Backups mit Obnam

Lars Wirzenius (liw@liw.fi) Übersetzung: Jan Niggemann (jn@hz6.de)

Version obnam-1.22-43-g6d9742cb

Inhaltsverzeichnis

Kapitel 1

Einleitung

> ... Backups? Hat hier jemand was von Backups gesagt? Ich bin sicher das hier irgendwer über Backups gesprochen hat. Backups! BACKUPS! BACKUPS SIND ABGEFAHREN!

Das ist ein Zitat direkt aus meiner IRC history. Ich finde Backups recht interessant, besonders die Implementierung, und manchmal bin ich sogar etwas besessen davon. Deswegen habe ich meine eigenes Backup-Programm geschrieben. Ich nenne es Obnam. Das ist sein Handbuch.

Ich bin ungewöhnlich: Leute finden Backups meistens nervtötend und allenfalls langweilig. Wenn ich mit Leuten über Backups rede, dann ist die Reaktion normalerweise "Ich weiß, ich sollte...". Es gibt viele Gründe dafür. Einer ist, dass Backups so ähnlich wie Versicherungen sind: Man muss zunächst Zeit, Arbeit und Geld 'reinstecken um hinterher von ihnen Gebrauch zu machen. Ein anderer Grund ist, dass das gesamte Thema beängstigend ist: Man muss darüber nachdenken was passiert wenn etwas richtig schief läuft, und das schreckt die Leute ab. Ein dritter Grund ist: Es gibt zwar eine Menge an Backup-Tools, aber es ist nicht immer einfach eines auszuwählen.

Dieses Handbuch ist für das Obnam Programm, aber es versucht für jeden nützlich zu sein, der über Backups nachdenkt.

1.1 Anmerkung des Übersetzers

Ich bin kein professioneller Übersetzer, sondern ein interessierter Obnam Anwender. Aus Erfahrung weiß ich, dass man sich beim Übersetzen von fremdsprachigen Texten immer wieder die Frage stellt, ob man Begriffe eindeutschen soll, oder nicht. Viele englische Begriffe werden auch im deutschen Umfeld verwendet, daher habe ich einige Begriffe nicht übersetzt von denen ich denke, dass sie dem Leser bekannt sind.

Sicherlich kann die eine oder andere Stelle anders, manchmal auch besser übersetzt werden. Für Anregungen und konstruktive Kritik bin ich dankbar und werde sie gegebenenfalls an Lars weiterleiten.

– Jan Niggemann

Kapitel 2

TL;DR: Zuerst Lesen: Eine kurze Übersicht über Obnam

Vermutlich müssen Sie nur dieses Kapitel lesen.

Dieses Kapitel gibt eine kurze Einführung in die wichtigsten Teile von Obnam. Der Rest des Buches ist im Grunde eine ausführliche Version dieses Kapitels. Sie sollten dieses Kapitel zuerst lesen und dann einfach behaupten, Sie hätten auch den Rest gelesen – auf Cocktailpartys wird Sie jeder ehrfürchtig ansehen. Ich verspreche, dass auch niemand sonst den Rest des Buches gelesen haben wird, es gibt also kein Risiko erwischt zu werden.

2.1 Konfiguration

Obnam benötigt keine Konfigurationsdatei, Sie können alles mittels Optionen auf der Kommandozeile konfigurieren. Aber natürlich können Sie auch eine Konfigurationsdatei verwenden, speichern Sie sie unter `~/obnam.conf` und geben Sie ihr z.B. folgenden Inhalt:

```
[config]
repository = sftp://your.server/home/youruser/backups/
log = /home/liw/obnam.log
```

Die folgenden Beispiele gehen davon aus, dass Sie eine Konfigurationsdatei erstellt haben, so dass Sie die Optionen nicht jedes Mal wiederholen müssen.

Vermutlich wollen Sie jetzt die `log` Einstellung aktivieren, damit Sie im Falle eines Problems alle verfügbaren Informationen zur Problemlösung in der Protokolldatei finden.

2.2 Das erste Backup

Ihr erstes Backup wird recht groß sein und eine ganze Weile dauern. Ein langes Backup kann abstürzen, aber das ist kein Problem: Obnam erstellt alle Paar hundert Megabytes einen **Checkpoint**, von dem aus es abgebrochene Vorgänge wieder aufnehmen kann.

```
obnam backup $HOME
```

2.3 Inkrementelle Backups

Wenn Sie Ihr erstes Vollbackup gemacht haben (eventuell in mehreren Schritten), sichern Sie sämtliche Änderungen einfach indem Sie Obnam nochmal aufrufen:

```
obnam backup $HOME
```

Dies wird alle neuen und geänderten Dateien sichern. Es wird ebenfalls aufgezeichnet, welche Dateien seit dem letzten Backup gelöscht wurden.

Sie können Obnam so oft ausführen, wie Sie mögen. Es werden immer nur die Änderungen zum letzten Backup gesichert.

2.4 Mehrere Clients in einem Repository

Sie können mehrere Clients in ein einzelnes Repository sichern, indem Sie die Option `--client-name=<identifier>` beim Programmaufruf mitgeben. Die Sicherungssätze werden getrennt gespeichert, aber die Deduplizierung läuft über alle Sätze.

2.5 Alte Generationen entfernen

Irgendwann wird Ihr Backup Repository so groß, das Sie einige alte Generationen entfernen wollen. Diese Operation wird “forget” genannt:

```
obnam forget --keep=30d
```

Dieser Befehl behält ein Backup von jedem der letzten 30 Kalendertage, beginnend mit dem neuesten Backup (nicht der aktuellen Uhrzeit). Wenn Sie mehrmals am Tag gesichert haben, wird nur die letzte Generation des Tages behalten.

Alle Daten, die zu einer Generation gehören die behalten wird, bleiben im Repository. Jegliche Daten die ausschließlich einer Generation angehören die vergessen wird, wird entfernt.

2.6 Daten wieder herstellen

Hoffentlich werden Sie das nie benötigen, aber der einzige Grund warum man Backups anlegt ist, dass man eines Tages die Daten wieder herstellen kann, falls ein Unglück geschieht:

```
obnam restore --to=/var/tmp/my-recovery $HOME
```

Dieser Befehl wird Ihr gesamtes Home-Directory aus der letzten Generation nach `/var/tmp/my-recovery` wiederherstellen. Wenn Sie nur einzelne Verzeichnisse oder Dateien benötigen, können Sie diese stattdessen angeben:

```
obnam restore --to=/var/tmp/my-recover $HOME/Archive/receipts
```

Sollten Sie Sich nicht mehr an den Dateinamen erinnern, benutzen Sie zuerst `obnam ls`:

```
obnam ls > /var/tmp/my-recovery.list
```

Dies wird den Inhalt der Generation in einem Format ähnlich `ls -lAR` ausgeben. Speichern Sie den Inhalt in einer Datei und sehen Sie sie durch. (Das ist ein eher langsames Kommando, daher ist es komfortabler die Ausgabe in eine Datei zu speichern.)

2.7 Verschlüsselung nutzen

Obnam kann Backups mittels GnuPG verschlüsseln. Um dies einzuschalten müssen Sie einen PGP-Schlüssel besitzen (oder erzeugen) und Obnam dann erklären was es machen soll.

```
[config]
encrypt-with = CAFEBABE
```

In diesem Fall ist `CAFEBABE` die **Key-ID** Ihres Schlüssels, so wie GnuPG sie ausgibt. Im Moment benötigen Sie entweder `gpg-agent` oder etwas ähnliches, denn Obnam hat keine Möglichkeit nach dem Passwort zu fragen.

