Скачиваем OpenSSH, ZLib и OpenSSL:

```
nastya@linux:~/openssh$ git clone https://github.com/openssh/openssh-portable.git
Cloning into 'openssh-portable'...
remote: Enumerating objects: 69785, done.
remote: Counting objects: 100% (567/567), done.
remote: Compressing objects: 100% (152/152), done.
remote: Total 69785 (delta 497), reused 417 (delta 415), pack-reused 69218 (from 3)
Receiving objects: 100% (69785/69785), 31.36 MiB | 3.78 MiB/s, done.
Resolving deltas: 100% (53713/53713), done.
```

```
nastya@linux:~/openssh/openssl$ git clone https://github.com/openssl/openssl.git --branch OpenSSL_1_1_1-stable
Cloning into 'openssl'...
remote: Enumerating objects: 518697, done.
remote: Counting objects: 100% (46/46), done.
remote: Compressing objects: 100% (21/21), done.
remote: Total 518697 (delta 34), reused 25 (delta 25), pack-reused 518651 (from 2)
Receiving objects: 100% (518697/518697), 289.61 MiB | 2.67 MiB/s, done.
Resolving deltas: 100% (381132/381132), done.
nastya@linux:~/openssh/openssl$
```

```
nastya@linux:~/openssh/zlib$ git clone https://github.com/madler/zlib.git --branch v1.2.1.2
Cloning into 'zlib'...
remote: Enumerating objects: 7297, done.
remote: Counting objects: 100% (1/1), done.
remote: Total 7297 (delta 0), reused 0 (delta 0), pack-reused 7296 (from 1)
Receiving objects: 100% (7297/7297), 4.33 MiB | 5.22 MiB/s, done.
Resolving deltas: 100% (5170/5170), done.
Note: switching to '7a6955760ba950eb82f57929f8f6c9847c65f0af'.
```

Для ZLib выполняем команду configure с флагом -s для компиляции динамической библиотеки:

```
nastya@linux:~/openssh/zlib/zlib$ ./configure -s
Checking for gcc...
Checking for shared library support...
Building shared library libz.so.1.2.1.2 with gcc.
Checking for unistd.h... Yes.
Checking whether to use vs[n]printf() or s[n]printf()... using vs[n]printf()
Checking for vsnprintf() in stdio.h... Yes.
Checking for return value of vsnprintf()... Yes.
Checking for errno.h... Yes.
Checking for mmap support... Yes.
```

Выполняем make с указанием нужного кросс-компилятора:

```
nastya@linux:~/openssh/zlib/zlib$ make CC=arm-linux-gnueabihf-gcc
arm-linux-gnueabihf-gcc -fPIC -O3 -DUSE_MMAP   -c -o example.o example.c
arm-linux-gnueabihf-gcc -fPIC -O3 -DUSE_MMAP   -c -o adler32.o adler32.c
arm-linux-gnueabihf-gcc -fPIC -O3 -DUSE_MMAP   -c -o compress.o compress.c
arm-linux-gnueabihf-gcc -fPIC -O3 -DUSE_MMAP   -c -o crc32.o crc32.c
arm-linux-gnueabihf-gcc -fPIC -O3 -DUSE_MMAP   -c -o gzio.o gzio.c
arm-linux-gnueabihf-gcc -fPIC -O3 -DUSE_MMAP   -c -o uncompr.o uncompr.c
arm-linux-gnueabihf-gcc -fPIC -O3 -DUSE_MMAP   -c -o deflate.o deflate.c
arm-linux-gnueabihf-gcc -fPIC -O3 -DUSE_MMAP   -c -o trees.o trees.c
arm-linux-gnueabihf-gcc -fPIC -O3 -DUSE_MMAP   -c -o zutil.o zutil.c
arm-linux-gnueabihf-gcc -fPIC -O3 -DUSE_MMAP   -c -o inflate.o inflate.c
arm-linux-gnueabihf-gcc -fPIC -O3 -DUSE_MMAP   -c -o infback.o infback.c
arm-linux-gnueabihf-gcc -fPIC -O3 -DUSE_MMAP   -c -o inftrees.o inftrees.c
arm-linux-gnueabihf-gcc -fPIC -O3 -DUSE_MMAP   -c -o inffast.o inffast.c
```

