Hochschule der Medien
Druck und Medien
Praxissemester
Prof. Dr. Ansgar Gerlicher
Sommersemester 2015

Per Guth

10. Januar 2016

Praxissemesterbericht

Einleitung

Ich verbrachte mein Praxissemester im Sommersemester 2015 bei einer zwei Mann starken Firma in Bietigheim-Bissingen. Die Firma Waldmann EDV, namentlich Dipl.-Ing. Gerhard Waldmann und Dipl.-Inform. Thomas Waldmann, existiert in ihrer jetzigen Rechtsform GbR seit über 10 Jahren. Zeitweilig umfasste dir Firma auch einen dritten Gesellschafter.

Entsprechend der beiden Slogans auf der Firmenwebseite (https://waldmann-edv.de) „Ihr Partner für EDV-Dienstleistungen aller Art.“ und „Ihre Quelle für EDV-Produkte aller Art.“ steht die Firma hauptsächlich dem lokalen Mittelstand, Kleinunternehmen und Privatpersonen zu Diensten.

Da die Firma durch Thomas Waldmann auch in verschiedenen Open Source Software Projekten investiert ist, ergeben sich mit gewisser Regelmäßigkeit über das Jahr hinweg auch internationale Auftrage bei denen Kunden um Assistenz bei der Einrichtung, Wartung und Anpassung von Software bitten.

Auswahl der Praxissemester-Stelle

Eigentlich plante ich darauf, mein Praxissemester im Ausland bei einer der mittlerweile unzähligen Firmen zu machen, die sich aktuell im Internetzugangs-Anonymisierungsbereichs (Stichwort: VPN-Anonymizer) tummeln. Ich hatte hierfür auch schon eine kleine Vorauswahl an Kandidaten getroffen und verfolgte regelmäßig deren Blogs und Entwicklungen im Themenfeld.

VPN-Anonymizer sind sowohl technisch als auch rechtlich eine spannende Angelegenheit. Technisch wird oftmals eine internationale Infrastruktur an angemieteten Servern betrieben. Aber auch in den Fällen, in denen sich die Infrastruktur auf ein Land beschränkt, sehen sich die Dienstleister häufig großen Angriffen gegenüber gestellt. Als gutes Beispiel kann die spannende

Darstellung einer „DNS amplification“ Attacke durch iPredator aus dem Jahr 2013 dienen: https://blog.ipredator.se/2013/11/dns-amplification-attacks-weapons-of-internet-mass-destruction.html

Alternativ liebäugelte ich auch immer mit einer Stelle bei der Free Software Foundation, kurz FSFE, in Berlin. Das Thema über das Praxissemester hinweg wäre dort die Open Source Advocacy gewesen: die Zusammenarbeit mit NGOs und die Öffentlichkeitsarbeit zur Verbreitung von Open Source Software.

Am Ende blieb ich dann doch im Raum Stuttgart. Denn so verlockend die internationale Bühne auch klingt, so abgehoben erscheint es auch, wenn man sich die über das eigene Studium hinweg reichende Zukunft vorstellt.

Fasziniert an der Firma Waldmann EDV hatte mich sowohl die kleine Größe (zwei Mann, GbR) und der Bezug zur Open Source Bewegung. Ich hielt es für mich oft aus grundsätzlichen Überlegungen heraus für ausgeschlossen, bei einer großen Firma zu arbeiten und so war ich sehr daran interessiert, das Leben eines Selbstständigen kennen zu lernen.

Und dass bei Waldmann EDV die Aussicht darauf bestand, mich über das Semester hinweg weiter in die Open Source Bewegung vertiefen zu können - und vor allem mit Open Source Software arbeiten zu können - besiegelte meinen Vorsatz dort anzuheuern.

Die Arbeitsumgebung

Waldmann EDV war angesiedelt im Industriegebiet von Bietigheim-Bissingen, unweit vom dortigen Bahnhof. Das Büro umfasste 2 Arbeitsräume, 1 Serverraum, einen Schulungsraum und einen Rezeptionsbereich. Der Server wurde mit Debian Linux betrieben. Die Arbeitscomputer wurden sowohl zum Teil mit Windows 7 als auch mit Ubuntu Linux 14.04 LTS betrieben. Das Büro ist über eine DSL-16000er Leitung mit dem Internet verbunden.

Die Arbeitsaufgaben

Zusammenbau und Konfiguration von Computern

Eine sich immer wieder stellende Aufgabe war das Montieren von Desktopcomputern. Hierbei

wurde besonders auf fachmännische Verkabelung im Gerät geachtet. So wurden alle losen Kabel noch einmal extra gebunden und von den Herstellern so gelieferte Kabelbinder entfernt, sofern diese Metall beinhalteten. Die Systeme wurden von den Komponenten her immer so angelegt, dass entweder gleich eine Spiegelung der Festplatte (RAID 0) möglich war, oder das System entsprechend ergänzt werden konnte.