Wenn das geschafft ist, wird Obnam von da an automatisch ver- und entschlüsseln.

Wenn Sie Ihre Backups verschlüsseln, sollten Sie unbedingt auf anderem Weg eine Sicherheitskopie Ihres GPG Schlüssels anfertigen. Ohne den Schlüssel können Sie keine Dateien wieder herstellen, daher können Sie sich nicht auf das gleiche Obnam Backup verlassen um den Schlüssel zu sichern. Sichern Sie Ihren GPG-Schlüssel irgendwo anderes und stellen Sie sicher, eine ausreichend starke Passphrase zu verwenden, um offline Brute-Force Attacken stand zu halten. Denken Sie daran: Sollten Sie Ihren GPG Schlüssel verlieren oder nicht mehr darauf zugreifen können, wird Ihre gesamte Backup unbrauchbar.

Wenn Sie die Verschlüsselung erst nachträglich aktivieren, müssen Sie mit einem neuen Repository von vorne beginnen. Sie können keine verschlüsselten und unverschlüsselten Backups im gleichen Repository mischen.

(Es gibt eine Reihe von Befehlen für die Verwaltung von Obnams Verschlüsselung. Normalerweise benötigen Sie sie nicht, es sei denn mehrere Ihrer Clients teilen sich das gleiche Repository. In diesem Fall sollten Sie die manpage lesen.)

Kapitel 3

Sie wissen, dass Sie sollten...

Dieses Kapitel beschreibt Philosophie und Theorie von Backups. Es erläutert, warum Sie sichern sollten, verschiedene Konzepte rund um Backups, Dinge an die Sie bei der Einrichtung von Backups denken sollten und was langfristig zu tun ist (z.B. Verifizierung usw.). Außerdem werden einige Annahmen erläutert die Obnam macht, und welche Einschränkungen es gibt.

3.1 Warum sichern?

FIXME: Add some horror stories here about why backups are important. With references/links.

3.2 Backup-Konzepte

Dieser Abschnitt behandelt Kernkonzepte von Backups und legt einige Begriffe fest, die in diesem Buch verwendet werden.

Live-Daten sind die Daten, mit denen Sie arbeiten oder die Sie aufbewahren, die Dateien auf Ihrer Festplatte: Dokumente, die Sie schreiben, Fotos, die Sie speichern, der unvollendete Roman, von dem Sie wünschen das Sie ihn fertig schrieben.

Die meisten Live-Daten sind **kostbar**, Sie wären aufgeschmissen wenn sie verloren gingen. Einige Live-Daten sind es allerdings nicht, zum Beispiel Ihr Browser-Cache. Durch diese Unterscheidung können Sie die Datenmenge begrenzen, welche Sie sichern müssen – was die Kosten des Backups erheblich senken kann.

Ein **Backup** ist eine Ersatz-Kopie Ihrer Live-Daten. Sollten Sie einige oder alle Ihre Live-Daten verlieren, können Sie sie aus Ihrem Backup wieder herstellen (**restore**). Die Sicherungskopie ist notwendigerweise älter als Ihre Live-Daten, aber wenn Sie die Sicherung erst kürzlich erstellt haben werden Sie nicht viel verlieren.

Manchmal ist es sinnvoll, mehr als eine Backup-Kopie Ihrer Live-Daten zu haben. Eine Abfolge von Backups, die zu verschiedenen Zeiten erstellt wurden, wird **Sicherungsverlauf** genannt. Jede Kopie Ihrer Live-Daten in Ihrem Sicherungsverlauf ist eine **Generation**. Dadurch können Sie jetzt eine Datei wieder herstellen, die Sie schon vor langer Zeit gelöscht hatten, von der Sie

aber nicht wussten, dass Sie sie heute benötigen. Wenn Sie nur eine einzige Backup-Version vorhielten, könnten Sie die Datei nicht zurückbekommen. Aber wenn Sie, sagen wir, einen Monat lang ein tägliches Backup behalten, dann haben Sie einen Monat Zeit zu erkennen, das Sie eine Datei noch brauchen, bevor sie für immer verloren ist.

Der Ort an dem Ihre Backups gespeichert werden wird **Repository** genannt. Sie können viele Arten von **Backup-Medien** verwenden um Ihr Backup zu speichern: Festplatten, Bänder, optische Medien (DVD-R, DVD-RW, usw.), USB-Sticks, Online-Speicher und so weiter. Jedes Medium hat dabei verschiedene Eigenschaften: Größe, Geschwindigkeit, Komfort, Zuverlässigkeit, Preis. Diese Eigenschaften sollten Sie bei der Auswahl Ihrer Backup-Lösung in Erwägung ziehen.

Sie sollten mehrere Backup-Repositories benützen, eines davon sollte sich **off-site** ("außerhalb") befinden, also wo anders, als Ihre Computer in der Regel stehen. Andernfalls verlieren Sie alle Ihre Backups, wenn z.B. Ihr Haus abbrennt.

Sie müssen **überprüfen**, dass Ihre Backups auch wirklich funktionieren. Es wäre schade um den Aufwand für das Erstellen von Backups, wenn Sie im Fall der Fälle dann nicht in der Lage wären, Ihre Daten wieder herzustellen. Sie möchten vielleicht sogar Ihr **Desaster Recovery** testen. Tun Sie so als wäre all Ihr Computer-Zeug weg, bis auf die Backup-Medien. Können Sie alles wieder herstellen? Sie sollten dies von Zeit zu Zeit tun, um sicher zu stellen das Ihr Backup-System einsatzfähig ist.

Es gibt eine sehr große Auswahl an **Backup-Tools**. Sie können sehr einfach und händisch sein: Sie können Ihre Dateien alle Jubeljahre mit dem Dateimanager auf ein USB-Laufwerk kopieren. Sie können auch sehr komplex sein: Enterprise Backup-Produkte, die riesige Mengen an Geld kosten und mit mehrtägigen Schulungspaketen für Ihr Sysadmin-Team begleitet werden, und nur dann vernünftig funktionieren, wenn das gesamte Team vernünftig funktioniert.

Sie müssen eine **Backup-Strategie** definieren, die alles zusammenhält. Welche Live-Daten sind zu sichern, auf welchem Medium, mit welchen Tools, welche Art von Backup-History wollen Sie behalten, und wie überprüfen Sie, dass sie auch funktionieren.

3.3 Backup Strategien

Sie haben ein Backup-Repository eingerichtet und Sie erstellen seit einem Monat Tag für Tag ein Backup. Ihre Backup-History wird langsam lang genug, um nützlich zu sein. Können Sie Sich jetzt zufrieden zurück lehnen?

Willkommen in der Welt der Bedrohungsmodellierung. Backups sind Ihre Versicherung, sie mildern kleine und große Katastrophen, aber Katastrophen können auch Backups zustoßen. Wann sind Sie so sicher, dass keine Katastrophe Ihnen mehr schaden können?

Es gibt immer eine größeres Problem, das darauf wartet zu zuschlagen. Wenn Sie Ihre Daten auf eine USB-Laufwerk auf Ihrem Schreibtisch sichern, und jemand bricht ein und stiehlt Ihren PC und auch das USB-Laufwerk, dann haben Sie nichts gewonnen.

