Netz39 Blog

An dieser Stelle verkünden wir aktuelle Neuigkeiten zum Stand des Projekts. Schaut ab und zu rein, oder abonniert den RSS Feed.

Logoideen anyone?

Mit wachsender Bekanntheit in der Stadt und darüber hinaus wächst der Bedarf nach einem Signet zur einfachen visuellen Wiedererkennung unseres Vereins aka einem Logo. Wir haben seit langem eine Seite dazu hier im Wiki und es gab online wie offline schon hitzige Diskussionen über das wer, wann, wie, warum und wieviele. Fakt ist: wir wollen eins und wir wollen es bald und so gab es auf dem letzten Stammtisch den Entschluss, noch bis zum Ende dieses Monats Ideen zu sammeln und dann eine Abstimmung zu machen, welche Idee dann zu unserem offiziellen Logo werden wird. Wer sich inspiriert und befähigt fühlt (vulgo: alle dürfen mitmachen, egal ob Mitglied oder nicht), möge also bis 30.04.2013 Vorschläge einreichen. Das muss nicht fertig perfekt ausgearbeitet sein, aber die Idee soll klar erkennbar sein. Einreichungen bitte im Wiki oder bei GitHub. Bei Fragen stehen wir gern zur Verfügung.

~~LINKBACK~~

2013-04-16 17:11 · alex · 0 Kommentare
logo

phpmyadmin mit nginx unter Debian Wheezy

Eigentlich wollte ich ja heute einen status.net Server aufsetzen, aber da der gerne MySQL und auch noch gern eine eigene Datenbank hätte und das auf dem designierten Server noch nicht installiert war, schob ich erstmal die Installation von MySQL und phpMyAdmin ein. Das Debian-Paket *phpmyadmin* bringt Beispielkonfigurationsdateien für Apache2 und lighttpd mit, für den hier eingesetzten *nginx* leider nicht. Das Web ist voll von HowTos zu dem Thema, aber keins passt so richtig (beispielsweise weil mir das reicht phpmyadmin in 'nem Unterordner zu haben) und deswegen gibt's jetzt noch HowTo, dieses hier. Voraussetzung ist ein bereits fertig eingerichteter nginx mit *php5-fpm* und ein MySQL-Server. In der entsprechenden Config des nginx steht irgendwo etwa folgendes für den php5-fpm:

```
upstream php {
    server unix:/var/run/php5-fpm.sock;
}
```

Dann gibt es sicher noch einen Abschnitt für den HTTPS-Server und da fügt man dann folgendes ein:

```
# phpmyadmin
location /phpmyadmin {
    alias   /usr/share/phpmyadmin;
    index   index.php;
```

```
}

location ~ ^/phpmyadmin/libraries {
    deny all;
}

location ~ ^/phpmyadmin/setup/lib {
    deny all;
}

location ~ ^/phpmyadmin/setup/(.+\.php)$ {
    auth_basic              "phpMyAdmin Setup";
    auth_basic_user_file    "/etc/phpmyadmin/htpasswd.setup";
    alias                   /usr/share/phpmyadmin/setup/$1;
    fastcgi_split_path_info ^(.+\.php)(/.+)$;
    fastcgi_pass            php;
    fastcgi_index           index.php;
    include                 fastcgi_params;
}

location ~ ^/phpmyadmin/(.+\.php)$ {
    alias                   /usr/share/phpmyadmin/$1;
    fastcgi_split_path_info ^(.+\.php)(/.+)$;
    fastcgi_pass            php;
    fastcgi_index           index.php;
    include                 fastcgi_params;
}
```

Damit entspricht das recht genau den von Debian mitgelieferten Configs für die anderen Webserver. Der vorletzte Abschnitt schützt das Setup von phpmyadmin. Ein Passwort würde man mit dem Tool `htpasswd` setzen können, das ist bekanntermaßen im Paket *apache2-utils* enthalten. Aber: wenn man phpmyadmin über den Debian-Paketmanager installiert hat, kann man sich das sparen, wenn die Konfiguration gleich bei der Installation mit Hilfe von *dbconfig-common* gemacht wurde. Die Doku in `/usr/share/doc/phpmyadmin/README.Debian.gz` sagt dazu:

> Since 3.0.0, phpMyAdmin can be configured using dbconfig-common. It creates a phpmyadmin database and control user on the chosen server and configures phpMyAdmin to use cookie authentication on this server. The database autoconfiguration might fail if you do not have local MySQL server installed or you have configured too high priority of which questions should debconf ask. To rerun the configuration just invoke:
>
> ```
> dpkg-reconfigure -plow phpmyadmin
> ```
>
> phpMyAdmin also provides a web-based setup script available at http://localhost/phpmyadmin/setup/index.php

Betonung auf »also«, d.h. wenn man dbconfig-common benutzt hat, ist man bereits fertig.

