sirjofri.de changeblog

Sat, 10 Jan 26 13:56:54 CET :

1. Building and Running Purgatorio on Windows 11

I wanted to try building purgatorio (inferno) on a Windows system, and with some effort I got it compile using Visual Studio. Even more than that, I was able to set up Visual Studio in a way that I can use IntelliSense and the debugger.

Here are a few notes:

- make sure that git **doesn't change line endings to CRLF!** This is important!
  - to "fix" files later, change the file and use `git restore` to restore it after setting `git config core.autocrlf false`
- it's easiest to open folder in VS, then open the programmer command line. This sets up `%PATH%` etc.
  - Tools → Command Line → Developer Command Prompt
- **Do not** try to run `makemk.sh` etc. Just use the NT binaries that are shipped with the repo (`Nt/386/bin`).
- adjust `mkconfig`
- set `%PATH%` to include `Nt/386/bin`: `set PATH=%PATH%;...`
- mkfiles: cl/link/lib args replace `-` with `/` — mostly optional
  - the tools sometimes print warnings that the dash options will be deprecated. I think for the future we should change the mkfiles for Nt. `/` *is* the windows way for options.
- `libinterp`: first manually `mk *.h` (individual headers)
- `utils/mkfile`: windows can't build mk properly:
  - multiple arguments are bundled to a single argument, which can't be understood by cl
  - mk is already built and likely won't change, so you can just comment out mk in the mkfile (`NOTPLAN9`)
    - I assume the issue is this line: `CFLAGS=$CFLAGS -I../include -DROOT'="'$ROOT'"'`, and `$CFLAGS` is included as a single option, not a list of options.

1.1. Setting up Visual Studio

Store these files in the root folder of the project:

1.1.1. `launch.vs.json`: Needed for launching. Pick `emu` in the selection.

```
{
  "version": "0.2.1",
  "configurations": [
    {
      "name": "emu",
      "project": "emu\\Nt\\iemu.exe",
      "args": [ "-r", "${workspaceRoot}", "/dis/wm/wm.dis" ]
    }
  ]
}
```

1.1.2. `CppProperties.json`: Needed for IntelliSense.

```
{
  "configurations": [
    {
      "inheritEnvironments": [
        "msvc_x86"
      ],
      "name": "x86-Debug",
      "includePath": [
        "${env.INCLUDE}",
        "${workspaceRoot}\\**"
      ],
      "defines": [
        "WIN32",
        "_DEBUG",
        "UNICODE",
        "_UNICODE"
      ],
      "intelliSenseMode": "windows-msvc-x86"
    }
  ]
}
```

1.1.3. `tasks.vs.json`: If you want right-click → build for mkfiles.

```
{
  "version": "0.2.1",
  "tasks": [
    {
      "taskLabel": "runmk",
      "appliesTo": "mkfile",
      "contextType": "build",
      "type": "launch",
      "command": "mk",
      "args": [ "install" ],
      "envVars": {
        "PATH": "${workspaceRoot}\\Nt\\386\\bin;${env.PATH}"
      }
    }
  ]
}
```

Mon, 18 Aug 25 19:38:48 CES :

2. Encrypted File Store on Plan 9 using cryptsetup

Sometimes you just need a little writable filesystem that is encrypted and stored in a single file. Turns out there are multiple ways to do that, besides the obvious ones.

This post describes a simple way to do that using cryptsetup and gefs. It is worth noting that I won't go into details about configuring gefs to do exactly what you want. Also, gefs is still considered experimental and should be used with care. Especially if you store sensitive data in that file, you should have a proper backup.

2.1. Cryptsetup

Using gefs on a file is trivial, so we start with the more complicated things: cryptsetup. Cryptsetup uses fs(3) to expose the unencrypted file as a simple disk filesystem. The stored file itself is encrypted. First, we have to create a file we can use. We use `dd` for that:

```
dd -if /dev/zero -bs 1024 -count 524288 > mydisk
```

This generates a file `mydisk` with a size of roughly 500 MB (512 * 1024 = 524288). You can use `hoc` to calculate the perfect size for you.