Eine Lösung wäre, zwei USB-Laufwerke zu verwenden. Eines bleibt bei Ihrem Computer, das andere wandert in ein Bankschließfach. Das ist recht sicher, außer wenn ein Erdbeben neben Ihrem Haus auch die Bank zerstört.

Eine Lösung wäre, dass Sie Online-Speicherplatz in einem anderen Land mieten. Das ist recht sicher, außer es gibt einen Fehler im Betriebssystem (das nicht nur Sie, sondern auch der Provider

benutzen) und Hacker dringen ein und löschen all Ihre Daten.

Eine Lösung wäre, dass Sie mit einem 3D Drucker Betonblöcke drucken, auf denen Ihre Daten mittels QR Code verewigt sind. Das ist recht sicher, außer ein Meteorit trifft die Erde und zerstört unsere gesamte Zivilisation.

Eine Lösung wäre, dass Sie Satelliten mit Kopien Ihrer Daten in stationäre Umlaufbahnen um alle 9 Planeten (Pluto ist auch ein Planet!) des Sonnensystems schicken. Das ist - auch wenn Sie vom Meteoriteneinschlag gestorben sind - recht sicher, außer die Sonne wird zur Supernova.

Es gibt immer eine noch größere Katastrophe. Sie müssen entscheiden, welche davon so wahrscheinlich sind, das Sie sie in Ihrem Szenario berücksichtigen und welche Kosten Sie für den Schutz dagegen akzeptieren.

Eine kurze Liste von Bedrohungsszenarien zum Bedenken:

- Was wäre, wenn Sie Ihren Computer verlieren?
- Was wäre, wenn Sie Ihr Heim und alles was drin ist verlieren?
- Was wäre, wenn die Gegend in welcher Sie leben zerstört wird?
- Was wäre, wenn Sie Ihr Land verlassen müssten?

Diese Fragen sind nicht abschließend, aber ein Anfang. Fragen Sie sich:

- Können Sie mit dem Verlust Ihrer Daten leben? Wenn Ihre Daten verloren sind, bedeutet es einen Verlust von Erinnerungen, oder ein Paar Unbequemlichkeiten im täglichen Leben, oder wird es Ihnen fast unmöglich, wieder normal zu leben und zu arbeiten? Welche Daten sind Ihnen am wichtigsten?
- Wie viel ist es Ihnen wert, Ihre Daten wieder zu bekommen, und wie schnell soll das passieren? Wie viel Geld und Mühe sind Sie bereit für das erste Backup zu investieren, und wie viel für weitere Backups? Und für die Wiederherstellung, wie viel sind Sie da bereit zu zahlen? Ist es besser für Sie, weniger für Backups auszugeben, auch wenn das Restore dann langsamer und teurer ist und mehr Aufwand bedeutet? Oder ist es umgekehrt?

Dieses Bedrohungsmodell berücksichtigt Unfälle und Naturkatastrophen. Modelle die Angriffe und Feinde berücksichtigen sind ähnlich aber auch etwas anders, und werden in der Nächsten Episode ein Thema sein, wenn es wieder heißt: Sie Abenteuer von Bac-Kup.

3.4 Backups und Sicherheit

Sie sind nicht der einzige, der sich um Ihre Daten sorgt. Eine Vielzahl von Regierungen, Unternehmen, Kriminellen und allzu neugierigen Schnüfflern sind wahrscheinlich ebenfalls interessiert (... und es ist manchmal schwer, diese auseinander zu halten). Sie könnten Beweise gegen Sie (er)finden wollen, Sie erpressen wollen, oder einfach nur neugierig darauf sein, was Sie mit Freunden besprechen.

Sie könnten Ihre Daten aus statistischen Gründen interessant finden und überhaupt kein Interesse an Ihnen persönlich haben. Oder sie könnten ausschließlich an Ihnen interessiert sein.

Statt Ihre Dateien und eMails zu lesen oder Ihre Fotos und Videos anzusehen könnten sie Interesse daran haben, Ihnen den Zugang dazu zu versperren, oder einfach Ihre Daten ganz zerstören. Sie könnten sogar Ihre Daten korrumpieren, indem sie Kinderpornographie in Ihr Foto-Archiv ablegen.

Sie schützen Ihren Computer so gut Sie können damit diese und andere schlimme Dinge nicht passieren. Ihre Sicherungen sollten Sie mit der gleichen Sorgfalt behandeln.

Wenn Sie auf ein USB-Laufwerk sichern, sollten Sie das Laufwerk verschlüsseln, genau wie auch Online-Speicher. Es gibt viele Arten von Verschlüsselung und ich bin nicht qualifiziert Ihnen Rat zu geben, aber jegliche gängige, moderne Verschlüsselung sollte ausreichen – außer für besonders entschlossene Angreifer.

Anstatt oder zusätzlich zur Verschlüsselung können Sie den physischen Zugang zu Ihren Backup-Medien absichern. Lagern Sie Ihr USB-Laufwerk z.B. in einem Safe oder Schließfach.

Die Vielzahl von Backups die Sie benötigen um sich gegen Erdbeben, Flutkatastrophen und marodierende Gangs dreirad-fahrender Clowns zu schützen, sind auch ein guter Schutz gegen Angreifer. Sie können Ihre Live-Daten und die Backups bei Ihnen zu Hause korrumpieren, aber vermutlich könnten Sie nicht an das USB-Laufwerk herankommen, das in Beton gegossen an einem geheimen Ort vergraben ist, den nur Sie kennen.

Auf der anderen Seite möchten oder müssen Sie vielleicht Anderen Zugriff auf Ihre gesicherten Daten geben. Wenn Sie zum Beispiel von der Clown Gang entführt wurden, sollte Ihr Partner in der Lage sein, Ihren MI6-Verbindungsmann zu kontaktieren, damit der Geheimdienst Sie retten kann. Den sicheren Zugang zu (Teilen) Ihres Backups herzustellen ist ein interessantes Problem für sich, für das es verschiedene Lösungen gibt: Geben Sie Ihrem Partner das Verschlüsselungspasswort, oder geben Sie es einem Freund dem Sie vertrauen, oder einem Anwalt. Sie könnten auch so etwas wie libgfshare verwenden, um die Schlüssel auf sichere Weise treuhänderisch zu hinterlegen.

3.5 Betrachtung von Backup-Medien

Dieser Abschnitt behandelt Backup-Medien, ihre unterschiedlichen Charakteristiken, und wie Sie für sich selbst etwas Passendes auswählen.

Es gibt viele verschiedene Speicher-Medien. Die wohl bekanntesten sind:

- verschiedene Arten von Magnetbändern
- Festplatte: interne oder externe, drehende Magnetscheiben oder SSDs oder USB-Sticks
- Optische Medien: CD, DVD, Blu-ray
- verschiedene Online-Speicher
- Papier

Die exotischeren und / oder ungebräuchlichen Dinge wie Microfiche überspringen wir mal.

Magnetbänder sind traditionell die wohl gebräuchlichste Form von Backup-Medien. Die Bänder an sich sind (je GB) preiswert, aber für das Laufwerk ist eine hohe Anfangsinvestition fällig. Vieles in der Backup-Terminologie ist Magnetbändern entlehnt, z.B. volle Sicherung / inkrementelle Sicherung. Obnam unterstützt keine Bandlaufwerke.