~~LINKBACK~~

2013-03-25 18:02 · alex · 1 Kommentar
debian, nginx, phpmyadmin, howto

Network UPS Tools (NUT) unter Debian Wheezy mit einer MGE Ellipse 600

Uninterrupted Power Supply (UPS) oder zu deutsch Unterbrechungsfreie Stromversorgung (USV) ist das, was man am Server haben will um sich gegen kurze oder längere Stromausfälle zu schützen, genauer um seine Hardware vor den Auswirkungen derselben zu schützen. Dicker Akku, bisschen Elektronik und schon läuft der Server weiter, wenn mal kurz der Strom weg ist. Zutaten für das Rezept heute: Dell PowerEdge 1750 mit installiertem Debian 7.0 aka Wheezy und eine gespendete MGE Ellipse 600. Als Software werden wir NUT installieren, das ist laut Internet wohl Weapon of Choice.

Warum noch ein HowTo: nun ja, neue Version des Betriebssystems, Doku passt nicht, bisschen Rumbasteln nötig, Ihr kennt das.

Hardware anschließen

Wie man die Kabel da zusammensteckt, muss ich hier nicht erklären, jedenfalls lässt sich das Gerät per USB anschließen und meldet sich dann auf `lsusb` wie folgt:

```
Bus 003 Device 004: ID 0463:ffff MGE UPS Systems UPS
```

Software installieren

Okay, Gerät bekannt, prima soweit, dann erstmal NUT installieren. Entweder das metapackage `nut` oder `nut-client` und `nut-server` einzeln, ich hab auch noch die Dokumentation installiert:

```
aptitude install nut nut-doc
```

udev einrichten

Zu allererst geh ich mal in die Doku, will sagen in `/usr/share/doc/nut/README.Debian.gz` geschaut und siehe:

```
For USB devices, permissions are automatically set by the
/lib/udev/rules.d/52-nut-usbups.rules udev rules file.
```

Aber die Datei gibt's gar nicht, dafür gibt es `/lib/udev/rules.d/52-nut-usbups.rules` und die enthält:

```
# This file is generated and installed by the Network UPS Tools package.
ACTION!="add|change", GOTO="nut-usbups_rules_end"
SUBSYSTEM=="usb_device", GOTO="nut-usbups_rules_real"
SUBSYSTEM=="usb", GOTO="nut-usbups_rules_real"
```

```
SUBSYSTEM!="usb", GOTO="nut-usbups_rules_end"
LABEL="nut-usbups_rules_real"
#  various models  - usbhid-ups
ATTR{idVendor}=="0463", ATTR{idProduct}=="ffff", MODE="664", GROUP="nut"
LABEL="nut-usbups_rules_end"
```

Ah da haben wir ja Vendor- und Produkt-ID und das stimmt auch mit der Ausgabe von `lsusb` überein, wird nur irgendwie von udev noch nicht automatisch geladen. 😕 Es gibt dazu diverse Bugreports[1)] in Debian, alle geschlossen, obwohl es nicht auf Anhieb tut – naja sagen wir, obwohl es falsch oder unverständlich beschrieben ist. Jedenfalls dann nach `/etc/udev/rules.d` gehen und

```
ln -s /lib/udev/rules.d/52-nut-usbups.rules
```

ausführen, udev neu starten, neu booten oder USB-Gerät ab- und wieder anstecken oder beides, was da jetzt genau geholfen hatte, weiß ich grad nicht mehr. Das war der unkomplizierte Teil. Die Konfiguration von NUT verteilt sich dann nämlich auf mehrere Dateien, weil man mit der Software auch abgefahrene Konfigurationen mit mehreren USV und Rechnern aufbauen kann, bunte Bildchen hier.

