

Аварийная прошивка OpenWrt через консоль Breed для роутеров с разметкой NAND-flash от Xiaomi Mi Router 3G

inflabz • November 19, 2020

1. Скачиваем nginx с <https://nginx.org/ru/download.html>

Распаковываем архив в C:\nginx\

В порт LAN1 роутера подключить компьютер (скорость UART 115200, если используете терминал)

Зажать на роутере Reset 5-10 сек. и подключить питание к роутеру.

В браузере (в режиме инкогнито) открываем <http://192.168.1.1/index.html> и проверяем вход в Breed

Запускаем PuTTY

IP 192.168.1.1

Telnet, порт 23

и подключаемся к Breed:

```
Breed for Beeline SmartBox GIGA
```

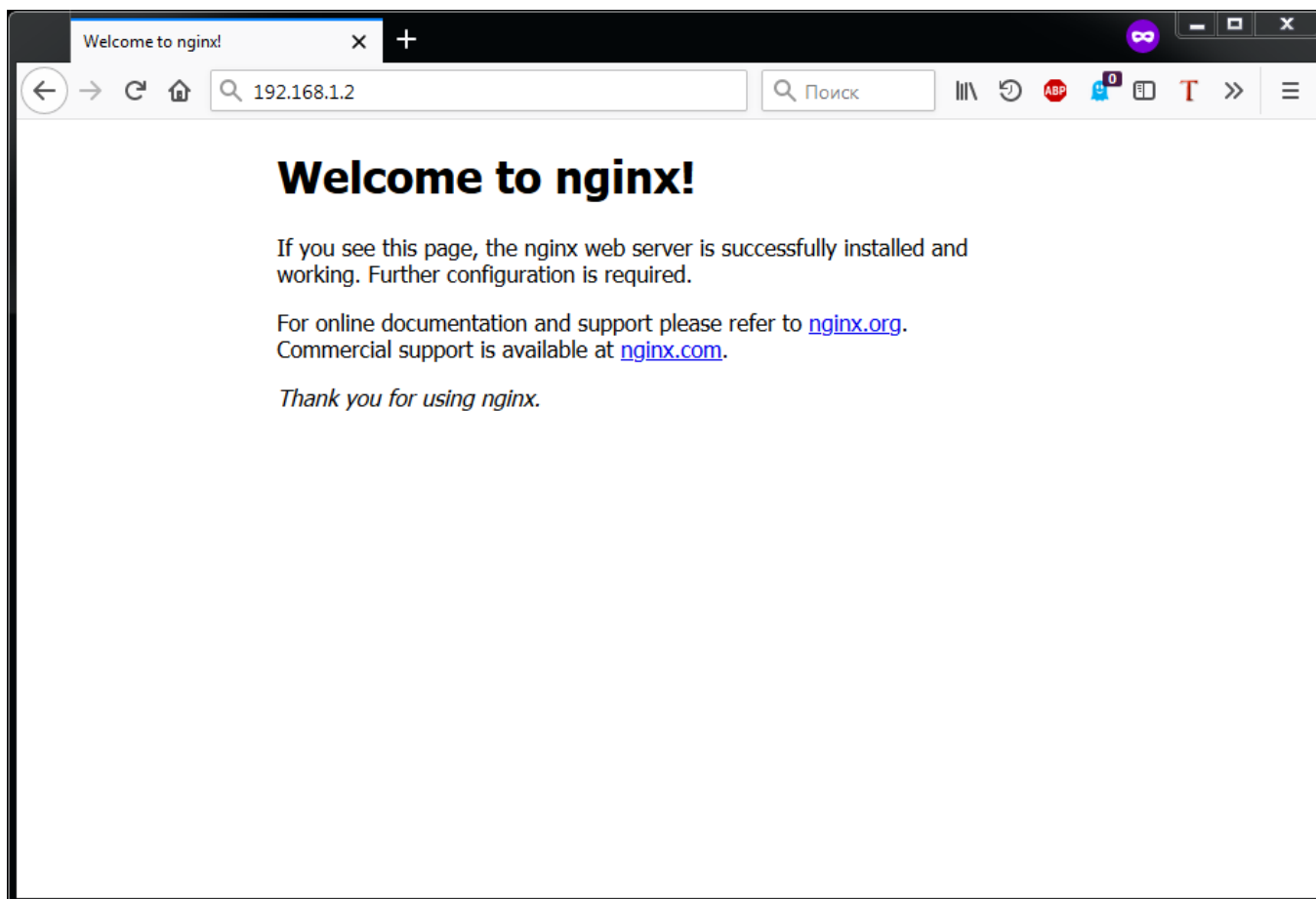
```
Starting breed built-in shell
```

```
breed>
```

Запускаем nginx из C:\nginx\nnginx.exe

Смотрим в сетевых подключения IP-адрес компьютера (обычно Breed выдает 192.168.1.2 или 192.168.1.3 и т.п.)