Note that gefs has a minimum file size requirement.

We want to encrypt this file with cryptsetup. To do that, we first initialize the file, then make it available in `/dev/fs`:

```
# set up file for encryption. Set password.
disk/cryptsetup -f mydisk
# make file available as /dev/fs/mydisk
disk/cryptsetup -i mydisk
```

After doing that, the decrypted disk file will be available as `/dev/fs/mydisk`.

2.2. gefs

With our virtual disk available in `/dev/fs/mydisk`, let's use it:

```
# ream the disk, with $user as the owner
gefs -f /dev/fs/mydisk -r $user
# srv the disk as /srv/mydisk and /srv/mydisk.cmd
gefs -f /dev/fs/mydisk -n mydisk
```

With that set up, we can mount the disk and use it:

```
mount -c /srv/mydisk /n/mydisk
# do something
```

2.3. Shutting down the filesystem

To shut down the disk and remove it from `/dev/fs`, we first have to remove the only process that accesses the file `/dev/fs/mydisk` by shutting down gefs, then we can remove it from `/dev/fs`.

```
unmount /srv/mydisk
# stop gefs
echo halt > /srv/mydisk.cmd
# remove from /dev/fs
echo del mydisk > /dev/fs/ctl
```

When gefs is still running while you remove the disk from `/dev/fs`, fs(3) will wait until the file is not used anymore, and then remove it.

Regarding actual use: I haven't used this system yet. It is possible that it's very slow, but I doubt that. Gefs could eat your data, so have a good backup solution.

Sun, 23 Feb 25 10:57:05 CET :

3. ChromaGun 2: Dye Hard

Disclaimers: Right now, there's only the demo freely available, so the review is about that only. Furthermore, since ChromaGun is about colors, it *has* a specific colorblind mode, which I didn't test, and I'm also not the right person to test this.

ChromaGun 2 (Demo!) is a puzzle platformer which surprised me with a good story and challenging puzzles, especially for a demo. Often enough, the story and the complexity of the puzzles in a demo are reduced to a bare minimum, not showing too much to make people buy the game. In this case however, the demo is perfect for exploring what the game is about. It even has enough replay value due to achievements and collectibles.

The demo teases a great and deep **story** which can't hide behind other great puzzle platformers, most notably *Portal*. While the story shares some similarities with Portal ("Testing"), it is clearly not a clone. In the story, the player is guided through different dimensions with very distinct styles. Each dimension has its own sections with puzzles to solve, and a different narrator. Judging from the demo, it seems like there's not much choice the player has about the story; there are no branches, no different endings. It would be easy to spoil more parts of the story, but it's even easier to say: Just play it—it's worth it!

Gameplay-wise, it looks very similar to the first part *ChromaGun*, though I can't really tell as I haven't played it. The player has the ability to color specific plates and objects in the level using the Chroma Gun. These colored plates will behave like magnets for various gameplay objects with the same color. The challenge is to color the correct plates in the correct order to manipulate the level so we can finish this section and continue.

The colors themselves are oriented around three primary colors (red, yellow, blue) and their secondary colors (orange, green, magenta). Those secondary colors can be achieved by mixing the primary colors accordingly.

It's worth noting that at some point the plates and objects are "overcolored" and just turn black, deactivating any magnetic behavior. This, and the fact that some objects can't change their color, is used as an additional puzzle challenge.

Throughout the demo there are challenges of varying degrees. While some challenges can be solved very easily by just looking around and coloring two objects, others require careful thinking and also thinking outside the box. One challenge even let me pause playing to continue at a later point. Yes, the demo is not just made for a single testing session—you can enjoy it multiple times!

The **graphics** of the game are kept simple, there's no over-realism or hyper-realism. The image is clear to read, there are no strange lighting artifacts or blurring happening. Various effects and animated objects fill the otherwise almost sterile levels with life. And did I notice some very well known Niagara effect somewhere?

Other than that, it's hard to judge the game graphics based on the demo and in total. Each dimension has its own art and graphical style, one going into a more organic direction and the other looks like a comic. Each style however looks very well thought out and matches the dimension.

