UpTimes

Mitgliederzeitschrift der
German Unix User Group

Interview zum Vorstandswechsel

SoX - Mischpult per CLI

Limoncellis 'Operations Report Card'

2014-2

Inhaltsverzeichnis

Grußwort vom Vorstand: Moin liebe Mitglieder!
von Dirk Wetter, Vorstandsvorsitzender **3**

Gipfelruhe: Doppel-Interview zum Vorstandswechel
Interview: Anika Kehrer **5**

Aufruf: GUUG-Vorstandswahl 2015
von Christian Lademann, Wahlleiter GUUG e.V. **9**

Eines Tages im Linuxhotel: Erstes Redaktionstreffen der UpTimes
von Anika Kehrer **10**

Das Auge denkt mit: Paketfilterregeln von *iptables* visualisieren
von Mathias Weidner **15**

Wer nicht fragt bleibt dumm: *OpsReportCard* – Fragen für Sysadmins
Übersetzung: Mathias Weidner **18**

Mixpult per Tastatur: Der CLI-Sound-Exchanger SoX
von Jürgen Plate **31**

Shellskripte mit Aha-Effekt IV: Ein Interface für das Shell-Shuffle
von Jürgen Plate **37**

Loser, Learner und O'lala: So war die LinuxCon Europe
von Nils Magnus **41**

Hilfreiches für alle Beteiligten: Autorenrichtlinien **44**

Über die GUUG: German Unix User Group e.V. **47**

Impressum **48**

Grußwort vom Vorstand
Moin liebe Mitglieder!

And now for something completely different. . . Nein, jetzt kommt kein Monty-Python-Sketch. Hier schreibt heute nur jemand anderes das Vorwort, als sonst.

von **Dirk Wetter, Vorstandsvorsitzender**

Wir sind Verein.

Dirk Wetter

Ihr habt es vielleicht mitbekommen: Auf der letzten Mitgliederversammlung haben wir Reise nach Jerusalem gespielt. Zum Sitzen kam ein komplett neues Team. Deren Gesichter findet Ihr auf der Webseite [1], ein paar eher oberflächliche Infos zu den Personen im Protokoll der Mitgliederversammlung. Anika und die restliche UpTimes-Meute werden wohl dem einen oder anderen gegenwärtigen und vielleicht zukünftigen Mitglied des Vorstands in den nächsten Ausgaben weitere Details zu den Personen abnötigen.

Für mich und die anderen, die sich dankenswerterweise innerhalb weniger Tage bereit erklärt haben, der GUUG frischen Wind einzuhauchen, tat eine Ablösung Not (siehe Interview weiter hinten im Heft). Ich möchte nicht nach hinten blicken, sondern nach vorn. Dafür engagiere ich mich.

Wir haben ziemlich viele Aufgaben zunächst einmal betrieblicher Natur auf dem Zettel, und die Zeit bis zu den nächsten Wahlen ist – gemessen daran – kurz. Immerhin hatten wir rund fünf Wochen nach unserer Wahl einen ersten Erfolg vorzuweisen: Termin, Ort und einige Sprecher für unser Frühjahrsfachgespräch (FFG) 2015 stehen fest. Wir sind vom 24. bis zum 27. März 2015 an der Uni Stuttgart zu Gast. Der nächste drängende Punkt: Wir arbeiten an der Einziehung und Rechnungsstellung der Mitgliederbeiträge 2014. Das klingt nun einfach. Leider steckt der Teufel im Detail. Der trägt den Namen IBAN und hat sich geforkt, sodass er in mehreren Inkarnationen auftritt. Immerhin sind wir erheblich weiter als bei Amtsantritt, und auch an den Rest kommt dieser Tage ein Haken. Dann werden wir rückständige Beiträge von 2013 und 2012 anmahnen.

So viel zu den drängenden Alltagsdingen – die Pflicht, die es unter anderem zu erledigen gilt. Spannender im Sinne des Vereins und als Community sind die gestalterischen Punkte.

Keiner ist wie wir

Ich finde, kein Verein in Deutschland vertritt so gut wie die GUUG das Thema Open Source und Unix in technischer Hinsicht. Das und unsere Konferenz – gegenwärtig eine, nämlich das FFG – können sich blicken lassen! Soweit ich die Nachbarländer aus meiner Zeit als Linux-Kongress-Strippenzieher kenne: Wir stehen echt gut da, was unsere Themen und auch unsere Verbreitung angeht. Allerdings ist das kein Anlass sich entspannt zurückzulehnen. Die Erde dreht sich weiter, und es gibt leider viele, die die GUUG gar nicht kennen, sie aber aufgrund ihres fachlichen Hintegrunds eigentlich kennen sollen.

Kurzum: Was wir haben, müssen wir fortentwickeln und das, von dem wir denken, dass es unseren Verein ausmacht, mehr herausdestillieren. Wir müssen es auf die Straße bringen. Dazu nenne ich vier Ideen – die der Vorstand in der verbleibenden Zeit allerdings maximal anschubsen kann:

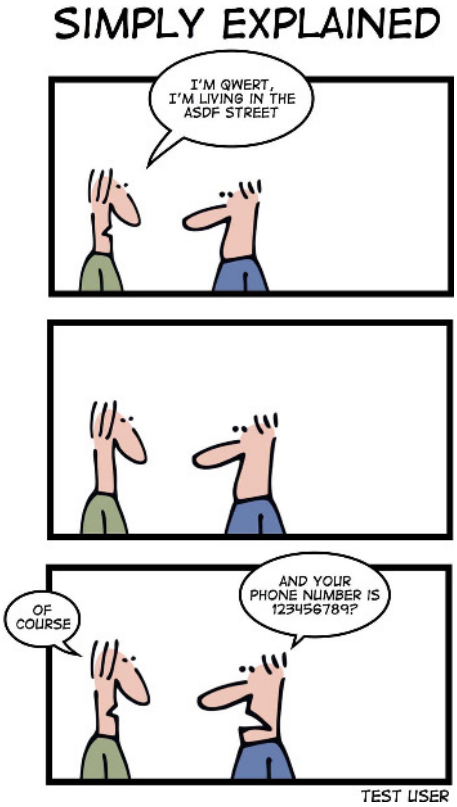
- **Lokale Gruppen lebendiger machen.** Für viele – das weiß ich von meinem Stammtisch in Hamburg – sind lokale Vorträge eine im Wortsinn gern gesehene Sache, und für unseren Verein sind sie ein offener Außenposten. Es macht schlicht Spaß, bei einem Vortrag dazuzulernen, ähnlich Gesinnte kennenzulernen und Technik-Schnack beim Bier danach zu betreiben. Wir haben in Deutschland viele tolle Sprecher, zum Beispiel bei unseren FFGs, die auch bereit wären, gegen Unkostenerstattung aus dem Vereinsbudget einen Vortrag an beliebigen Orten zu halten. Davon wird noch zu wenig Gebrauch gemacht. Dies muss übergeordnet in die Hand genommen werden. Auch "Wandertutorien"hatten wir mal diskutiert.
- **Die *Zukunftswerkstatt GUUG* ausrollen.** Die Idee enstand 2013 auf dem FFG in Frankfurt.

Hier muss nach Vorbereitung ein moderiertes Treffen her. Danach müssen die Ergebnisse mitgenommen, weiterverfolgt und die resultierenden Aktionen herausarbeitet werden.

- **Geld ausgeben**. Wie ihr den Kassenberichten entnehmen könnt, haben wir gegenwärtig ein Geldpolster. Wir müssen uns Gedanken machen, wie wir unser Geld sinnvoll einsetzen, das heißt zweckgebunden. Da gibt es erste Ideen. Ein Beispiel wäre wie in alten Zeiten, FOSS-Projekte gezielt zu sponsoren. Wenn wir dies wollen, brauchen wir Antworten auf die Frage: Was sind die Kriterien?

- **Transparenz leben.** *we're open* ist ein knackiger Spruch für die GUUG, der es gut trifft. Für die Governance ist dies noch ausbaufähig und muss sich besser finden. Ein Hinweis dazu: Alles, was ihr auf dem Herzen habt oder diskutieren wollt zum Thema Verein, das äußert Ihr auf *guug-member at guug dot de* – das ist die richtige Anlaufstelle dafür. Und wenn ihr tiefere Einblicke wünscht, was Eure Mitstreiter vom Vorstand so treiben: Dafür ist nur eine Mail an *vorstand at guug dot de* erforderlich. Über Mailingliste, Wiki und Vorstandsprotokolle könnt Ihr Euch über das Vorstandsgeschehen informieren. Dies empfiehlt sich besonders, wenn Ihr euch für einen Vorstandsposten interessiert. Denn so bekommt ihr einen Eindruck, was Euch erwartet.

A propos Vorstandsposten: Die nächste zweijährige Amtsperiode startet Ende März 2015. Auch bei den Wahlen wünschen wir uns mehr Transparenz. Es soll Interviews geben, die den Wählern – Euch allen – ein besseres Urteil erlauben. Und by the way, es wäre schön, wenn der Vorstand ab dem nächsten Jahr personell besser besetzt wäre. Drei fehlende Schaffende machen sich durchaus bemerkbar, nicht nur aufgrund des Nachholbedarfs an Aufgaben.



Wer ist 'wir'?

Ich habe nicht selten das Wort 'wir' verwendet, oder mich mit Passiva um Personalpronomen herumgemogelt, was Anika wahrscheinlich nur mit Schmerzen erträgt *(Anm. d. Red.: Anika fand sich durchschaut)*. Damit möchte ich aber etwas Bestimmtes ausdrücken. Nämlich, dass wir alle die Zukunft der GUUG bestimmen. *(Anm. d. Red.: Ach so. Na gut.)*

'Wir' sind **wir alle**. Ihr seid herzlich eingeladen, Euch an unserem Verein zu beteiligen! Äußert Eure Ideen, diskutiert sie auf *guug-members*, und macht mit bei der Umsetzung. Wenn ihr Zeit und Engagement mitbringt, vielleicht interessiert Euch, ein Projekt unter die eigenen Fittiche zu nehmen. Es ist unser aller Verein, er ist es wert – also lasst uns was daraus machen!

Ran ans Werk. Und viele weitere Wünsche mögen Euch so in dreieinhalb Wochen in Erfüllung gehen.

[1] GUUG-Vorstand: http://www.guug.de/verein/vorstand/index.html

Wer Dirk ist . . .

. . . erfahrt Ihr im Doppelinterview zum Vorstandswechsel. So viel Zeit muss sein. :-)

Gipfelruhe
Doppel-Interview zum Vorstandswechel

Auf der Mitgliederversammlung der GUUG am 24. September 2014 führte der Rücktritt des gesamten bisherigen Vorstands zu einem neuen Interim-Vorstand bis zur regulären Wahl 2015. Der alte und der neue Vorsitzende stellten sich zu diesem Geschehen den Fragen der UpTimes.

Interview: **Anika Kehrer**

Wir ist mehr
als der Plural von Ich.
Wir ist das Präsens von Futur.

Karl-Heinz Karius

Wolfgang, wie kam es zu dem Rücktritt?

Wolfgang: Das Frühjahrsfachgespräch haben wir erst in den Herbst verschoben, dann sehr kurzfristig abgesagt. Daraufhin kamen Rufe aus der Mitgliederschaft – einige zahm, andere heftiger – dass das so nicht gehe. Natürlich haben wir das auch intern diskutiert. Der Vorstand war in seiner damaligen Zusammensetzung schon relativ lange aktiv, zumindest der harte Kern aus vier, fünf Leuten. Auch war der Vorstand bereits dezimiert, da zwei Vorstandsmitglieder ausgetreten waren. Die übrigen wollten bei der nächsten Wahl nicht mehr antreten. Also sagte ich mir, wir machen Platz für andere. Ich wusste zu dem Zeitpunkt nicht, ob es überhaupt andere gibt, die bereit dazu wären. Den nächsten, die da vielleicht kommen, wollten wir auf jeden Fall keine Steine in den Weg legen.

Dirk, wie hast Du das erlebt?

Dirk: Ich war einer der Schuldigen. Ich habe einige Sachen einfach nicht gut gefunden und mich darüber beklagt. Ich finde, dem alten Vorstand gebührt Respekt, dass er diesen Schritt getan hat, und dass der Wechsel so reibungslos geklappt hat – insbesondere Wolfgang.

Wolfgang: Danke!

„Dem alten Vorstand gebührt Respekt“

Wieviele Stimmen forderten denn den Rücktritt, Wolfgang?

Wolfgang: Drei oder vier. Es waren die üblichen Unkenrufe. Es war auch nicht der erste Ruf. Für mich war dann das Maß voll.

Dirk, Du warst einer derer, die den Vorstand kritisierten?

Dirk: Ja, öfter mal. Wolfgang weiß das auch. Und ich finde, dass ein Wechsel jetzt gut tut. Nachdem klar war, dass der Rücktritt angeboten wird, versuchte ich ein Team zu finden, das bereit war im Vorstand mitzuarbeiten. Was sich natürlich in der kurzen Zeit nicht einfach gestaltete. Um nicht zu sagen, schwer. Es gab da schon Tage, an denen ich nicht viel arbeitete, sondern mailte und telefonierte.

Wolfgang: Ich wäre froh gewesen, ein paar der Leute gehabt zu haben, die sich jetzt fanden. Wir haben auch im alten Vorstand versucht, neue Leute zu rekrutieren. Wir haben zum Beispiel die Vorstandskommunikation öffentlicher gemacht, und ein paar Leute haben sich auch in die Mailingliste eingetragen. Sie haben dann aber nur ganz selten Kommentare abgegeben. Und oft waren es kritische Kommentare. Ein paar Tage vor der Mitgliederversammlung schickten wir dann schließlich die Mail an den ganzen Mitgliederverteiler, dass der Vorstand zurücktritt und die Mitglieder jetzt aufgefordert sind, einen neuen zu wählen.

Dirk: Ja, diese Mail war auch der Auslöser für mich, mir über einen neuen Vorstand Gedanken zu machen.

Welche Resonanz bekamst Du auf die Rücktrittsankündigung, Wolfgang?

Wolfgang: Zwei Leute sagten, dass sie mitarbeiten würden. Den Rest hat Dirk dankenswerterweise zusammengetrommelt.

Dirk, kannst Du die Entscheidung des alten Vorstands nachvollziehen?

Dirk: Ja, ich begrüße sie auch sehr. Wie gesagt: Ich begrüße den Mut, das so zu machen. Es ist das einzig richtige, was sie tun konnten.

„Vereinsarbeit bedeutet, es nie allen recht machen zu können“

Wolfgang, Du sprichst von mehrfachen Rücktrittsforderungen in der Vergangenheit. Fandest Du die Kritik in der Sache berechtigt?

Wolfgang: Nein. Hätte ich das so gesehen, dann wäre ich die Punkte angegangen. Meiner Meinung nach waren es oft Kleinigkeiten. Vereinsarbeit bedeutet, dass man es nie allen recht machen kann.

Stand der Vorstand Eurer Meinung nach richtiggehend unter Beschuss?

Wolfgang: Nein. Also, zuerst schon. Zumal ich im alten Vorstand die Funktion des Sprachrohrs hatte. Denn manche konnten sich nicht so gut ausdrücken, andere wollten es einfach nicht. Ich war der, der reagiert, wenn niemand anders reagiert. So entstand der Eindruck, ich sei das Sprachrohr. Viele der Vorwürfe hatten leider einen persönlichen Unterton. Ich habe immer versucht, es auf eine sachliche Ebene zu heben. Und wenn ich dann mit ein paar Tagen Abstand darauf sah, konnte ich besser sehen, dass es nicht um mich persönlich geht, sondern um Kritik an der Vorstandsarbeit. Die wichtigste Lehre, die ich aus meiner Vorstandsarbeit ziehe, ist, dass man Kritik nicht persönlich nehmen darf.

Dirk: Es wurde immer wieder Kritik geübt. Aber unter Beschuss stehen, das würde ich nicht sagen.

Wie gehts Dir denn jetzt als Vorsitzender, Dirk? Denkst Du: "Yeah, lasst uns koslegen"?

Dirk: Die erste Zeit war es so. Ich kann nicht sagen, dass Ernüchterung eingetreten ist. Aber ich merke, dass es ein ganzes Stück Arbeit ist. Ich merke auch, dass es als Vorstandsvorsitzender nicht einfach ist.

Woran machst Du das fest?

Dirk: Wir haben ja vieles in Aussicht gestellt, und stellt sich als wirklich ganz schön viel Arbeit heraus. Da fühle ich mich in der Pflicht, das auch zu liefern. Ich habe bei solchen Sachen auch relativ viel Ehrgeiz. Bei meinen Kundenprojekten habe ich sehr viel selbst in der Hand. Aber sobald es um Leute geht, die ehrenamtlich mitarbeiten, ist das Ganze schon ein bisschen schwieriger. Man hat eben nicht acht Stunden am Tag, in denen man sie ansprechen kann und in denen sie etwas erledigen können.

Wolfgang, was ist aus Deiner Sicht schwierig für den Vorstandsvorsitzenden?

Wolfgang: Einerseits ist es schwer, Leute im Ehrenamt zu motivieren. Andererseits gibt es aber auch das Problem, dass einige Leute etwas machen wollen, bei dem man schon vermutet, dass sie sich damit übernehmen oder dass sie nicht denselben Qualitätsanspruch haben wie man selbst. So jemandem möchte man seine Aufgabe nicht unbedingt wegnehmen. Einerseits, weil demjenigen vielleicht sehr viel daran liegt, auch wenn es nicht besonders gut läuft. Andererseits möchte man auch niemanden aus dem Vorstand vertreiben. Man ist froh um jeden, der mitarbeitet. Wenn alles auf Ehrenamt basiert, hast du wenig in der Hand, wo sich ein Hebel ansetzen ließe.

Dirk: Highfive, Wolfgang.

„Es kann nicht sein, dass alles beim Vorstand kleben bleibt“

Siehst du Deine Kritik von damals jetzt anders, Dirk?

Dirk: Ich wusste damals schon um die Implikationen, und ich wusste auch, dass man es sich als Außenstehender leicht machen kann. Ich hatte mir das Recht herausgenommen, Außenstehender zu sein. Aber wenn wir das in der Summe alle machen... Ich denke, wir müssen es als Verein schaffen, mehr Mitglieder einzubeziehen. Es kann nicht sein, dass alles beim Vorstand kleben bleibt. Es ist oft, als würden Leute für ein Fernsehprogramm bezahlen und dann hindurch-zappen wollen. Das ist einerseits gut, denn das Geld braucht der Verein, um etwas zu bewegen. Aber der Verein braucht auch Leute, die anpacken. Aber Leute, die sagen: „Ich mach das jetzt“, müssen am Ende auch liefern. Und zwar nicht nur der Vorstand, sondern auch Mitglieder, denn als Vorstand verlässt du dich ja auch darauf. Es ist einfach Arbeit

für den Verein – wer genau es am Ende macht, ist egal.

Wie geht es Dir jetzt mit dem Rücktritt, Wolfgang?

Wolfgang: Ich bin nicht traurig. Ich habe eine Menge gelernt, auch viele professionelle Unixer kennengelernt. Diese Verbindungen möchte ich auf jeden Fall weiter nutzen. Im Nachhinein denke ich, dass ich einiges hätte besser machen können.

Was gibst Du dem neuen Vorstand mit auf den Weg?

Wolfgang: Mehr Mitgliederbeteiligung. Es sind genügend Ideen da, was man aus dem Verein oder mit dem Verein machen kann. Dafür braucht es helfende Hände. Der alte Vorstand hat es nicht geschafft, die helfenden Hände zu motivieren. Vielleicht haben wir auch etwas falsch gemacht. Es gibt aber Dinge, die man auch als Mitglied besser machen kann.

Dirk: Wir möchten versuchen, dass es besser wird. Ob das binnen der verbleibenden vier Monaten klappt, ist natürlich unklar. Aber ein paar Sachen haben wir schon in Angriff genommen, etwa das FFG, die Mitgliederverwaltung und die IT. Da kam auch schon einiges an Feedback zurück. Das wird jetzt intern zur Diskussion gestellt. Es ist ein gutes Zeichen dafür, dass es da draußen Mitglieder gibt, die sich kümmern. Dass mehr sich berufen fühlen, ist wichtig für das Fortkommen des Vereins. Es ist egal, ob uns das noch gelingt, oder dem Vorstand ab 2015.

Dirk, würdest Du was anders machen als Wolfgang?

Dirk: Ich würde wohl mal auf den Tisch hauen. *(Lacht)*

Wolfgang: Das habe ich auch gemacht. Aber vielleicht zu spät. *(Lacht)*

Würdest Du Wolfgang gern mal was fragen, was sich Dir bisher nicht erschloss?

Dirk: Einiges ist hier schon zur Sprache gekommen. Es ist auch mir passiert, dass Vorstandsmitglieder nicht liefern, was sie versprochen haben. Manche mögen das nicht einsehen und finden mich ungeduldig. Aber ich denke, wenn man nicht an den richtigen Stellen ungeduldig ist, kann man Dinge nicht vorantreiben.

„Es tut gut, auch mal positiv angesprochen zu werden“

Habt Ihr Euch eigentlich mal generell über Vorstandsarbeit ausgetauscht – was man besser machen kann oder was man nicht machen sollte – oder habt Ihr das vor?

Dirk: So konkret haben wir das nicht gemacht. Wolfgang hat Übergangspunkte gesammelt. Das sind aber eher die wichtigen betrieblichen Sachen, die für den gegenwärtigen Vorstand wichtig sind.

Wolfgang: Ich denke, dass wir beide die Sachen nicht wesentlich anders machen würden. Dirk hat den Vorteil eines neuen, frischen Teams. Ich stehe für Fragen weiter auf dem Vorstandsverteiler.

Dirk: Ich denke, dass ich Wolfgangs Anspruch teile. Ich halte es außerdem für einen Vorteil, dass ich nicht von scratch anfange, sondern schon einmal vier Jahre im Vorstand war. Auch, als ich pausierte, las ich im Vorstandsverteiler mit.

Was ist jetzt Dein Hauptanliegen während Deiner Amtszeit, Dirk?

Dirk: Das Frühjahrsfachgespräch 2015. Da bin ich froh, dass wir schon den Termin eingetütet haben.

Gibt es etwas, das Ihr beide direkt den Mitgliedern sagen möchtet?

Wolfgang: Kriegt den Hintern hoch. *(Lacht)* Und es ist tatsächlich so, dass es gut tut, auch mal positiv angesprochen zu werden, wenn auf der Mitgliederliste wieder einmal vergleichsweise motzige Mails kamen. Ich wurde auch schon angesprochen, dass doch alles so weit ganz gut ist und dass es immerhin ehrenamtlich ist. Es tut gut, auch mal Dank zu bekommen.