Festplatten sind eine gängige moderne Alternative zu Magnetbändern, besonders für alle, die nichts für ein Bandlaufwerk ausgeben möchten. Festplatten haben den Vorteil, das auf jedes Bit des Backups mit gleicher Geschwindigkeit zugegriffen werden kann, was das Auffinden einer alten Datei schneller und einfacher macht. So werden auch **Snapshots** möglich, die von Obnam benutzt werden.

Verschiedene Typen Festplatten haben verschiedene Eigenschaften z.B. was Verlässlichkeit, Geschwindigkeit und Preis angeht, und diese Eigenschaften variieren häufig von Woche zu Woche und Jahr zu Jahr. Wir wollen nicht ins Detail aller möglichen Eigenschaften gehen. Aus Sicht von Obnam ist alles, das sich wie eine Festplatte verhält (drehender Rost, SSD, USB Flash-Speicher oder Online-Speicher) brauchbar um Backups abzulegen, Hauptsache es ist wiederbeschreibbar.

Optische Medien, besonders welche die nur einmal beschrieben werden können, können für Backups benutzt werden. Sie eignen sich aber eher für Komplettsicherungen, die lange Zeit aufbewahrt werden als für aktiv genutzte Backup-Repositories. Alternativ können Sie als eine Art Bandsicherung verwendet werden, in der jedes Band nur einmal beschrieben wird. Obnam unterstützt keine optischen Medien als Backup-Speicher.

Papier funktioniert ebenso gut für die Archivierung, allerdings nur für kleine Datenmengen. Trotzdem kann ein Papier-Backup, das mit Archivier-Tinte auf gutem Papier gedruckt wurde, Jahrzehnte oder sogar Jahrhunderte halten. Das macht es zu einer guten Option für kleine, aber wichtige Datenmengen. Beispiele wären persönliche Finanz-Unterlagen, geheime Chiffrierschlüssel und Liebesbriefe Ihres Partners. Diese können ganz normal gedruckt werden (am Besten in einer Schriftart die einfach zu OCRen ist), oder als zweidimensionale Barcodes (z.B. QR). Obnam unterstützt auch dies nicht.

Obnam unterstützt ausschließlich Festplatten und alles das sich wie eine (beschreibbare) Festplatte benimmt, z.B. Online-Speicher – erstaunlicherweise scheint das den Meisten zu reichen.

3.6 Glossar

- **Backup**: Eine getrennte Sicherheitskopie Ihrer Live-Daten, die intakt bleibt, auch wenn das Original zerstört, gelöscht oder ungewollt geändert wird.
- **corruption**: Unerwünschte Veränderung Ihrer (Backup-)Daten
- **disaster recovery**: Was Sie tun wenn etwas schief lief
- **full backup**: Eine komplette Sicherung Ihrer wertvollen Livedaten
- **generation**: Ein Backup in einer Serie von Backups der selben Livedaten, eine historische Sicht auf selbige
- **history**: Alle generations
- **incremental backup**: Eine Sicherung jeglicher Änderungen (Neue Daten, geänderte Daten, gelöschte Daten) im Vergleich zu einer früheren Generation (entweder die vorhergehende Vollsicherung oder eine vorhergehende inkrementelle Sicherung) Normalerweise kann eine Vollsicherung nur dann entfernt werden, wenn auch alle inkrementellen Sicherungen die darauf aufbauen mitgelöscht werden)
- **live data**: Alle Daten die Sie haben
- **local backup**: Eine Sicherung, deren Repository physisch in der Nähe der Livedaten gespeichert ist.
- **Medium**, **Backup-Medium**, **Speichermedium**: Das, wo ein Backup Repository drauf gespeichert wird
- **off-site backup**: Eine Sicherung, deren Repository physisch weit weg von den Livedaten ist
- **precious data**: Alle Daten, die Ihnen etwas bedeuten, vgl. “live data”
- **Repository**: Der Ort an dem Backups gespeichert werden
- **restore**: Daten aus einem Repository wieder herstellen

- **root**, **backup root**: Ein Verzeichnis das gesichert werden soll, inkl. aller Dateien und Verzeichnisse darunter
- **Snapshot-Sicherung**: Eine Alternative zu kompletter/inkrementeller Sicherung, bei der jede Generation effektiv ein Vollbackup der Livedaten darstellt, das einzeln wieder hergestellt oder gelöscht werden kann
- **strategy**, **backup strategy**: Ein Plan um Ihre Daten abzusichern, sogar wenn die Dinosaurier in Raumschiffen zurückkommen um die Welt zu übernehmen, jetzt wo die Eiszeit vorüber ist
- **verification**: Sicherstellen das ein Backup-System auch funktioniert und Daten aus der Sicherung wieder hergestellt werden können und das die Sicherung nicht beschädigt ist

Kapitel 4

Installation

Dieses Kapitel beschreibt die Installation von Obnam. Es ist bisher keine sehr umfangreiche Anleitung. Insbesondere bietet es im Moment nur Informationen für Debian Benutzer. Hinweise für andere Systeme sind sehr willkommen.

4.1 Debian

Obnam auf einem Debian System zu installieren ist am einfachsten. Wenn Sie `Wheezy` oder ein neueres Release einsetzen ist Obnam bereits enthalten, Sie können es mit einem einfachen Befehl installieren:

```
apt-get install obnam
```

Auf der Webseite des Autors könnte es eine neuere Version geben, der Rest dieses Kapitels beschreibt wie das Repository des Autors eingebunden und dann von dort installiert wird.

Fügen Sie die folgende Zeile Ihrer `/etc/apt/sources.list` hinzu:

```
deb http://code.liw.fi/debian wheezy main
```

Führen Sie dann als root diese Befehle aus:

- `apt-get update`
- `apt-get install obnam`

Sie werden eine Fehlermeldung über fehlende PGP Schlüssel erhalten, mit welchen das Archiv signiert ist. Sie können diese Meldung entweder ignorieren, oder – nach geeigneter Prüfung – den Schlüssel von der Webseite http://code.liw.fi/apt.asc Ihren Schlüsseln hinzufügen.

4.2 Andere Systeme

Auf anderen Systemen müssen Sie Obnam aus den Quellen installieren. Hinweise dazu finden Sie in der Datei `README` in den Quellen.

Kapitel 5

Sichern

Dieses Kapitel beschreibt verschiedene Aspekte der Erstellung von Backups mit Obnam.

5.1 Ihr erstes Backup

OK, dann lasst uns mal ein Backup machen! Um den Beispielen zu folgen benötigen Sie Live-Daten, die Sie sichern können. Diese Beispiele benutzen Dateinamen, die Sie an Ihre eigenen Dateinamen anpassen müssen. Die Beispiele gehen davon aus, dass Ihr Home-Verzeichnis `/home/tomjon` ist und Sie Dokumente in einem weiteren Verzeichnis names `Documents` in Ihrem Home-Verzeichnis haben. Weiterhin wird davon ausgegangen das Sie ein USB-Laufwerk unter `/media/backups` gemounted haben und das Sie das Verzeichnis `tomjon-repo` auf diesem USB-Laufwerk als Backup-Repository benutzen.