Nach dem Zusammensetzen der Teile wurde immer das BIOS geprüft und zusammen mit dem RAID vorkonfiguriert. Hierbei war die Versionsnummer zu prüfen als verschiedene Systemparameter zu konfigurieren. Üblicherweise entsprachen sich die ausgelieferten BIOS Versionen so dass rein die Konfiguration notwendig war.

Die Konfiguration umfasste zum Beispiel das Einstellen der Lüfter auf „leise“ als auch das setzen von Benachrichtigungsereignissen für mögliche Fälle von Systemüberhitzung.

Wartung von Kundencomputern (Privatkunden)

Häuft hatten wir Laptops vor Ort, die uns Kunden zur Wartung vorbei brachten. Teilweise handelte es sich um Hardwareprobleme, teilweise um Software oder Konfigurationsprobleme. Je nach Fall gab es Einweisungen vor Ort oder eine Problembehebung durch uns über mehrere Tage hinweg in Abwesenheit des Kunden. Da es sich ausschließlich um Windows Systeme handelte bestand immer ein großer Aufwand, was die Systemaktualisierung anbetraf. Wir wählten hier, wie auch bei anderen Aufgaben solcher Art immer den Weg, parallel zu arbeiten. Damit konnten wir uns sowohl die eigene Vernunft bewahren (Wahnsinn und Windows beginnen sicherlich nicht zufällig mit dem gleichen Buchstaben), als auch den Geldbeutel des Kunden schonen.

Wartung von entfernten Systemen

Waldmann EDV war für verschiedene Firmennetzwerke und deren Backup-Infrastruktur zuständig. Bei Problemfällen oder bei Veränderungswünschen wurden die Kunden entweder vor Ort aufgesucht oder es wurden Fernwartungssitzungen durchgeführt. Der betreute Einzugsbereich reichte während meines Praxissemesters bis Heilbronn. Es kam auch zu ein paar Fernwartungssitzungen, in denen wir Computer im Ausland betreuten.

Zu den üblichen Problemen zählten hierbei funktionsuntüchtige Festplatten, die wir über ihren SMART Status auf Fehler prüften.

Schulung von Kunden

Gerade bei Vertretern der älteren Generation wurde gerne die Möglichkeit der Schulung durch uns wahrgenommen. Es ging dabei zumeist um einfache Problemstellungen und Nutzungsfälle, die die Kunden sowohl technisch gelöst als auch in ihrer Lösung vorgeführt bekommen wollten. Man denke zum Beispiel an das Nutzen von Tablets in Verbindung mit Desktopcomputern: wie bekommt man die Bilder vom mobilen Gerät auf den Standcomputer? Welcher Mobilfunktarif bietet sich für den skizzierten Nutzungsfall an?

Weiterentwicklung der eigenen Infrastruktur

Da mit jedem neuen Computer wieder und wieder die gleichen Systemupdates herunter geladen werden mussten, hatten wir die Idee, ein Caching-Proxy im eigenen Netzwerk einzurichten. Hierfür bot sich die Verwendung eines Raspberry-Pis 2 an, der mit einer 64GB Speicherkarte ausgestattet wurde. Hierdurch konnte die Downloadgeschwindigkeit an den Endgeräten im Caching-Fall maßgeblich verbessert werden. Es zeigte sich, dass Windows ohne Probleme auf unsere Infrastrukturänderung anspringen konnte, Ubuntu jedoch Probleme hatte. Auch Firefox bedurfte besonderer Aufmerksamkeit, wenn er den Cache ansprechen sollte.

Softwareentwicklung (für intern)

Viele der Kunden von Waldmann EDV kamen der Empfehlung nach, ihre Daten regelmäßig zu Backupen. Beziehungsweise unternahmen zumindest den ersten Anlauf. Die einzigen tauglichen Softwareangebote in diesem Bereich konnten uns aber leider nicht ausreichend zufrieden stellen. Auch der jüngste Anbieter am Markt konnte die Nutzungsprobleme der Vorgänger nicht ganz ausräumen. Hinzu kam ein aggressiveres Vorgehen des Herstellers zur Sammlung von Nutzerdaten.

Aus all diesen Gründen verfolgte Thomas Waldmann schon seit längerer Zeit ein Open Source Software Projekt namens attic (https://attic-backup.org/). Dabei handelte es sich um eine deduplizierende, mit einem rolling Hash arbeitende Backupsoftware die sowohl mit hoher Geschwindigkeit als auch sauberem Code glänzen konnte.