Dirk: Der Punkt ist einfach: Es ist unser aller Verein. Nicht nur der des Vorstands. Also lasst uns was draus machen.

Über Dirk

Dirk Wetter ist seit 2003 selbständiger IT-Sicherheitsberater, eine kurze Angestelltenpause ausgenommen. Um 2006 herum ist er in die GUUG eingetreten und startete 2006 den Hamburger GUUG-Stammtisch. Von 2009 bis 2013 stand er erfolgreich erst als Beisitzer, dann als stellvertretender Vorsitzende zur Wahl. Dirk ist außerdem in der OWASP aktiv, was ihm in puncto Offenheit sowie Community bei der GUUG-Arbeit hilft. Seit September 2014 ist Dirk Vorsitzender des Interim-Vorstandes der GUUG.

Über Wolfgang

Wolfgang Stief arbeitet als freier Berater für Solaris und Storage-Systeme. 2012 hat er mit zwölf weiteren Freiberuflern im IT-Bereich die sys4 AG gegründet. Wolfgang ist 2002 in die GUUG eingetreten und übernahm 2003 die Organisation des Admin-Stammtisches in München. Von 2005 bis 2013 stellte sich Wolfgang erfolgreich erst als Beisitzer, dann als stellvertretender Vorsitzender des GUUG-Vorstandes zur Wahl. Als der Vorsitzende Martin Schulte 2008 vorzeitig zurücktrat, wählte ihn die Mitgliederversammlung in den Vorstandsvorsitz. Im September 2013 trat Wolfgang zusammen mit dem restlichen Vorstand als Vorsitzender zurück.

Aufruf
GUUG-Vorstandswahl 2015

Die Wahlkommission ermutigt alle Mitglieder, für einen Posten im Vorstand zu kandidieren.

von **Christian Lademann, Wahlleiter GUUG e.V.**

Wahlen allein machen noch keine Demokratie.

Barack Obama

Liebe GUUG-Mitglieder,

zu Beginn des kommenden Jahres steht wieder eine Vorstandswahl an, die wir entsprechend unserer Satzung per Briefwahl durchführen. Zu wählen sind:

- ein Vorsitzender,
- zwei Stellvertreter,
- ein Schatzmeister,
- bis zu fünf Beisitzer.

Die Wahlkommission möchte hiermit alle Mitglieder ermutigen, für einen Posten im Vorstand zu kandidieren. Berechtigt hierzu ist jedes persönliche Mitglied. Informationen über Aufgaben und zeitlichen Aufwand der Vorstandsmitglieder erteilt der amtierende Vorstand gern.

Ihr reicht Eure Kandidatur unter Angabe des angestrebten Amtes, Eures Namens und der Mitgliedsnummer bei der Wahlkommission ein. Sie muss bis zum **11.01.2015** eingegangen sein. Der Eingang wird durch den Wahlleiter bestätigt.

Eine Einreichung ist möglich als Brief oder E-Mail an:

GUUG e.V. Wahlkommission
c/o Christian Lademann
Neue Str. 6
56412 Nomborn
E-Mail: <kandidatur@guug.de>

Bis zum 14.02.2015 sollten Kandidaten außerdem eine Kurzvorstellung mit maximal 1.500 Zeichen und Bild als Beilage zu den Wahlunterlagen in elektronischer Form einreichen, ein Formatvorschlag wird zur Verfügung gestellt. Geplant ist auch ein Interview, in dem die Kandidaten unter anderem zu aktuellen Themen und Herausforderungen Stellung nehmen können.

Aus den gültigen Kandidaturen erstellt die Wahlkommission in Zusammenarbeit mit den Kandidaten die Unterlagen zur Briefwahl, die voraussichtlich am 21.02.2015 verschickt werden und dann bis zum 21.03.2015 an die Wahlkommission zurückgeschickt werden müssen.

Die Bekanntgabe des Ergebnisses sowie die Amtsübergabe an den neuen Vorstand erfolgen auf der Mitgliederversammlung am 25.03.2015 in Stuttgart.

Satzung und Wahlordnung finden sich auf unseren Webseiten unter http://www.guug.de/verein/satzung/.

Mit freundlichen Grüßen, Christian Lademann

Eines Tages im Linuxhotel
Erstes Redaktionstreffen der UpTimes

Am Samstag, 27. September 2014 traf sich das Redaktionsteam für einen Workshop-Tag im *Linuxhotel* in Essen. Das Redaktionsteam fand sich 2012, und so war es Zeit, sich persönllich kennenzulernen und am eigenen Tun zu tüfteln.

von **Anika Kehrer**

In zweifelhaften Fällen
entscheide man sich für das Richtige.

Karl Kraus

Wo steht das Projekt *UpTimes – Mitgliederzeitschrift der GUUG* [1] und wie möchte das Team (Abbildung 1) es weiterführen: Um das herauszufinden, reichte die Agenda von neun bis 18 Uhr und umfasste Einheiten zu *Rückschau*, *Status Quo* und *Zukunft*. Die drei Anwesenden beantworteten sich in den Einheiten Fragen, zum Beispiel *Wie findet Ihr das Erreichte und den Weg dorthin?*, *Wer macht was?*, *Was können die GUUG-Mitglieder im Moment von der UpTimes erwarten, und soll sich das ändern?* oder *Sollen wir von LaTeXnach AsciiDoc o. ä. umstellen?* Die Fragen, die wir uns stellen wollten, sammelten die Beteiligten vorher im Wiki. Für den Workshop selbst gab es eine Agenda, eine Ergebnisliste und einen Aktionsplan, sodass das Besprochene nachhaltig Niederschlag findet. Die komplette Dokumentation des Workshops ist im Wiki hinterlegt [2].

Abbildung 1: Das Redaktionsteam, von links: Mathias Weidner, Anika Kehrer, Robin Schröder. Fehlt: Titelgestalterin Hella Breitkopf. (Foto: Fremder Linuxhotel-Besucher)

Das Besprochene hat nach Ansicht der Verfasserin dieser Zeilen den gewünschten Zweck erfüllt. So ließ sich feststellen, dass das Projekt – in diesem Fall die UpTimes – an Struktur gewonnen habe, dass der Workflow neu entstanden sei und habe auch schon eine Reihe Verbesserungen erfahren hat. Die UpTimes war ja im Sommer 2012 in neuen Händen, in neuer Optik und im neuen, nämlich digitalen Medium praktisch ganz neu entstanden, nachdem die zuvor gedruckte Publikation einige Jahre pausiert hatte. Die erste Aufgabe war damals, ein neues Layout in LaTeXumzusetzen. So war es zu der heutigen Optik mit der schwarzen und grauen Fläche im Kopf jedes Artikels gekommen, zu der umrandenden Ameisenlinie aus der GUUG-eigenen Corporate-Design-Guide und zu den Rubriken-Labels rechts unten am Rand jeder ersten Artikelseite, die ebenfalls dem Guide entsprechen. Das alles wollte erst implementiert werden.

Workflow und Potenzial

Den Workflow stufen die Beteiligten mit heutigem Stand als relativ effizient ein: Robin Schröder (LaTeX-Setzer) und Mathias Weidner (ePub-Ersteller) berichteten, dass sie über den reinen Satz-Vorgang hinaus kaum mehr etwas zu tun hätten, weil sie ihre Tools optimiert haben und Anika Kehrer (Chefredakteurin und derzeit einzige Redakteurin bei der UpTimes) die Texte weitgehend passend getaggt anliefert. Im einzelnen fallen dennoch stets Prüfungen und Korrekturen beim Kompilieren und Schönmachen an. Die Redaktion sieht derzeit aber keinen Grund, von LaTeXwegzugehen, da sich der Artikel-Workflow gut eingespielt hat (Abbildung 2), sodass keine Vorteile einer Umstellung ersichtlich sind.

Das Herzstück war herauszufinden, wie die Beteiligten das Projekt sehen, was daran Spaß mache und wo es Entwicklungsmöglichkeiten sieht. Spaß

macht die UpTimes den Beteiligten, weil man daran herumhacken könne und etwas Anguckbares dabei herauskäme. Die UpTimes in der derzeitigen Form erreicht nach Meinung der Anwesenden vorwiegend technisch orientierte Leser, bei denen Unix- und Linux-Grundlagen vorhanden sind. Auf Autorenseite sei sie für solche Autoren interessant, die etwas sehr Spezielles oder sehr Technisches beschreiben möchten. Geeignete Inhalte seien Interesse-Artikel, Technologiegrundlagen, Rechtsfragen, Tooltipps, Vorstellung aktueller Software, sowie beispielsweise Geschichtsartikel, weil es hier dann mal nicht um Technik geht, was entspannend wirke.

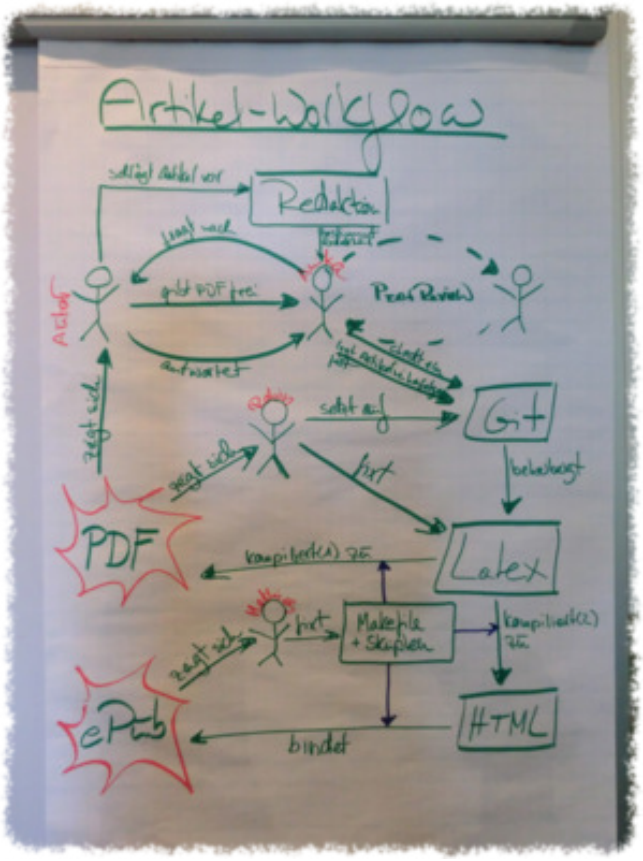


Abbildung 2: Artikel-Workflow der UpTimes.

Die redaktionelle UpTimes, von der hier die Rede ist, erscheint regulär zwei Mal im Jahr, was intern als Sommer- und Winterausgabe gilt. Mit dem Konzept aus Vereinsleben plus Fachartikel zeigten sich die Beteiligten zufrieden, wobei noch mehr Vereinsleben in Form von Nähe und Anschaulichem erwünscht war. Gut sind immer konkrete Zahlen: Bei dem Workshop lagen den Beteiligten die Zahlen der Aufrufe auf die URLs der einzelnen Ausgaben vor. Die Zugriffszahlen beziehen sich sowohl auf das PDF als auch auf das ePub, sodass die Zahlen nicht nur Seitenaufrufen, sondern Downloadzahlen entsprechen dürften. Demnach wurde das PDF der letzten Ausgabe 2014-1 (http://www.guug.de/uptimes/2014-1/uptimes_2014-01.pdf) 3.125 mal aufgerufen, die ePub-Ausgabe 323 mal (http://www.guug.de/uptimes/2014-1/uptimes_2014-01.epub). Offen bleibt lediglich die Frage, wieviele von den Zugriffen auf Webcrawler zurückzuführen sind. Die Zugriffszahlen zeigt Tabelle 1.

Ausgabe	ePub	PDF
2014-1	323	3.125
2013-3	533	6.014
2013-2	2.728	15.117
Zum Vergleich: 2013-1 (Proceedings)	–	4.950
2012-3	1.192	30.037
2012-2	1.226	6.937

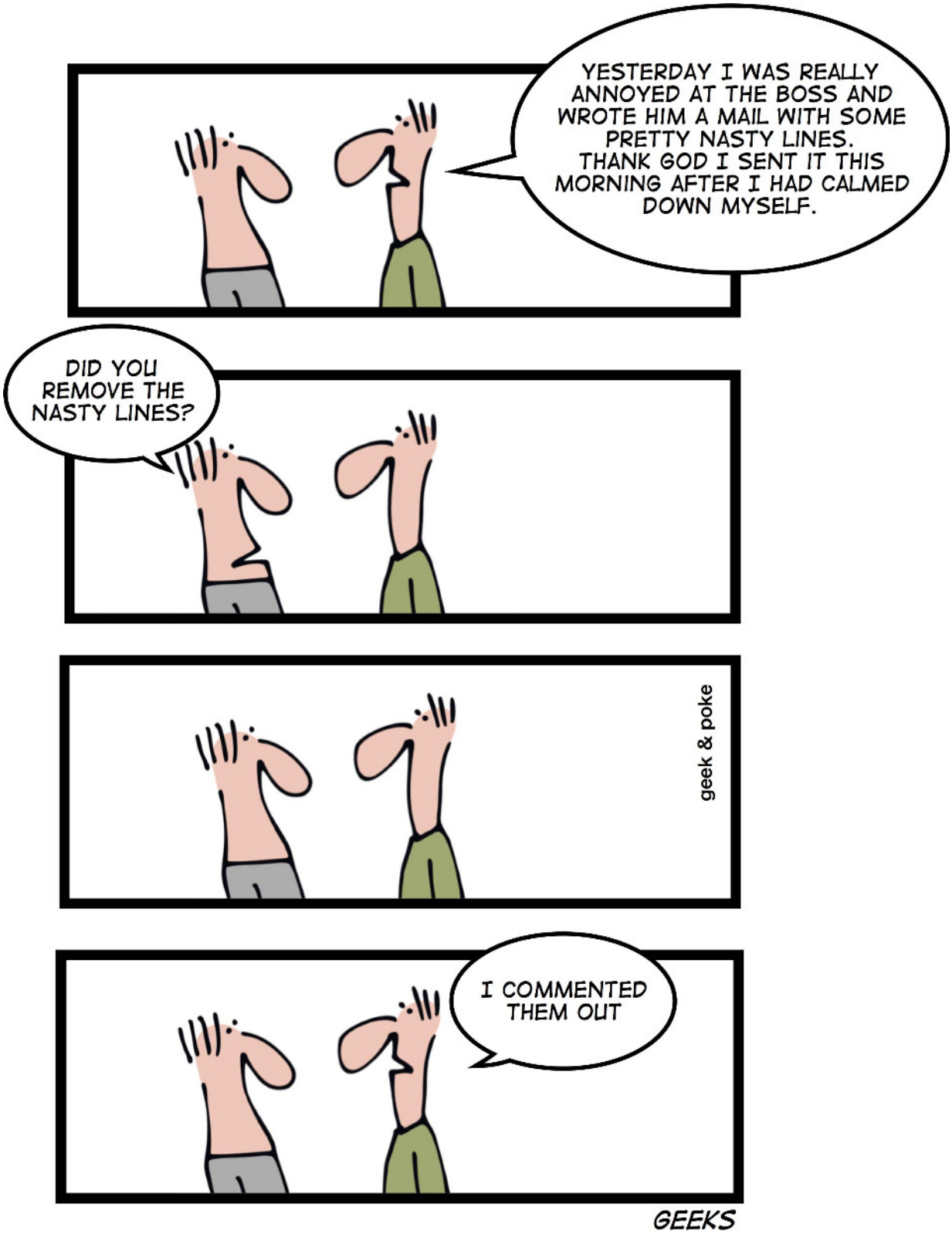
Tabelle 1: Zugriffe auf URLs der UpTimes-Ausgaben seit der ersten redaktionellen Ausgabe im Sommer 2012 laut Serverlog. Zu berücksichtigen: Webcrawler sind nicht herausgefiltert.

Das wichtigste Entwicklungspotenzial sieht das Team derzeit bei mehr Beteiligungsmöglichkeiten. Dazu gehören Fachbeiträge, aber auch die Möglichkeit für GUUG-Mitglieder, als Redakteure oder Übersetzer mitzuarbeiten. Weiter kam die Idee auf, einen Peer-Review-Prozess einzubinden. Dieser geschieht zwischen Redaktion und Nichtredaktionsmitgliedern punktuell bei einzelnen Artikeln. Eine Zusammenstellung zeigt der Kasten *Beteiligungsmöglichkeiten bei der UpTimes*.

Abbildung 3: Abendlicher Besuch im Essener *Unperfekthaus*.

Der Ort des Treffens, das *Linuxhotel* [3] in Essen, bietet mit seinem umliegenden Parkanlagen eine sehr schöne Arbeitsumgebung. Es besteht aus einer ehemaligen Industriellenvilla, die danach 30 Jahre lang als Kloster diente, bis der heutige Besitzer Reinhard Wiesemann sie 1994 kaufte. 2010

haben er und ein Team den größten Teil der Anlage als Schulungszentrum eröffnet. Ebenfalls ein Projekt von Reinhard Wiesemann ist das *Unperfekthaus* [4] in Essen, wo die Workshop-Teilnehmer nach getaner Arbeit den Abend verbrachten (Abbildung 3).



Beteiligungsmöglichkeiten bei der *UpTimes*

Autor werden

Dies ist die klassischste und definierteste Form der Beteiligung an einer Publikation. Ein Autor der *UpTimes* verfasst Fachbeiträge oder Beiträge aus dem Vereinsleben. Fachbeiträge können aus beliebigen Bereichen stammen, die im Interessenbereich der GUUG liegen. Praxisberichte, Alltagsabenteuer, Vorstellung interessanter Software oder sonstiger Technologien sowie Einführungen in neue Themen sind interessant, Basiswissen im Stil von *Einführung in die Bourne Shell* hingegen nicht. Daneben sind Berichte oder Kommentare aus dem Vereinsleben, Buchrezensionen, Konferenzberichte, humoristische Formen und Leserbriefe ausdrücklich erwünscht. Näheres enthalten die *Autorenrichtlinien* im hinteren Teil der UpTimes.

Übersetzer werden

Dies ist auf eine Anregung aus dem Leserkreis zurückzuführen, die in den Workshop eingeflossen ist. So entstand folgende Beteiligungsmöglichkeit: Übersetzer finden im Web oder in Büchern interessante fremdsprachige Texte, die sie für die UpTimes ins Deutsche übersetzen möchten. Sie schlagen einen solchen Text der Redaktion vor, so wie sie einen eigenen Artikel vorschlagen würden. Bei Zusage der Redaktion nehmen sie Kontakt mit dem Autoren auf und fragen die Erlaubnis an, den Text zu übersetzen und unter der CC-BY-SA-Lizenz in der UpTimes veröffentlichen zu dürfen. Bei Zusage des Autoren übersetzen sie den Text. Diesen reichen sie wie einen eigenen Artikel bei der Redaktion ein.

Vermittler werden

Eine Erkenntnis des Workshops war, dass man sich mit dieser Beteiligungsmöglichkeit nur sehr punktuell, relativ einfach und doch höchst effektiv bei der UpTimes einbringen kann. Die Akquise von Artikeln ist ja die Voraussetzung einer Publikation. Für jemanden, die oder der sich vielleicht nicht umfänglich oder dauerhaft einbringen möchte, braucht sie nur aus einem Ansprechen potenziell interessanter Autoren zu bestehen. Es genügt, zum Beispiel Kollegen, Speaker oder Bekannte auf ein bestimmtes Thema anzusprechen und ihnen vorzuschlagen, sich bei der Redaktion zu melden. Idealerweise geht dieses Ansprechen möglicher Autoren über ein *Willste nicht mal . . .* etwas hinaus, denn je konkreter und Willkommen-heißender jemand angesprochen wird, desto höher ist die Wahrscheinlichkeit, dass er oder sie Ja sagt. Doch über diese Anbahnung hinaus kann alles weitere auf der Redaktionsmailingliste passieren, was die Task des Vermittlers sehr überschaubar macht.

Redakteur werden

Dies ist die lehrreichste und aufwändigste Beteiligungsmöglichkeit an der UpTimes. Der Redaktionsprozess passiert zwischen Redakteur und Autor. GUUG-Mitglieder können bei der UpTimes selbst Redakteur sein: Sie schauen sich nach interessanten Themen um, sprechen Autoren an, korrespondieren mit der Chefredaktion und mit dem Autoren und stehen diesem ggf. bei der Themenentwicklung zur Seite. Sie sind der erste Testleser: Sie prüfen und bearbeiten gegebenenfalls einen eingegangenen Beitrag hinsichtlich Konsistenz (bleiben Fragen offen?), Argumentation (ist es plausibel?) und Lesbarkeit (zu lange Sätze?). Sie achten am Ende auf die UpTimes-eigenen LaTeX-Formate und pipen den Beitrag an die Chefredaktion (das Sicherheitsnetz). Wichtig ist zu erwähnen, dass ein UpTimes-Redakteur nicht alle genannten Aufgaben erfüllen und schon gar nicht alles perfekt können muss.

Peer-Reviewer werden

Diese Beteiligungsmöglichkeit setzt voraus, dass ein Redakteur das Handling übernimmt (Anika selbst implementiert aus Zeitgründen kein Peer Review). Das Peer Review passiert zwischen Redakteur und Peer-Reviewer, nicht etwa zwischen Autor und Peer-Reviewer. Es bezieht sich auf den fachlichen Inhalt eines Artikels. Bevor der Redakteur beginnt, sich mit einem geliefertem Artikel zu Prüfzwecken auf Inhaltsebene zu beschäftigen, kann er das Feedback eines Peer-Reviewers anfragen. Er kann den Input dann in seine Autorenkorrespondenz einfließen lassen, etwa um Fragen zu klären oder punktuelle Ergänzungen vorzuschlagen. Auf diese Weise erhält die Qualitätssicherung eine weitere Dimension. Der Redakteur ist jedoch nicht verpflichtet, den Input des Peer-Reviewers einzubeziehen.

Die UpTimes-Redaktion ist unter der E-Mail-Adresse <redaktion@uptimes.de> zu erreichen. Redaktionsmitglieder sind Robin Schröder (LaTeX-Satz), Mathias Weidner (ePub-Satz, Autor, Redakteur), Hella Breitkopf (Titelbild), Anika Kehrer (Chefredaktion, <kehrer@guug.de>), Dirk Wetter (Vorstandsvorsitzender, verantwortlich im Sinne des Presserechts) und Wolfgang Stief (ehemals Vorstandsvorsitzender, Initiator der 2012-er Neuauflage der UpTimes). Die Entstehung jeder UpTimes ist für GUUG-Mitglieder öffentlich im GUUG-Wiki dokumentiert (Beispiel Frühlingsausgabe 2014-2 [5], generell [6]).