As a game developer, it would be quite interesting to take a look behind the scenes and see how they achieved those different styles. And of course, I noticed a few things that I'd have done differently.

In total, the ChromaGun 2: Dye Hard demo is a wonderful demo that showcases what a demo in 2025 can be. It gives a good taste of the story, and has a lot of replay value for a demo by making use of achievements and collectibles. The gameplay covers a nice range of easy and complex puzzles, I sure hope they can find the right balance in the final game. Pixel Maniacs, good job!

- https://store.steampowered.com/app/2982340/ChromaGun_2_Dye_Hard/

  Comment: `https://pleroma.envs.net/notice/ArPM0sC5Xp2tGkgQgS`

Fri, 21 Feb 25 16:07:46 CET :

4. Mail Server DKIM

Some mail providers want it, others demand it: DKIM.

Upas is quite an old mail system, but it *has* dkim support. However, documentation for upas in general is rare, so I'll try to note down how to sign your outgoing mail in a 9front mail system. This post ist not only for you, but also for me in five years.

4.1. Theory: DKIM on Plan 9

Upas is distributed with an additional tool `upas/dkim`, which we will use here. The tool expects the private key in factotum. How you get the key into the factotum is up to you as it depends on various factors. I'll just show you which key to generate and how to use it.

DKIM uses your domain and a specific *selector* as an identifier. While it is pretty clear what the domain is, the selector is just a name for a specific key. It is possible to have multiple DKIM keys, and this is sometimes needed when rotating your keys.

Everything else is just calling `dkim` in your `remotemail`.

4.2. Implementation

To generate keys, run the following commands:

```
auth/rsagen -b 2048 -t 'service=dkim role=sign hash=sha256 domain=example.com'
  > dkimprivatekey
auth/rsa2asn1 -f spki dkimprivatekey | auth/pemencode DKIM >dkimpubkey
```

This will generate the private key you should feed into the factotum, as well as a public key file in PEM format.

We don't need the PEM format specifically, but it's an easy way to create a Base64 encoded version of the public key, which is what we need. Just forget about the specific and only copy the key itself to the DNS entry.

The DNS entry must be a TXT entry named `SELECTOR._domainkey.example.com` with the content: `v=DKIM1; k=rsa; p=YOURPUBLICKEY`.

This DNS entry will be used by the receiving servers to verify your mail. Keep note of the *SELECTOR* as it is the name of this specific key, and you'll use it to tell the receiving server which key you used for signing.

To sign your mails, open your `/mail/lib/remotemail` file and edit the call to `smtp` with something similar to this:

```
/bin/upas/smtp -f -C -s -h $fd $addr $sender $*
   | /bin/upas/dkim -s SELECTOR -d example.com
   | /bin/upas/smtp -C -s -h $fd $addr $sender $*
```

You can see, your mail is processed by two calls to `smtp`, with a call to `dkim` in between. The first call doesn't *send* the mail, it only processes it (the `-f` flag) to add additional headers.

The call to `dkim` then processes the headers and adds the DKIM signature header to your mail.

Last, the second call to `smtp` finally sends the processed mail to the receiving server.

Comment: `https://pleroma.envs.net/notice/ArLe4cGFkHavYUj4lM`

Mon, 16 Aug 21 10:36:56 CES :

5. HTTPS on 9front

I was able to switch my website hosting to 9front completely! This is thanks to ori's aclient (acme client which works with letsencrypt). This change makes my website deployment much easier since I'm fully writing on 9front and also deploying to 9front. The only thing missing is that the source repository is on github, but that's fine since we also have git9.

I'm writing this short blog post on my smartphone via drawterm for android, and I'm sure we'll get a proper documentation for how aclient works (btw the man page is very well documented), so I'll just add some short note about how to use the certificate with tcp80 and tlssrv.

`/rc/bin/service/tcp443`:

```
#!/bin/rc
/bin/tlssrv -c/sys/lib/tls/acmed/mydomain.tls.crt /bin/tcp80
```

Fri, 18 Dec 20 15:31:32 CET :

6. Automatically save sent files in "Sent"

Since it is what many mail clients do it might be helpful for other people to have this. As for me, I like to have all my outgoing mails automatically saved in a *Sent* directory, so that's what I want to do.

