited Apr 9, 2019 at 23:31 asked Mar 27, 2019 at 12:28

Follow

Rico
57.1k ● 12 ● 108 ● 138

Dave Sargrad
45 ● 2 ● 10

2 Answers

Sorted by:
Highest score (default)

2

The `no_proxy/NO_PROXY` environment variables should be the way to go. However, you don't need to add every single IP to the string you can just add the whole subnet. Also, looks like you are missing `10.96.0.0/12` from the list.

For example (assuming `10.93.98.0/24` is your LAN subnet):

```
export no_proxy="10.96.0.0/12,10.93.98.0/24,10.244.0.0/16,127.0.0.1"
```

Also, make sure `Docker` noProxy is configured if you are using Docker.

Share Improve this answer Follow

answered Apr 10, 2019 at 0:43
Rico
57.1k ● 12 ● 108 ● 138

2

The CIDR/IP range does not work in no_proxy in many environments/applications.

We can make a reasonable assumption that we don't access network nodes outside web-proxy thru IP address. In other words, we use FQDN to access, say python.com, google.com, github.com, but not directly using their IP addresses.

With this assumption, we can bypass web-proxy for all direct IP address access.

```
export no_proxy=localhost,.svc
printf -v allip '%s,' .{0..255}
export no_proxy="$no_proxy,${allip%,}"
```

This adds `.0,.1,.2,...,.255` to the no_proxy env variable. There is no magic here. We just treat IP address as FQDN, so a suffix match works as FQDN no_proxy setting. Say, .120 would match all IP addresses x.x.x.120.

Share Improve this answer Follow edited Nov 17, 2019 at 6:17

answered Sep 8, 2019 at 17:11
B.Z.
388 ● 4 ● 11

Are HTTP_PROXY, HTTPS_PROXY and NO_PROXY environment variables standard?

Asked 7 years, 7 months ago Modified 1 month ago Viewed 91k times

45

It seems that a lot of programs are designed to read these environment variables to decide what proxy to go through in order to connect to a resource on the internet. Those programs may also have their own, individual proxy settings, but if those are not set, they'll happily use these environment variables...

- HTTP_PROXY
- HTTPS_PROXY
- NO_PROXY

I just want to know:

- Are these environment variables standard?
- Is there a written specification (may be by the OS manufacturers?) that recommends the use of these environment variables?

environment-variables http http-proxy

Share Improve this question Follow

edited Jan 13, 2017 at 12:57
Piotr Dobrogost
5,272 ● 14 ● 56 ● 78

asked Jul 24, 2015 at 19:44
Niko Bellic
1,102 ● 3 ● 13 ● 20

1 I do not know no_proxy, but http_proxy (written lowercase) is standard – Uwe Burger Jul 24, 2015 at 20:21

@UweBurger perhaps you can state which programs use it.. And that goes for the questioner too. I've seen it used on wget – barlop Jul 25, 2015 at 3:23

Documented (in a wiki) for Arch Linux: wiki.archlinux.org/... – Brent Bradburn Jun 29, 2020 at 1:33

They are a *de facto* standard. Despite their unixy origin, even .NET uses them. There is some variability in the details of their support, though, as the excellent accepted answer explains. – Palec Feb 8, 2022 at 18:41

3 Answers

Sorted by: Highest score (default)

34

I agree with BillThor's statement that *This is more a convention than a standard*.
I don't know the origin of these variables but in case of HTTP on *nix many conventions seem to originate from behavior of libcurl HTTP library and curl command line program.

At https://curl.haxx.se/docs/manual.html there's description of environment variables related to using HTTP proxy which libcurl/curl understands:

> ENVIRONMENT VARIABLES
>
> Curl reads and understands the following environment variables:
> `http_proxy, HTTPS_PROXY, FTP_PROXY`
>
> They should be set for protocol-specific proxies. General proxy should be set with
> `ALL_PROXY`
>
> A comma-separated list of host names that shouldn't go through any proxy is set in (only an asterisk, '*' matches all hosts)
> `NO_PROXY`
>
> If the host name matches one of these strings, or the host is within the domain of one of these strings, transactions with that node will not be proxied.