Links

[1] *UpTimes*, Mitgliederzeitschrift der GUUG: http://www.guug.de/uptimes/
[2] Dokumentations-PDF des Redaktionstreffens (ganz oben unter *Ergebnisse*, zugänglich für GUUG-Wiki-Accountinhaber): https://wiki.guug.de/uptimes/redaktionstreffen-2014
[3] Linuxhotel: http://linuxhotel.de
[4] Unperfekthaus: http://www.unperfekthaus.de
[5] Orga-Seite einer UpTimes-Ausgabe im GUUG-Wiki, Beispiel Frühlingsausgabe 2014-2 (zugänglich für GUUG-Wiki-Accountinhaber): https://wiki.guug.de/uptimes/2014-2
[6] Übersicht der Orga-Seiten im GUUG-Wiki (unter *Organisation der einzelnen Ausgaben*, zugänglich für GUUG-Wiki-Accountinhaber): https://wiki.guug.de/uptimes/index

Über Anika

Anika Kehrer arbeitet seit acht Jahren als Fachautorin und Redakteurin mit dem Schwerpunkt IT und Technik, vier davon als freie Journalistin in München und Berlin. Seit der Wiederauflage der UpTimes im Sommer 2012 gestaltet sie als Chefredakteurin die UpTimes als Fachmagazin und Vereinszeitschrift der GUUG in Zusammenarbeit mit einem Redaktionsteam aus GUUG-Mitgliedern.

Das Auge denkt mit
Paketfilterregeln von *iptables* visualisieren

Der kleine Bericht aus der Praxis zeigt, wie man sich die komplexen Iptables-Filterregeln verständlich(er) machen kann, indem man sie veranschaulicht. Dazu dienen *iptables-save*, *dot* und das eigene Perl-Modul *App::Iptables2Dot*.

von **Mathias Weidner**

Wessen Auge nicht sieht,
dessen Mund nicht wässert.

Chinesisches Sprichwort

Für eines meiner Bücher [1] beschäftigte ich mich mit Hardware für selbstgebaute Router und entdeckte so *OpenWrt*, das sich gut als Betriebssystem für diesen Zweck eignet. Was mich faszinierte, war die konsistente Konfiguration des gesamten Systems mit *UCI* (Unified Configuration Interface) beziehungsweise *LuCI* (ein in Lua geschriebenen Web-Configuration-Interface). Für das Buch wolle ich einerseits beschreiben, welche Einstellungen in LuCI oder UCI welche Auswirkungen auf die Paketfilter haben, andererseits, was ich eingeben muss, um bestimmte Ziele zu erreichen. Mein Plan war, minimale Änderungen am Webinterface zu machen, und diese mit dem CLI (UCI) und der Ausgabe von *iptables-save* zu vergleichen.

Als erstes stellte ich fest, dass ich überhaupt keine Ahnung hatte, wofür OpenWrt so viele Filterketten verwendet. Bereits bei zwei Firewall-Zonen (LAN, WAN) erzeugte die Firewall-Konfiguration von *OpenWrt Backfire* 23 benutzerdefinierte Filterketten in Iptables, auf die es die Regeln verteilt. Bei drei Zonen (LAN, WAN, DMZ) sind es bereits 30 Ketten. Ich fand keine Dokumentation, aus der ich hätte lernen können, welchen Zweck sie haben. So blieb nur, es selbst herauszufinden. Dafür galt es, die Bedeutung der Iptables-Ketten zu erkennen, damit ich die Regeln in ihrem Kontext sehen konnte.

Regeln als Graph-Diagramm

Zu diesem Zweck lässt sich die Netfilter-Firewall als *gerichteter Graph* darstellen, dessen Knoten die Regelketten sind. Die Kanten zwischen den Knoten sind die Sprungregeln, durch die die Auswertung in einer anderen Kette fortfährt. Diesen Graph zu Papier zu bringen hilft, ein Verständnis für das Zusammenspiel der Regeln zu entwickeln, die es in Iptables nur als Liste gibt.

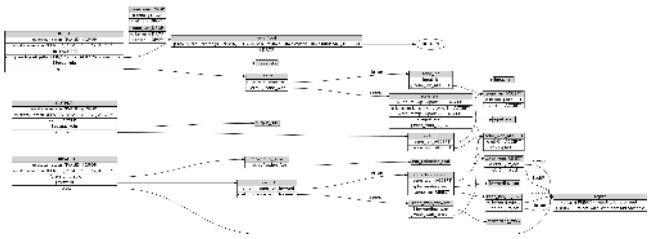


Abbildung 1: Graph der IPtables-Filterregeln. Man muss die Einträge nicht lesen können – es gilt nur zu zeigen, wie komplex die Darstellung sämtlicher Sprungregeln ist.

Ich brauchte also etwas, das mir die drögen Iptables-Tabellen als Graphen darstellt. Dieses Etwas fand ich in dem Programm *dot* aus dem Paket *GraphViz*. Dot liest Textdateien, die einen Graphen beschreiben, und gibt den Graphen als Bild in verschiedenen Formaten aus – zum Beispiel PDF, falls ich es ausdrucken und neben den Bildschirm legen möchte, oder PNG, falls ich es in einer HTML-Seite verwenden will. Das Ergebnis sieht aus wie in Abbildung 1.

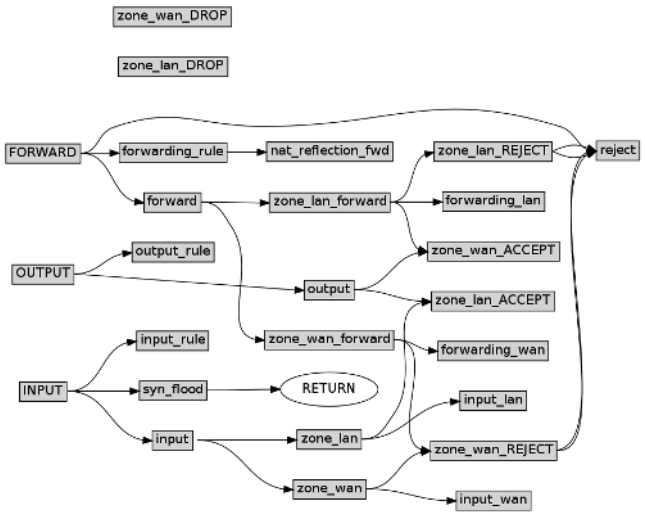


Abbildung 2: Der gleiche Graph ohne Sprungregeln, um Übersicht zu gewinnen.

Da ich bereits einige Netfilter-Firewalls gesehen habe, deren Regeln in die Hunderte gehen, wollte ich außer der kompletten Darstellung aller Regeln mit allen Sprüngen, wie in Abbildung 1, auch eine vereinfachte Darstellung. Diese sollte nur die Ketten mit jeweils einer Kante zwischen zwei Ketten enthalten – egal, wie viele Sprungregeln es gibt. Das Ergebnis ist Abbildung 2.

Noch anschaulicher wird das Modell, wenn ich gleichartige Ketten zusammenfasse und mit Platzhaltern für austauschbare Namensbestandteile versehe. Dazu sind die Graphenbeschreibungen für Dot von Hand zu ändern. Das Ergebnis zeigt Abbildung 3.

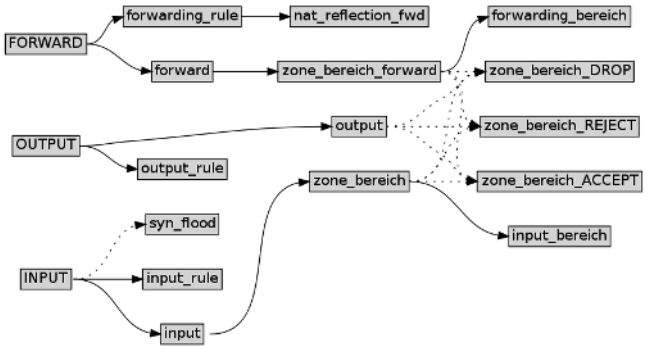


Abbildung 3: Manuell bearbeiteter Graph mit Platzhalterbezeichnungen und gepunkteten Linien. Letztere weisen auf Knoten hin, die bei bei manchen Firewall-Konfigurationen nicht zum Einsatz kommen.

Mit dieser Darstellung konnte ich gut arbeiten: In den Knoten steht zum Beispiel der Bezeichnungsteil ‚bereich, für eine beliebige Zone wie WAN oder LAN. Die gepunkteten Linien bei einigen Kanten bedeuten, dass die betreffenden Knoten bei manchen Firewall-Konfigurationen nicht erreicht werden und dann aus der Betrachtung herausfallen können.

Werkzeuge

Iptables-save listet die Firewall-Regeln und Ketten auf. Dot von GraphViz erzeugt aus geeigneten Beschreibungen Bilder in verschiedenen Formaten. Was fehlt, ist ein Programm, das aus der Tabelle von Iptables-save die Beschreibung für Dot generiert. Das erledigt das Programm *iptables2dot*, Bestandteil meines dafür geschriebenen Perl-Moduls *App::Iptables2Dot* [3]. Das kam so: Für die Arbeit am eingangs erwähnten Buch reichte zunächst ein einfaches Skript. Im Laufe der Arbeit daran hatte ich aber immer wieder etwas zu ergänzen, sodass ich schließlich das Perl-Modul daraus machte, um die Weiterentwicklung in geordnete Bahnen zu lenken.

Das Aussehen der erzeugten Graphen lässt sich mit Optionen für Iptables2dot beeinflussen Tabelle 1). Bei allen Listen in den Optionen werden die einzelnen Elemente durch Komma getrennt.

Option	Beschreibung
`-edgelabel`	Damit erhalten die Kanten zu den Sprungzielen Label, die das Input- oder Output-Device angeben.
`-omittargets targetlist`	Lässt einzelne Sprungziele – also Knoten im Graphen – aus, um den Graphen übersichtlicher zu halten. *RETURN* ist ein Kandidat dafür.
`-noshowrules`	Unterdrückt die Anzeige der Regeln bei den Knoten.
`-showunusednodes`	Zeigt auch Regelketten, zu denen keine Sprünge führen. Diese Ketten werden sonst nicht angezeigt.
`-tables tablelist`	Zeigt die Ketten aller angegebenen Tabellen. Mögliche Tabellen sind *filter*, *mangle*, *nat*, *raw*). Ohne Angabe werden nur die Regeln der Tabelle *filter* angezeigt.

Tabelle 1: Optionen für *iptables2dot*

Der typische Arbeitsablauf mit dem Modul sieht so aus:

```
sudo iptables-save \
| iptables2dot -noshowrules -table filter \
iptables-filter-overview.dot
dot -Tpdf iptables-filter-overview.dot  \
-o iptables-filter-overview.pdf
```

Oder so, für den detaillierten Graphen:

```
sudo iptables-save \
| iptables2dot -edgelabel -table filter \
iptables-filter.dot
dot -Tpdf iptables-filter.dot  \
-o iptables-filter.pdf
```

Im Laufe der Zeit stellte sich ein Problem mit dem Programm heraus. Es kennt nicht alle Optionen von Iptables und stolpert somit in einigen Fällen über die Ausgabe von Iptables-save. Für diesen Fall gibt es die Option `-addoptdef optdef`, die dem Skript über die Kommandozeile die vorher unbekannte Option von Iptables bekannt macht. Das Modul verwendet intern *Getopt::Long*, um die Ausgabe von Iptables-save zu parsen. Alle mit `-addoptdef optdef` angegebenen Optionen werden an *Getopt::Long* als zusätzliche Optionen für die Funktion `GetOptions()` durchgereicht und dann ignoriert, weil zur Aufbereitung für den Graphen nur die Sprünge herangezogen werden.

Übrigens: *CPAN*, das Online-Repository für Perl-Module, erweist sich wieder einmal als sehr nützlich. Man braucht hier nur ein Modul hochzuladen, schon stürzen sich Hunderte von Testern

darauf und probieren es mit den verschiedensten Versionen von Perl und unterschiedlichen Betriebssystemen. Ich kann mir keine bessere Qualitätskontrolle denken. Sie zwingt außerdem zu sorgfältiger Arbeit, wenn man nicht mit einem ‚FAIL, der CPAN-Tester leben will.

Andere Lösungen

Im Internet fand ich seinerzeit nur *gressgraph* [3], das sich für meine Zwecke nicht eignete. Erst, als ich *App::Iptables2Dot* fast fertig hatte, stolperte ich über ein Python-Programm [4], das ähnlich vorgeht und ähnliche Ausgaben erzeugt wie Iptables2dot.

Links

[1] Buch, für das ich das gemacht habe:
http://buecher.mamawe.net/buecher/headless-linux/
[2] Perl-Modul: http://search.cpan.org/~mamawe/App-Iptables2Dot/
[3] Gressgraph: http://jekor.com/gressgraph/
[4] Python-Programm zur Visualisierung von Iptables-Ketten:
http://www.vitavonni.de/blog/200704/2007040901-visualizing-iptables.html

Über Mathias

Mathias Weidner studierte Ende der 1980-er Jahre Automatisierungstechnik in Leipzig. Nach verschiedenen Stellen in der Software-Entwicklung landete er bei einem kleinen Rechenzentrum in Wittenberg als Administrator für Unix/Linux und Netzwerke. Seit einiger Zeit schreibt er hin und wieder Bücher zum Thema, die gedruckt, als unter http://buecher.mamawe.net als PDF oder ePUB erhältlich sind.

Wer nicht fragt bleibt dumm
OpsReportCard – Fragen für Sysadmins

Thomas Limoncelli, Autor mehrerer guter Bücher zum Thema Systemadministration, hat analog zum Joel-Test für Software-Entwickler einen Fragenkatalog für Systemadministratoren zusammengestellt: die *Operations Report Card*. Dieser Artikel übersetzt die Fragen gekürzt ins Deutsche.

Übersetzung: **Mathias Weidner**

Ob ein Mensch klug ist,
erkennt man an seinen Antworten.
Ob ein Mensch weise ist,
erkennt man an seinen Fragen.

Nagib Machfus

Der Fragenkatalog *The Limoncelli Test: 32 Questions for Your Sysadmin Team* von **Thomas Limoncelli** entstand im Jahr 2011 in Anlehnung an *The Joel Test – 12 Steps to Better Code* von Joel Spolsky aus dem Jahr 2000 [1]. Die so genannte 'Operations Report Card' ist inzwischen auf einer eigenen Webseite beherbergt [2], nachdem sie vorher auf Thomas Limoncellis Homepage lag. Die Fragen sind nützlich bei Einstellungsgesprächen, um die Firma besser einzuschätzen. Sie helfen aber auch zu sehen, wo und wie man etwas bei seiner Arbeit in der derzeitigen Firma besser machen kann. Für die UpTimes erfolgte die Übersetzung mit Genehmigung des Autoren.

A. Öffentlich sichtbare Praktiken

1. Verfolgt ein Ticketsystem die Benutzeranfragen?

Das ist so grundlegend, dass es weh tut, das zu erklären. Menschen können sich nicht so gut erinnern wie Computer. Von Systemadministratoren zu erwarten, dass sie sich an alle Benutzeranfragen erinnern, ist der direkte Weg, um Anfragen unter den Tisch fallen zu lassen.

Anforderungen in einer Datenbank zu erfassen verbessert den Zugriff innerhalb des Teams. Es verhindert, dass zwei Leute in der gleichen Angelegenheit gleichzeitig tätig werden. Es ermöglicht den Systemadministratoren, die Arbeit untereinander aufzuteilen. Es ermöglicht es, eine Aufgabe von einem auf den nächsten zu übertragen ohne den Überblick über das schon Getane zu verlieren. Es ermöglicht anderen Systemadministratoren für jemand einzuspringen, der ausser Haus ist, unabkömmlich oder im Urlaub.

Es ermöglicht besseres Zeitmanagement. Jede Unterbrechung, die ein Systemadministrator erfährt, setzt ihn in seiner Arbeit um sieben Minuten zurück. Ein Ticketsystem verhindert Unterbrechungen von Benutzern, die eine Anforderung haben oder nach dem Status der Bearbeitung fragen wollen. Es läßt den Systemadministrator seine Arbeit priorisieren, anstatt auf den lautesten Beschwerdeführer zu antworten.

Abbildung 1: Ticketsysteme sind für die Problemschau effizienter als Einzel-E-Mails oder Glaskugeln. (Foto: Eva K. @ Wikipedia, CC-BY-SA)

Es lässt auch Manager bessere Manager sein. Steckengebliebene Anforderungen werden sichtbar, so dass der Manager eingreifen kann. Es offenbart Trends und häufige Anfragen, sodass diese

durch Automatisierung oder Prozessverbesserung eliminiert werden können. Mikromanager können die Informationen, die sie haben wollen, von der Software bekommen, anstatt den Systemadministrator zu belästigen. Es hilft zum Beispiel, systematische Probleme zu identifizieren: Ein ehemaliger Chef befragte das Ticketsystem und entdeckte, dass drei von 1000 Benutzern zehn Prozent aller Tickets geöffnet hatten. Eine kleine Untersuchung und wir waren in der Lage, die grundlegende Angelegenheit zu lösen, die das verursacht hatte. Und es ersetzt jammernde Benutzer durch evidenzbasierte Diskussion: Wenn eine Anforderung wirklich zu lange gebraucht hat, kann der Manager die Sache angehen. Wenn es dem Benutzer lediglich so erscheint, dass die Sache zu lange dauert, wird ihn der Beweis zur Ruhe bringen. Wenn Benutzer sagen: „Das Problem besteht schon monatelang, aber ich öffnete erst gestern ein Ticket“, kann der Manager den Benutzer belehren über die Nichtexistenz von Zeitreisen, Gedankenlesern und anderen übernatürlichen Kräften (Abbildung 1).

2. Sind die *drei Entscheidungsrichtlinien* definiert und veröffentlicht?

Es gibt drei Richtlinien, die nötig sind, wenn Systemadministratoren überhaupt irgendwelche Arbeit schaffen können sollen. Es geht ebenso um den Dienst am Kunden, wie um die Effizienz des Teams. Wenn ein Manager das Gefühl hat, ein Team ist nicht gut im Zeitmanagement, liegt es vielleicht daran, dass er diese drei Richtlinien nicht durchgesetzt hat:

- Die zulässigen Methoden für Benutzer, um Hilfe zu bekommen.
- Die Definition eines *Notfalles*.
- Der Zuständigkeitsbereich eines Dienstes: Wer, was und wo.

Ein Dokument kann alle drei Dinge in weniger als einer Seite erläutern. Dieses gehört auf die Website der Abteilung oder auf Postern an der Wand, sodass es ersichtlich ist. Diese Richtlinie muss auch von der Geschäftsführung getragen werden. Das bedeutet, dass ein Benutzer ein Nein zur Antwort bekommt, wenn er nach einer Ausnahme fragt. Der Prozess für die Ausnahme sollte keine Temposchwelle sein, sondern eine solide Wand.

3. Erfasst das Team monatliche Metriken?

Daten müssen zur Hand sein, wenn Entscheidungen anstehen oder die oberen Ebenen des Managements zu beeinflussen sind (Abbildung 2). Ein Beispiel: Die Richtlinie für ein PC-Bereitstellungsteam lautet, einen standardisierten High-End-PC für jeden Benutzer ab dem ersten Tag bereitzustellen, den es alle drei Jahre zu branchenüblichen Betriebskosten ersetzt. Die Metriken dafür könnten sein:

- Anzahl der Wochen, seit die Standardkonfiguration aktualisiert wurde;
- Kosten der aktuellen Standardkonfiguration;
- Anzahl der neuen Angestellten pro Monat;
- Kapitalaufwendungen;
- Betriebsaufwendungen;
- wieviele Tage neue Angestellte auf ihre Maschine warten (*vor der Ankunft, 1. Tag, 2., 3., 4., 5. Tag, mehr als 5 Tage*);
- Alter der Flotte (*< 1 Jahr, 1-2 Jahre, 2-3 Jahre, > 3 Jahre*);
- Anzahl der Maschinen, die vom Standard abweichen.

Alternativ sind hier ein paar Starter-Metriken, die jeder ab heute einsetzen kann. Zeichnen Sie diese jeweils am Ersten des Monats auf. Packen Sie sie in eine Kalkulationstabelle. Sie können diese zur Bestimmung des Budgets verwenden, oder für Präsentation, wenn Sie erläutern, was Ihre Gruppe macht:

- Wie viele Systemadministratoren haben wir?
- Für wie viele Benutzer bieten wir Dienste an?
- Wie viele Maschinen verwalten wir?
- Wie viel Plattenplatz insgesamt? RAM? CPU-Kerne?
- Wie viele offene Tickets sind gerade im Ticketsystem?
- Wie viele neue Tickets gibt es seit dem letzten Monat?
- Was war der Durchschnitt an Tickets pro Administrator im letzten Monat?
- Wie viel Internet-Bandbreite wurde letzten Monat verbraucht?
- Nehmen Sie vier bis fünf wichtige SLAs und zeichnen Sie auf, wie nahe Sie ihrer Erfüllung kommen.

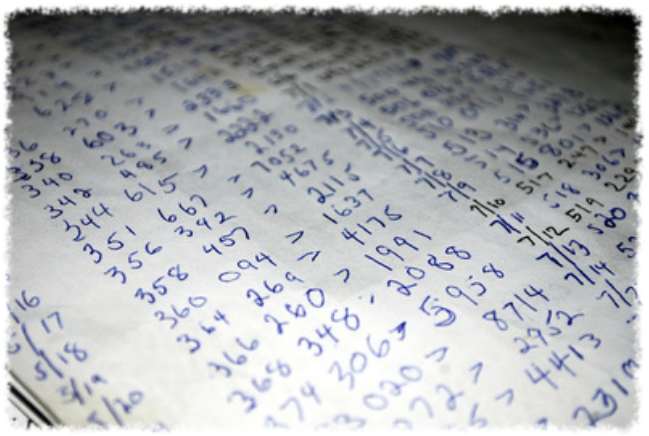


Abbildung 2: Daten helfen, Entscheidungen zu treffen oder das obere Management zu beeinflussen.

B. Moderne Praktiken der Gruppenarbeit

4. Gibt es ein Wiki für Richtlinien und Abläufe?

Ein Team braucht ein Wiki, um Richtlinien (was soll getan werden) und Abläufe (wie wird es gemacht) zu dokumentieren.