There are multiple ways to do this. I want to present a very simple way without sending your mail to yourself or something like that.

Outgoing mail is processed via `upas/send` or `/mail/box/$user/pipefrom`, if that exists. We use this feature to build or own little filter script for outgoing mails.

The script itself is very simple. We just need a temporary place to store the mail message, then save it in the `Sent` directory and forward it to the normal send routines:

```
#!/bin/rc
rfork en
# see /sys/src/cmd/upas/filterkit/pipefrom.sample .
bind -c /mail/tmp /tmp
TMP=/mail/tmp/mine.$pid
cat > $TMP
/bin/upas/mbappend /mail/box/<yourusername>/Sent < $TMP
/bin/upas/send $* < $TMP
```

Of course you need to create the directory `/mail/box/<yourusername>/Sent` and exchange `<yourusername>` with your `$user` and make this script executable.

The `Sent` directory needs read and write access for your own user, which should be fine with the defaults.

If you want unauthenticated users to send mails to that directory you need to make this directory world-writable.

With these adjustments you can send mails with acme/marshal and they are automatically saved in your `Sent` mailbox.

Tue, 15 Dec 20 11:33:54 CET :

7. 9front on Lenovo Thinkpad Twist

A few weeks ago I removed archlinux from my remaining machine. I noticed how the new lenovo keyboards aren't good and the trackpoint is crap. That's why I still prefer the Thinkpad T61, even without battery.

Anyways, I'll try to describe the process of the installation. The installation itself went according to the FQA, I'll just add some notes.

7.1. Process

First I had to disable UEFI completely and switch to legacy BIOS. I know 9front can handle UEFI somehow, but I never got it working on any machine. To make 9front work with legacy BIOS I had to change the SSD layout from GPT to MBR. This was possible, just remove all partitions and use the command line to create DOS partitions. Then the SSD was detected as MBR/non-GPT and I could proceed with default installation.

After installation I needed to get WIFI working. Thanks to 9front developers I was able to use BSD drivers as documented in the FQA. In my case I just grabbed the `iwn-2030` drivers, placed them in `/lib/firmware` and built the kernel from scratch.

Enabling ACPI in the `plan9.ini` and starting `aux/acpi` went without errors, only `bad opcode` warnings showed up. Still, everything works as expected, so I didn't investigate further.

7.2. Issues and Troubleshooting

I had exactly **two** issues. The first is the `bad opcode` as described earlier.

The second is a big surprise. Backlight controls work out of the box! I know older machines handle this directly without using the operating system, but this was a modern machine. Still it worked with only one tradeoff:

It always prints *lapicerrors* on the console. I didn't find a good way to disable them, so I just added a hidden window in my riostart that just `cat /dev/kprint` so the errors don't fill the screen.

7.3. Bonus: conntosrv

As a bonus I have a small script that saves me lots of installation time. I have a server with my `$home` directory, including some configuration in my `lib/profile`. On my terminals (laptops) I just work in my `$home` like it was right there on my machine.

To make this happen I placed the little script in my terminal's `/cfg/$sysname/conntosrv` and called in the `/cfg/$sysname/termrc`.

The script contains:

```
#!/bin/rc

echo -n 'connect to server: '
server=`{read}

if(~ $#server 0){
        echo not connecting to services >[1=2]
        exit
}

if(! test -e /net/dns)
        ndb/dns -r

auth/factotum
for(i in $server){
        rimport -Cc $i /n/$i
}
bind -c '/n/'^$server(1)^'/usr/'^$user /usr/$user
```

As you can see, the scrips connects to all servers you input at the prompt. It takes the first to be your `$home`, all others are imported to `/n`.

Wed, 29 Jul 20 10:32:43 CES :

8. Restrict RCPU User Access to Groups

This is how to restrict user access to groups. You can use this to enable `rcpu` access for all users of a specific group. All other groups will not be allowed.