Diesen Annahmen folgend sichern Sie Ihre Dokumente so:

```
obnam backup -r /media/backups/tomjon-repo ~/Documents
```

Das ist alles. Wenn Sie viele Dokumente haben dauert es eine Weile, aber am Ende sieht es dann ungefähr so aus:

```
Backed up 11 files (of 11 found), uploaded 97.7 KiB in 0s at 647.2 KiB/s average speed
```

Das bedeutet, das Obnam insgesamt 11 Dateien gefunden hat, von denen alle 11 gesichert wurden. Die Dateien waren zusammen ungefähr 100 Kilobyte groß und die Upload Geschwindigkeit für diese Daten war über 600 Kilobytes pro Sekunde. Für die Eindeutigkeit sind Einheiten mit IEC Präfixen versehen (Basis 2), weitere Informationen finden Sie unter [Wikipedia on kibibytes].

Ihr erster Sicherungslauf sollte eher wenige Daten enthalten, so können Sie prüfen das alle Einstellungen korrekt sind ohne lange zu warten. Anstatt Ihres gesamten Home-Verzeichnisses könnten Sie mit einem kleineren Unterordner beginnen.

5.2 Ihr zweites Backup

Nach Ihrem ersten Backup möchten Sie irgendwann einmal ein Weiteres erstellen. Das tun Sie auf die gleiche Weise:

```
obnam backup -r /media/backups/tomjon-repo ~/Documents
```

Beachten Sie das Sie Obnam nicht mitteilen müssen ob Sie ein Vollbackup oder ein inkrementelles Backup erstellen wollen. Obnam sorgt dafür das jede Generation ein Snapshot der Daten zur Zeit des Backups ist. Somit wird nicht zwischen Vollbackup und inkrementellem Backup unterschieden. Jede Generation ist eine vollständige Sicherung, was aber nicht heißt, dass jede einzelne Generation sämtliche Daten separat vorhält. Obnam sorgt dafür das mit jeder neuen Generation nur die Daten gesichert werden, die bisher nicht im Repository waren. Obnam sucht die Daten in jeder Datei und jeder vorausgehenden Generation aller Clients, die sich das Repository teilen.

Wir kommen später auf das Thema zurück, wie Generationen gelöscht werden können, Sie werden dabei sehen das Obnam jede beliebige Generation löschen kann, auch wenn sie sich mit anderen Generationen Daten teilt. Die anderen Generationen werden dabei natürlich keinerlei Daten verlieren.

Nachdem Sie das zweite Backup erstellt haben, können Sie die Generationen ansehen:

```
$ obnam generations -r /media/backups/tomjon-repo
2   2014-02-05 23:13:50 .. 2014-02-05 23:13:50 (14 files, 100000 bytes)
5   2014-02-05 23:42:08 .. 2014-02-05 23:42:08 (14 files, 100000 bytes)
```

Wir sehen zwei Generationen, die die Kennungen 2 und 5 haben. Die Bezeichner der Generationen sind nicht unbedingt eine einfache Folge wie 1, 2, 3. Dies liegt daran wie einige der internen Datenstrukturen in Obnam umgesetzt werden, und in keinster Weise daran das der Autor Spaß daran hat, Menschen zu verwirren.

Die beiden Zeitstempel zeigen wenn der Backup-Lauf begann und wann er endete. Darüber hinaus wird für jede Generation die Anzahl der Dateien in dieser Generation ausgegeben (insgesamt, nicht nur neue oder geänderte Dateien), und die summierte Größe der Dateien.

5.3 Auswählen was zu sichern ist – und was nicht

Obnam muss wissen, was gesichert werden soll. Dazu übergeben Sie eine Liste von Verzeichnissen, die backup roots genannt werden. Bisher haben wir in den Beispielen dieses Kapitels das Verzeichnis `~/Documents` als backup root benutzt (das ist das Verzeichnis `Documents` in Ihrem Home Verzeichnis). Es darf aber auch mehrere backup roots geben:

```
obnam -r /media/backups/tomjon-repo ~/Documents ~/Photos
```

Alles in den backup root Verzeichnissen wird gesichert – außer es ist explizit ausgeschlossen. Es gibt mehrere Möglichkeiten um etwas von Backups auszuschließen:

- Die `--exclude` Option verwendet reguläre Ausdrücke, die dem kompletten Pfadnamen jeder Datei bzw. jeden Verzeichnisses entsprechen. Wenn der Pfadname übereinstimmt, werden die Datei oder das Verzeichnis nicht gesichert; Obnam tut so als wären die Dateien / das Verzeichnis nicht vorhanden. Wird ein Verzeichnis ausgeschlossen, dann werden alle

Dateien und Unterverzeichnisse ebenfalls ausgeschlossen. Um beispielsweise alle MP3-Dateien auszuschließen, verwenden Sie (`--exclude='\.mp3$'`).

- Die `--exclude-caches` Option schließt Verzeichnisse aus, die eine spezielle Datei namens `CACHEDIR.TAG` enthalten, die mit einer bestimmten Byte-Sequenz anfangen muss. Eine solche Datei könnte z.B. im Cache-Verzeichnis Ihre Browsers abgelegt werden. Die Daten in diesen Verzeichnissen sind meist nicht wichtig und brauchen nicht gesichert zu werden und es ist einfacher, das gesamte Verzeichnis mittels der Spezialdatei für den Ausschluss zu markieren, als einen regulären Ausdruck für `--exclude` zu schreiben.
- Die `--one-file-system` Option schließt alle mount points und den Inhalt der gemounteten Dateisysteme aus. Das ist praktisch um z.B. virtuelle Dateisysteme wie `/proc`, remote Dateisysteme (z.B. NFS-mounts) und mittels `obnam mount` eingebundene Obnam Repositories auszuschließen (letzteres behandeln wir im nächsten Kapitel).

Generell ist es besser zu viel zu sichern als zu wenig. Sie sollten auch genau wissen, was gesichert wird und was nicht. Die Option `--pretend` bewirkt das Obnam ein Backup anfertigt, das Repository dabei aber nicht verändert, es ist also schnell durchgelaufen. So können Sie sehen was gesichert würde und die excludes entsprechend anpassen.

5.4 Backups "außer Haus" speichern

Vermutlich möchten Sie mindestens ein Backup auf einem entfernten Rechner "offsite" (außer Haus) speichern. Obnam kann Backups mittels SFTP (Teil von ssh) über das Netzwerk machen. Sie benötigen folgendes um dies zu tun:

- Ein **Server**, auf den Sie über SFTP zugreifen können. Dies könnte ein eigener Server sein, ein gemieteter virtueller Server ("VPS"), oder etwas Ähnliches wie z.B. der Rechner eines Freundes, der Ihnen Plattenplatz auf seinem System zur Verfügung stellt (und umgekehrt!).

- Ein **ssh key** ("Schlüssel"), um sich beim Server anzumelden. Ein Login mittels Passwort wird zur Zeit von Obnam nicht unterstützt.

- Genügend Speicherplatz auf dem Server, um Ihre Backups vorzuhalten.

Um auf den Server zuzugreifen verwendet Obnam lediglich den SFTP-Teil der ssh-Verbindung. Um zu prüfen ob das funktioniert können Sie diesen Befehl ausführen:

```
sftp USER@SERVER
```

Passen Sie `USER@SERVER` entsprechend Ihren Zugangsdaten an. Sie sollten die Meldung `Connected to SERVER` sehen und in der Lage sein, `ls -la` aufzurufen um den Verzeichnisinhalt der Gegenseite auszugeben.