Automatisierung ist großartig, aber bevor jemand etwas automatisieren kann, muss er in der Lage sein, es von Hand zu tun. Den Ablauf von Hand für etwas zu dokumentieren ist eine Vorbedingung für Automatisierung. Einstweilen ermöglicht es konsistente Abläufe quer durch das Team, und Leiter erlangen die Fähigkeit zu delegieren. Wenn es dokumentiert ist, kann es jemand anders machen. Das Inhaltsverzeichnis für das Wiki sollte die allgemeinen Routineaufgaben enthalten, insbesondere die Anschluss-, Änderungs- und Lösch-Prozeduren, die jeder im Team ausführen können sollte. Beispiel für eine Verfahrensliste:

- Wenn ein neuer Mitarbeiter anfängt;
- Wenn ein Mitarbeiter die Firma verlässt;
- Wenn ein Mitarbeiter entlassen wird;
- Wenn eine neue Maschine zu installieren ist;
- Wenn eine Maschine stillzulegen ist;
- Wie man eine Person zum VPN-Dienst hinzufügt oder davon entfernt;
- Wie man eine Platte im RAID-System wechselt;
- Wie man das Kennwort von root auf allen Maschinen ändert.

Erfasst sind hier drei Kategorien: Dinge, die Sie konsistent haben wollen; Dinge, die sie selten tun und bei denen Sie keine Zeit verschwenden wollen, sich an den Prozess zu erinnern; und schließlich Dinge, die Sie in Panik tun und bei denen Sie nicht über die nächsten Schritte nachdenken wollen. Diese Sachen können alle mit einfachen Schritt-für-Schritt-Checklisten dokumentiert werden. Das schafft auch ein Trainingsprogramm für neue Mitarbeiter. Es ist außerdem dienlich, um die Stellenbeschreibung für den Assistenten zu schreiben, den Ihre Firma für Sie einstellen soll, damit er Ihre ganze Arbeit macht.

Für alle Aufgaben will man vermutlich verschiedene *Prozess*- und *Ablauf*-Dokumente haben. Prozess ist, was das Management definiert: Alle neuen Benutzer bekommen eine drahtlose Maus. Ablauf ist, wie die Aufgabe abgearbeitet wird: Drahtlose Mäuse sind im dritten Behälter, lade sie auf und teste sie mit den folgenden Schritten. Prozesse ändern sich nur mit Zustimmung des Managements. Abläufe werden hingegen durch die Techniker geändert.

Viele Systemadministratoren hassen es, Dokumentationen zu schreiben. Aber Schritt-für-Schritt-Checklisten sind nicht so schlimm. Sie in einem Wiki vorzuhalten, ist wichtig: Jeder kann es korrigieren und verbessern. Selbst, wer nicht in einem Team arbeitet oder Aufgaben hat, die nur er tut, für den hat Dokumentieren Vorteile. Man braucht weniger nachzudenken, wenn eine Aufgabe ansteht. Genauso wahr wie das Sprichwort „Wir automatisieren, weil wir faul sind“ ist „Wir dokumentieren, weil wir ungeduldig sind“.

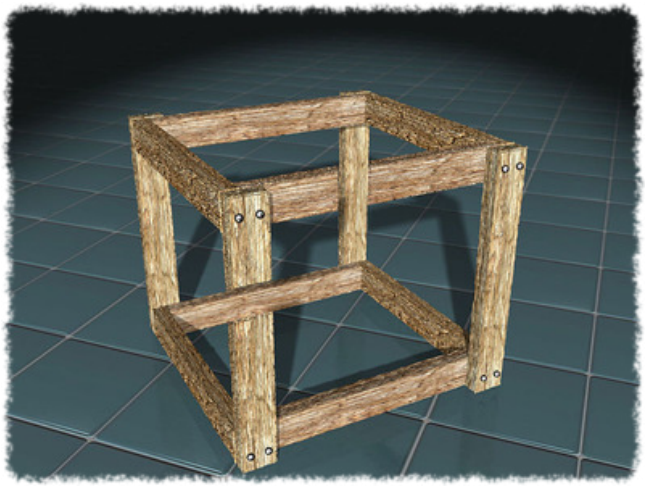


Abbildung 3: Als Passwort-Tresor reicht oft schon eine sichere Kiste. Es muss ja nicht gerade die *unmögliche Kiste* von Richard Gregory sein. (Grafik: Dominique Toussaint @ Wikipedia, CC-BY-SA)

5. Gibt es einen Passwort-Safe?

Er zeigt, dass jemand Kennworte auf durchdachte Weise verwaltet. Es gibt viele Software-basierte Passwort-Tresor-Systeme. Dabei ist ein Umschlag in einer sicher verschlossenen Blechkiste oft ausreichend (Abbildung 3). Das Problem ist oft die

Verifizierung: Woher wissen Sie, dass kein böswilliger Mitarbeiter die falschen Kennworte in den Umschlag legt? Wenn Sie so paranoid sind, lassen Sie eine andere Person alle neuen Kennworte prüfen.

6. Wird der vom Team geschriebene Code mit Versionsmanagement gepflegt?

Wenn die Installation einer neuen Maschine ein API-Aufruf ist, sind wir jetzt alle Programmierer. Und Programmierer verwenden Quellcode-Verwaltung. Dinge, die in das Repository gehören, sind Skripten, Programme, Konfigurationsdateien, Dokumentation und eigentlich fast alles. Im Zweifelsfall lautet die Antwort ‚Ja, es gehört hinein". Die Konfigurationsdateien in die Quellcode-Verwaltung zu bringen, fühlt sich zunächst wie Luxus an und rettet Ihnen am Ende die Haut.

7. Verwendet das Team eine Fehlerdatenbank für selbstgeschriebenen Code?

Fehlerdatenbanken unterscheiden sich von Ticketsystemen. Wenn es nur gelegentlich Fehler gibt – vielleicht schreibt ein Team nur wenig Code – dann reicht es, Tickets für sich selbst zu öffnen. Falls es dem Team jedoch ernst ist mit dem Schreiben von Code, startet es eine eigene Fehlerdatenbank. Fehlerdatenbanken haben einen anderen Arbeitsablauf als Ticketsysteme. Ein Ticketsystem ist ein Kommunikationswerkzeug zwischen Admin und Benutzer. Fehlerdatenbanken hingegen verfolgen den Lebenszyklus eines Fehlers (*Melden, Verifizieren, Zuweisen, Beheben, Schließen, Verifizieren*).

8. Hat Stabilität Vorrang vor neuen Features bei Fehlern oder Tickets?

Neue Features hinzufügen macht mehr Spaß, als Fehler zu beseitigen. Leider geht es hier nicht um Spaß. Stabilität ist zu sichern, bevor neue Feature kommen. Sicherheitsfragen sind vorrangige Fragen der Stabilität. Bewährte Teams priorisieren ihre Fehler also auf folgende Weise:

- Sicherheit (am höchsten)
- Stabilität
- Fehler
- Performance
- neue Feature (am niedrigsten)

Einige Änderungen fügen neue Feature hinzu, während andere die Stabilität verbessern. Die Reihenfolge sollte sein: *Feature, Stabilität, Feature, Stabilität, Feature, Stabilität.* Nicht: *Feature, Feature, Feature, OMG!, Stabilität, Stabilität, Stabilität.* Machen Sie die Dinge solide, bevor Sie mit dem nächsten beeindruckenden Feature weitermachen. Ärzte verstehen das. In der Notaufnahme stabilisieren sie den Patienten zuerst. Sie kurieren niemanden zuerst von der Grippe, wenn dieser am Verbluten ist.

9. Schreibt das Team Entwurfsdokumente?

Gute Teams von Systemadministratoren denken, bevor sie handeln. In einem größeren Team ist es wichtig zu kommunizieren, was jemand zu tun beabsichtigt oder getan hat. Ein Entwurfsdokument ist ein standardisiertes Format, um neue Dinge vorzuschlagen oder bestehende Dinge zu beschreiben. Ein vorab gestaltete Muster und kann überall Verwendung finden. Die Überschriften könnten folgendes enthalten:

- Überblick
- Ziele
- Nicht-Ziele
- Hintergrund
- Vorgeschlagene Lösung
- In Betracht gezogene Alternativen
- Sicherheit
- Wiederherstellung im Notfall
- Kosten

Das Entwurfsdokument kann mehr Überschriften haben oder weniger, einige können optional sein, andere notwendig. Es empfiehlt sich eine Kurzform und eine Langform. Das Format kann für einen 20-Seiten-Plan zur Restrukturierung des Netzwerks dienen. Es kann ein fünf-Seiten-Dokument werden, das den Aufbau eines Prototypen beschreibt. Es kann für eine ein-Seiten-Notiz zum Einsatz kommen, die die Namen erläutert, welche jemand für einen neuen Verzeichnisbaum auf dem Server plant. Verwenden Sie es, um ein System zu dokumentieren, nachdem es als Referenz für andere in Ihrem Team aufgebaut wurde. Zum Teufel, verwenden Sie es, um das Team-Picknick zu beschreiben, das Sie planen.

Der Punkt ist, das das Team damit einen Mechanismus hat, um vor dem Handeln zu denken; einen Weg, um Pläne zu kommunizieren, der über

ein Gespräch auf dem Flur oder im Chat hinausgeht; und ein System, welches Artefakte hinterlässt, die andere verwenden können, um zu verstehen, wie etwas gemacht wurde. Eine Vorlage zu haben erleichtert es, anzufangen und vermeidet das Leere-Seite-Syndrom.

Abbildung 4: Wer Post-mortem-Autopsien als Pranger versteht, versteht sie falsch. (Foto: Michael Hoefner (http://www.zwo5.de) @ Wikimedia, CC-BY-SA

10. Gibt es einen Post-mortem-Prozess?

Schreibt jemand nach einem Ausfall auf, was passiert ist, so dass jemand anderes davon lernen kann, oder hofft man nur, dass es unbemerkt vorbei geht? Kann das Monitoringsystem die Situation erkennen, sodass der Admin eher davon weiß als der Benutzer? Oft haben Systeme eine Möglichkeit zum Testen neuer Konfigurationen, bevor diese eingesetzt werden (in Quellcode-Verwaltungen sind das zum Beispiel *pre-submit scripts*). Lassen sich Tests hinzufügen, die Tippfehler erkennen, welche zum Ausfall führen?

Eine gute Post-mortem-Autopsie (PM) beinhaltet eine Zeitleiste dessen, was passierte, wer betroffen war, was unternommen wurde, um es zu beheben, wie das Tagesgeschäft beeinträchtigt war und eine Liste von vorgeschlagenen Lösungen um das Problem in Zukunft zu verhindern. Jeder Ausfall bekommt mindestens eine vorbeugende Maßnahme. Solche PMs konsequent durchzuführen, schafft eine stabilere Umgebung.

Beim PM geht es nicht um Schuld und Schande. In einer guten Kultur der Systemadministration macht es nichts aus, wenn jemandes Name im *Was lief falsch*-Abschnitt auftaucht. Wenn ein Manager das PM verwendet um herauszufinden, wer zu bestrafen ist, versteht er nicht, das es beim operativen Geschäft nicht darum geht, Dinge perfekt zu tun, sondern jeden Tag ein bisschen besser. Jeder Manager, der eine Person wegen eines nicht böswillig herbeigeführten Ausfalls feuert, fährt seine Firma vor die Wand.

Das PM sollte für alle öffentlich sein. Sie mögen in Verlegenheit kommen und besorgt sein, dass Sie quasi die schmutzige Wäsche des Teams öffentlich waschen. Aber die Wahrheit ist: Wenn Sie das konsistent tun, respektieren Ihre Benutzer sie mehr. Transparenz gebiert Vertrauen. Natürlich – um wirklich Vertrauen zu entwickeln, müssen alle im Ergebnis eingereichten Fehler und Tickets auch wirklich bearbeitet werden.

C. Operative Praktiken

11. Gibt es für jeden Dienst eine operative Dokumentation (OpsDoc)?

Ihr DNS-Server fällt aus. Sie bauen ihn wieder auf, weil Sie wissen, wie es geht. Cool, stimmt's? Sie müssen die neueste Version von *BIND* kompilieren und installieren, und Sie tun es, weil Sie wissen, wie es geht. Stimmt's? Wenn das Monitoring diesen Fehler anzeigt, der dann und wann auftritt, wissen Sie, wie es repariert wird. Stimmt's? Sie wissen, wie man all das macht. Warum sollten Sie es aufschreiben (Abbildung 5)?

Darum: Werden Sie sich in 6 Monaten erinnern, wie man all das macht? Werden Sie sich auch dann erinnern, wenn Sie unter Druck stehen? Wie sieht es mit neuen Teammitgliedern aus – Sollen sie diese Dinge lernen, indem sie Sie beobachten und belästigen, oder können sie von selbst lernen? Und was ist, wenn Sie nicht in der Nähe sind: Können Sie einen entspannenden Urlaub nehmen, wenn Sie etwas belastet? Sie können sich nicht darüber klagen, dass Sie überarbeitet sind und Ihre Arbeit nicht auf andere verteilen können, wenn Sie keinen Weg bauen, die Arbeitslast zu teilen. Und schließlich: Wie kann Sie Ihr Manager befördern oder an ein neues und interessanteres Projekt setzen, wenn Sie die einzige Person mit einem bestimmten Wissen sind?

Also sollte jeder Dienst eine Mini-Website mit folgenden sieben Reitern bekommen. Der achte gilt nur, wenn inhäusig Software entwickelt wird:

- **1. Überblick:** Was ist es für ein Service; warum haben wir ihn; wer sind die Hauptkontaktpersonen; wie werden Fehler gemeldet; wo liegen Entwurfsdokumente und andere relevante Information?
- **2. Aufbau:** Wie wird die Software zusammengestellt, die den Service ausmacht; wo ist

die Download-Quelle; wo ist das Quellcode-Repository; was sind die Schritte für die Kompilierung und Paketierung oder andere Mechanismen zur Verteilung?

- **3. Einrichten:** Wie wird die Software oder ein Server eingerichtet? RAM-/Platten-Anforderungen; Version des Betriebssystems und Konfiguration; welche Software muss installiert sein. Wenn dies mit einem Konfigurationsmanagementwerkzeug automatisiert ist, dann geben Sie das hier an.
- **4. Alltägliche Aufgaben:** Schritt-für-Schritt-Instruktionen für alltägliche Dinge, wie Bereitstellung (hinzufügen/ändern/entfernen), allgemeine Probleme und ihre Lösungen.
- **5. Pager-Spielplan:** Eine Liste aller Alarmsignale, die Ihr Monitoringsystem für diesen Dienst generieren könnte und eine Schritt-für-Schritt Anleitung für „Was ist zu tun, wenn . . . „ für jede davon.
- **6. DR (Disaster Recovery):** Krisenplan und Ablauf. Wenn eine Maschine ausfällt, wie schalten Sie auf die Reserve um?
- **7. SLA:** Der Dienstleistungsvertrag, den Sie mit ihrem Kunden abschließen. Typischerweise Verfügbarkeitsziel sowie Richtwerte für Wiederherstellungspunkte und -zeiten.
- **8. Release-Management:** Falls das Team eigene Software entwickelt, dann steht hier, wie man eine Entwicklungsumgebung aufsetzt, wie man Integrationstests macht und wie eine Release freigegeben wird.

Wer ein Held ist, erzeugt eine solche Schablone, die der Rest des Teams nutzen kann. Dann dokumentiert er einen grundlegenden Dienst wie DNS, um loszulegen. Dann einen größeren Dienst. So entsteht ein Gerüst, in dem andere die fehlenden Stücke ausfüllen können.

Abbildung 5: *OpsDoc*: Was man nicht aufschreibt, erzeugt Mehrarbeit.

12. Wird jeder Dienst angemessen überwacht?

Es ist kein Dienst, wenn ihn niemand überwacht. Wenn es keine Überwachung gibt, dann läuft nur eine Software. Die Überwachung sollte auf dem SLA der operativen Dokumentation basieren. Wer kein SLA hat, für den ist ein einfacher oben/unten-Alarm das Minimum.

13. Gibt es einen Bereitschaftsplan?

Oder sind Sie der Trottel, der einfach immer in Bereitschaft ist? Ein Bereitschaftsplan dokumentiert, wer zu welchen Zeiten den Pager mitführt (oder für Alarme und Notfälle verantwortlich ist).

Man kann buchstäblich den sprichwörtlichen Pager weiterreichen, indem man ihn periodisch an die nächste Person weitergibt. Oder jeder hat einen eigenen Pager und das Monitoringsystem verwendet die Einteilung, um zu bestimmen, wen es anpiepsen soll. Es ist am besten, eine generische E-Mail-Adresse zu haben, die an die derzeitige Person geht, so dass die Kunden den Plan nicht kennen müssen.

Dieser Plan dient vielen Leuten: Die Rotation verbessert den Kundendienst, weil es die chaotische Panik eliminiert, einen Systemadministrator zu finden. Er dient dem Management, weil er das Vertrauen gibt, dass der nächste Notfall nicht genau dann passiert, wenn alle weg sind. Er dient der Personalabteilung, sofern eine Firma Ausgleichszeiten gibt oder entsprechend den Gesetzen zahlt. Wenn der Plan in maschinenlesbarer Form vorliegt, kann ihn ein einfaches Script lesen, um einen Bericht für die Lohnbuchhaltung zu generieren.

Er dient auch Ihnen, weil Sie so ein Leben außerhalb der Arbeit planen können. Wenn Sie glauben, Sie hätten keinen Plan, dann heißt der Plan: *24x7x365 und Sie sind ein Trottel*.

14. Haben Sie verschiedene Systeme für Entwicklung, QA und Produktion?

Entwickler machen ihre Arbeit auf den Entwicklungsservern. Wenn sie glauben, dass sie fertig sind, schnüren sie Packages und installieren sie auf dem QA-System. Wenn QA (Quality Assurance) und UAT (User Acceptance Test) zustimmen, werden die gleichen Packages verwendet, um die

Software auf den Produktivsystemen zu installieren.

Das ist Systemadministrator-Einmaleins, oder? Warum treffe ich dann ständig Systemadministratoren, deren Management sie das nicht tun lässt? Wenn in Management sagt, "Das kostet zu viel", dann ist bei denen Hopfen und Malz verloren. QA ist nicht teuer. Wissen Sie, was teuer ist? Ausfallzeit. Experimentelle Änderungen an Produktivservern sind auch nicht nur schlecht: Unter *SOX* sind sie illegal (*Sarbanes-Oxley Act*, in Europa entspricht dem *EuroSOX* [3]).

Das QA-System muss nicht so teuer sein, wie sein Gegenstück im Produktivbereich. Es kann weniger RAM haben, weniger Plattenplatz und weniger CPU-Leistung. Das können virtuelle Maschinen sein, die sich eine große Maschine teilen. Nur falls Skalierung und Antwortzeiten wichtig sind, ist ein QA-System nötig, das dem Produktivsystem ähnelt.

Abbildung 6: Kanarienvögel dienten in Kohlebergwerken als Frühwarnsignal. (Foto: 4028mdk09 @ Wikimedia Commons, CC-BY-SA)

15. Gibt es einen Kanarienvogel-Prozess bei der Auslieferung auf viele Maschinen?

Angenommen, eine Änderung auf 500 Maschinen steht an. Vielleicht ist es ein neuer Kernel, vielleicht ist es nur eine kleine Fehlerbeseitigung. Führen Sie die Änderung auf allen 500 Maschinen aus? Nein. Sie führen es auf einer kleinen Anzahl von Maschinen aus, um zu sehen, ob es Probleme gibt. Keine Probleme? Führen Sie es auf mehr Maschinen aus. Dann mehr und mehr, bis Sie fertig sind.

Diese ersten Maschinen nennt man Kanarienvögel (Abbildung 6). Kanarienvögel sind das klassische Beispiel eines Wächter-Tieres, das in Kohlebergwerken als Frühwarnsignal für giftige Gase diente. Einige Kanarienvogel-Techniken:

- *Eine, ein paar, viele.* Erledigen Sie eine Maschine (vielleicht Ihr Arbeitsplatzrechner), dann ein paar Maschinen (vielleicht die Ihrer direkten Mitarbeiter), schließlich viele Maschinen (größere und größere Gruppen, bis Sie fertig sind).
- *Cluster-Kanarienvögel.* Aktualisieren Sie eine Maschine, dann ein Prozent aller Maschinen, dann eine Maschine pro Sekunde bis alle Maschinen durch sind (typisch bei Google und in Betrieben mit großen Clustern).

D. Praxis der Automatisierung

16. Werden Konfigurationsmanagementwerkzeuge wie *CFEngine*, *Puppet* oder *chef* verwendet?

Software für Konfigurationsmanagement (Configuration Management, CM) koordiniert die Konfiguration von Maschinen. Sie könnte das Betriebssystem steuern, die Software, die angebotenen Dienste oder alles zusammen. Vor CM erledigten Systemadministratoren Änderungen an den Maschinen von Hand. Wenn sie 100 Maschinen hatten, hatten sie 100 Aufgaben von Hand zu erledigen. Gewiefte Systemadministratoren automatisieren solche Änderungen. Noch gewieftere Systemadministratoren begreifen, dass allgemeine Werkzeuge für solcherart Automatisierung noch besser wären. Sie erfanden Automatisierungssysteme mit Namen wie *CFEngine*, *bcfg2*, *Puppet* oder *chef*.

Das Kennzeichen eines CM-Systems ist, dass der Admin beschreibt, was er will, und die Software findet heraus, wie das zu tun ist. Was er will, ist in beschreibenden Anweisungen spezifiziert wie „hostA ist ein Webserver“ und „Webserver haben die folgenden Packages und Attribute“. Die Software verwandelt das in ausführbare Befehle. Ein anderer wichtiger Aspekt ist, das die Anweisungen generisch sind („Installiere die Befehle in `foo.sh` als Cron-Job„), das CM-System jedoch das richtige für das Betriebssystem des betreffenden Computers macht (es wählt aus zwischen `/etc/crontab` gegenüber `/var/spool/cron`).

17. Laufen automatisierte Aufgaben unter Funktionskonten?

Oft richten wir automatische Prozeduren ein, die zu vorbestimmten Zeiten laufen, zum Beispiel ein Skript, dass eine Datenbank einmal pro Nacht validiert. Mancherorts laufen diese Skripten mit den Rechten eines der Systemadministratoren. Wenn

dieser Systemadministrator die Firma verlässt, stirbt das Skript. Gute Betriebe lassen diese Skripten unter einem Funktionskonto laufen, oft *root*. Es ist jedoch sicherer, sie unter einem Konto mit weniger Privilegien laufen zu lassen.