Выполняем make install с указанием директории для установки:

```
nastya@linux:~/openssh/zlib/zlib$ make install prefix=$PWD/_install
cp zlib.h zconf.h /home/nastya/openssh/zlib/zlib/_install/include
chmod 644 /home/nastya/openssh/zlib/zlib/_install/include/zlib.h /home/nasty
cp libz.so.1.2.1.2 /home/nastya/openssh/zlib/zlib/_install/lib
cd /home/nastya/openssh/zlib/zlib/_install/lib; chmod 755 libz.so.1.2.1.2
cd /home/nastya/openssh/zlib/zlib/_install/lib; if test -f libz.so.1.2.1.2;
  rm -f libz.so libz.so.1; \
  ln -s libz.so.1.2.1.2 libz.so; \
  ln -s libz.so.1.2.1.2 libz.so.1; \
  (ldconfig || true)  >/dev/null 2>&1; \
fi
cp zlib.3 /home/nastya/openssh/zlib/zlib/_install/share/man/man3
chmod 644 /home/nastya/openssh/zlib/zlib/_install/share/man/man3/zlib.3
```

Для OpenSSL выполняем команду configure с указанием архитектуры, директории для установки и префикса кросс-компилятора:

```
nastya@linux:~/openssh/openssl/openssl$ ./Configure linux-armv4 --prefix=$PWD/_install --cross-compile-prefix=arm-linux-
gnueabihf
Configuring OpenSSL version 1.1.1x-dev (0x10101180L) for linux-armv4
Using os-specific seed configuration
Creating configdata.pm
Creating Makefile

**********************************************************************
***                                                                ***
***   OpenSSL has been successfully configured                     ***
***                                                                ***
***   If you encounter a problem while building, please open an    ***
***   issue on GitHub <https://github.com/openssl/openssl/issues>  ***
***   and include the output from the following command:           ***
***                                                                ***
***       perl configdata.pm --dump                                ***
***                                                                ***
***   (If you are new to OpenSSL, you might want to consult the    ***
***   'Troubleshooting' section in the INSTALL file first)         ***
***                                                                ***
**********************************************************************
```

Выполняем make:

```
nastya@linux:~/openssh/openssl/openssl$ make
/usr/bin/perl "-I." -Mconfigdata "util/dofile.pl" \
    "-oMakefile" include/crypto/bn_conf.h.in > include/crypto/bn_conf.h
/usr/bin/perl "-I." -Mconfigdata "util/dofile.pl" \
    "-oMakefile" include/crypto/dso_conf.h.in > include/crypto/dso_conf.h
/usr/bin/perl "-I." -Mconfigdata "util/dofile.pl" \
    "-oMakefile" include/openssl/opensslconf.h.in > include/openssl/opensslconf.h
make depend && make _all
make[1]: Entering directory '/home/nastya/openssh/openssl/openssl'
make[1]: Leaving directory '/home/nastya/openssh/openssl/openssl'
make[1]: Entering directory '/home/nastya/openssh/openssl/openssl'
arm-linux-gnueabihf-gcc  -I. -Iinclude -fPIC -pthread -Wall -O3 -DOPENSSL_USE_NODELETE -DOPENSSL_PIC -DOPENSSLDIR="\"/ho
me/nastya/openssh/openssl/openssl/_install/ssl\"" -DENGINESDIR="\"/home/nastya/openssh/openssl/openssl/_install/lib/engi
nes-1.1\"" -DNDEBUG  -MMD -MF apps/app_rand.d.tmp -MT apps/app_rand.o -c -o apps/app_rand.o apps/app_rand.c
arm-linux-gnueabihf-gcc  -I. -Iinclude -fPIC -pthread -Wall -O3 -DOPENSSL_USE_NODELETE -DOPENSSL_PIC -DOPENSSLDIR="\"/ho
me/nastya/openssh/openssl/openssl/_install/ssl\"" -DENGINESDIR="\"/home/nastya/openssh/openssl/openssl/_install/lib/engi
nes-1.1\"" -DNDEBUG  -MMD -MF apps/apps.d.tmp -MT apps/apps.o -c -o apps/apps.o apps/apps.c
arm-linux-gnueabihf-gcc  -I. -Iinclude -fPIC -pthread -Wall -O3 -DOPENSSL_USE_NODELETE -DOPENSSL_PIC -DOPENSSLDIR="\"/ho
me/nastya/openssh/openssl/openssl/_install/ssl\"" -DENGINESDIR="\"/home/nastya/openssh/openssl/openssl/_install/lib/engi
nes-1.1\"" -DNDEBUG  -MMD -MF apps/bf_prefix.d.tmp -MT apps/bf_prefix.o -c -o apps/bf_prefix.o apps/bf_prefix.c
```