Sobald all das richtig eingerichtet ist, kann Obnam den Server über eine SFTP-URL als Repository benutzen und so auf den Server sichern.

```
obnam -r sftp://USER@SERVER/~/mein-wertvolles-backup
```

Einzelheiten über SFTP-URLs finden Sie im nächsten Abschnitt.

5.5 URL Syntax

Wann immer Obnam eine URL akzeptiert, kann immer ein lokaler Pfadname oder eine SFTP-URL angegeben werden. SFTP-URLs haben die Form:

```
sftp://[user@]domain[:port]/path
```

Dabei ist `domain` ein normaler Domainname der einen Server bezeichnet, `user` der Benutzername auf diesen Server, `port` ein optionaler numerischer port und `path` ein gültiger Pfad auf dem Server.

Entgegen dem SFTP-Standard (aber wie bei bzr) ist der Pfad absolut, außer wenn er mit `/~/` beginnt (dann ist er relativ zum home-Verzeichnis des Benutzers auf dem Server).

Beispiele:

- `sftp://liw@backups.pieni.net/~/backup-repo` ist das Verzeichnis `backup-repo` im home-Verzeichnis des Benutzers `liw` auf dem Server `backups.pieni.net`. Beachten Sie das wir nicht den absoluten Pfad des Home-Verzeichnisses benötigen.
- `sftp://root@my.server.example.com/home` ist das Verzeichnis `/home` (absoluter Pfad) auf dem Server `my.server.example.com`, wobei der Benutzername `root` verwendet wird, um auf den Server zuzugreifen.
- `sftp://foo.example.com:12765/anti-clown-society` ist das Verzeichnis `/anti-clown-society` auf dem Server `foo.example.com`, Zugriff über Port 12765.

Sie können SFTP URLs für das Repository und die Live-Daten (`--root`)verwenden. Aufgrund von Beschränkungen im SFTP-Protokoll und seiner Implementierung in der paramiko-Bibliothek funktionieren einige Dinge beim Zugriff auf Live-Daten über SFTP nicht so gut.

Dabei besonders hervorzuheben wäre die Handhabung von Hardlinks, weswegen die URL beim Zugriff auf Live-Daten nicht mit `/~/` enden sollte. Fügen Sie in diesem besonderen Fall einen Punkt am Ende hinzu.

5.6 Pull Backups

Obnam kann die Live-Daten auch über SFTP statt über das lokale Dateisystem sichern. Das heißt Sie können Obnam auf, sagen wir, Ihrem Desktop zur Sicherung Ihres Servers oder auf Ihrem Laptop zur Sicherung Ihres Telefons laufen lassen (vorausgesetzt, Sie bekommen den SSH-Server auf dem Telefon zum laufen). Manchmal ist es nicht möglich, Obnam auf der Maschine zu installieren, auf der die Live-Daten liegen. Dann ist es sinnvoll, stattdessen ein **Pull Backup** zu machen: Sie lassen Obnam auf einem anderen Rechner laufen und lesen die Live-Daten über das SFTP-Protokoll.

Um dies zu tun spezifizieren Sie die Backup-Quelle (`root` im config file oder als Kommandozeilen-Argument zu `obnam backup`) mittels einer SFTP-URL. Sie sollten auch den Client-Namen explizit angeben. Ansonsten wird Obnam den Hostnamen des Rechners verwenden auf dem es läuft. Dies kann höchst verwirrend sein: Angenommen der Client-Name ist `my-laptop` und der des Servers ist `down-with-clowns`, dann speichert Obnam die Backups als ob die Daten zu `my-laptop` gehörten.

Wenn Sie Ihr Laptop ebenfalls in das gleiche Backup-Repository sichern, wird es noch schlimmer. Obnam speichert dann sowohl Server- und Laptop-Daten mit dem gleichen Client-Namen, was zu viel Verwirrung für alle führt.

Beispiel:

```
obnam backup -r /mnt/backups sftp://server.example.com/home \
    --client-name=server.example.com3
```

5.7 Konfigurationsdateien: Eine kurze Einführung

Zu diesem Zeitpunkt werden Sie vielleicht bemerkt haben, dass Obnam eine ganze Reihe von konfigurierbare Einstellungen hat, die Sie auf vielfältige Art verändern können. Auf der Kommandozeile ist das immer möglich, aber dann wird das Kommando doch recht lang. Stattdessen könnten Sie auch eine Konfigurationsdatei verwenden.

Jede Option die Obnam kennt, kann auch in einer Konfigurationsdatei verwendet werden. Später in diesem Handbuch gibt es ein ganzes Kapitel, das alle Details der Konfigurationsdateien und verschiedenen Einstellungen, die Sie verwenden können, umfasst. Hier geben wir erstmal eine kurze Einführung.

Eine Obnam Konfigurationsdatei sieht so aus:

```
[config]
repository = /media/backup/tomjon-repo
root = /home/liw/Documents, /home/liw/Photos
exclude = \.mp3$
exclude-caches = yes
one-file-system = no
```

Diese Form der Konfigurationsdatei ist als “INI file” bekannt, vielen z.B. von Microsoft Windows. Jede Obnam-Option wird in den Abschnitt `[config]` geschrieben, und jede Einstellung hat den gleichen Namen wie die Kommandozeilen-Option (ohne die Doppel-Minuszeichen). Demnach wird `--exclude` auf der Kommandozeile und `exclude` im Config-File verwendet.

Einige Optionen können mehrere Werte annehmen, z.B. `exclude` und `root`, die Werte werde mittels Komma getrennt. Wenn die Anzahl der Werte zu groß wird können Sie sie über mehrere Zeilen verteilen; die zweite und weitere Zeilen müssen dann mit Leerzeichen oder TAB eingerückt werden.

Jetzt sollten Sie genug für den Anfang haben, Details finden Sie im Kapitel “Obnam Konfigurationsdateien und Einstellungen”.

5.8 Wenn Ihre wertvollen Daten sehr groß sind

Wenn Ihre wertvollen Daten sehr groß ist, kann die erste Sicherung sehr lange dauern. Dito, wenn Sie viele neue wertvolle Daten in eine späteres Backup aufnehmen. In diesen Fällen müssen Sie sehr geduldig sein und dem Backup Zeit lassen. Oder Sie können klein beginnen und dann nach und nach Sicherungen hinzufügen.

Die Option “Geduld” ist einfach: Sie lassen Obnam alles sichern, starten die Sicherung und warten, bis es fertig ist, auch wenn das Stunden oder Tage dauert. Sollte die Sicherung vorzeitig abbrechen, z. B. wegen einer ausgefallenen Netzwerkverbindung, brauchen Sie Dank Obnams Checkpoint-Unterstützung nicht von Grund auf neu beginnen. Jedes Gigabyte oder so (standardmäßig) erzeugt Obnam eine Checkpoint Generation. Wenn die Sicherung später abstürzt, können Sie Obnam einfach erneut ausführen und es wird beim letzten Checkpoint weitermachen. Das alles passiert vollautomatisch. Wie oft die Checkpoints erstellt werden sollen können Sie in der Option `--checkpoint` verändern.