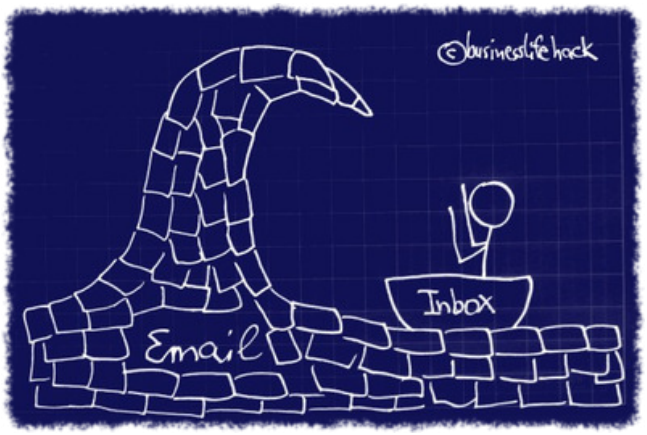


Abbildung 7: Heute schon in Mails ertrunken? So kann niemand arbeiten. (Grafik: Alexander Witt @ Flickr, CC-BY

18. Senden automatisierte Prozesse E-Mail nur dann, wenn sie etwas zu melden haben?

Meine Regeln sind einfach:

- Wenn ein menschlicher Eingriff genau jetzt nötig ist: Sende einen Pieps an den Pager oder eine SMS.
- Wenn innerhalb 24 Stunden reagiert werden muss: Erzeuge ein Ticket.
- Wenn es informatorisch ist: Protokolliere es in eine Datei.
- Gib nichts aus, wenn es keine Information gibt.

Während das Senden eines Pieps oder das Anlegen eines Tickets durch E-Mail erledigt werden könnte, ist der springende Punkt, dass E-Mail kein guter Mechanismus dafür ist. Beide könnten eine CC-Mail an Sie versenden, aber E-Mail ist nicht der primäre Mechanismus. Der schlimmste Fall ist ein System, dass Lognachrichten an jeden im Team der Systemadministratoren sendet. Ihr E-Mail-System ist kein gutes Archiv für die Protokolle.

Eine wahre Geschichte: Ein Freund in New York City arbeitete in einem Betrieb, wo alle automatisierten Prozesse ihre Ausgabe als E-Mail an *root@the-company-domain* sandten. *root* war die Mailingliste, die an alle Systemadministratoren in der Firma ging. Es war eine konstante Flut von Nachrichten. Die Systemadministratoren in dieser Firma konnten buchstäblich keine E-Mail lesen (Abbildung 7). Keine noch so gute Filterung war gut genug. Im Ergebnis nutzten die Systemadministratoren dieser Firma ihre privaten E-Mail-Konten für alle Kommunikation, sogar für Sachen, die die Arbeit betrafen. Welche Firma das war? Ein großer E-Mail-Provider, der nicht länger im Geschäft ist. (Ich frage mich, welche anderen schlechten Entscheidungen noch halfen, ihn aus dem Geschäft zu bringen.)

E. Flottenmanagement

19. Gibt es eine Datenbank aller Maschinen?

Jeder Betrieb sollte wissen, welche Maschinen er hat. Die Datenbank sollte wenigstens einige grundlegende Attribute speichern: Betriebssystem, Hauptspeicher, Größe der Platten, IP-Adresse, Eigentümer, wer über Wartungsarbeiten informiert werden sollte. Die Daten sollten automatisch eingesammelt werden. Ein sehr kleiner Betrieb kann auhc mit einer Kalkulationstabelle oder einer Wiki-Seite auskommen.

Eine solche Datenbank ermöglicht, über alle Maschinen zu automatisieren. In der Lage zu sein, einen Befehl präzise auf den Maschinen mit einer bestimmten Konfiguration abzuarbeiten, ist der Schlüssel für viele alltägliche Abläufe. Wer ein Inventar hat, kann außerdem Entscheidungen auf der Basis von Daten treffen, um Problemen vorzubeugen.

20. Ist die Installation des Betriebssystem automatisiert?

Automatische Installationen des Betriebssystems sind schneller und konsistenter. Wenn das Betriebssystem automatisch installiert wird, dann sind am Anfang alle Maschinen gleich. Entropie zu bekämpfen ist schwer genug. Wenn jede Maschine von Hand installiert wird, ist es unmöglich. Wer das Betriebssystem von Hand installiert, verschwenden seine Zeit doppelt: einmal, während er die Installation ausführt, und jedesmal wieder, wenn er einen Fehler sucht, den konsistent konfigurierte Maschinen verhindert hätten.

Wenn zwei Leute Betriebssysteme von Hand installieren, ist die Hälfte verkehrt, aber keiner weiß, welche Hälfte. Beide mögen behaupten, dass sie dieselbe Prozedur verwenden, aber ich versichere Ihnen, dass dem nicht so ist. Stecken Sie jeden in einen anderen Raum und lassen Sie sie ihre Prozedur niederschreiben. Dann zeigen Sie jedem Systemadministrator die Liste des anderen. Das wird eine Prügelei geben.

Benutzer sehen Inkonsistenzen außerdem als Inkompetenz. Wenn neue Maschinen mit einer Einstellung eintreffen, die ihnen nicht passt, wissen sie, wie sie die Einstellung ändern müssen und sind glücklich. Wenn in der Hälfte der Fälle diese Einstellung so eingestellt ist und in der andern Hälfte anders, verlieren sie das Vertrauen in den Systemadministrator. Sie fragen sich: „Welche Deppen installieren diesen Kram?“

Nicht in der Lage zu sein, eine Maschine einfach zu reinigen und wieder zu laden, ist schließlich eine Sicherheitsfrage. Eine Maschine sollte gelöscht und neu installiert werden, wenn ein gebrauchter Computer von einem Benutzer an einen anderen weiter gegeben wird. Wenn dieser Prozess nicht reibungslos funktioniert, gibt es die Versuchung Zeit zu sparen, indem man es unterlässt.

21. Können Sie Software automatisch über alle Systeme aktualisieren?

Wenn die Installation des Betriebssystems automatisiert ist, dann starten alle Maschinen gleich. Wenn Aktualisierungen automatisiert sind, dann bleiben alle Maschinen aktuell. Konsistenz ist eine gute Sache.

Aber Sicherheitsaktualisierungen sind außerdem wichtig, weil die Betriebssicherheit der Systeme sie erfordert. Auch nicht-sicherheitsbezogene Aktualisierungen sind wichtig, weil die Funktionsfähigkeit des Systems sie erfordert und weil sie neue Features für Kunden bringen. Aktualisierungen zurückhalten ist wie Eltern, die Liebe verweigern. Wer hat Sie aufgezogen?

22. Gibt es eine Richtlinie für die Erneuerung der Hardware?

Wenn Sie keine Richtlinie haben, wann Laptops und Arbeitsplatzrechner zu ersetzen sind, werden diese niemals ersetzt. Im Serverraum macht man sich üblicherweise mehr Gedanken darüber, wann ein Gerät zu ersetzen ist. PC-Umgebungen hingegen benötigen eine Art reproduzierbaren, zyklischen Prozess, damit sie frisch bleiben. Ohne das werden die Dinge alt und ununterstützbar, oder die Leute erhalten neue Geräte als Statussymbol, und dann wird es politisch.

Ein bestimmter Anteil einer Flotte sollte alt sein, das ist einfach ökonomisch. Jedoch sind extrem alte Maschinen teurer zu unterhalten, als zu ersetzen. Es ist Zeitverschwendung für den Admin, einen Workaround zu erarbeiten, damit neue Software auf untermotorisierten Maschinen läuft. Es ist auch Zeitverschwendung für den Benutzer, auf langsame Computer warten zu müssen (Abbildung 8).

Wer keine Richtlinie hat, hier ist eine einfache für den Beginn: Alle Computer haben einen drei-Jahres-Abschreibungsplan. Jedes Jahr enthält das Budget Posten, um ein Drittel aller Maschinen zu ersetzen. Am ersten Tag jeden Quartals werden genügend Maschinen bestellt, um neun Prozent der ältesten Maschinen in der Flotte zu ersetzen.

CFOs mögen das, weil sie Vorhersagbarkeit mögen. In einer Firma war der CFO ganz aufgeregt, als ich ihm die Kontrolle darüber gab, in welchem Monat die Aktualisierungen passieren. Wir einigten uns, dass ein Viertel aller Aktualisierungen in jedem Quartal geschehen, und er konnte den Monat aussuchen, in dem es losgeht. Er konnte es sogar in individuelle monatliche Lose aufteilen.

F. Vorbereitung auf ein Desaster

Abbildung 8: Nicht zu alte Personalcomputer nutzen Anwendern und Admins gleichermaßen.

23. Arbeiten die Server weiter, auch wenn eine Platte kaputt geht?

Es war üblich, dass ein Komponentenfehler einen Ausfall bedeutete: Eine Platte stirbt und Sie verbrachten den Tag damit, sie zu ersetzen und die Daten vom Bandlaufwerk wieder herzustellen. Eine kaputte Platte ruinierte ihren ganzen Tag; gerade wie ein Atomkrieg.

Heute sind die Dinge anders. Wie bauen überlebensfähige Systeme. Wenn eine Platte Teil eines

gespiegelten Paares ist, dann kann eine dieser Platten ausfallen und es ist kein Gesamtausfall. Es gibt nur einen Ausfall, wenn der gespiegelte Partner auch noch ausfällt. Statistisch gibt das Stunden, möglicherweise Tage, um die kaputte Platte zu ersetzen, bevor ein Ausfall für die Benutzer sichtbar wird. Damit entkoppeln Sie Komponentenfehler von Ausfällen. Das Leben wird besser.

RAID war teuer und selten. Ein Luxus für die Reichen. Jetzt ist es alltäglich, billig und oft frei (wenn es in Software realisiert wird). Sagte ich „alltäglich"? Ich meinte: obligatorisch. Den Tag mit der Wiederherstellung von Daten zu verbringen, ist nicht nur ein Zeichen schlechter Planung, es ist schlechtes Zeitmanagement. Den Tag damit zu verbringen, einen verzweifelten Benutzer zu trösten, der Jahre, Monate oder auch nur Stunden von Arbeit verloren hat, ist nicht heldenhaft – es ist schlechte Systemadministration.

Plattenfehler sind nicht selten. Warum haben Sie ein System gebaut, dass annahm, sie sind es?

24. Ist der Kern des Netzwerks N+1?

Der Ausfall des Netzwerkes ist für eine Person eine Schande. Ein Ausfall für viele Leute ist inakzeptabel. So wie redundante Platten nun ein Minimum sind, ist es die doppelte Anbindung des Netzwerks. Es war mal ein teurer Luxus für die Reichen. Nun ist es eine Minimalanforderung.

Ja, es wird noch erwartet, dass da eine Verbindung von der Arbeitsstation zum ersten Switch ist. Aber danach sollte alles N+1 redundant sein. Als Minimum sollten alle Stammleitungen zwei Netze haben. Am besten ist es, wenn jede beliebige Uplink-Verbindung, jede beliebige Karte oder jeder beliebige Netzwerk-Router und -Switch ausfallen könnte – und die Pakete trotzdem durchkommen.

LANs werden im Allgemeinen so entworfen: Die Laptops und Arbeitsstationen in einem Büro sind an der Wandsteckdose angeschlossen. Diese verbinden zu *Zugangs-Switches*, die viele Ports haben. Diese Zugangs-Switche haben *Nervenstämme*, die zu einer Hierarchie von *Kern-Switches* verbinden, welche die Pakete zur richtigen Stelle sausen lassen, möglicherweise zu Ausgängen (Verbindungen zu anderen Gebäuden oder dem Internet).

Wenn ein Betrieb nicht so konfiguriert ist, dann lebt er in den Tagen, als Computer noch ohne Netzwerk nützlich waren. Ausnahme: Betriebe, die klein genug sind, dass sie keinen Kern haben. Selbst dann sollten alle Stämme redundant sein und ein Austauschsatz sollte für jede Art von Gerät bereitstehen.

Abbildung 9: O nein, das Haus ist abgebrannt. Backups an einem anderen Ort sind die einzige Hoffnung.

25. Sind Backups automatisiert?

Diese Frage setzt voraus, das Sie Backups machen. Sie machen doch Backups, ja? Sie brauchen sie aus vier Gründen:

- (1) Ups, ich habe eine Datei gelöscht.
- (2) Ups, die Hardware ist ausgefallen.
- (3) O nein, das Haus ist abgebrannt.
- (4) Archive.

Situation (1) wird kurzfristig, aber nicht langfristig durch Snapshots gelöst. Manchmal wird eine Datei gelöscht und muss erst viel später wieder hergestellt werden. Einfache Speicherauszüge werden da nicht helfen. Auch RAID hilft in dieser Situation nicht, denn es ist kein Backup-Mechanismus. Wenn jemand eine Datei aus Versehen löscht, wird RAID pflichtbewusst dieses Versehen auf allen Spiegeln replizieren. Sie werden ein redundantes Array von fehlerhaften Daten haben.

Situation (2) klingt, als ob RAID helfen würde. Aber erinnern Sie sich, dass Fehler an zwei Platten bedeuten kann, dass Sie den ganzen RAID1-Spiegel oder das komplette RAID5-Set verloren haben? RAID10 und RAID6 liefern alle Daten bei Fehlern an drei Platten. Diese Dinge passieren. Sie sind nur einen ungeschickten Elektriker weit entfernt von einer Zerstörung aller Platten auf einmal. Wirklich.

Situation (3) wird oft *Disaster Recovery* genannt. Backups an einem anderen Ort, ob auf Platte oder Band, sind hier die einzige Hoffnung.

Situation (4) gibt es oft auf Grund geltender Vorschriften. Die Technologie für das Backup ist oft die gleiche wie in Situation (3), aber die Rückhaltezeit ist üblicherweise unterschiedlich. Wenn eine andere Abteilung auf Grund geltender Vorschriften ein Archiv fordert, sollte diese für die Medien bezahlen.

Für jeglichen dieser Gründe muss der Prozess automatisiert werden. Während das Haus abbrennt, wollen sie nicht das Management informieren, dass die Daten verloren sind, weil „ich in Urlaub war" oder „ich es vergessen habe". *(Anm. d. Red.: Eine zur Implementierung von Backup hilfreiche Übersicht über Strategien und Tools enthalten [4] und [5].)*

26. Werden Krisenpläne regelmäßig getestet?

Die letzte Sektion war ein wenig gelogen. Es gibt nicht vier Gründe, um Backups zu machen. Es gibt stattdessen vier Gründe, um Dateien wiederherzustellen. Niemand interessiert sich für Backups. Die Leute interessieren sich nur für die Wiederherstellung. Wenn Sie herausfinden, wie man Daten ohne vorheriges Backup wiederherstellt, werde ich mich beim Nobelpreis-Komitee für einen Preis für Systemadministratoren einsetzen, nur damit Sie der Erste sein können, der ihn bekommt.

Sie wissen nicht, ob Backups gültig sind, bis Sie sie testen. Glaubensbasierte Backupsysteme sind nicht gut. Hoffnung stärkt uns, aber sie ist keine IT-Strategie.

Ein voller Test beinhaltet einen Totalausfall und eine volle Wiederherstellung. Sie wissen nicht, wie viel Zeit Sie dafür wirklich brauchen, bis Sie es ausprobieren. Wiederherstellungen vom Band dauern oft zehn mal länger als das Backup. Wenn Sie ein volles Backup für Ihren Lohnbuchhaltungsserver in acht Stunden machen können, dann bereiten Sie sich darauf vor, dass es 80 Stunden keine Lohnzahlungen geben wird, falls Sie das System von Grund auf wiederherstellen müssen.

Wenn man überhaupt keine Tests macht, dann ist etwas Testen besser als gar nichts. Schreiben Sie ein kleines Skript, das zufällig einen Server auswählt, dann zufällig eine Platte auf diesem Server, dann zufällig eine Datei auf dieser Platte. Dieses Skript erzeugt ein Ticket, das darum bittet, diese Datei wiederherzustellen, so wie sie vor sechs Wochen aussah. Lassen Sie das Skript jede Woche laufen. Damit haben sie eine gute Chance, einen Server oder eine Platte zu finden, die noch nicht im Backup-Plan war. Außerdem: Wenn Sie glauben, dass diese Wiederherstellungen eine Menge Arbeit für Sie ist, hier ist ein Geheimnis. Es braucht überhaupt keine Arbeit von Ihnen, wenn ihre Mitarbeiter das Ticket abarbeiten. Erzeugen Sie das Ticket mit genug zufälligem Text, sodass sie nicht wissen, dass es eine Übung ist.

Um das einen Schritt weiter zu führen, planen Sie einen Spieltag, bei dem die Krisenpläne wirklich getestet werden (Abbildung 10). Nehmen Sie an, das bestimmte Leute tot sind und stellen Sie sicher, dass die verbleibenden Leute wissen, wie sie die Dienste umstellen. Entweder rufen Sie wirklich Ausfälle hervor (ziehen sie das Strom- oder Netzwerkkabel), oder spielen Sie die Szene – die 'tote' Person kann den Test zum Beispiel beaufsichtigen. Eine andere Methode ist es, Ihrem CEO zu erlauben, in das Datenzentrum zu spazieren und irgendein Kabel nach Belieben zu ziehen.

Abbildung 10: Einen großen roten Aus-Knopf drücken darf man bei einem Spieltag, bei dem die Krisenpläne getestet werden. (Foto: Stahlkocher @ Wikimedia, CC-BY-SA)

27. Haben die Maschinen im Datenzentrum Fernzugang für Strom und Konsole?

Das benötigt kaum Erklärung. Fernkonsolen (IP-basierte KVM-Umschalter) sind billig, gute Server haben sie eingebaut. Fernsteuerung für die Stromzufuhr ist kein Luxus, wenn der Computer mehr als ein paar Kilometer entfernt ist. Die Ausnahme zu dieser Regel sind Grid-Computersysteme mit Hunderten oder Tausenden von identischen Maschinen. Wenn eine ausfällt, nimmt eine andere ihren Platz ein.

G. Sicherheitspraxis

28. Läuft auf den Desktops, Laptops und Servern sich selbst aktualisierende, stille Antischadsoftware?

Viren und Schadprogramme sind eine Tatsache. Jede Maschine braucht heute Schutz-Software. Jeder Angriff von Schad-Software bedeutet Arbeit: ausgefallene Maschinen aufräumen, Daten wiederherstellen, Benutzer wegen ihres Datenverlustes trösten. Es ist Zeitverschwendung und eine unentschuldbare Störung für Benutzer. Es ist schlechtes Zeitmanagement.

Antischad-Software sollte sich stillschweigend aktualisieren. Es ist Aufgabe des Admins, die Sicherheitsrichtlinien durchzusetzen, und Aktualisierungen herunterladen ist Teil dieser Richtlinie. Diese Verantwortung auf den Benutzer zu delegieren, ist falsch und möglicherweise unethisch. Sie fragen keine Fußgänger: „Soll ich auf die Bremse treten, um Sie nicht zu überfahren?“, und Sie fragen auch keinen Benutzer, ob er weniger sicher sein möchte. Aus ähnlichen Gründen sollten Benutzer die Antischad-Software nicht einfach abschalten können. Benutzer werden sie aus den absurdesten Gründen abschalten.

29. Gibt es eine schriftliche Sicherheitsrichtlinie?

Es ist entscheidend, einen schriftliche Sicherheitsrichtlinie zu haben, bevor man sie umsetzt. Existierende Sicherheitsrichtlinien anzuschauen hilft, Ideen zu bekommen [6].

30. Gibt es regelmäßige Sicherheitsüberprüfungen?

Das braucht kaum Erläuterung. Wer seine Sicherheit nicht testet, weiß nicht, wie angreifbar er ist.

31. Kann ein Benutzerkonto auf allen Systemen innerhalb einer Stunde deaktiviert werden?

Eine einzige Authentifizierungsdatenbank zu haben, die alle Systeme nutzen, ist nicht nur ein *wäre nett zu haben*. Es ist ein *muss ich haben*. Wenn Sie glauben, dass Sie es nicht brauchen bis Sie größer sind, werden Sie herausfinden, dass Sie keine Zeit haben es zu installieren, während Sie damit beschäftigt sind zu wachsen.

Die beste Praxis ist, Managementsysteme für den Lebenszyklus von Benutzerkonten zu verwenden. Ein solches System erzeugt, verwaltet und steuert Nutzerkonten von vor der Anstellung über Änderungen der Lebensumstände bis zur Kündigung und darüber hinaus: Was ist, wenn sich der Name eines Benutzers ändert? Was, wenn jemand wieder in der Firma angestellt wird? Was, wenn jemand wieder eingestellt wird, der Name sich aber geändert hat? Es gibt eine Menge Grenzfälle, die ein Admin handhaben können sollte.

32. Können alle privilegierten Kennworte (*root*) innerhalb einer Stunde geändert werden?

Die Antwort auf diese Frage zeigt, ob ein administrativer Zugang wohlverwaltet ist. Wenn Sie sie mit Nein beantworten, erzeugen Sie auf einer Wiki-Seite eine Checkliste aller Stellen, die geändert werden müssen. Ändern Sie das Kennwort global, indem Sie dieser Liste folgen, wobei Sie sie ergänzen, wenn Sie sich an andere Geräte erinnern. Für obskure Systeme dokumentieren Sie den exakten Befehl oder Prozess, um das Kennwort zu ändern.

Wenn Sie sie mit Ja beantworten, erzeugen Sie eine Wiki-Seite, die dokumentiert, wie dieser Prozess zu aktivieren ist. Und dann listen Sie alle Ausnahmen auf, die immer noch von Hand zu erledigen sind.

Links

[1] Anregung für die DevOpsCard: *The Joel Test – 12 Steps to Better Code*:
http://www.joelonsoftware.com/articles/fog0000000043.html
[2] Vorlage für die Übersetzung: *The Operations Report Card*: http://opsreportcard.com
[3] Zur Information über den Einfluss von EuroSOX auf die IT:
https://www.datenschutz-praxis.de/fachartikel/eurosox-und-datenschutz/
[4] Stefan Schumacher:Wenn der GAU kommt; Datensicherungsstrategie erarbeiten und umsetzen. UpTimes 2012-3, S. 21:
http://www.guug.de/uptimes/2012-3/index.html
[5] Stefan Schumacher: Wenn der Mini-GAU kommt; Kleine Programme zur Sicherung einzelner Rechner. UpTimes 2013-2, S. 17:
http://www.guug.de/uptimes/2013-2/index.html
[6] Bibliothek von Beispielen für Sicherheitsrichtlinien:
http://www.sans.org/security-resources/policies/

Mixpult per Tastatur
Der CLI-Sound-Exchanger SoX

Weitgehend unbekannt, doch reich an Features liest, schreibt und konvertiert SoX alle denkbaren Audioformate. Dazu noch lässt SoX den Dateien Effekte angedeihen und arbeitet prima Hand in Hand mit Skripten: Hier ein Appetitmacher für Sound-Experimente auf der Kommandozeile.

von **Jürgen Plate**

I understand the inventor of bagpipes
was inspired when he saw a man
carrying an asthmatic pig.
Unfortunately, the man-made sound never equalled
the purity of the sound achieved by the pig.