Please notice that `http_proxy` is spelled lowercase as the only one among these variables. Some libraries/programs look for lowercase names of these variables whereas others look for upppercase names. To be *safe* one should define both lowercase and uppercase versions of each variable.

Another issue is that cited description of how host names are matched against `NO_PROXY` is not precise and does not answer the following questions:

- Should values be fully qualified domain names (FQDN) thus ending with a dot like `foo.example.com.` or not?
- Should `foo.example.com` match only this one domain or should it also match any subdomain like `bar.foo.example.com`? If the latter then should it also match any subdomain in any subdomain like `bar.baz.foo.example.com`?
- Is `.foo.example.com` (dot at the beginning) allowed and if so then what should it match ?
- Is asterisk (`*`) allowed as part of value (`*.example.com`, `*example.com`) and if so then how is it treated?

Lack of formal specification leads to confusion and bugs. Here one has to mention *libproxy* library which aims to provide correct and consistent support for proxy configuration. From project's home page:

> libproxy exists to answer the question: Given a network resource, how do I reach it? It handles all the details, enabling you to get back to programming.

Further reading:

- Issue I raised against curl – Please document syntax and semantics of NO_PROXY environment variable.
- Python's issue – urllib: no_proxy variable values with leading dot not properly handled and related SO question – Python 2.7.13 does not respect NO_PROXY and makes urllib2.urlopen() error with "Tunnel connection failed: 403 Forbidden"

Share Improve this answer Follow edited Mar 5, 2018 at 20:59 answered Jan 13, 2017 at 10:31

Piotr Dobrogost
5,272 ● 14 ● 56 ● 78

What does libproxy have to say about the questions you raise? The one I'm interested in: "Should .foo.example.com match foo.example.com or not?" – Robin Winslow Mar 5, 2018 at 14:30

I have no idea. I encourage you to ask at github.com/libproxy/libproxy/issues – Piotr Dobrogost Mar 5, 2018 at 21:01

16

This is more a convention than a standard. It is likely supported by one or more protocol handler libraries which actually make the connections. Java uses similar properties in its protocol libraries.

Understanding and using common conventions makes development much simpler. It also helps implement the principle of least surprise and make programs more likely to `just work`.

Share Improve this answer Follow answered Jul 25, 2015 at 1:02

BillThor
10.7k ● 2 ● 24 ● 23

8

There is no real standard.

Different tools interpret these variables similarly but subtly differently. For example the *case* of the recognised environment variables and the *case-precedence* varies between tools like `curl` and `wget` and languages like Ruby, Python and Go: (This table is taken from the excellent article We need to talk: Can we standardize NO_PROXY?)

	`curl`	`wget`	**Ruby**	**Python**	**Go**
`http_proxy` (lowercase)	Yes	Yes	Yes	Yes	Yes
`HTTP_PROXY` (Uppercase)	No	No	Yes (warning)	Yes (if REQUEST_METHOD not in env)	Yes
`https_proxy` (lowercase)	Yes	Yes	Yes	Yes	Yes

	`curl`	`wget`	**Ruby**	**Python**	**Go**
`HTTPS_PROXY` (Uppercase)	Yes	No	Yes	Yes	Yes
Case precedence	lowercase	lowercase only	lowercase	lowercase	Uppercase

For more details (particularly about the NO_PROXY variable) and some history look at the mentioned complete article.

Share Improve this answer Follow

answered Dec 1, 2021 at 1:21

halloleo
223 ● 2 ● 10

Could you also add the no_proxy table from about.gitlab.com/blog/2021/01/27/we-need-to-talk-no-proxy to this answer in case the article goes away? – balder Jan 12 at 12:04

@balder Makes sense. Happy for you to add it to this answer. :) – halloleo Jan 13 at 2:18