Da attic jedoch mit einer nur sehr langsamen Geschwindigkeit fortentwickelt wurde und auch keine Communitybeteiligung gewünscht wurde, kam es zu einem Fork des Projekts durch Thomas

Waldmann unter dem Namen BorgBackup (https://github.com/borgbackup/borg).

Es stellte sich die Frage, wie die neue Community organisiert werden sollte und welche Standards eingehalten werden sollten. Wir entschlossen und für ein simples Schema nach dem mit jedem neuen Major-Release die Abwärtskompatibilität gebrochen werden konnte. So hofften wir, eine hohe Entwicklungsgeschwindigkeit ermöglichen zu können. Das Projekt gewann auch schnell an Popularität und ist heute mit 435 Stars auf Github versehen.

Da wir langfristig den Einsatz beim Kunden vor Ort planten, entwickelten wir eine simple, auf einem lokal laufenden Webserver basierende Oberfläche, die der regelmäßigen Überwachung des Backupzustands dienen sollte (BorgWeb, https://github.com/borgbackup/borgweb).

Neben BorgBackup und dem Frontend gab es immer wieder kleinere Programmieraufgaben um die eigenen Arbeitsabläufe zu verbessern.

Softwareentwicklung (für extern)

Wir wurden von einem schweizer Unternehmen beauftragt einen PDF für das Web so anzupassen, dass er dem Nutzer nicht mehr die Möglichkeit bietet, die PDF zu speichern. Es kam dann auch noch Watermarking und ein rudimentärer Schutz vor dem einfachen Herauslösen der PDF aus dem Wiedergabeprogramm hinzu. Hier hatte ich zum ersten aber auch einzigen mal explizite moralische Bedenken was die innere Zielsetzung unserer Arbeit anbelangte („Einschränken statt Helfen“).

Wie bei dem BorgBackup Frontend war unsere Arbeitsaufteilung hier die, dass ich mich um die JavaScript Seite und Thomas Waldmann sich um die Backend Python Seite kümmerte.

Dokumentation von Lösungen

Wunderbar war auch, dass ich mir immer wieder die Zeit nehmen konnte, um Gelerntes in kurzen Texten google-gerecht abzupacken. Dies diente sowohl unserer eigenen Dokumentation von Lösungswegen als auch unserem Interesse, unsere Erkenntnisse, die wir selbst oft nur durch googlen gewinnen konnten, wieder mit der Allgemeinheit zu teilen.

Die von mir verfassten Texte finden sich im Anhang dieses Dokuments und sind immer aus einem konkreten Anwendungs- oder Problemfall vor Ort entstanden.

Schluss

Es hat mir ganz gut gefallen, bei Waldmann EDV in Bietigheim-Bissingen. Ich hatte eine schöne Zeit mit vielen technischen Diskussionen und interessanten Aufgabenstellungen. Thomas Waldmann, mit dem ich die meiste Zeit verbracht hatte, war eine sehr umgängliche und angenehme Person. Besonders gefiel mir auch, dass wir in der Zeit auch über die Arbeitszeit hinaus einige Aktivitäten gemeinsam wahr nahmen. Zum Beispiel besuchten wir Linux- und Computertreffs wie auch Computermessen. Ich habe vor allem auch einen Eindruck von der Welt der Freiberufler in diesem speziellen Bereich erhalten können, davon welchen Problemen sie gegenüber stehen aber auch welche Freiheiten sie genießen und habe jetzt mehr als Vorher ein Gefühl dafür, was es heißt, in diesem Bereich auf eigenen Beinen zu stehen.

Ob ich mit meiner Wahl im Kontrast zu eventuellen Möglichkeiten bei der FSFE oder bei VPN-Anonymizer Dienstleistern zufrieden sein kann? Ich denke, das wird die Zeit zeigen. Aktuell bin ich es. Gleichzeitig sehe ich aber auch, wie für viele Kommilitonen die Praxissemesterstelle bereits eine Anstellungsmöglichkeit für die Zeit nach dem Studium eröffnet hat. Dies kann ich sicherlich als tatsächlichen Nachteil anführen. Der Gewinn hingegen ist aber, dass ich eben *gerade* eine kleine Firma von innen kennen lernen durfte.

Measure real network speed (between two machines)

April 28, 2015 | 0 Comments | Tags: technology, linux, ubuntu, network, speed, measurement, bandwidth, RaspberryPI, wedv, berichtsheft

Sometimes you want to measure the real network speed between two machines without the interference of possibly slow storage device reading speeds. The following two tools will help you with that.

But when doing this on a RaspberryPI or similar device keep in mind what Jeff Geerling wrote:

> *However, for many real-world use cases, the Pi's other subsystems (CPU and disk I/O especially, since I/O is on a single, shared USB 2.0 bus) will limit the available bandwidth.*

Contents

Using preinstalled `netcat`

On Linux/Ubuntu you can use `netcat` for that:

1. On one machine open a data sink:
   `nc -v -l PORT > /dev/null`
2. On the other machine send some data:
   `dd if=/dev/zero bs=1M count=10 | nc -v HOST PORT`
3. Results will be displayed on the senders side.