To allow access only to `sys` group members: adjust your `/rc/bin/service/tcp17019`

```
#!/bin/rc
userfile=/adm/users
fn useringroup{
        grep $1 $userfile | {
                found=0
                while(~ $found 0 && line=':{read}){
                        if(~ $line(2) $2){
                                found=1
                        }
                }
                if(~ $found 1)
                        status=''
                if not
                        status='not found'
        }
}
if(~ $#* 3){
        netdir=$3
        remote=$2!'{cat $3/remote}
}
fn server {
        ~ $#remote 0 || echo -n $netdir $remote >/proc/$pid/args
        rm -f /env/'fn#server'
        . <{n='{read} && ! ~ $#n 0 && read -c $n} >[2=1]
}
exec tlssrv -a /bin/rc -c 'useringroup $user sys && server'
```

This checks if the user is in group `sys` and only then calls the `server` function. Otherwise the connection is terminated.

This is especially useful if you want a CPU server to expose filesystems *and* have cpu access for administrators only.

Thu, 16 Jul 20 09:44:56 CES :

9. lib/profile quick hack

Some smaller change that can change your life.

There are reasons why you not run *rio* in your lib/profile. For me the main reason would be: You can no longer use `rcpu -c commands` in your shell. Rio opens and there you are, stuck in front of a gray background.

My solution:

```
case cpu
   # … lots of stuff …
   rcpucmd=‘{cat /mnt/term/env/cmd >[2]/dev/null}
   if(~ $#rcpucmd 0)
      rio
   # … lots of stuff …
```

Now I can `rcpu` and have my rio, or `rcpu -c command` and run the command without leaving my shell.

Thu, 16 Jul 20 08:41:14 CES :

10. Mail Server Configuration

Recently I installed my mail server on 9front. Most of the time I followed the guide in the FQA, but still there are things to explain. In this document I'll go through the section of the FQA and annotate things.

Right at the beginning the FQA mentions how the executing user needs write permissions for the mailboxes. This is *very important*! If upas can't write the mailboxes the mail server will *not* accept incoming mail!

In my setup I can skip all DNS stuff, because I have my DNS hosted somewhere else. Make sure to add proper MX records as well as (at least) an SPF record.

10.1. /mail/lib/smtpd.conf

To make things short, here are the necessary lines in my setup. The server handles authenticated incoming mail for sending to other providers as well as incoming mail for local accounts.

```
defaultdomain    sirjofri.de
norelay          on
verifysenderdom  on
saveblockedmsg   off
ourdomains       sirjofri.de
```

Note that the server is no relay for unauthenticated/untrusted requests, it will still relay if you authenticate.

At this point it might be a good idea to check your user password. Use `auth/changeuser` to add *Inferno/POP secrets* to your user accounts. Use these passwords to authenticate to the smtp server.

10.2. /mail/lib/rewrite

The program that handles sending mail uses this file to rewrite mail addresses. This file is responsible for filtering out local mail as well as sending other mails to the mailer.

In my setup I added three aliases:

```
pOsTmAsTeR    alias postmaster
aBuSe         alias abuse
wEbMaStEr     alias webmaster
```

Use regular expressions to define your domain:

```
de!(.*)   alias 1
sirjofri.de!(.*)       alias 1
```

For translating mails I added one more rule for mail address *tags*. These tags are in the form of *user+tag@example.com*. Official specifications say that everything behind that "+" must be ignored, but it can be used to automatically sort incoming mail into folders. I do this, by the way, so I describe here, how.

We need rules for those plus signs:

```
# The other translate rules are default
```

For delivering local mails, I added extra rules:

```
local!(.+)+(.+)  |  "/bin/test -d /mail/box/1/2  /bin/upas/mbappend /mail/box/1/2 || /bin/upas/mb
local!"(.+)+(.+)  |  "/bin/test -d /mail/box/1/2  /bin/upas/mbappend /mail/box/1/2 || /bin/upas/m
# leave the other rules untouched.
```

With this settings, mails to user+*tag* will be checked. If a mailbox folder for *tag* exists, mail is sent to this folder. Otherwise it is sent to the user's default inbox. **Note:** I tested, but this *does not work* with aliased mail. If my aliasmail changes *userA* to *userB*, mails to *userA+tag* will be rejected! If you know how I can make this work, feel free to send me a mail.