Alfred Hitchcock

In der Basisversion handelt es sich bei dem „SOund-eXchanger“ (*SoX*) um ein reines Kommandozeilen-Tool, das vom Benutzer den exzessiven Einsatz von Kommandozeilenparametern fordert [1]. Deshalb wird dieser Artikel nur ganz leicht an der Oberfläche kratzen. Er soll Appetit auf eigene Experimente machen. SoX ist bei vielen Distributionen in den Paketlisten enthalten, wird aber nicht automatisch installiert. Bei Debian genügt `apt-get install sox`. Das Programm aus dem Quellcode zu erstellen, ist beispielsweise dann nötig, wenn der Paketmaintainer das Binary ohne MP3-Unterstützung kompiliert hat.

Auf der Projekt-Homepage befinden sich neben vielen Infos die Programmversionen für Linux, Windows und Mac OS zum Herunterladen. Das Programm läuft problemlos unter Linux und anderen Unix-Systemen. Unter Umständen fehlen einige Sound-Formate. Die meisten von ihnen lassen sich durch die Installation von `libsox-fmt-all` nachträglich integrieren [2].

Da SoX auf der Kommandozeile arbeitet, ist es extrem flexibel in Skripten einsetzbar, aber manchmal auch unkomfortabel. Schon die allgemeine Syntax von SoX ist komplex:

```
sox <Allgemeine Optionen>\
 <Format-Optionen> <Eingabedatei(-Liste)>\
 <Format-Optionen> <Ausgabe-Datei>\
 [<Effekt>] [<Effekt-Optionen>]
```

Die Formatoptionen für die Ein- und Ausgabedatei stehen also jeweils vor dem zugehörigen Dateinamen. Was auf den ersten Blick übersophisticated erscheint, reduziert sich für eine simple Audiokonvertierung auf:

```
sox <Eingabedatei> <Ausgabedatei>
```

Dabei erkennt SoX auch gleich noch das Format der Sound-Dateien an der Endung. Schlimmstenfalls muss man den Typ mit dem Schalter `-t` einstellen. So wandelt beispielsweise der Aufruf `sox Musik.mp3 Musik.ogg` die Datei *Musik.mp3* von einer MP3- in eine OGG-Datei um. Soll die Eingabedatei rückwärts in die Ausgabe wandern, erledigt das ei hinten angehängtes `reverse`.

Unter Linux erleichtern beigelegte Shellskripte wie `play` und `record` den Einstieg. Zudem greifen viele grafische Programme auf SoX als Backend zurück, da die Programmierer nicht einsehen, warum sie das Rad noch einmal erfinden sollten. Außerdem ist SoX das Schweizer Taschenmesser unter den Sound-Bearbeitungsprogrammen: Es beherrscht auch außergewöhnliche Datei-Formate, mit denen nur wenige Programme zurecht kommen, darunter *Apple/SGI AIFF*, *Mac HCOM* oder *SoundBlaster VOC*. Eine Übersicht der Formate zeigt der Aufruf `sox -h` . Mit dem SoX-eigenen Format *.dat* für die Zieldatei werden die Audiodaten sogar als Gleitpunktzahlen in eine Textdatei geschrieben.

Beispiele zum Aufwärmen

Option `-v` ändert die Lautstärke. Der Wert nach `-v` wirkt als Multiplikationsfaktor. Dabei handelt es sich um eine Eingabeoption, weil sich der Faktor auf die Eingabedatei bezieht, um daraus die Ausgabe zu erzeugen. Um also eine doppelt so laute Kopie zu erzeugen, dient das Kommando:

```
sox -v 2.0 input.wav output.wav
```

Zusammen mit der Statistikfunktion trimmt das die Ausgabedatei auf maximale Lautstärke ohne Verzerrungen. Das Flag `-n` unterdrückt unerwünschte Ausgaben:

```
# Lautstaerkemaximum ermitteln
VOL=$(sox input.wav -n stat -v 2>&1)
# Datei konvertieren
sox -v $VOL input.wav output.wav
```

Folgendes trennt eine Eingabedatei mit zwei Kanälen so auf, dass jeder Kanal in einer separaten Datei gespeichert ist. `-c 1` bedeutet, nur ein Kanal. Die in SoX enthaltene `mixer`-Anweisung erhält so die Information, den zweiten Kanal zu unterdrücken und den ersten Kanal beizubehalten (1,0), anschließend umgekehrt (0,1):

```
sox stereo.wav -c 1 links.wav mixer 1,0
sox stereo.wav -c 1 rechts.wav mixer 0,1
```

Sollen aus einer Mono-Aufnahme zwei Stereo-Kanäle werden, geht das ebenso leicht:

```
sox mono.wav -c 2 stereo.wav
```

Fast so einfach sind Ausschnitte von Dateien anzufertigen. Um eine zehn Sekunden lange Musiksequenz mit Einblenden (Fade-in) und Ausblenden (Fade-out) zu erstellen, geht man folgendermaßen vor:

```
sox musik.wav out.wav fade 0:0.5 0:10 0:0.5
```

Das Format-Schema lautet *hh:mm:ss.frac*und zeigt hier zum Beispiel `0:0.5`für 0,5 Sekunden. Beide Fades dauern so jeweils 0,5 Sekunden. Der Aufruf speichert also die ersten zehn Sekunden einer Datei in `out.wav`, blendet sie am Anfang eine halbe Sekunde ein und am Ende eine halbe Sekunde aus. Will man nicht einblenden, aber ausblenden, setzt man 0 für den ersten Wert ein.

Liegt ein Musikstück beispielsweise in zwölf Minuten Länge vor, von dem aber nur die bei 4.15 beginnende Passage relevant ist, lässt sich die Eingabedatei zunächst mit der *trim*-Anweisung zerstückeln. Folgender Befehl schneidet aus der Datei *musik.wav* ab 4:15 die darauffolgenden 30 Sekunden heraus – und versieht die neue Datei dann mit dem gerade gezeigten Fade-out. Mit den Schneid-Fading-Effekten lassen sich beispielsweise aus längeren Musikstücken passende Häppchen als Mobil-Klingelton erstellen:

```
sox musik.wav klein.wav trim 4:15 30
sox klein.wav out.wav fade 0:0.5 0:10 0:0.5
```

Integration mit anderen Tools

SoX eignet sich vorzüglich zur Kombination mit anderen Programmen, da die Sound-Daten über die Standardein- und -ausgabe laufen können. Ein typisches Beispiel wäre der Einsatz des MP3-Players *mpg123* zusammen mit SoX, um MP3-Dateien ins *.wav*-Format zu kopieren:

```
mpg123 -b 10000 -s datei.mp3 |                  \
 sox -t raw -r 44100 -s -w -c 2 - datei.wav
```

Besonders beliebt sind Echo-Effekte, die einer trocken klingenden Aufzeichnung etwas Raum vermitteln. Auch zarte Stimmchen von an sich unbegabten Interpreten lassen sich so aufhübschen. Dazu einige Beispiele mit dem SoX-Player gleich zum Anhören. Die Kommandos *play* und *echo* sind im Paket enthalten. Die vier Zahlenwerte des *echo*-Befehls stehen für den Eingangspegel, den Pegel des Echos, die Verzögerung in Millisekunden und die Abklingzeit in Millisekunden:

```
# Scheinbare Verdoppelung der Instrumente,
# vollerer Sound
play file.xxx echo 0.8 0.88 60.0 0.4

# Sehr kurzes Echo ergibt einen metallischen,
# maschinellen Klang
play file.xxx echo 0.8 0.88 6.0 0.4

# Ein laengeres Echo macht ein Kammerorchester
# zum Open-Air-Erlebnis
play file.xxx echo 0.8 0.9 1000.0 0.3
```

Der Compander-Effekt ermöglicht Kompression oder Expansion des Dynamikbereichs eines Signals. Für die meisten Situationen sollte die Angriffszeit – wenn also die Lautstärke schnell ansteigt – kürzer sein als die Abklingzeit, weil unsere Ohren empfindlicher auf plötzliche Lautstärke reagieren als auf plötzlich leiser werdende Musik (typische Beispiele: Gongs oder Trommelschläge). Bei Situationen, wo die lauten Stellen eines Musikstücks zu laut, die leisen Stellen aber – etwa wegen Umgebungsgeräuschen – zu leise sind, lässt sich der Dynamikumfang eingrenzen. Oder man macht die Datei insgesamt leiser oder lauter. Das folgende Kommando normalisiert die Eingabedatei um 10 Dezibel:

```
sox --norm=-10 input.wav output.wav
```

SoX kann auch eine Aufnahme resamplen, zum Beispiel für eine Wideband-Samplingrate von 16 Kilohertz:

```
sox datei.wav -r 16000 datei-wide.wav resample
```

Die Effekte *stretch* und *speed* beschleunigt oder bremst das Abspielen, wenn etwa der Minutenwalzer auch mal zwei Minuten dauern darf. Allein die Manpage und die Dokumentation des kleinen Tools bieten genügend Lesestoff mit anschließenden Akustikexperimenten für lange Winterabende.

Zusammen mit SoX kommt bei Installation auch das Tool *soxmix* auf die Platte. Dieses Programm führt mehrere Dateien zu einer Datei zusammen, wobei es die Audiodaten übereinander legt. So lässt sich zum Beispiel der Gesang über die

Hintergrundmusik mischen oder besondere Geräuscheffekte erzielen, denn alle Effekte von SoX stehen auch hier zur Verfügung:

```
soxmix eingabedatei1 eingabedatei2 ... ausgabedatei
```

Bei neueren Versionen ist diese Funktion mit dem Schalter *-m* sogar in SoX integriert (`sox -m ...`).

Verarbeitung mit Bash und Perl

Koppelt man SoX mit einer Skriptsprache wie Perl, lassen sich aufwändige Dinge anstellen. Zudem macht das Skript drumrum die Bedienung leichter, fehlerfreier und erspart jedesmal das Nachschlagen von zahllosen Optionen. Das erste Beispiel demonstriert die Vorgehensweise zunächst mit der ganz normalen Shell.

Alle erdenklichen Informationen über eine Soundadatei liefert der Aufruf:

```
sox sounddatei -n stat stats
```

Man kann diese Infos verwenden, um Sounds automatisch an bestimmte Gegebenheiten anzupassen. So ließen sich beispielsweise durch Auswerten der maximalen Amplitude mehrere Dateien per Skript auf etwa gleiche Lautstärke bringen. Neben der Konvertierung, dem Abspielen und dem Manipulieren von Sounddateien beherrscht SoX auch die Synthetisierung akustischer Signale.

Weil es so schön ist, erzeugt das Beispiel aus einem nervigen Fernsehwerbung-Jingle eine *.wav*-Datei. Das Skript erzeugt mittels `synth` dafür zuerst fünf Sinustöne im Raw-Format (Option `-t raw` in Zeilen 03 bis 07). Das Raw-Format ist praktisch, weil sich die Töne einfach in der Datei aneinander hängen lassen. Das *synth*-Kommando hat drei Parameter: die Länge des Sounds im Format *hh:mm:ss.frac*, dann den Typ des Sounds und als drittes die Frequenzangabe in Hertz. Als Frequenzangabe dient auch ein Bereich aus Anfangs- und Endfrequenz – zum Beispiel `300-8000`. In diesem Fall würde SoX einen Sweep erzeugen.

Im Beispiel erzeugt die Angabe `synth` *`Zahl-x`* `sine` *`Zahl-y`* ein Sinussignal von x Sekunden Dauer mit der Frequenz y. Der Effekt *gain* legt die Lautstärke fest. Zeile 09 konvertiert den Raw-Sound in eine Wav-Datei, Zeile 10 entsorgt die nicht mehr benötigte Raw-Datei. Die letzte Zeile 11 spielt das Ergebnis gleich mal ab – Sox kann das Erzeugnis nämlich auch gleich ausgeben, indem die Option `-d` die Standard-Soundausgabe in Anspruch nimmt:

```
01 my $SOX = '/usr/local/bin/sox';
02
03 $SOX -U -r 8000 -n -t raw - synth 0.2 \
       sine 1244 gain -3 > t.ul
04 $SOX -U -r 8000 -n -t raw - synth 0.2 \
       sine 1244 gain -3 >> t.ul
05 $SOX -U -r 8000 -n -t raw - synth 0.2 \
      sine 1244 gain -3 >> t.ul
06 $SOX -U -r 8000 -n -t raw - synth 0.2 \
      sine 1760 gain -3 >> t.ul
07 $SOX -U -r 8000 -n -t raw - synth 0.2 \
      sine 1244 gain -3 >> t.ul
08
09 $SOX -c 1 -r 8000 t.ul jingle.wav
10 rm t.ul
11 $SOX jingle.wav -d
```

Das Perl-Programm im Kasten *Listing: Weißes Rauschen* zeigt, wie ein wenig Rechnerei über das Erzeugen einzelner Sinustöne hinausführt. Es produziert für Messzwecke eine Wav-Datei mit verschiedenenen Rauschsignalen, also weißem Rauschen. Dafür entstehen zwischenzeitlich 32 Dateien mit Rauschsignalen, bei denen Dauer, Mittenfrequenz und Bandbreite innerhalb vorgegebenen Grenzen zufällig sind. Die Anzahl der Dateien und die Ober- und Untergrenzen von Dauer, Mittenfrequenz und Bandbreite lassen sich einstellen. Die Anzahl der Sounddateien ist im Beispiel allerdings auf 32 beschränkt, da bei mehr die Samples kaum noch unterscheidbar werden.

Zeilen 06 und 07 setzen ein Temporärverzeichnis, das neu angelegt wird, wenn es nicht besteht. Die Zeilen 09 bis 15 legen die Grenzen fest, aus denen per Zufallsauswahl die Segment-Dateien mit Rauschsignalen entstehen. Die Anzahl der Dateien ist aus praktischen Gründen auf 32 begrenzt (Zeile 09), ihre Länge soll zwischen 0,25 und 1 Sekunde liegen (Zeilen 10 und 11), die Mittenfrequenz soll zwischen 500 und 8000 Hertz liegen (Zeilen 12 und 13) und die Bandbreite zwischen 100 und 1000 (Zeilen 14 und 15). Die Zeilen 17 und 47 bis 53 bauen eine Help-Funktion ein. Die gemahnt zum Beispiel gegebenenfalls daran, dass laut Zeile 19 ein Dateiname von der Kommandozeile für die Ergebnis-Datei am Ende gefordert ist.

Zeile 26 speichert die Dateinamen der für die Ergebnis-Datei erzeugten Soundsegmente. Die For-Schleife in Zeile 28 erzeugt die Segmente: Sie läuft so lange durch, wie in Zeile 09 festgelegt. In ihr erzeugen die Zeilen 30 bis 34 per Zufallsprinzip Parameter aus den festgelegten Grenzen für Dauer und Frequenz. Zeilen 35 und 36 kümmern sich um die Dateinamen. Zeilen 38 bis 42 erzeugen endlich die Segmentsound-Dateien. Und ganz am Ende mixt Zeile 45 alle Segmente für die selbst benannte Ergebnis-Sounddatei zusammen, indem der `system()`-Aufruf unser SoX einbindet.

Listing: Weißes Rauschen

```
01 use strict;
02 use warnings;
03
04 my $SOX = '/usr/local/bin/sox';
05
06 my $tmp_dir = 'noise';
07 mkdir($tmp_dir) if(! -e $tmp_dir);
08
09 my $segs = shift || 32;
10 my $min_dur = shift || 0.25;
11 my $max_dur = shift || 1.0;
12 my $min_center = shift || 500;
13 my $max_center = shift || 8000;
14 my $min_bw = shift || 100;
15 my $max_bw = shift || 1000;
16
17 usage() if($ARGV[0] =~ /-[h?]/);
18
19 my $outfile = shift;
20 unless (defined $outfile)
21   {
22   print "Keine Ausgabedatei angegeben\n";
23   usage();
24   }
25
26 my $filelist = '';
27
28 for my $i (1..$segs)
29   {
30   my $dur = rand($max_dur) + $min_dur;
31   my $center = int(rand($max_center
32               - $min_center)) + $min_center;
33   my $bw = int(rand($max_bw - $min_bw))
34          + $min_bw;
35   my $segfile = "$tmp_dir/temp$i.wav";
36   $filelist .= "$segfile ";
37
38   my $params = " -n " . $segfile
39                . " synth " . $dur
40                . " noise bandpass "
41                . $center . " " .$bw;
42   system("$SOX $params");
43   }
44
45 system("$SOX -m $filelist $outfile");
46
47 sub usage
48   {
49    print "Usage: $0 <outfile> <segs (<=32)> " .
50          "<min dur> <max dur> <min cent> " .
51          "<max cent> <min bw> <max bw>\n";
52   exit;
53   }
```

Computer als Quasselstrippe

SoX alleine ist schon ein steter Quell der Freude. Richtig lustig wird es aber, wenn der Computer vor sich hin quasseln kann wie der *hitchBOT*, der per Anhalter quer durch Kanada fuhr und sich mit den Mitreisenden unterhielt [3]. Nötig ist dazu das Paket *festival* mit den Programm *text2wave* von der University of Edinburgh [4], das bei vielen Linux-Distributionen über die Paketverwaltung abrufbar ist.

Der Befehl `text2wave` kann, wie sich zu Recht vermuten lässt, eine Textdatei als Wav-Datei ausgeben – also quasi vorlesen. Zum Beispiel so:

```
text2wave /etc/motd -o /tmp/motd.wav
```

Dieses Kommando liest die Datei */etc/motd* ein, wandelt den Text per Phonemsynthese in das Wav-Format um und legt die generierte Sounddatei als */tmp/motd.wav* ab. Wer nun diese Datei mit dem SoX-Player abspielt, dem liest der Computer brav den Inhalt der Datei vor.

Das Ganze klingt wegen der Phonemsynthese zwar etwas blechern. Der Vorteil liegt aber darin,

dass sich aus rund drei Duzend Lauten alle denkbaren Wörter zusammensetzen lassen. Die Quelltext darf zur Verbesserung der Verständlichkeit etwas mogeln und beispielsweise ‚Kompjuta, statt ‚Computer, schreiben. Auch Worte mit ‚ü, bereiten Probleme. Eine Art Abhilfe schaffen ‚u, oder ‚i,.

Das Programm hat per default eine männliche, englischsprachige Stimme. Eine deutsche Stimme gibt es noch nicht, aber mit der oben angedeuteten Mogelei kann man pseudo-englische Texte erzeugen, die dann recht deutsch klingen – halt mit starkem Akzent. Längere Textdateien benötigen auch eine gewisse Zeit zum Konvertieren. Man sollte Standardtexte vorher erzeugen und durch aktuelle Einschübe ergänzen.

Mit einem kleinen Skript lassen sich Erzeugung und Ausgabe variabler Texte vereinfachen. Alles, was auf der Kommandozeile steht, wird in eine Wav-Datei gepipt (Zeile 03) und dann als Sound ausgegeben (Zeile 04). Die Shellvariable `$$` (Prozess-ID) in Zeile 01 sorgt hoffentlich für einen halbwegs eindeutigen Dateinamen. Neben Wav beherrscht das Programm etliche andere Formate wie *alaw*, *snd* oder *riff*, die sich mit der Option `-otype` festlegen lassen.

```
01 WAV=/tmp/talk.$$.wav
02
03 echo $@ | text2wave -o $WAV
04 play -q $WAV
```

Achtung, dunkle Töne

Bei Sound-Anwendungen ist zur Vorsicht zu raten, wie eine Anekdote belegt. Die Serverausstattung meines Labors bestand vor vielen, vielen Jahren mal aus zwei SUN-Sparc-10-Systemen, an die sich Windows-3.1-PCs über einen Telnet-Client andockten. Immerhin kostete so eine Kiste damals einen fünfstelligen DM-Betrag. Bei einem der Server war auch noch das Sound-System angeschlossen – Mono, 8 Kilohertz Samplingrate (Abbildung 1). Bei einem Server gibt es natürlich wenig Anlass zum Dudeln. Lediglich die Zeitanzeige lief. Irgendwann haben wir ein Start-Stop-Skript gebastelt, das beim Hochfahren des Rechners „Here comes the sun“ von den Beatles abspielte und beim Herunterfahren Schwarzeneggers Terminator-Spruch „Hasta la vista, baby“ zum besten gab.

Abbildung 1: Der Lautsprecher der SUN Sparc 10 war ein komplettes Sound-Interface inklusive Verstärker und Digitalisierer.

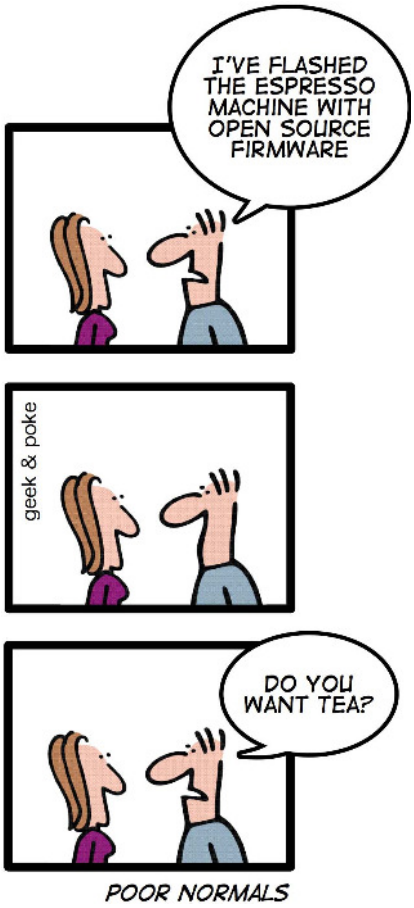
Als die Kiste mal neu zu booten war, verlegten wir das wegen des Tagesbetriebs in die Abendstunden. Die Wartungsarbeiten erledigte ich vom Büro aus per Telnet. Nachdem ich mich überzeugt hatte, dass niemand mehr eingeloggt war, startete ich den Reboot. Aber es war noch eine Studentin im Labor, die an einem der PCs arbeitete. Sie hat beinahe einen Herzkasperl bekommen, als das „Hasta la vista . . . “ aus dem dunklen Hintergrund ertönte.