Example results:

```
[18:11] pguth@pc ~ $ dd if=/dev/zero bs=1M count=10 | nc -v proxy 1234
Ncat: Version 6.47 ( http://nmap.org/ncat )
Ncat: Connected to 10.10.10.9:1234.
```

```
10+0 records in
10+0 records out
10485760 bytes (10 MB) copied, 5.12357 s, 2.0 MB/s
Ncat: 10485760 bytes sent, 0 bytes received in 5.19 seconds.
```

Using installable `iperf`

On Linux/Ubuntu you can get `iperf` from the default repositories:

1. On both machines install `iperf`:
   `sudo apt-get install iperf`
2. Make one machine play server:
   `iperf -s`
3. On the other machine start the measurement:
   `iperf -c HOST`
4. Results will be presented on both sided.

Example results:

```
[18:21] pguth@pc ~ $ iperf -c proxy
------------------------------------------------------------
Client connecting to proxy, TCP port 5001
TCP window size: 85.0 KByte (default)
------------------------------------------------------------
[  3] local 10.10.10.192 port 57176 connected with 10.10.10.9 port 5001
[ ID] Interval       Transfer     Bandwidth
[  3]  0.0-10.1 sec  21.1 MBytes  17.6 Mbits/sec
```

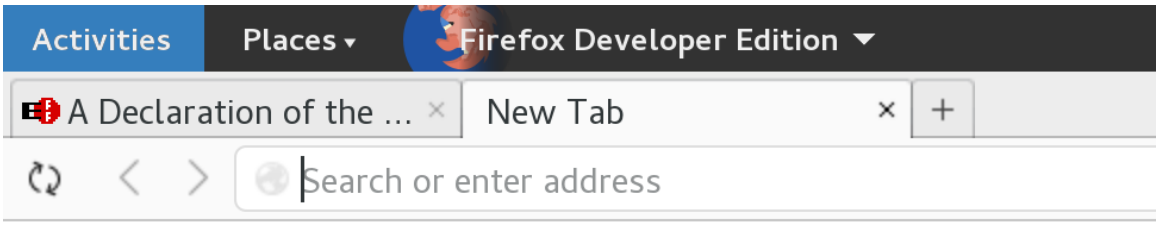
References

- Measuring Network Speeds with Netcat and Dd
- Getting Gigabit Networking on a Raspberry Pi 2 and B+

Gnome: Remove title bar when maximized

June 8, 2015 | 0 Comments | Tags: technology, linux, ubuntu, Gnome, shell, UI, customization, Fedora, Korora, berichtsheft

Hide the useless titlebar when a window/application is maximized:[src]



1. Edit `metacity-theme-3.xml`:
   `sudo nano /usr/share/themes/Adwaita/metacity-1/metacity-theme-3.xml`
2. Press `CTRL` + `w` and type `frame` and press `ENTER`.
3. Your file may differ but see that the found section in your file matches the following snippet with regard to the highlighted values:

```
<frame_geometry name="max" has_title="false" title_scale
="medium" parent="normal" rounded_top_left="false" round
ed_top_right="false">
     <distance name="left_width" value="0" />
     <distance name="right_width" value="0" />
     <distance name="left_titlebar_edge" value="0"/>
     <distance name="right_titlebar_edge" value="0"/>
     <distance name="title_vertical_pad" value="0"/>
     <border name="title_border" left="0" right="0" top=
"0" bottom="0"/>
     <border name="button_border" left="0" right="0" top
="0" bottom="0"/>
     <distance name="bottom_height" value="0" />
</frame_geometry>
```

- Ubuntu: Press `ALT` + `F2`,
  Fedora/Korora: Press `ALT` + `F2` and enter `r`, press `ENTER`.

Proxy auto-configuration for Windows and Linux

June 8, 2015 | 0 Comments | Tags: technology, linux, Windows, proxy, setup, networking, wedv, berichtsheft

Operation systems generally can digest and look for a `PAC file` (="proxy auto-config"). They look for it using `WPAD` (="web proxy auto-discovery protocol").

Windows 7 and 8 do that. Ubuntu can do that (can be activated within the `Network` settings). For Fedora and Korora it's the same.

The `PAC file` can look like this one which provides "PAC with network and domain whitelisting".

For `WPAD` to work automatically the file should be served using the the MIME type `application/x-ns-proxy-autoconfig`. Its location should be announced via `DHCP` and `DNS`.