Falls Obnam nicht bis zum Ende durchläuft und Sie es neu starten müssen, dann beginnt der Scan der Quelldateien wieder von vorne. Die Checkpoint Generationen enthalten nicht genügend Zustandsinformationen über die gescannten Quelldateien, um Obnam einfach bei der aktuellen Datei im Checkpoint weiter laufen zu lassen: Es wäre ein sehr komplizierter Zustand, der außerdem leicht von Dateisystem-Änderungen invalidiert würde. Stattdessen scannt Obnam erneut alle Dateien, wobei die meisten hoffentlich schon in einer Checkpoint Generation enthalten sind und seitdem nicht geändert wurden, so dass der Scan recht schnell gehen sollte.

Das Problem mit der Option “Geduld” ist, dass Ihre wichtigsten Daten nicht gesichert werden, während alle Ihre großen, aber weniger wertvollen Daten gesichert werden. Zum Beispiel können Sie eine große Menge an heruntergeladenen Videos von Konferenz-Präsentationen haben, was schön ist, aber nicht enorm wichtig. Während diejenigen gesichert werden, bleiben Ihre eigenen Dokumente ungesichert.

Sie können dies umgehen, indem Sie zunächst alles außer Ihren wertvollsten Daten vom Backup ausschließen. Wenn diese dann gesichert sind, können Sie nach und nach die Ausschlüsse reduzieren, bis Sie alles gesichert haben. Zum Beispiel könnte Ihr erstes Backup folgende Konfiguration haben:

```
obnam backup -r /media/backups/tomjon-repo ~ \
    --exclude ~/Downloads
```

So werden alle Downloads ausgeschlossen. Im nächsten Lauf schließen Sie nur noch Video-Dateien (mp4) aus:

```
obnam backup -r /media/backups/tomjon-repo ~ \
    --exclude ~/Downloads/'.*\.mp4$'
```

Dann reduzieren Sie den Ausschluss auf einzelne Videos, deren Namen mit einem bestimmten Buchstaben beginnen:

```
obnam backup -r /media/backups/tomjon-repo ~ \
    --exclude ~/Downloads/'[^b-zB-Z].*\.mp4$'
```

Reduzieren Sie die Ausschlüsse immer weiter bis alle Videos gesichert sind.

5.9 Deduplizierung

Obnam de-dupliziert die Daten die es sichert über alle Dateien in allen Generationen für alle Clients, die sich das Repository teilen. Dies geschieht durch Aufteilen der Dateidaten in “chunks” genannte Teile. Jedes Mal wenn Obnam eine Datei liest und ein chunk zusammenkommt, schaut es im Repository nach, ob ein identischer chunk bereits vorhanden ist. Wenn ja muss Obnam den chunk nicht hochladen, was Platz, Bandbreite und Zeit spart.

Deduplizierung in Obnam ist in verschiedenen Situationen hilfreich:

- Wenn Sie zwei identische Dateien haben. Sie haben vielleicht verschiedene Namen und liegen in verschiedenen Verzeichnissen, enthalten aber die selben Daten.
- Wenn Dateien wachsen, aber neue Daten nur am Ende angehängt werden, was z.B. für Logfiles typisch ist. Falls die ersten chunks unverändert sind, müssen nur die neuen Daten gesichert werden.
- Wenn eine Datei oder ein Verzeichnis umbenannt oder verschoben wird. Wenn Sie meinen das der englische Begriff `Photos` für das Verzeichnis unpassend ist und Sie stattdessen lieber das finnische `Valokuvat` möchten, können Sie das Verzeichnis einfach umbenennen. Ohne Deduplizierung müssen Sie dann aber alle Fotos nochmal sichern.
- Wenn ein Team mit den gleichen Daten arbeitet und demnach jeder eine Kopie der Daten vorhält, muss das Repository statt eine Kopie pro Team-Mitglied nur eine einzelne Kopie der Daten vorhalten.

Die Deduplizierung in Obnam ist nicht perfekt. Die Granularität des Findens doppelter Daten ist recht grob (vgl. Option `--chunk-size`) , daher kann Obnam oft keine Überschneidungen finden, wenn die Änderungen nur klein sind.

In den folgenden Fällen ist Deduplizierung nicht sinnvoll:

- Änderungen an Dateien, indem größere Teile innerhalb der Datei verschoben werden. Die (jetzige) Implementierung der Deduplizierung basiert auf nicht überlappenden Teilen (chunks), die vom Anfang der Datei an gebildet werden. Wenn Daten eingefügt werden, kann Obnam nicht bemerken, dass der Rest der Daten nur etwas “nach hinten” gerutscht ist. Das könnte zum Beispiel in Disk- oder ISO-Images der Fall sein.

- Dateien mit doppelten Daten, die aber nicht an einer Chunk-Grenze liegen. Dies könnte zum Beispiel der Fall sein, wenn eine eMail mit großem Anhang an mehrere Empfänger (=Mailboxen auf dem Host) gesendet wird. Jeder Empfänger erhält verschiedene `Received` Kopfzeilen, was den Mailbody und die Anhänge mehr oder weniger verschiebt. Aus diesem Grund kann Obnam die doppelten Daten nicht erkennen und dementsprechend nicht deduplizieren.

- Wenn Daten komprimiert vorliegen, zum Beispiel als `.zip` oder `.tar.xz`. Obnam weiss nichts von der Kompression und “sieht” nur die komprimierten Daten, die naturgemäß nicht deduplizierbar sind.

Künftige Versionen werden hoffentlich einen besseren Deduplizierungsalgorithmus enthalten. Sollten Sie diesen optimistischen Absatz in einer 2017 oder später erschienenen Version von Obnam finden, informieren Sie bitte die Maintainer. Vielen Dank.

5.10 Deduplizierung und Sicherheit gegen Prüfsummen-Kollisionen

Dieses Thema ist ein bisschen beängstigend, aber es wäre unehrlich, es überhaupt nicht zu behandeln. Sie dürfen gern später auf diesen Abschnitt zurück kommen.

Obnam verwendet den MD5-Algorithmus zur Erkennung von doppelten chunks. MD5 hat den Ruf, unsicher zu sein: Menschen haben Dateien gebaut die zwar unterschiedlich sind, aber zu der

gleichen MD5-Prüfsumme führen. Es stimmt – MD5 ist nicht für sicherheitskritische Anwendungen geeignet.

Jeder Prüfsummen-Algorithmus kann Kollisionen haben. Obnam auf, sagen wir, SHA1, SHA2, oder den neuen SHA3 Algorithmus umzustellen, würde nicht die Möglichkeit von Kollisionen ausschließen. Es reduzierte die Chance von zufälligen Kollisionen, aber die Chance ist bereits so klein, dass sie mit MD5 vernachlässigt werden kann. Oder, anders ausgedrückt: Wenn Sie über zufällige MD5-Kollisionen besorgt sind, sollten Sie ebenfalls über versehentliche SHA1, SHA2 oder SHA3 Kollisionen besorgt sein.

Abgesehen von zufälligen Kollisionen gibt es zwei Fälle, in denen Sorgen um Prüfsummen-Kollisionen angebracht sind (unabhängig von Algorithmus).

Erstens: Wenn Sie einen Gegner haben der Ihre gesicherten Daten korrumpieren möchte, kann er einige der gesicherten Daten mit anderen Daten mit der gleichen Prüfsumme austauschen. Auf diese Weise werden Ihre Daten beschädigt ohne das Obnam es merkt und Sie können sie nicht wieder herstellen.