Links

[1] SoX: http://sourceforge.net/projects/sox/
[2] Installationsanleitung für SoX:
http://www.deimeke.net/dirk/blog/index.php?/archives/2065-SoX-und-MP3-....html
[3] *hitchBOT* – eine Reise quer durch Kanada:
http://www.hitchbot.me/
[4] *festival*, it vollem Namen *Festival Speech Synthesis System*:
http://www.cstr.ed.ac.uk/projects/festival/

Über Jürgen

Jürgen Plate ist Professor für Elektro- und Informationstechnik an der Hochschule München. Er beschäftigt sich seit 1980 mit Datenfernübertragung und war, bevor der Internetanschluss für Privatpersonen möglich wurde, in der Mailboxszene aktiv. Unter anderem hat er eine der ersten öffentlichen Mailboxen – TEDAS der mc-Redaktion – programmiert und 1984 in Betrieb genommen.



Shellskripte mit Aha-Effekt IV
Ein Interface für das Shell-Shuffle

Was bisher geschah: Folge III von *Shellskripte mit Aha-Effekt* stellte ein Skript vor, das denjenigen Tastendruck akustisch quittiert, welchen mangels Tastatur eine Buzzer-Taste an den Server schickt und hier eine Aktion auslöst. Dieser Artikel spielt weiter: Der Buzzer betreibt – mit einem Umweg über Lötkolben und *C* – einen Soundserver.

von **Jürgen Plate**

Between two evils, I always pick
the one I never tried before.

Mae West

Selber Ort, andere Zeit: Eine Linux-Anwendung im Labor soll auf Tastendruck eine Aktion des Servers auslösen. Am Server sind weder Tastatur noch Monitor für Input und Output angeschlossen. Der Tastendruck wird also behelfsweise vollzogen. Immer noch sitzt dafür eine schöne, große, rote Pilztaste an einer Statusleitung der seriellen Schnittstelle – die Sorte, die auch Politiker gern mal drücken (Abbildung 1. Es wird also nicht etwa eine Tastatur simuliert, sondern die serielle Schnittstelle abgefragt, die es bei Servern immer noch gibt (und die den Administrator mitunter aus höchster Not rettet).

Ein kleines C-Programm bewachte die Statusleitungen. Sobald der Taster einige Sekunden gedrückt wurde, startete es das Shellskript aus Folge III. Denn was fehlte, war eine Rückmeldung der Form „Tastendruck akzeptiert“. Jenes Skript wählte aus einer Menge von Sound-Dateien zufällig eine aus und übergab den Namen an das aufrufende Skript [1].

Abbildung 1: Dieses Kind hat viele Namen: Pilztaster aka Grobmotorikertaste aka Buzzer aka Fuß- und Grobhandtaster

Und jetzt muss irgendwie der Pilztaster an die Schnittstelle heran. Nicht erschrecken: Im Folgenden kommt das Wort ‚löten , vor. Doch das bisschen bekommt MacGyver – ein Held wie wir – mit dem Feuerzeug oder dem Küchen-Flammenwerfer hin.

Wir brauchen eine neunpolige SUB-D-Buchse (das Ding mit dem Löchern), ein Stück Kabel (da geht alles, was zwei Adern hat) und natürlich den Pilztaster, der offiziell ‚Fuß- und Grobhandtaster, heißt. Gibt es alles bei Amazon. Die beiden Adern des Kabels sind am Taster und an die Pins 7 (*RTS*, Request to Send) und 9 (*RI*, Ring Indicator) der seriellen Buchse anzulöten – wie in Abbildung 2. Falsch machen kann man fast nichts, denn die Pinnummern sind in die Buchse eingeprägt.

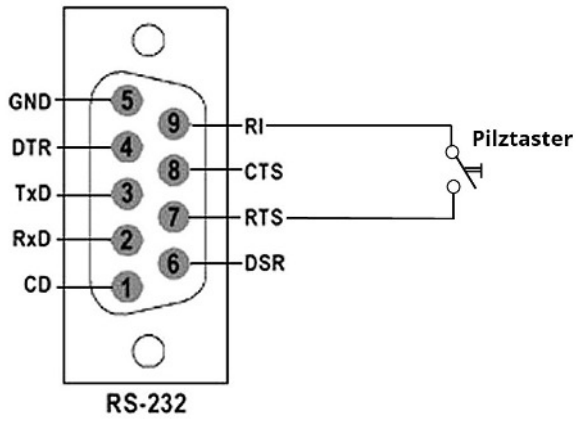


Abbildung 2: Die Belegung der seriellen Buchse.

Der Trick ist, dass ich den Ausgangspin RTS als Spannungsversorgung verwende. Software setzt ihn logisch auf 1, und schon kommen etwa 12 Volt aus der Buchse. Über den Taster geht es dann an den RI-Eingang, wo vor gefühlten 100 Jahren das Modem einen ankommenden Anruf signalisierte. Damit steht die Hardware.

Zunächst C statt Bash

Die Software dazu ist ausnahmsweise kein Shellskript. Aber das Programm folgt der Unix-Prämisse, nur eine Aufgabe zu erledigen – in diesem Fall die Tastenabfrage. Damit nicht jede und eventuell zufällige kurze Betätigung etwas auslöst, akzeptiert das Programm nur länger dauernde Betätigungen der Taste. Das Programm wartet einfach so lange, bis die Taste länger als zwei Sekunden unten ist und beendet sich dann. Da es intern `sleep()` aufruft, stellt es trotzt des aktiven Wartens keine Belastung des Betriebssystems dar.

Gesteuert wird das Ganze über Standard-IO-Control-Aufrufe in C, es gibt also keine Kernel-Magie oder Ähnliches. Übrigens kann man auf die gleiche Weise mittels `TIOCM_DSR` auch die Leitung *DTR* (Data terminal ready) und bei der PC-Architektur sogar die *TX*-Leitung schalten (Transmitter). Wie RI taugen auch die Leitungen CD (Aufruf: `TIOCM_CAR`), *DSR* (Aufruf: `TIOCM_DSR`) und *CTS* (Aufruf: `TIOCM_CTS`) als Eingang. Erweiterungen des Prinzips sind also möglich [2].

Listing: `taste.c`

```
01 #include <stdio.h>
02 #include <stdlib.h>
03 #include <sys/ioctl.h>
04 #include <fcntl.h>
05 #include <unistd.h>
06
07 int main(int argc, char** argv)
08   {
09   int Stat = 0;  /* fuer RI-Detektion */
10   int fd = 0;    /* Filedescriptor */
11   int AktStat;   /* fuer ioctl-Aufrufe */
12
13   if(argc != 2)
14     {
15     fprintf(stderr, "Aufruf: taste <dev>\n");
16     return 1;
17     }
18
19   if((fd = open(argv[1],O_RDWR|O_NDELAY)) < 0)
20     {
21     fprintf(stderr,
22        "Kann Device \"%s\" nicht oeffnen.\n",
23          argv[1]);
24     return 2;
25     }
26
27   AktStat = TIOCM_RTS;
28   ioctl(fd, TIOCMSET, &AktStat);
29
30   while(Stat < 2)
31     {
32     ioctl(fd, TIOCMGET, &AktStat);
33     if((AktStat & TIOCM_RNG)) Stat++;
34     else Stat = 0;
35     sleep(1);
36     }
37
38   AktStat = 0;
39   ioctl(fd, TIOCMSET, &AktStat);
40   close(fd);
41   return 0;
42   }
```

Das Programm aus Kasten *Listing: `taste.c`* erwartet eine Gerätedatei als Parameter. In der Regel ist dies `/dev/ttyS0`. Fehlt der Parameter, bricht das Programm mit Rückgabewert 1 ab (Zeilen 13 bis 17). Andernfalls versucht es, die Datei zu öffnen. Geht das schief, endet das Programm mit Rückgabewert 2 (Zeilen 19 bis 25). Danach setzt es *RTS* auf 1 (High), woraufhin sich der Pin 5 des Steckers an der Serverrückseite auf etwa +12 Volt einstellt (Zeilen 27 und 28). In der While-Schleife von Zeile 30 bis 36 liest das Programm den Schnittstellenstatus und endet nach mindestens zwei Sekunden dauerndem 1er-Wert an *RI* mit Rückgabewert 0. Das bedeutet, solange die Taste nicht oder

nicht lange genug gedrückt wird, läuft das Programm weiter. Durch den `sleep(1)`-Aufruf in der Schleife bewegt sich die Beanspruchung der Rechnerleistung im Bereich des Hintergrundrauschens. Am Ende machen Zeilen 38 bis 41 das Licht aus.

Nach dem Starten beispielsweise mit `taste /dev/ttyS0` wartet das Programm also, dass jemand lange genug den Buzzer drückt, um sich anschließend still zu beenden. Folgende Kommandozeile übersetzt das Ganze:

```
gcc -Wall -o taste taste.c
```

Pilz-Bashing

Nun haben wir ein neues Tool, das in beliebigen Skripten eingesetzt werden kann. Zurück in die Bash! Im Labor hat sich inzwischen eine Anwendung dafür ergeben: Ein übrig gebliebenes Barebone-System mit Atom-CPU inspirierte zusammen mit zwei bis dahin kaum verwendeten, fest montierten Aktiv-Boxen aus der Laboreinrichtung die Idee, den Barebone als Sound-Server zu installieren. Und schon stellte sich die Frage des Interfaces zum Abspielen der Musik.

Die im Labor ansässigen Praktikums-PCs haben zwar alle die übliche grafische Oberfläche, jedoch neigt der Laborleiter zu einfachen Lösungen zu Gunsten der Faulheit. Server bekommen sowieso kein *X*, zum Administrieren reicht ein SSH-Login. Damit fallen schon mal alle Klicki-Bunti-Interfaces durchs Raster. Übrig blieb der gute alte *MOC* (Music On Console), ein Audio-Player, der über die Konsole gesteuert wird [3]. Von der Bedienung her erinnert er an den *Midnight-Commander*, und er unterstützt Playlists sowie alle wichtigen Dateiformate.

Nur ein – an sich ideales – Feature macht manchmal Probleme: Hat man seine Musikstücke oder Playlist ausgewählt, ist es nicht nötig, die SSH-Shell offen zu lassen. Der Player wird mit der *q*-Taste verlassen und dudelt als Server fröhlich weiter. Man kann dann später die Session wieder aufnehmen. Immer wieder loggen sich aber Schussel abends aus und merken erst an der Tür, dass da noch Geräusche aus den Boxen kommen. Also: Jacke aus, Arbeitsplatz-PC booten, wieder einloggen, Player killen, ausloggen, PC runter fahren, Jacke an, S-Bahn weg.

So bekam der Soundserver eine neue Pilz-Taste und ein Shellskript, das beim Booten automatisch aus */etc/rc.local* startet. Hier ist nur zu beachten, dass man das Programm in den Hintergrund verlagert, weil sonst die Abarbeitung von */etc/rc.local* stoppt. Damit käme der ganze Bootprozess ins Stolpern. Also wandert zusätzlich die folgende Zeile in `/etc/rc.local`:

```
nohup /home/sound/killer &
```

Ob der Befehl `nohup` wirklich nötig ist oder ob das Ampersand zum Verlagern in den Hintergrund reicht, ist derzeit nicht verifiziert.

Fehlt also nur noch das Killer-Skript. Dessen Aufbau ist recht einfach. Nach der Definition einiger Variablen (was spätere Änderungen vereinfacht) prüft es, ob es selbst eventuell schon oder noch läuft. In diesem Fall beendet sich das Skript, ohne weiterzumachen (Zeile 08). Das verhindert, dass der Killer mehrfach läuft. So etwas kann eigentlich nur vorkommen, wenn jemand das Killer-Skript von Hand gestartet hat – nach dem Motto ‚Mal sehen, was passiert‚.

Normalerweise, also beim ersten und einzigen Aufruf, landet die Ausführung in einer Endlosschleife (Zeilen 10 bis 13). Hier wartet das Skript durch Aufruf des oben beschriebenen `taste.c` in Zeile 12 auf die Pilztaste. Wurde diese betätigt, ermittelt es in Zeile 15 die Prozessnummern aller laufenden *mocp*-Prozesse. Diese beendet es, indem es zuerst mit `-HUP` höflich nachfragt und dann renitente Reste mit `-9` versenkt (Zeile 21).

Nach dem Löschen der *mocp*-Prozesse erzeugt das Skript einen Quittungston (Zeile 24 und 25), den das in Folge III von *Shell-Skript mit Aha-Effekt* zufällig auswählt (vgl. Zeile 06). Abschließend erzeugt es eine Pause, die nicht mit dem *sleep*-Befehl, sondern durch Abspielen eines entsprechenden Soundfiles realisiert wird (Zeile 27, vgl. Zeile 04). Das verhindert, dass Spielkinder durch andauerndes Drücken der Taste alle Zufallssounds in Folge abspielen: Die relativ lange Pause von fünf Sekunden wirkt als Spaßbremse.

Wer keine Sounddatei mit sekundenlanger Stille herumliegen hat, freut sich auf einen entsprechenden zukünftigen Artikel und genießt bis dahin die Hintergrundinformation in [4].

```
01 PLAY="/usr/bin/play -V0"
02
04 SILENCE="/home/sound/psst.wav"
05
06 RND="/home/sound/shuffle"
07
08 ps -C killer && exit
09
10 while :
11   do
12   /home/sound/taste /dev/ttyS0
13   test $? -ne 0 && exit
14
15  PIDS=$(ps -C mocp -o pid=)
16
17   if [ -n "$PIDS" ]
```

```
18      then
19      kill -HUP $PIDS > /dev/null 2>&1
20      sleep 3
21      kill -9 $PIDS > /dev/null 2>&1
22      fi
23
24    SOUND_TO_PLAY=$($RND)
25    $PLAY -v5 $SOUND_TO_PLAY > /dev/null 2>&1
26
27    $PLAY -v 0.5 $SILENCE > /dev/null 2>&1
28
29    done
```

Nachtrag: Leuchten statt Piepsen

Manch einer mag vielleicht die akustische Rückmeldung nicht – etwa weil es an der Soundausgabe gebricht, oder weil der interne Lautsprecher komplett wegoptimiert wurde. Mit dem konnte man wenigsten noch piepsen. Auch da gibt es Abhilfe in Form einer optischen Rückmeldung. Einziger Nachteil: Es muss wieder gelötet werden.

An zusätzlicher Hardware ist nur ein Widerstand von 4.7 Kiloohm und eine sogenannte Duo-LED nötig. Letztere besteht aus zwei LEDs in einem Gehäuse, die antiparallel geschaltet sind, wie Abbildung 3 zeigt. Zur Not lassen sich auch einfach zwei LEDS passend verlöten. Wichtig ist nur, dass man eine Low-Power-LED verwendet, die schon bei 2 mA Strom ordentlich hell leuchtet.

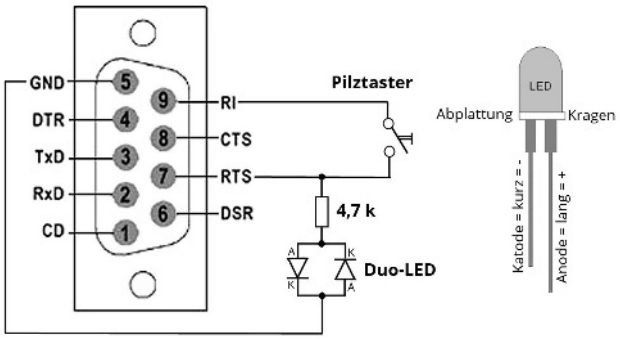


Abbildung 3: Ein Widerstand und eine LED erweitern die Funktion.

LEDs sind gepolte Bauteile. Sie leuchten nur, wenn der Strom in der richtigen Richtung durch die LED fließt. Deshalb zeigt Abbildung 3 rechts eine einzelne LED. Die Kathode – in der Schaltung mit ‚K‚ bezeichnet – ist der kürzere Anschlussdraht. Bei der Duo-LED entscheidet die Polung darüber, welche Farbe wann leuchtet.

Außerdem muss stets ein Widerstand den Strom begrenzen, der durch eine LED fließt. Mit dem vorgeschlagenen Wert fließen etwa zwei Milliampere durch die Diode. Vom Widerstand hängt auch die Helligkeit ab: Hier lässt sich mit Werten zwischen 1 Kiloohm und 10 Kiloohm experimentieren.

Was passiert nun bei unserer aufgebohrten Schaltung? Anfangs ist der RTS-Pin auf logisch 0, was bedeutet, dass der Pin eine Spannung von zwischen -10 und -12 Volt gegenüber dem GND-Pin aufweist. Es leuchtet also die LED, deren Kathode in Richtung RTS-Pin liegt (in Abbildung 3 ist das die rechte LED). Dann startet `taste` und setzt RTS auf logisch 1. Nun sind es zwischen +10 und +12 Volt gegenüber GND, und es leuchtet die andere LED (in Abbildung 3 ist das die linke LED). Wenn das Programm nach dem Tastendruck endet, wird RTS wieder auf logisch 0 gesetzt und die LEDs wechseln sich wieder ab. So hat man eine generelle Bereitschaftsanzeige und man sieht, wann das Programm wartet.

Die LEDs passen in den Unterbau der Pilztaste: Einfach zwei fünf Millimeter große Löcher in den Sockel bohren und die Lämpchen einkleben. Die Verdrahtung erfolgt frei innerhalb des Sockels. Für die Anbindung zum RS232-Stecker des Servers eignet sich ein handelsübliches Telefonkabel, das gibt es sogar im Baumarkt. Es hat vier Adern – so bleibt sogar noch eine Ader frei für Erweiterungen.

Links

[1] Jürgen Plate, Shellskripte mit Aha-Effekt III: Shell-Shuffle für die Ohren. In: UpTimes 1-2014, Frühlingsausgabe, S. 41:
http://www.guug.de/uptimes/2014-1/
[2] Mehr zur Programmierung der seriellen Schnittstelle:
http://www.netzmafia.de/skripten/hardware/Seriell/index.html
[3] MOC – Music on Console: http://moc.daper.net/
[4] „Dr. Murkes gesammeltes Schweigen„:
http://www.youtube.com/watch?v=-JQ6NzeMsGU

Loser, Learner und O'lala
So war die LinuxCon Europe

Vom 13. bis 17. Oktober 2014 habe ich die LinuxCon Europe, Cloud Europe und Linux Plumbers Conference in Düsseldorf besucht. Hier sind meine Erkenntnisse.

von **Nils Magnus**

Wichtig sind menschliche Tester.

Linus Torvalds

Veranstalter aller drei Teilkonferenzen war die Linux Foundation. Gleichzeitig gab es auch noch die Embedded Linux Conference und ein gefühltes gutes Dutzend weitere "Co-Located Subconferences". Alles fand im Congress Center Düsseldorf (CCD) statt, in ziemlicher Nähe zum Flughafen. Mindestens für mich war es mitunter etwas schwierig, die einzelnen Teile voneinander zu unterscheiden, denn alle hatten mehr oder minder große Überschneidungen. Die LinuxCon befasste sich im Kern etwas mehr mit echten Kernel-Angelegenheiten, die Cloud Europe mit Cloudthemen, IaaS und PaaS. Die so genannte Plumbers Conference beschäftigte sich mit allem irgendwo dazwischen. Die ersten beiden waren etwas geschäftsorientierter, die Plumbers etwas hemdsärmeliger.

Abbildung 1: Jim Zemlin ist der CEO der Linux Foundation und somit der Arbeitgeber von Linus Torvalds. Foto: Linux Foundation

Das CCD ist so, wie diese Conference Center so sind: Groß, durchaus praktisch, etwas unübersichtlich – was wohl den überall vorherrschenden 60-Grad-Winkeln geschuldet ist – und vergleichsweise weit draußen. Das Publikum bestand mehrheitlich aus erfahrenen System Engineers, die Sprecher waren durchaus international. Konferenzsprache war englisch, aber auch mit deutsch kam man gut zurecht, zumindest außerhalb der Vorträge.

Die LinuxCon ist eine der wenigen Veranstaltungen, zu denen Linus Torvalds kommt und auftritt – vermutlich, weil sein Arbeitgeber, die Linux Foundation, ihn dazu zwingt (Abbildung 1). Zumindest habe ich jetzt zum wiederholten Male diesen Eindruck gewonnen. Er macht jedes Jahr mit Intels Dirk Hohndel die gleiche Show: Dirk fragt mehr oder weniger Allgemeinplätze ab, Linus antwortet mit kleinen Sticheleien. Technisches oder Strategisches kam da noch nie bei heraus, und das war auch diesmal nicht anders (Abbildung 2).

En vogue: Docker, OpenShift, OpenStack

Es gab eine Menge Vorträge zu allen möglichen Themen, die mal tief-technisch, mal oberflächlich-marketinglastig waren. Zumindest für mich waren die Gespräche mit den Leuten im Gang daher ein ganz wesentlicher Punkt. Da waren Kernel-Maintainer Greg Kroah-Hartman und Christoph Hellwig zu treffen, RT-Maintainer Thomas Gleixner, Chris Schlaeger, der CEO von Amazon WS Development, Thorsten Leemhuis von der c't, der den Kernel-Log schreibt, und der rebellige Systemd-Entwickler Lennart Poettering. Die Partys am Abend (und es gab viele Abende) waren ganz gut (und lang). Bei den Themen habe ich mich schwerpunktmäßig mit Cloud- und systemnahen Fragen befasst.

Docker war auf der Konferenz allgegenwärtig. Die Vorträge behandelten mehrheitlich Aspekte, die deutlich über die Anwendungsfragen hinausgehen, die in meinem Day-Job anfallen. Mir scheint Docker in eine Art Vakuum zwischen PaaS

und IaaS zu stoßen. PaaS mag es ja als Konzept theoretisch geben, ich habe es aber noch nirgendwo live und in Farbe im Einsatz gesehen. Das, was sich oft IaaS nennt, sind in Wirklichkeit nur virtualisierte Individualserver, die nur wenig automatisierbar und auch nur wenig elastisch skalierbar sind.

So ganz im Produktionsbetrieb angekommen ist Docker aber wohl noch nicht: Verschiedentlich haben Sprecher über Stabilitätsprobleme gerade unter Last geklagt. Das betrifft wohl vor allem die Overlay-Dateisysteme. Das Security-Thema ist darüber hinaus überaus trickreich und beansprucht genaueste Beobachtung. Dafür hat man bei Docker noch genügend Kontrolle über die Umgebung und es bietet den Rahmen für eine einheitliche Form des Deployments.

Abbildung 2: Linux-Erfinder Linus Torvalds und Linux-Obmann bei Intel, Dirk Hohndel. Foto: Linux Foundation

Das manifestiert sich in einer Reihe von Tools: Gerade Red Hat promotet **OpenShift** sehr intensiv. Das ist eine Art **Management-Plattform**, in der sich Docker- und PaaS-Komponenten verwalten lassen. Ich hatte den Eindruck, dass das keine Kleinigkeit ist, sondern intensive Einarbeitungszeit erfordert.