The `DNS` should announce either an `A record` (="host record") or an `CNAME` for the domain name `wpad` which should resolv to the IP of the machine that serves the `PAC file`. All in all it should be possible to access it over port `80` using `http://wpad.[local_domain]/proxy.pac`.

Using `DHCP` the file can be made available using any address and port. On a Linux machine the responsible configuration file `/etc/dhcp/dhcpd.conf` could look like this:

```
option local-proxy-config code 252 = text;
...
subnet 192.168.0.0 netmask 255.255.255.0 {
  range 192.168.0.100 192.168.0.200;
  ...
  option local-proxy-config
"http://www.example.org/proxy.pac";
}
```

References

- Web Proxy Autodiscovery Protocol (wikipedia.org)
- Configuring Web proxy clients... (technet.microsoft.com)

X2Go bugfix `x2golistsessions: not found`

June 8, 2015 | 0 Comments | Tags: technology, linux, X2Go, bugfix, wedv, berichtsheft

X2Go throws the following **error** while trying to establish a connection:

```
Connection failed sh: 1: x2golistsessions: not found
```

Solution:[src] You maybe just skipped installing the core software and only got the GUI:

```
sudo apt-get install x2goserver x2goserver-xsession
```

XFCE HiDPI fix `everything too small`

June 8, 2015 | 0 Comments | Tags: technology, linux, UI, bugfix, XFCE, HiDPI, wedv, berichtsheft

Mate Desktop can do that right out of the box *and* features better UI/UX. Spare yourself the pain of XFCE - we gave up an used the HiDPI ready Ubuntu Mate.

What you can achieve

XFCE in general

1. **HiDPI theme**: http://opendesktop.org/content/show.php/Pupi+%28Xfce+High+DPI+%2B+Retina+Theme%29?content=165525

Firefox

1. **about:config**: layout.css.devPixelsPerPx
2. Adjust default zoom level

Thunderbird

1. **about:config**: layout.css.devPixelsPerPx

Ressources

- HiDPI: Xfce (wiki.archlinux.org)

Backup, extract serial and remove/replace Windows

June 9, 2015 | 0 Comments | Tags: self-defense, technology, Windows, proprietary, serials, backup, berichtsheft

Maybe you have bought a "`'puter`", maybe even a mobile one, and most certainly they have force-sold you a operation system that fits the dull-wittedness of the word. But then somehow reason creeps back in and starts anoying you... and then you just cannot can't switch to linux.

So here you go:

1. Spare you the hassle of talking to those "`'puter`"-guys at the local computer-discounter-monster-store in the case of warranty issues and backup the disk image of your freshly bought and so called ~~self-enslavement~~ operation system:
    - Plug a bootable USB stick and boot your favorite Linux.
    - Plug/mount/use the current USB stick to `dd` your drive: `sudo dd if=[DISK_PATH] of=[BACKUP_DRIVE_PATH] bs=4M`
2. Get your Windows serial out[1] - remember what they say: "`one mans trash...`"
3. ***Hopefully never:*** Restore the diskimage if you have a warranty case. Just repeat step one and swap `if=` and `of=` values.

And remember: always have a second device that runs SomaFMs Defcon Radio. 👍

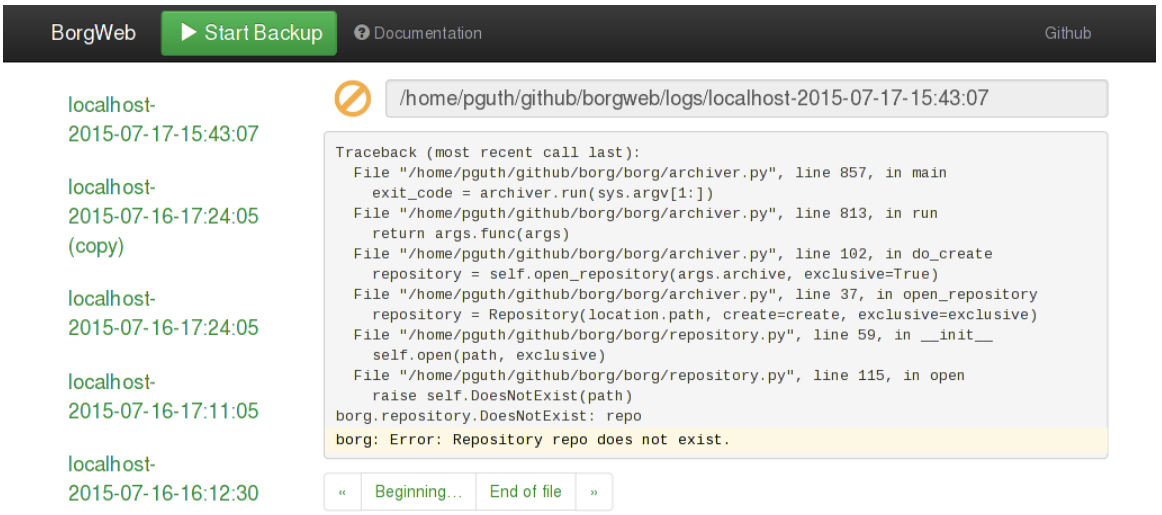
---

[1]: `sudo cat /sys/firmware/acpi/tables/MSDM`

Borgweb frontend development

July 22, 2015 | 0 Comments | Tags: technology, wedv, berichtsheft, development, coding, open source, JS