10.3. /mail/lib/names.local

This file is pretty easy. Just add your alias mail addresses:

```
postmaster  sirjofri
webmaster   sirjofri
abuse       sirjofri
```

10.4. /mail/lib/remotemail

```
#!/bin/rc
shift
sender=$1
shift
addr=$1
shift
fd=‘{/bin/upas/aliasmail -f $sender}
switch($fd){
case *.*
    ;
case *
    fd=sirjofri.de
}
exec /bin/upas/smtp -h $fd $addr $sender $*
```

10.5. SMTP over TLS

I don't use port 587. I use 25 for this. Mail servers relay mails to this port by default, so it makes sense.

/rc/bin/service/tcp25

```
#!/bin/rc
user=‘{cat /dev/user}
exec /bin/upas/smtpd -f -E -r -c /sys/lib/tls/cert -n $3
```

Don't forget to create your TLS certificate!

10.6. IMAP4 over TLS

I did this exactly like the FQA. See there.

10.7. No.

At this point I stopped. I did not configure ratfs and have no spam handling right now. It doesn't really matter for me, because nobody knows me and I don't use that mail address to register anywhere.

Links:

→ https://fqa.9front.org/fqa7.html#7.7

Wed, 1 Jul 20 18:30:46 CES :

11. Guided Replica

Today I installed *replica*(1) on my VPS. I noticed that I can write some helper scripts around it and here they are.

You can download them from `https://sirjofri.de/files/guidedreplica`.

You can install it like that:

```
# bind your client $home to /n/rclient
# bind your server $home to /n/rserver
hget https://sirjofri.de/files/guidedreplica/guidedreplica.rc | rc
# follow the prompts
```

This will also install two helper scripts to `$home/bin/rc/replica/`. Reproto copies one proto over the other. You can choose which one you want to keep. Reupdate is helpful if there are update-update errors. It should automatically solve them (untested, but should work).

Update: *replica*(1) has issues. Often it does a bad job tracking changes, leaving removed files there and vice versa. I never encountered data loss, only inconsistencies in the copies.

Many people use *mkfs*(8), which does not overwrite changed files. At some point I will build some scripts around it and use that instead of *replica*(1).

(Files: `https://sirjofri.de/files/guidedreplica/README https://sirjofri.de/files/guidedreplica/guidedreplica.rc`)

Mon, 29 Jun 20 18:39:39 CES :

12. 9front on Netcup VPS

Today I installed 9front on a Netcup VPS. Here are some notes if you want to do it yourself.

I used the smallest VPS option. Currently, that's "VPS 200 G8". It costs like 2.69 Euro, but you might be able to find some way to make it cheaper.

After ordering it might take some time until the server is up and ready. By default debian was installed in a GPT, we can ignore that.

Before we can install our custom ISO we first must upload it somewhere. This is done via FTP (you get the access data from the SCP), I used windows default file explorer (`ftp://user@address`, enter password). Copy the 9front ISO in `/cdrom`. This will take some time.

Meanwhile you can delete the virtual disk and create a new one. You need your SCP password for this. This step is necessary to remove the GPT. Of course you could manually reformat the disk, but deleting the disk will save time.

In the settings you can virtually insert the iso as a DVD and verify the boot order (DVD first). Start up the machine and switch to the web VNC display.

At this point you can proceed with the default 9front installation described in the fqa. Don't forget to install the MBR and activate the partition. Otherwise there are no additional special steps besides manually configuring the `/lib/ndb/local` after installation. In my case I made an auth server.

Currently it seems to work fine. I installed the machine today, so there might be some issues I didn't find yet.

Tue, 23 Jun 20 15:00:45 CES :

13. changeblog feed — social media[2]

RSS is still a thing.

Yes, there are more modern alternatives, like Atom or fancy json feeds. What I want to say is, feeds are still a thing.