Eine Nummer kleiner, aber auch von Red Hat am Rande von OpenShift entwickelt, ist **Cockpit**, eine Art Web-Frontend für Hosts, auf denen Docker läuft. Damit kann man unter anderem Container starten, stoppen, Images verwalten und die Last beobachten. Es erscheint praktisch als OpenNebula oder Rex.IO für Docker. Mir kam das relativ übersichtlich vor. Es ist aber auf einen Host begrenzt, was bedeutet, dass Container-Migrationen über Hostgrenzen hinweg noch nicht vorgesehen sind.

Open Stack als IaaS-Statthalter war auch stark vertreten. Da sah ich aber wenige Neuigkeiten (OpenStack Juno war erst kurz nach der Konferenz veröffentlicht worden). Das Thema scheint mittlerweile weitestgehend angekommen zu sein: Eine riesige Zahl von Leuten nutzen das Framework oder die Software, oder sie evaluieren zumindest deren Einsatz. **OpenNebula** war auch vertreten, allerdings war das Gebotene etwas arg ernüchternd: Ein marketinglastiger Überblick vor zwölf Besuchern, den ich schon vor zwei Jahren gehört habe, und ein Workshop zum Aufsetzen eine ONE-Cloud vor acht Personen. Die Sprecher wiesen etwas verzweifelt auf ihre guten Kunden gerade im Wissenschaftsbereich hin, zum Beispiel das CERN, aber im echten Business will das offenbar kaum jemand nutzen.

Aufsteigend: Systemd, Network Functions Virtualization

Ein relevantes Thema ist meiner Meinung nach **Systemd**. Mir ist einmal mehr bewusst geworden, dass Systemd erheblich mehr ist, als bloß ein SysV-Init-Dropin-Replacement. Seine Entwickler bauen Stück für Stück das komplette Unix bzw. Linux an der Schnittstelle zwischen Anwendungen und Kernel um. Das umfasst Ansätze für Container (*Nspawn*), Netzwerk (DNS, DHCP, Networkmanager, VPNs, DNSSEC), das Starten von Anwendungen (ähnlich *inetd*), Anbindung an Container und noch eine Menge mehr. Cool war der Ansatz von so einer Art Port-Activated-Containern – den offiziellen Namen müsste ich noch einmal nachschlagen: Man konfiguriert sich ja gern mal 100 unterschiedliche Container für alles mögliche, die aber alle nicht laufen. Erst wenn man dorthin eine Netzverbindung etwa über SSH oder HTTP/S eröffnet, schnappt Systemd sich die Verbindung und fährt den Container hoch. Ist die Verbindung abgebaut, fährt Systemd ihn wieder herunter.

Ein letztes Thema, das boomt, Praktiker aber vermutlich nicht ganz so schnell stark beschäftigt, ist das virtualisierte Netz. Bis letztes Jahr hieß das ja noch **SDN** (Software Defined Networking), aber in diesem Jahr wurde das Akronym **NFV** (Network Functions Virtualization) für mehr oder weniger dasselbe Thema gesetzt. Telekommunikationsanbieter versprechen sich dadurch eine erhebliche Einsparungen bei ihren extrem teuren Spezial-Appliances. Sie hoffen, dass man mittels NFV dann auf einer Kiste – etwas überspitzt gesagt – sowohl eine GSM-Basisstation als auch einen TCP-Loadbalancer betreiben kann; zu Weihnachten tauschen sie dann alles durch SMS-Gateways aus. Da scheint es noch viel Euphorie zu geben, aber mich erinnert das alles ein wenig an die Cloud-Aufbruchstimmung vor fünf Jahren. Für normale Rechenzentren sehe ich da aktuell keine große Revolution, wobei da sicher der eine oder andere Technologiebrocken dafür abfällt.

tl;dr: Eine sehr intensive Woche, die ich durchaus empfehlen kann. Die nächste LinuxCon Europe & Co. findet 2015 voraussichtlich in Dublin statt.

Über Nils

Diplominformatiker, Perl-Jünger und Bash-Basher Nils Magnus hat nach zehn Jahren Security-Beratung einen Ausflug als Redakteur in die Linux-Magazin-Redaktion unternommen. Seit rund drei Jahren hat ihn das System- und Netzwerk-Handwerk wieder, das er als System Engineer der inovex GmbH ausübt.

Hilfreiches für alle Beteiligten
Autorenrichtlinien

Selbst etwas für die UpTimes schreiben? Aber ja! Als Thema ist willkommen, was ein GUUG-Mitglied interessiert und im Themenbereich der GUUG liegt. Was sonst noch zu beachten ist, steht in diesen Autorenrichtlinien.

Der Schriftsteller ragt zu den Sternen empor,
Mit ausgefranstem T-Shirt.
Er raunt seiner Zeit ihre Wonnen ins Ohr,
Mit ausgefranstem T-Shirt.

Frei nach Frank Wedekind, Die Schriftstellerhymne

Wir sind an Beiträgen interessiert. Wir, das ist diejenige Gruppe innerhalb der GUUG, die dafür sorgt, dass die UpTimes entsteht. Dieser Prozess steht jedem GUUG-Mitglied offen. Der Ort dafür ist die Mailingliste <redaktion@uptimes.de>.

Welche Themen und Beitragsarten kann ich einsenden?

Die UpTimes richtet sich als Vereinszeitschrift der GUUG an Leser, die sich meistens beruflich mit Computernetzwerken, IT-Sicherheit, Unix-Systemadministration und artverwandten Themen auseinandersetzen. Technologische Diskussionen, Methodenbeschreibungen und Einführungen in neue Themen sind für dieses Zielpublikum interessant, Basiswissen im Stil von *Einführung in die Bourne Shell* hingegen eher nicht. Wer sich nicht sicher ist, ob sein Thema für die UpTimes von Interesse ist, kann uns gern eine E-Mail an <redaktion@uptimes.de> schicken.

Neben Fachbeiträgen sind Berichte aus dem Vereinsleben, Buchrezensionen, Konferenzberichte, humoristische Formen und natürlich Leserbriefe interessant. Wer nicht gleich mehrseitige Artikel schreiben möchte, beginnt also mit einem kleineren Beitrag.

Fachbeiträge sind sachbezogen, verwenden fachsprachliches Vokabular und anspruchsvolle Erläuterungen, besitzen technische Tiefe und ggf. auch Exkurse. Berichte aus dem Vereinsleben greifen aktuelle Themen auf oder legen Gedankengänge rund um die GUUG und ihre Community dar. Konferenzberichte zeigen, welche Veranstaltungen jemand besucht hat, was er/sie dort erfahren hat und ob die Veranstaltung nach Meinung des Autoren beachtenswert oder verzichtbar war. Unterhaltsame Formen können ein Essay oder eine Glosse sein, aber auch Mischformen mit Fachartikeln (Beispiel: der „Winter-Krimi“ in Ausgabe 2013-3). Auch unterhaltsame Formen besitzen jedoch inhaltlichen Anspruch. Denn die UpTimes ist und bleibt die Mitgliederzeitschrift eines Fachvereines.

In der UpTimes legen wir daher auch Wert auf professionelle publizistische Gepflogenheiten und einheitliche Schreibweisen. Dafür sorgt zum Beispiel ein einheitliches Layout der Artikel, oder etwa die grundsätzliche Vermeidung von Worten in Großbuchstaben (entspricht typografisches Schreien) oder von Worten in Anführungsstrichen zum Zeichen der Uneigentlichkeit (entspricht Distanzierung von den eigenen Worten). Wichtig sind außerdem beispielsweise Quellenangaben bei Zitaten, Kenntlichmachung fremder Gedanken, Nachvollziehbarkeit der Argumentation sowie Informationen zum Autor nach dem Artikel.

In welchem Format soll ich meinen Artikel einsenden?

ASCII: Am liebsten blanke UTF8-Texte. Gern mit beschreibenden Anmerkungen oder Hinweisen (zum Beispiel mit Prozentzeichen zum Kenntlichmachen der Meta-Ebene).

LaTeX: Wir setzen die UpTimes mit LaTeX. Weil wir – wie es sich beim Publizieren gehört – mehrspaltig setzen und ein homogenes Erscheinungsbild anstreben, verwenden wir für die UpTimes bestimmte Formatierungen. Es ist nicht erwünscht, eigene Layoutanweisungen einzusenden. Wir behalten uns vor, Texte für die Veröffentlichung in der UpTimes umzuformatieren. Eine Vorlage mit den von uns verwendeten Auszeichnungen für Tabellen, Kästen und Abbildungen gibt es unter http://www.guug.de/uptimes/artikelvorlage.tex.

Listings: Der mehrspaltige Druck erlaubt maximal 45 Zeichen Breite für Code-Beispiele, inklusive 1 Leerzeichen und einem Zeichen für den Zeilenumbruch innerhalb einer Code-Zeile (Backslash). Breitere Listings formatieren wir um, verkleinern die Schriftgröße oder setzen sie als separate Abbildung.

Bilder: Wir verarbeiten gängige Bildformate, soweit ImageMagick sie verdaut und sie hochauflösend sind. Am besten eignen sich PNG- oder PDF-Bilddateien. Plant bei längeren Artikeln mit 1 Abbildung pro 3000 Zeichen. Das müssen nicht Bilder sein, sondern auch Tabellen, Listings oder ein Exkurskasten sind möglich. Verseht Eure Bilder nicht mit Rahmen oder Verzierungen, weil die Redaktion diese im UpTimes-Stil selbst vornimmt.

Wie lang kann mein Artikel sein?

Ein einseitiger Artikel hat mit zwei Zwischentiteln um die 2.700 Anschläge. Mit etwa 15.000 Anschlägen – inklusive 3 Abbildungen – landet man auf rund vier Seiten. Wir nehmen gern auch achtseitige Artikel, achten dabei aber darauf, dass der Zusammenhang erhalten bleibt und dass es genug Bilder gibt, damit keine Textwüsten enstehen.

Wer Interesse hat, für die UpTimes zu schreiben, macht sich am besten um die Zeichenzahl nicht so viele Gedanken – auch für kurze oder lange Formate finden wir einen Platz. Die Redaktion ist bei der konkreten Ideenentwicklung gern behilflich. Für eine Artikelidee an <redaktion@uptimes.de> reicht es, wenn Ihr ein bestimmtes Thema behandeln wollt.

Wohin mit meinem Manuskript?

Am einfachsten per E-Mail an <redaktion@uptimes.de> schicken. Das ist jederzeit möglich, spätestens jedoch vier Wochen vor dem Erscheinen der nächsten UpTimes. Zum Manuskript ist ein kleiner Infotext zum Autor wichtig, ein Bild wünschenswert.

Nützlich ist, wenn der Text vor Einsendung durch eine Rechtschreibkorrektur gelaufen ist. `aspell`, `ispell` oder `flyspell` für Textdateien sowie die von LibreOffice bieten sich an. Wenn Ihr Euren Text an die Redaktion schickt, solltet Ihr also weitestmöglich bereits auf die Rechtschreibung geachtet haben: Nach der Einschickung ist Rechtschreibung und Typo-Korrektur Aufgabe der Redaktion. Die Texte in der UpTimes folgen der neuen deutschen Rechtschreibung.

Wie verlaufen Redaktion und Satz?

Wir behalten uns vor, Texte für die Veröffentlichung in der UpTimes zu kürzen und zu redigieren. Das bedeutet, dafür zu sorgen, dass der Artikel nicht ausufert, versehentliche Leeraussagen wegfallen, Syntax und Satzanschlüsse geglättet werden, dass Passiva und Substantivierungen verringert und Unklarheiten beseitigt werden (die zum Beispiel Fragen offen lassen oder aus Passivkonstruktionen resultieren, ohne dass der Schreibende das merkt). Manchmal ist dieser Prozess mit Nachfragen an den Autoren verbunden.

Die endgültige Textversion geht jedem Autoren am Ende zur Kontrolle zu. Dabei geht es um die inhaltliche Kontrolle, ob sich durch den Redaktionsprozess Missverständnisse oder Falschaussagen entstanden sind. Danach setzt die Redaktion die Artikel. Wenn der Satz weitgehend gediehen ist – also ein *Release Candidate* als PDF vorliegt – erhalten die Autoren als erste diesen RC. Danach wird die UpTimes dann veröffentlicht.

Gibt es Rechtliches zu beachten?

Die Inhalte der UpTimes stehen ab Veröffentlichung unter der CC-BY-SA-Lizenz, damit jeder Leser die Artikel und Bilder bei Nennung der Quelle weiterverbreiten und auch weiterverarbeiten darf. Bei allen eingereichten Manuskripten gehen wir davon aus, dass der Autor sie selbst geschrieben hat und der UpTimes ein nichtexklusives, aber zeitlich und räumlich unbegrenztes Nutzungs- und Bearbeitungsrecht unter der CC-BY-SA einräumt.

Bei Fotos oder Abbildungen Dritter ist es rechtlich unabdingbar, dass der Autor sich bei dem Urheber die Erlaubnis zu dieser Nutzung einholt, und fragt, wie die Quelle genannt zu werden wünscht. Die Frage nach der CC-BY-SA ist hierbei besonders wichtig.

An Exklusivrechten, wie sie bei kommerziellen Fachzeitschriften üblich sind, hat die UpTimes kein Interesse. Es ist den Autoren freigestellt, ihre Artikel noch anderweitig nach Belieben zu veröffentlichen.

Bekomme ich ein Autorenhonorar?

Für Fach- und literarische Beiträge zahlt die GUUG dem Autor nach Aufforderung durch die Redaktion und Rechnungstellung durch den Autor pro Seite 50 € zuzüglich eventuell anfallender USt. Beiträge für die Rubrik „Vereinsleben“,

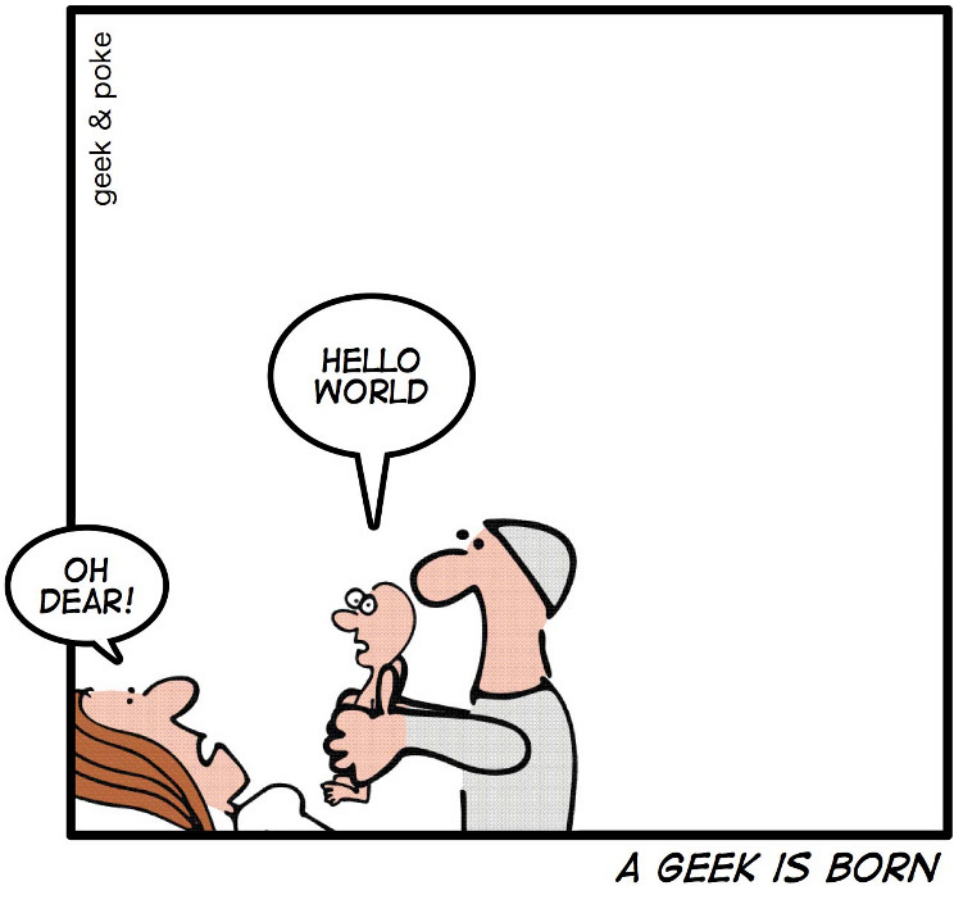
Buchrezensionen und Artikel bezahlter Redakteure sind davon ausgenommen. Gleiches gilt für Paper, wenn die UpTimes die Proceedings der Konferenz enthält.

Nächste Proceedings-Ausgabe:
Uptimes 2015-1, FFG-Ausgabe

- Redaktionsschluss: Zwei Wochen vor Konferenzbeginn (Freitag, 13. März 2015).
- Erscheinung: Zum Konferenzbeginn auf der Webseite (24. März 2015).
- Gesuchte Inhalte: Paper zum Frühjahrsfachgespräch 2015.
- Manuskripte an: Mailingliste <redaktion@uptimes.de>

Nächste redaktionelle Ausgabe:
Uptimes 2015-2, Sommer-Ausgabe

- Redaktionsschluss: Montag, 1. Juni 2015.
- Erscheinung: Samstag, 11. Juli 2015.
- Gesuchte Inhalte: Fachbeiträge über Unix und verwandte Themen, Veranstaltungsberichte, Rezensionen, Beiträge aus dem Vereinsleben.
- Manuskript-Template: http://www.guug.de/uptimes/artikel-vorlage.tex
- Fragen, Artikelideen und Manuskripte an: <kehrer@guug.de> oder an die Mailingliste <redaktion@uptimes.de>



Über die GUUG
German Unix User Group e.V.

Vereinigung deutscher Unix-Benutzer

Die Vereinigung Deutscher Unix-Benutzer hat gegenwärtig rund 700 Mitglieder, davon etwa 90 Firmen und Institutionen.

Im Mittelpunkt der Aktivitäten der GUUG stehen Konferenzen. Ein großes viertägiges Event der GUUG hat eine besondere Tradition und fachliche Bedeutung: In der ersten Jahreshälfte treffen sich diejenigen, die ihren beruflichen Schwerpunkt im Bereich der IT-Sicherheit, der System- oder Netzwerkadministration haben, beim *GUUG-Frühjahrsfachgespräch* (FFG).

Seit Oktober 2002 erscheint mit der *Uptimes* – die Sie gerade lesen – eine Vereinszeitung. Seit 2012 erscheint die Uptimes einerseits zu jedem FFG in Form einer gedruckten Proceedings-Ausgabe (ISBN), und andererseits im Rest des Jahres als digitale Redaktionsausgabe (ISSN). Daneben erhalten GUUG-Mitglieder zur Zeit die Zeitschrift *LANline* aus dem Konradin-Verlag kostenlos im Rahmen ihrer Mitgliedschaft.

Schließlich gibt es noch eine Reihe regionaler Treffen (http://www.guug.de/lokal): im Rhein-Ruhr- und im Rhein-Main-Gebiet sowie in Berlin, Hamburg, Karlsruhe und München.

Warum GUUG-Mitglied werden?

Die GUUG setzt sich für eine lebendige und professionelle Weiterentwicklung im Open Source-Bereich und für alle Belange der System-, Netzwerkadministration und IT-Sicherheit ein. Wir freuen uns besonders über diejenigen, die bereit sind, sich aktiv in der GUUG zu engagieren. Da die Mitgliedschaft mit jährlichen Kosten

Fördermitglied	350 €
persönliches Mitglied	90 €
in der Ausbildung	30 €

verbunden ist, stellt sich die Frage, welche Vorteile damit verbunden sind?

Neben der Unterstützung der erwähnten Ziele der GUUG profitieren Mitglieder auch finanziell davon, insbesondere durch die ermäßigten Gebühren bei den Konferenzen der GUUG und denen anderer europäischer UUGs. Mitglieder bekommen außerdem *c't* und *iX* zum reduzierten Abopreis.

Wie GUUG-Mitglied werden?

Füllen Sie einfach das umseitige Anmeldeformular aus und schicken Sie es per Fax oder Post an die unten auf dem Formular angegebene Adresse. Falls Sie die Seite nicht herausreißen wollen: Sie können den Mitgliedsantrag als PDF herunterladen, siehe URL auf dem Mitgliedsantrag.

Impressum

Uptimes – Mitgliederzeitschrift der
German Unix User Group (GUUG) e.V.
Herausgeber: GUUG e.V.
Ebertallee 16
D-22607 Hamburg
E-Mail: <redaktion@uptimes.de>
Internet: http://www.guug.de/uptimes/

Autoren dieser Ausgabe: Dirk Wetter, Anika Kehrer (Interview), Christian Lademann, Anika Kehrer, Mathias Weidner, Thomas Limoncelli (Englisches Original), Mathias Weidner (Übersetzung), Jürgen Plate, Nils Magnus
V.i.S.d.P: Dirk Wetter, Vorstandsvorsitzender, Anschrift siehe Herausgeber
Chefredaktion: Anika Kehrer
LaTeX-Layout (PDF): Robin Schröder
XHTML-Layout (ePub): Mathias Weidner
Titelgestaltung: Hella Breitkopf
Bildnachweis: Titelbild „Control Data deadstart panel,, CC-BY-SA, mit freundlicher Genehmigung von Wolfgang Stief. Comicreihe *geek & poke* CC-BY-SA, mit freundlicher Genehmigung von Oliver Widder. Andere Quellennachweise am jeweiligen Bild.
Titelbild: So genanntes *Deadstart Panel*, wie es für Mainframes der Control Data Corporation ab dem Modell CDC6600 (eingeführt 1964) bis zur Baureihe Cyber 170 (1979/1980) im Einsatz war. Mit dem Panel wurde die Boot-Sequenz zum Starten des Mainframes eingegeben, bis sich das System dann selbständig von einem anderen Gerät (normalerweise Band oder Platte) weiter initialisieren konnte. Die farbigen Klebepunkte sind ein manueller Patch der Operateure des Rechners, von dem das abgebildete Deadstart Panel kommt. Damit sind verschiedene Opcodes, Gerätenummern von Band- und Plattenlaufwerken oder I/O-Kanäle markiert, um den Deadstart-Prozess zu vereinfachen und nicht jedesmal in einem Handbuch nach den passenden Werten zu suchen. Ab der Modellreihe Cyber 180 wurde das *Deadstart Panel* durch Halbleiter-Elektronik abgelöst. Eine Liste mit allen Großrechnermodellen der Firma Control Data gibt es auf der Webseite http://www.cray-cyber.org.
Verlag: Lehmanns Media GmbH, Hardenbergstraße 5, 10623 Berlin
ISSN: 2195-0016

Wenn Sie Interesse an Anzeigen in der Uptimes haben,
wenden Sie sich bitte an <werbung@guug.de>.