Treiber konfigurieren

Bevor wir irgendwas anderes machen, einmal in die Datei `/etc/nut/nut.conf` gehen und die Zeile mit `MODE` suchen und anpassen, sonst startet der `upsd` später erst gar nicht. Hier sieht das so aus:

```
MODE=standalone
```

In `/etc/nut/ups.conf` konfigurieren wir dann den Treiber bzw. machen der Software bekannt, was wir für eine USV angeschlossen haben, indem wir einen Abschnitt hinzufügen:

```
[MGE_something]
        driver = usbhid-ups
        port = auto
```

Ganz einfach, wenn man den USB-Teil mit udev vorher klar hat. Theoretisch kann man da auch noch mehr Optionen angeben, wenn man nicht mehrere USV am Server hat, kann man sich das aber klemmen.

Kurztest

Zeit unser Gerät zu testen und zwar mit

```
upsc MGE_something@localhost | less
```

was dann ungefähr folgendes ausgibt:

```
battery.charge: 92
battery.charge.low: 30
battery.runtime: 1408
```

```
battery.type: PbAc
device.mfr: MGE OPS SYSTEMS
device.model: Ellipse 600
device.serial: BDBJ32016
device.type: ups
driver.name: usbhid-ups
driver.parameter.pollfreq: 30
driver.parameter.pollinterval: 2
driver.parameter.port: auto
driver.version: 2.6.4
driver.version.data: MGE HID 1.31
driver.version.internal: 0.37
input.transfer.high: 264
input.transfer.low: 184
outlet.1.desc: PowerShare Outlet 1
outlet.1.id: 2
outlet.1.status: on
outlet.1.switchable: no
outlet.desc: Main Outlet
outlet.id: 1
outlet.switchable: no
output.frequency.nominal: 50
output.voltage: 230.0
output.voltage.nominal: 230
ups.beeper.status: enabled
ups.delay.shutdown: 20
ups.delay.start: 30
ups.load: 20
ups.mfr: MGE OPS SYSTEMS
ups.model: Ellipse 600
ups.powups.productid: ffff
ups.serial: BDBJ32016
ups.status: OL CHRG
ups.timer.shutdown: -1
ups.timer.start: -10
ups.vendorid: 0463
```

Weitere Einrichtung

Gleich geschafft, nur noch zwei Dateien editieren. Erstmal `/etc/nut/upsd.users`

```
[admin]
        password = 12345
        actions = set
        actions = fsd
        instcmds = all

[monmaster]
        password = 12345
        upsmon master
```

```
[monslave]
        password = 12345
        upsmon slave
```

Die Erklärungen dafür kann man ausführlich in der Doku nachlesen. In kurz: wir haben verschiedene Nutzer, die verschiedene Dinge dürfen mit der USV und setzen da Berechtigungen und Passwörter[2)] und daher. Ja und zu guter letzt, warum wir den ganzen Kram hier eigentlich machen, wird dann in `/etc/nut/upsmon.conf` aktiviert, nämlich dass der Server auch sauber runterfährt, wenn der Strom zu lang weg bleibt. Da gibt's viele Erklärungen in der Datei, die defaults erschienen mir sinnvoll, so dass ich an passender Stelle nur eine Zeile hinzugefügt habe:

```
MONITOR MGE_something@localhost 1 monmaster 12345 master
```

Damit läuft das erstmal so, dass der Rechner automatisch runterfährt nach ungefähr 10 Minuten. Weitere Überwachung mit munin oder nagios oder oder oder ist dann advanced topic. Für einfach nur mal eben schnell USV zum Laufen kriegen ist das HowTo hier ja auch lang genug. 😉

~~LINKBACK~~

2013-02-27 09:54 · alex · 0 Kommentare
howto, usv

29C3 - Congress Everywhere im Netz39

Ab dem 27. Dezember geht es wieder los, der Netz39 wird von Donnerstag bis Sonntag den 29C3 in unserem Hackerspace übertragen.

Aber das ist noch lange nicht alles, wir werden wieder Hardware basteln, Software schreiben, zusammen kochen und viel Club-Mate trinken.