Выполняем make install:

```
nastya@linux:~/openssh/openssl/openssl$ make install
make depend && make _build_libs
make[1]: Entering directory '/home/nastya/openssh/openssl/openssl'
make[1]: Leaving directory '/home/nastya/openssh/openssl/openssl'
make[1]: Entering directory '/home/nastya/openssh/openssl/openssl'
make[1]: Nothing to be done for '_build_libs'.
make[1]: Leaving directory '/home/nastya/openssh/openssl/openssl'
created directory `/home/nastya/openssh/openssl/openssl/_install'
created directory `/home/nastya/openssh/openssl/openssl/_install/lib'
*** Installing runtime libraries
install libcrypto.so.1.1 -> /home/nastya/openssh/openssl/openssl/_install/lib/libcrypto.so.1.1
install libssl.so.1.1 -> /home/nastya/openssh/openssl/openssl/_install/lib/libssl.so.1.1
```

Для OpenSSH выполняем команду configure с указанием директории для установки, префикса кросс-компилятора, пути к ZLib, пути к OpenSSL, также указываем, где создать директорию var/empty и отключаем strip(т.к. указать нужную версию нельзя, по умолчанию используется strip системы, на которой выполняется компиляция):

```
nastya@linux:~/openssh/openssh-portable$ ./configure --prefix=$PWD/_install --host=arm-linux-gnueabihf --with-zlib=$PWD/
zlib --with-ssl-dir=$PWD/openssl --with-privsep-path=$PWD/_install/var/empty --disable-strip
checking for arm-linux-gnueabihf-cc... no
checking for arm-linux-gnueabihf-gcc... arm-linux-gnueabihf-gcc
checking whether the C compiler works... yes
checking for C compiler default output file name... a.out
checking for suffix of executables...
checking whether we are cross compiling... yes
checking for suffix of object files... o
checking whether the compiler supports GNU C... yes
checking whether arm-linux-gnueabihf-gcc accepts -g... yes
checking for arm-linux-gnueabihf-gcc option to enable C11 features... none needed
checking if arm-linux-gnueabihf-gcc supports C99-style variadic macros... yes
checking build system type... x86_64-pc-linux-gnu
```

Выполняем make:

```
nastya@linux:~/openssh/openssh-portable$ make
Makefile:728: warning: ignoring prerequisites on suffix rule definition
conffile=`echo sshd_config.out | sed 's/.out$//'`; \
/usr/bin/sed -e 's|/etc/ssh/ssh_config|/home/nastya/openssh/openssh-portable/_install/etc/ssh_config|g' -e 's|/etc/ssh/s
sh_known_hosts|/home/nastya/openssh/openssh-portable/_install/etc/ssh_known_hosts|g' -e 's|/etc/ssh/sshd_config|/home/na
stya/openssh/openssh-portable/_install/etc/sshd_config|g' -e 's|/usr/libexec|/home/nastya/openssh/openssh-portable/_inst
all/libexec|g' -e 's|/etc/shosts.equiv|/home/nastya/openssh/openssh-portable/_install/etc/shosts.equiv|g' -e 's|/etc/ssh
/ssh_host_key|/home/nastya/openssh/openssh-portable/_install/etc/ssh_host_key|g' -e 's|/etc/ssh/ssh_host_ecdsa_key|/home
/nastya/openssh/openssh-portable/_install/etc/ssh_host_ecdsa_key|g' -e 's|/etc/ssh/ssh_host_rsa_key|/home/nastya/openssh
/openssh-portable/_install/etc/ssh_host_rsa_key|g' -e 's|/etc/ssh/ssh_host_ed25519_key|/home/nastya/openssh/openssh-port
able/_install/etc/ssh_host_ed25519_key|g' -e 's|/var/run/sshd.pid|/var/run/sshd.pid|g' -e 's|/etc/moduli|/home/nastya/op
enssh/openssh-portable/_install/etc/moduli|g' -e 's|/etc/ssh/moduli|/home/nastya/openssh/openssh-portable/_install/etc/m
oduli|g' -e 's|/etc/ssh/sshrc|/home/nastya/openssh/openssh-portable/_install/etc/sshrc|g' -e 's|/usr/X11R6/bin/xauth|/us
r/bin/xauth|g' -e 's|/var/empty|/var/empty|g' -e 's|/usr/bin:/bin:/usr/sbin:/sbin|/usr/bin:/bin:/usr/sbin:/sbin:/home/na
```