Проверяем работу nginx открыв в браузере этот адрес



Проверка работы nginx

В директорию `c:\nginx\html\` помещаем файлы вашего роутера (пример для SmartBox GIGA)

`*squashfs-kernel1.bin`

`*squashfs-rootfso.bin`

для GIGA это будут

`openwrt-ramips-mt7621-beeline_smartbox-giga-squashfs-kernel1.bin`

`openwrt-ramips-mt7621-beeline_smartbox-giga-squashfs-rootfso.bin`

2. в PuTTY выполняем команды:

`wget http://192.168.1.2/openwrt-ramips-mt7621-beeline_smartbox-giga-squashfs-rootfso.bin`

вывод будет примерно таким

```
breed> wget http://192.168.1.2/openwrt-ramips-mt7621-beeline_smartbox-giga-squashfs-rootfso.bin
```

```
wget http://192.168.1.2/openwrt-ramips-mt7621-beeline_smartbox-giga-squashfs-
rootfs.bin
```

```
Connecting to 192.168.1.2:80... connected.
```

```
HTTP request sent, awaiting response... 200 OK
```

```
Length: 15335424/oxea0000 (14MB) [application/octet-stream]
```

```
Saving to address 0x80001000
```

```
[=====
=====] 100%
```

```
Transmission completed in 3.1s.
```

Если файл не скачивается, дальше не продолжаем !!!

3. Для 128 Мб флешки (GIGA, TURBO+ и оригинальный MIR3G)

Стираем флешь:

flash erase 0x2000000 0x7e00000

```
breed> flash erase 0x2000000 0x7e00000
```

```
flash erase 0x2000000 0x7e00000
```

```
Erasing flash bank 0 from 200000h , size 7e00000h
```

```
[=====> ] 30% mt7621-nfi.o: 1
error bit(s) corrected in page 20928
```

```
[=====
=====> ] 99% mt7621-nfi.o: Skipping
protected block 1023 (offset 0x07fe0000)
```

```
[=====
=====] 100%
```

```
Succeeded
```

В логe видно что сбойный блок NAND флeши скорректирован. Неисправные блоки невозможно скорректировать (только заменять флeшь)

4. Записываем файловою систему

flash write 0xA00000 0x80001000 15335424

где 15335424 это размер файла **squashfs-rootfs0.bin* из лога в п.2

```
breed> flash write 0xA00000 0x80001000 15335424
```

```
flash write 0xA00000 0x80001000 15335424
```

```
Writing flash bank 0 into a00000h from memory 80001000h, size ea0000h
```

```
[=====]  
=====] 100%
```

```
Succeeded
```

5. Записываем ядра

wget http://192.168.1.2/openwrt-ramips-mt7621-beeline_smartbox-giga-squashfs-kernel1.bin

```
breed> wget http://192.168.1.2/openwrt-ramips-mt7621-beeline_smartbox-giga-squashfs-  
kernel1.bin
```

```
wget http://192.168.1.2/openwrt-ramips-mt7621-beeline_smartbox-giga-squashfs-  
kernel1.bin
```

```
Connecting to 192.168.1.2:80... connected.
```

```
HTTP request sent, awaiting response... 200 OK
```

```
Length: 2365690/0x2418fa (2MB) [application/octet-stream]
```

```
Saving to address 0x80001000
```

```
[=====]  
=====] 100%
```

```
Transmission completed in 0.8s.
```

Записываем kernel0

flash write ox200000 ox80001000 2365690

где **2365690** это размер файла **squashfs-kernel1.bin* из лога в п.5

```
| breed> flash write ox200000 ox80001000 2365690
```

```
| flash write ox200000 ox80001000 2365690
```

```
| Writing flash bank 0 into 200000h from memory 80001000h, size 2418fah
```

```
| [=====
```

```
| =====] 100%
```

```
| Succeeded
```

Записываем kernel1

flash write ox600000 ox80001000 2365690

где **2365690** это размер файла **squashfs-kernel1.bin* из лога в п.5

```
| breed> flash write ox600000 ox80001000 2365690
```

```
| flash write ox600000 ox80001000 2365690
```

```
| Writing flash bank 0 into 600000h from memory 80001000h, size 2418fah
```

```
| [=====
```

```
| =====] 100%
```

```
| Succeeded
```

Если все прошло успешно перезагружаемся

reset

```
| breed> reset
```

Настройки запуска ядра Breed

<http://192.168.1.1/envedit.html>

Стандартные переменные окружения (Environment) для Xiaomi Mi Router 3G (и роутеров на базе его разметки):

```
autoboot.command = boot flash 0x600000
```

```
stock_kernel = boot flash 0x200000
```

однако, если **kernel1** не записалось в п.5 используйте специальную прошивку или попробуйте загрузиться из **kernel0** указав в

```
autoboot.command = boot flash 0x200000
```

Однако в этом случае будут проблемы с обновлением через OpenWrt