Du bist eingeladen dir mit uns den Congress anzusehen und unseren Space kennenzulernen.

4 Tage lang Netz39, wir würden uns freuen wenn du auch kommst.

https://events.ccc.de/congress/2012/wiki/Congress_everywhere#Magdeburg.2C_Netz39

~~LINKBACK~~

2012-12-26 17:29 · andreas · 0 Kommentare

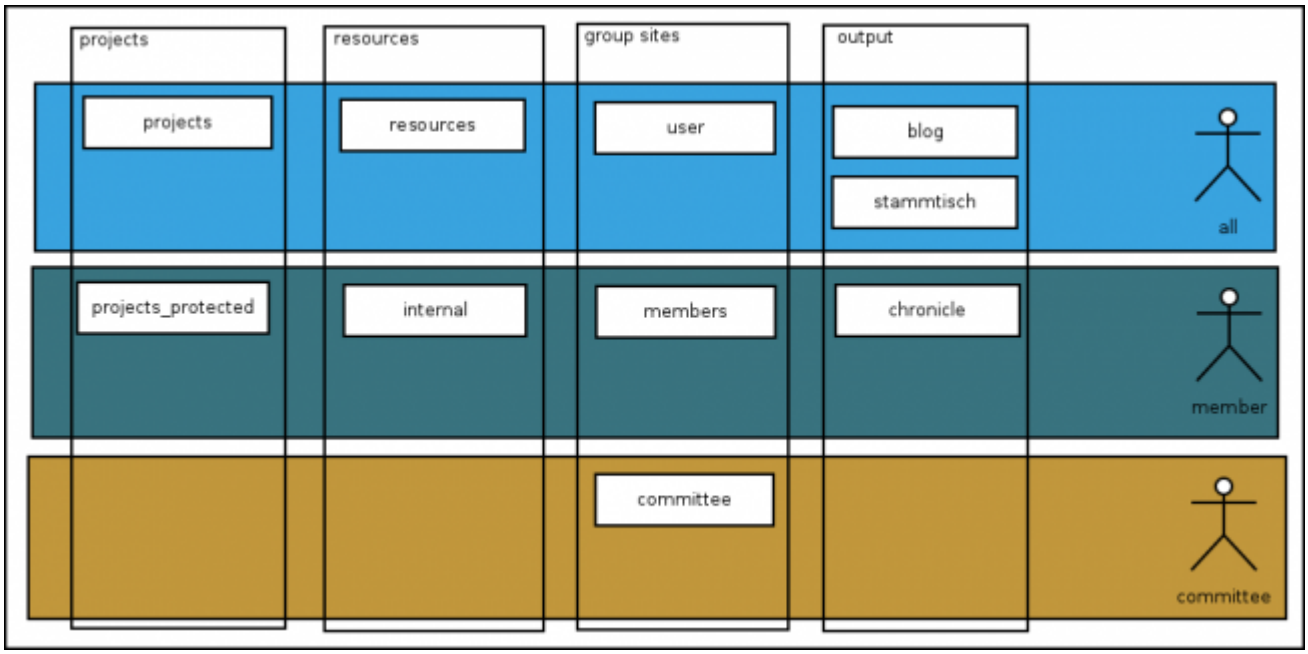
Neue Wiki-Organisation

Das Wiki wurde umgestaltet. Insgesamt ist jetzt alles flacher Organisiert. Das `archive` wurde eingestampft. Einige Schätze daraus wurden nach `projects` und `stammtisch` verschoben. Die `planning` wurde in `projects` integriert. `user`-Bereich wurde wieder der Öffentlichkeit gegeben. Wer dennoch das Bedürfnis hat seine kleine Welt zu pflegen, kann sich im `members`-Bereich

austoben.

Ziel dieser Umgestalltung ist, dass man Seiten schneller findet und neue Seiten leicht der richtigen Kategorie zuordnen kann. Alle Seiten sollen Permalinks sein und somit auch in Jahren noch gelten.

Sichtbarkeiten finden sich im Organigramm:



~~LINKBACK~~

2012-11-29 01:44 · frank · 1 Kommentar
wiki, orga

Ältere Einträge >>

1)
bspw. 660072
2)
Sichere Passwörter siehe https://www.youtube.com/watch?v=_JNGI1dI-e8