Having written Peertransfer last semester I now had the chance to try myself at another JS heavy-ish project: Borgweb a log file viewer for Borgbackup.



Borgwebs JS talks to a RESTful API to:

1. Get a list of log files (on the left).
2. Display the contents of log files.
3. Paginate between chunks of the log file (only request the part that is beeing displayed).
4. Display miscellaneous information about the log file.
    - Highlight erronous lines.
    - Indicate overall state of the backup that produced the log.
5. React to the height of the browser window and display only a suitable amount of lines.
6. React to height changes and re-render.

The first version I wrote was very complicated code-wise and it was quite hard and awkward to debug or even introduce new features. Having all code in one big file did not help. There was some aversion in our company

to use JS code structuring tools like Gulp and Browserify maybe because we underestimated the complexity these few features already introduce (for a beginner like me).

So a few discussion of algorithms later I decided to rewrite and use a more modular way of structuring my code and also tried hard to keep functions short.

Starting to use ES6 constructs I needed to transpile and decided to use Babelify. It's one of two contenders in that space but is the one that is community based. Building on previous attempts I could reuse/improve upon old build chain scripts and pretty much stayed with this version:

```
var lib = require('./gulpfile.lib.js')
...
var files = {
  js: './index.js',
  jsBndl: '../borgweb/static/bundle.js' }
...
gulp.task ('watch', function () {
  newBuildLog(); browserSync.init({ proxy: 'localhost:5000',
open: false })
  lib.generateBundle(files.js, files.jsBndl, true)

  gulp.watch([files.js, 'src/**/*.js'], function (ev) {
    if (ev.type == 'changed') {
      newBuildLog()
      log("Changed: " + ev.path)
      var ret = lib.generateBundle(files.js, files.jsBndl)
      setTimeout(function () { browserSync.reload() }, 150)
// todo
      return ret } }) })
```

What's neat for me about this is, that it watches all me used JS files and continuously rebuilds the `bundle.js` that is actually used by the browser. BrowserSync handles reloading the website when files changed. Since it spawns a local webserver I could use another machine remotely open the website via SSH (almost like shown here) and have BrowserSync handle the

reloading automatically - second monitor without having to plug cables *yay*.

When glueing code together I liked to keep it verbose so I had a separate export section at the end of every module like so:

```
module.exports = {
  noBackupRunning: noBackupRunning,
  pollBackupStatus: pollBackupStatus,
  ...
}
```

And also a section in my `index.js` where I made the functions globally available that needed to be callable from the frontend:

```
window.startBackup = borg.startBackup
window.switchToLog = logViewer.switchToLog
window.nextPage = logViewer.nextPage
...
```

Of course there are still open issues on Github. But we have a first working version and already have a PIP package.

So lessons learned here:

1. Write short functions.
2. Structure using modules.
3. Automatize if possible.

Fixing websites with Userscripts: Heise Preisvergleich

July 22, 2015 | Comments | Tags: technology, wedv, berichtsheft, Greasemonkey, Firefox, Userscripts, Open Web, questions

My boss uses Heises Preisvergleich a lot. It's a (to me somewhat awkward UI/UX-wise) german price comparison page. Typical german verbosity you might think. After all even in law *"Germans rarely find a rule they don't want to embrace"* as the Telegraph writes. That should give you a pretty good idea what our price comparison sites look like...

But we're not all that way :-) So we tried to fix it ourselves. It turns out there are some nice ways of doing that since the web is still free and open (will that die away with WebAssembly?).

We were mainly interested in the specification part of the site, where they list things like CPU and RAM. Since my boss is quite a Firefox fan I wrote a Greasemonkey script. It starts to run when the page has fully loaded and then replaces and transforms specification text passages using Regex. On my first try I made the error to really parse the HTML. After some time *Cpt. Obvious* hit me and made me use Regular Expressions to simply search and replace. That is a screenshot of the result (red marks changes introduced by my script; a dash means something got erased, just imagine a loat of bloat text where the dashes are):

ASUS Zenbook UX305FA-FB003H schwarz (90NB06X1-M00070)

— Intel Core M-5Y10, 2x 800MHz • RAM: 8GB • — 256GB SSD • **kein optisches Laufwerk** • — Intel HD Graphics 5300 (IGP), Micro HDMI • — 13.3", 3200x1800, non-glare, IPS • — 3x **USB3** • — WLAN 802.11a/b/g/n/ac, **BT** 4.0 • Cardreader: 4in1 (SD/SDHC/SDXC/MMC) • Webcam: 1.0 **MP** • — Windows 8.1 64bit • Akku: Li-Polymer • — 1.20kg • — lüfterlos

You can find all the code and some help setting up your development environment in the Github repository.