That’s why you are now able to read my changeblog as an Atom feed.

Now I just need to find enough time to write my posts.

Fri, 22 May 20 02:00:00 CES :

14. I use 9front

Today I want to share with you, that I use the plan9 distribution “9front” as my main computer.

Of course there are things that are almost impossible to do there, for example: all gamedev related stuff. This is of course an issue, because I am a game developer. I still have my windows machine with relevant tools, so I can still fiddle around with those complex things.

For gaming I also use my windows machine or some game console. Yes, there are a few games on plan9 systems.

Also most online services use javascript and heavy styling of webpages, so I also use a modern computer with a modern browser. Mothra is fine for doing basic research stuff, but in 2020 it’s almost impossible to actually do things on the web.

Anyways, let me tell you that I don’t really miss anything on plan9. I can write documents, check my email stuff, chat with people, and step by step it becomes more usable. The community is helpful and provides more applications. The system runs stable, the user interface is consistent and good to look at. Colors don’t jump in your eye and want to kill you and there’s catclock(1), our friendly companion.

Fri, 10 Jan 20 01:00:00 CET :

15. Revived

I updated my website to Uberspace 7, but not only this: I changed the whole webpage to make it more nine-friendly.

My whole webpage management system is completely 9 based. I use oridb's git9 implementation and plan9 tools, mk, sed, cat, …

I also decided to change the main language of the website to English.

Fri, 8 Jun 18 13:00:00 CES :

16. Hacknet Game Review (German)

Hacknet ist ein storylastiger Hacking-Simulator. Obwohl das „Hacking“ doch recht weit von der Realität entfernt ist, kommen manche Prozesse dem „echten“ Hacken doch sehr nahe. So muss der Spieler manche Dinge lernen (oder bereits im Vorfeld wissen), sich die Verwendung neuer Tools anlernen und diese Tools geschickt kombinieren, um an sein Ziel zu kommen. Manchmal reicht selbst das nicht aus, es gibt immer einen Hacker, der besser ist als man selbst. Dies ist vom Spiel aber gewollt und dient dem Fortschritt der Story.

Überhaupt ist das „Hacking“ recht einfach – es soll ja auch Spaß machen; der Schwierigkeitsgrad ändert sich Abschnittsweise und dient dem Spielerlebnis. An manchen Passagen gibt es auch ein gewisses Frustpotenzial, was jedoch der Realität noch näher rückt und das Spielvergnügen nicht mildert. Das stetige Befehle-eingeben und Systeme analysieren behindert das Erleben der Story im Übrigen überhaupt nicht, eher im Gegenteil: Die Tätigkeit lässt einen noch viel mehr in die Rolle des Protagonisten schlüpfen. Der Protagonist ist übrigens ein namenloser Hacker, der mit keiner Persönlichkeit ausgestattet ist. Für manche Spiele mag das ein Nachteil sein, da man nicht in die Rolle des Helden schlüpfen kann, in *Hacknet* jedoch lädt es den Spieler dazu ein, sich selbst in das Spiel hineinzusetzen. Selber zu hacken, statt den Helden hacken zu lassen.

Die Story von *Hacknet* hat auch vieles zu bieten: Fortschritt, Rivalität und die mysteriösen Handlungen eines großen Technik-Konzerns bieten dem Spieler Herausforderung und den Wunsch, mit seinem neu erlangten Können die Spielwelt zu verändern. Mehrere Nebenquests dienen dazu, sich in die Spielwelt einzufinden und die Arbeitsweise der Hackergruppen kennenzulernen.

Während des Spielens fiel mir auf, dass das Spiel durchaus an der vierten Wand gekratzt hat. Manchmal verschwamm die Grenze zwischen Spiel und Realität. Dafür ist die Länge des Spiel jedoch sehr hilfreich: Es ist trotz seiner Tiefe nicht sehr lange. Und der Preis scheint angemessen. Insgesamt ein sehr schönes Spiel, das den Spieler in ein technisches Abenteuer lockt und mit seinem Immersionspotenzial überrascht.

(Ursprünglich gepostet auf Steam.)