Выполняем make install(забыла сделать скриншот)

Затем в нашу файловую систему собранную с помощью busybox копируем содержимое всех директорий полученных после установки OpenSSH, ZLib и OpenSSL. Заново формируем файловую систему с помощью команд find . | cpio -o -H newc | gzip > initramfs.cpio.gz.

Запускаем нашу систему в эмуляторе QEMU:

```
nastya@linux:~/arm-compile$ QEMU_AUDIO_DRV=none qemu-system-arm -M vexpress-a9 -kernel zImage -dtb vexpress-v2p-ca9.dtb
-initrd initramfs.cpio.gz -append 'console=ttyAMA0 rdinit=/bin/ash' -nographic
[    0.000000] Booting Linux on physical CPU 0x0
[    0.000000] Linux version 6.14.0my-g38fec10eb60d-dirty (nastya@linux) (arm-linux-gnueabihf-gcc (Ubuntu 10.5.0-4ubuntu
2) 10.5.0, GNU ld (GNU Binutils for Ubuntu) 2.42) #1 SMP Sun Sep  7 19:55:53 UTC 2025
[    0.000000] CPU: ARMv7 Processor [410fc090] revision 0 (ARMv7), cr=10c5387d
[    0.000000] CPU: PIPT / VIPT nonaliasing data cache, VIPT nonaliasing instruction cache
[    0.000000] OF: fdt: Machine model: V2P-CA9
[    0.000000] Memory policy: Data cache writeback
[    0.000000] efi: UEFI not found.
[    0.000000] cma: Failed to reserve 64 MiB on node -1
```

Затем в QEMU создаем директории sys и etc, заполняем каталог dev с помощью команды mdev -s, в etc создаем файлы passwd и group, авторизуем пользователя root, затем выполняем команду ssh-keygen для проверки работы приложения:

```
~ # mkdir sys
~ # mount -t sysfs sysfs sys
~ # mdev -s
~ # mkdir etc
~ # mount -t tmpfs tmpfs etc
~ # echo "root:x:0:0:root:/root:/bin/ash" > /etc/passwd
~ # echo "root:x:0:" > /etc/group
~ # mkdir /etc/ssa
~ # ssh-keygen -t rsa -b 4096
[  225.654279] random: crng init done
Generating public/private rsa key pair.
Enter file in which to save the key (/root/.ssh/id_rsa): /etc/ssa/id_rsa
Enter passphrase for "/etc/ssa/id_rsa" (empty for no passphrase): hello
Enter same passphrase again: hello
Your identification has been saved in /etc/ssa/id_rsa
Your public key has been saved in /etc/ssa/id_rsa.pub
The key fingerprint is:
SHA256:Rmlsb4WD+5g0OwFF3SbVsISxNIH/b1YaRSTYr3qVGCQ root@(none)
The key's randomart image is:
+---[RSA 4096]----+
|       .oo=*+=...|
|       o.++E+oo..|
|      . B.+o=  o |
|       = o.o .  o|
|        S o.  oo.|
|       o O  ..o.o|
|        = .  o = |
|         .  . *  |
|            +    |
+----[SHA256]-----+
~ #
```