Redis data persistency on Uberspace

July 31, 2015 | 0 Comments | Tags: technology, berichtsheft, configuration, database, tutorial, Redis, Uberspace

Uberspace provides a nice german introduction on how to use Redis on their servers and also kindly points out that Redis resides in memory and will (when used with default settings) loose its state (speak: all your data **yikes**) when there is a crash or restart. So let us fix that:

Setting up Redis

Just a quick reharsal of their tutorial. Redis is beeing set up in two steps basically:

```
test -d ~/service || uberspace-setup-svscan
uberspace-setup-redis
```

Enabling persistency

There are several persistency models you can read up on here:. But I would recommend sticking with the most common way shown here, which strikes a good balane between performance and data safety.[1]

```
nano ~/.redis/conf
```

And then add the following lines:

```
dir /home/YOUR_UBERSPACE_USER_NAME/.redis/
appendonly yes
appendfsync everysec
```

```
# Restart redis
svc -du ~/service/redis
```

Verifying persistency

Now run the tool that will put data into Redis and make it do so. You can then check ① both that the persistency file grew to a size greater than `0` and ② also restart Redis and afterwards check whether your tool still sees the previously entered data.

```
# ① Check that the persistency file holds content
du -sch ~/.redis/appendonly.aof
```

```
# ② Restart Redis. This would kill all the contained data
# if the persistency setting would not work. Then check if
# the data is still available in your Redis using tool
svc -du ~/service/redis
```

Sources

1. Redis persistence demystified (oldblog.antirez.com)

Planned obsolescence through random misbehaviour

August 5, 2015 | 0 Comments | Tags: technology, berichtsheft, planned obsolescence, product design, closed source, proprietary software, closed source software, greed, abuse

Random misbehaviour could be a viable way to expire a closed source/proprietary software/hardware product without inflicting too much damage to the own brands name.

If products expire in an abrupt way it is obvious that negative resentments will rapidly accumulate in product forums, comment sections and by word of mouth. But what if the expiration happens in a gradual way? And what if the onset and the severity of the presented misbehaviour are also determined by a function of increasing probabilty?

I would argue that the consumer, because he can not pinpoint the cause of the misbehaviour due to the randomness, will first come up with theories in which he views himself as beeing able to temprarily fix or bypass the malfunctions. The genesis of such beneficial theories could also be nurtured by intelligent algorithms. Because the consumer finds himself in a situation which is, compared that of sudden expiration, harder to describe, he will, I hold, not so readily spread information that would damage the brand. Done right the consumer could ideally be given the impression that the malfunctions stem from ***other***, somehow connected/related devices. I would further argue that the consumers willingness to talk about the malfunction of a certain product decreases over time. This is why it should be beneficial to nurture the genesis of theories in which the ***other*** devices are the cause of the malfunction. That way the initial frustration and most of the negative communications during the "hot phase" of bad reputation generation are focused on multiple brands and/or technologies. When the frustration regarding the own brands device eventually reaches the replacement threshold, the willingness to further generate negative communications or dicuss malfunctions has already decreased and the own brand takes lesser damage.

Also it is valuable to note that the tech press would would find themselves in the same situation. Depending on the legal framework of the respective

country they would face the additional obstacle of possible lawsuits when publishing articles based on guesswork.

To give a concrete example of this concept:

Let's say a company produces a bluetooth headset. The bluetooth headset needs to be initially "paired" with other devices before it can relay audio for them. When switching a device on and off, the headset needs connect anew but must not be paired again. The proposed planned obsolescence would work in the following way:
The first day the device gets paired with another device an interal counter starts. The counter will, after a one or two year period has passed (depending on the guaranteed warranty) activate the random obsolescence function (ROF). Good malfunctions provided by the ROF could be to require the consumer to start the connection procedure multiple times when switching on the device or even to require a new pairing procedure to make it work again. Malfunctions like disconnects could be introduced when several bluetooth devices are connected at the same time. Unexpected behaviour like connecting successfully but still not relaying audio could also be introduced. The ROF could then, from week to week, increase the probability of such malfunctions in order to continually frustrate the consumer and to incentivice the product replacement process. Since the ROF acts only on a few connected devices the consumer will suspect other devices as possible sources of the malfunctioning and thus will also talk about them when discussing possibles solutions or giving product recommendations. When the consumer is eventually frustrated enough and replaces the headset he will no longer so willingly talk and theorize about the cause of the missbehaviours. The consumer would ideally be left with the impression that the malfuntions did stem from the technology standards used (eg. "Bluetooth *3.0*") or that they had to do with the combination of multiple brands.

Send files directly p2p and e2e encrypted

October 15, 2015

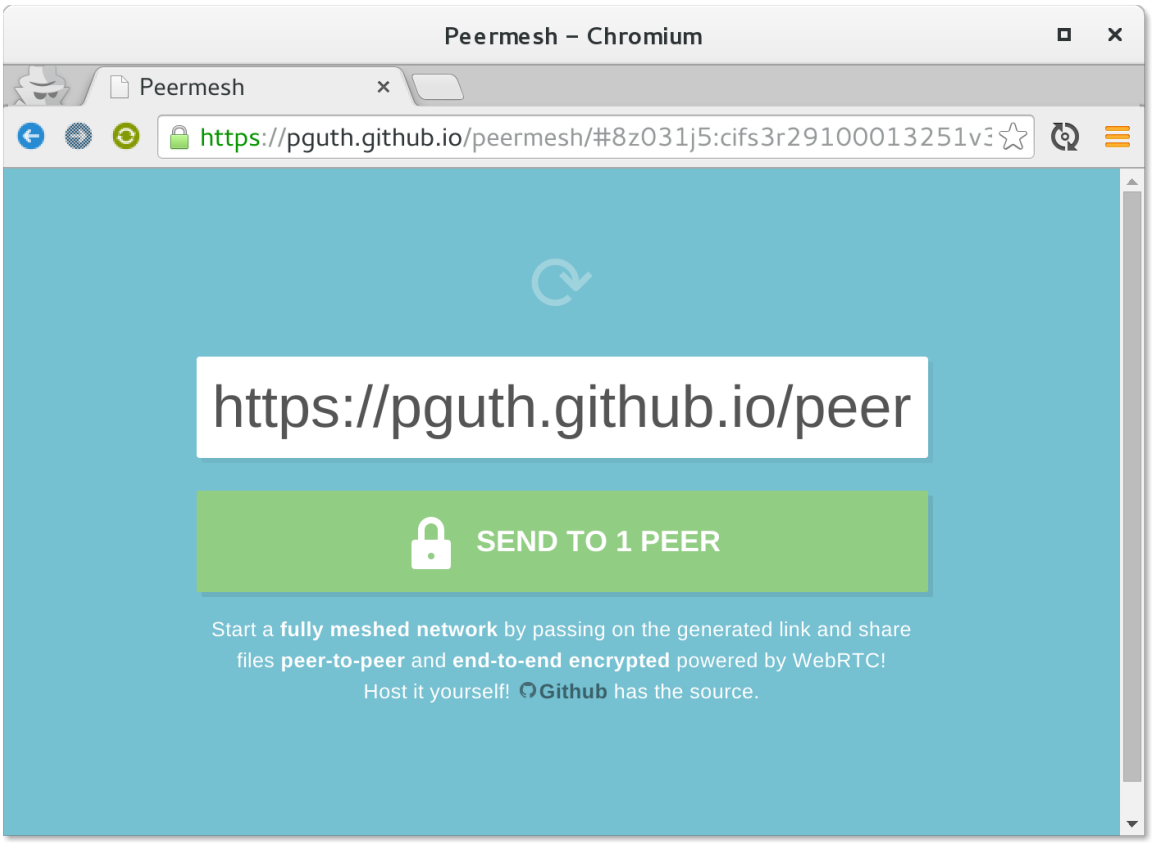
> *Start a **fully meshed network** by passing on the generated link and share files **peer-to-peer** and **end-to-end encrypted** powered by WebRTC!*

- Works fully in the browser using `WebRTC`.
- Mesh swarms can be started by opening the site. A "mesh URL" is generated to be passed around.
- The mesh URL contains a password. All files mesh will be sent `end-to-end encrypted`.
- Swarms can be joined by opening the mesh URL.
- Swarm form fully meshed networks (n:n) using `webrtc-swarm`.
- WebRTC signaling data is exchanged via `signalhub`.

Files will not be propagated among peers. The peers that initates a transfer will send the file to every connected peer individually.

The sourcecode is available at the Github repository.

Click the image to open the demo:



Related

- `peertransfer` Peertransfer is a (1:n) WebRTC based file transfer tool. Compared to `peermesh` it encodes a authentication code into the "sharing URL" that is passed around and will not initiate WebRTC signaling if the code is missing or wrong.

Credits

- Encrypt and decrypt content with Nodejs