Zweitens: Wenn Sie Prüfsummen-Kollisionen erforschen, haben Sie wahrscheinlich Dateien, deren Prüfsummen kollidieren. In diesem Fall werden Sie nach einer Katastrophe die Daten in ihrem Ursprungszustand wieder herstellen wollen, ohne das Obnam eine mit der anderen verwechselt.

Um mit diesen Situationen umzugehen besitzt Obnam drei Deduplizierungs-Modi, die Sie mit der `--deduplicate` Einstellung steuern:

- Der Standard-Modus `fatalist` geht davon aus, das keine Kollisionen auftreten. Das ist für die Meisten Anwender ein vernünftiger Kompromiss zwischen Geschwindigkeit, Schutz und Sicherheit.
- Der Modus `verify` geht nimmt an das Kollisionen passieren und stellt sicher, das bereits gesicherte chunks auch wirklich mit dem zu sichernden chunk übereinstimmen, indem die eigentlichen Daten verglichen werden. Um das zu tun muss der chunk aus dem Repository geladen werden, was recht langsam sein könne, da die Prüfsummen oft passen. Dieser Modus macht Sinn wenn Sie deduplizieren wollen und sehr schnellen Zugriff auf das Repository haben, z.B. wenn es auf einer lokalen Platte liegt.
- Mit `never` wird die Deduplizierung komplett abgeschaltet. Wenn Sie keine Deduplizierung benötigen oder Kollisionen fürchten, ist dies der richtige Modus für Sie.

Leider gibt es keine Deduplizierung die unverwundbar gegen Kollisionen ist und dabei auch noch schnell arbeitet, wenn der Zugriff auf das Repository langsam ist. Der einzige Weg um dagegen geschützt zu sein ist, die Daten zu vergleichen. Wenn das Herunterladen der Daten aus dem Repository langsam ist, dann wird der Vergleich natürlich erhebliche Zeit in Anspruch nehmen.

5.11 Locking

Mehrere Clients können sich ein Repository teilen. Um zu verhindern das sie sich dabei gegenseitig auf die Füße treten, sperren sie Teile des Repositories während des Vorgangs. Im Kapitel “Mehrere Clients in einem Repository” finden Sie Details.

Wenn Obnam abrupt abbricht kann die Sperre bestehen bleiben, auch wenn es nur einen einzelnen Client im Repository gibt. Neue Backups sind so nicht mehr möglich. Der Abbruch kann z.B.

durch eine abgebrochene Netzwerkverbindung oder aufgrund eines Fehlers in Obnam passieren, genau so wenn Obnam vom Benutzer unterbrochen wird, bevor es fertig ist.

Die Befehl `force-lock` kann diese Situation bereinigen, aber das ist gefährlich. Wenn Sie eine Sperre aufheben, die von egal welcher Obnam-Instanz auf egal welchem Client, der das Repository aktiv benutzt, manuell aufheben, dann wird Obnam wahrscheinlich ins Stolpern geraten. Im Extremfall kann sogar das Repository beschädigt werden. Seien Sie also vorsichtig.

Wenn Sie entschieden haben das es kein Problem darstellt, wird die Sperre wie folgt aufgehoben:

```
obnam -r /media/backups/tomjon-repo force-lock
```

Zum jetzigen Zeitpunkt ist es nicht möglich Eine Sperre manuell aufzuheben, die zu einem einzelnen Client gehört.

5.12 Konsistenz der Live-Daten

Das Erstellen einer Sicherungskopie kann eine ganze Weile dauern, und während das Backup läuft, könnte sich das Dateisystem ändern. Dies führt dazu, das der Snapshot, den Obnam als Generation sichert, in sich inkonsistent ist. Ein Beispiel: Sie haben zwei Dateien, A und B, die synchron gehalten werden müssen. Sie machen ein Backup, währenddessen Sie zuerst A und dann B ändern. Unglücklicherweise sichert Obnam File A bevor Sie Ihre Änderungen speichern und File B, nachdem Sie die Änderungen gespeichert haben. Die Sicherungsgeneration hat nun Versionen von A und B, die nicht synchron sind. Das ist schlecht.

Es gibt mehrere Möglichkeiten, mit diesem Problem umzugehen:

- Der Logical Volume Manager (LVM) ermöglichst Snapshots. Sie können Ihre Backups so einrichten, das zuerst ein Snapshot von jedem logischen Volume gemacht wird, das gesichert werden soll. Dann machen Sie das Backup und löschen danach den Snapshot wieder. Niemand kann die Daten im Snapshot verändern, während trotzdem normal weiter gearbeitet werden kann, während das Backup läuft.
- Ähnlich können Sie auch mit btrfs und seinen subvolumes vorgehen.
- Sie können das System herunterfahren und im Einzelbenutzermodus neu starten, das Backup machen, und dann wieder im Mehrbenutzermodus starten. Das ist nicht der schönste Weg, aber der sicherste, um ein konsistentes Backup zu erhalten.

Beachten Sie, dass Snapshots auf Dateisystemebene nicht wirklich eine konsistente Sicht auf die Live-Daten garantieren können. Eine Anwendung kann mitten im Schreiben einer Datei oder Gruppe von Dateien sein, wenn der Snapshot gemacht wird. Das ist gewissermaßen ein Bug in der Anwendung, aber das zu wissen hilft Ihnen nicht besser zu schlafen.

In der Regel haben die meisten Systeme aber ausreichend Leerlaufzeit, um ein konsistentes Backup während dieser Zeit zu erstellen. Zum Beispiel kann das Backup auf einem Laptop ausgeführt werden, während der Benutzer anderswo ist und nicht aktiv mit der Maschine arbeitet.

Wenn irgend möglich sollte Ihr Backup-Programm überprüfen, ob die Daten einer Backup-Generation in sich konsistent sind. Ansonsten werden Sie entweder darauf vertrauen müssen das die Anwendungen, die Sie verwenden, nicht zu buggy sind.

Wenn Sie diesen Abschnitt nicht verstanden haben: Keine Sorge, seien Sie glücklich und schlafen Sie gut.

Kapitel 6

Wiederherstellen von Backups

Das Schlimmste ist passiert! Ihre Katze hat das Katzenklo und Ihre Festplatte verwechselt! Ihre Ziege hat Ihr allerwichtigstes Dokument gelöscht! Wehe Dir!

Bleiben wir ruhig. Genau dafür haben wir ja Backups. Es gibt keinen Grund für Ausrufezeichen. Atmen Sie tief ein, trinken Sie eine Tasse Tee, und alles wird gut.

Es gibt zwei verschiedene Ansätze für die Wiederherstellung von Daten mit Obnam. Einer stützt sich auf das FUSE-Dateisystem. Dieses sehr schöne Stück Technik macht es möglich, dass Obnam Ihre Backups einfach als eine Verzeichnis darstellt. Es ist der bevorzugte Weg, aber nicht immer möglich, daher hat Obnam auch eine primitivere, weniger leicht zu bedienende Methode.

6.1 Wiederherstellen mit FUSE[1]

Mittels `obnam mount` können Sie Ihre Backups ansehen, als wären sie ein Verzeichnis wie jedes andere. Dazu müssen Sie allerdings FUSE installiert haben (vgl. Kapitel "Installation" für Details). Die meisten modernen Linux Desktops bringen FUSE schon mit.

```
mkdir ~/backups
obnam mount --to ~/backups
```

Führen Sie den Befehl oben aus und schauen Sie dann in das Verzeichnis `~/backups`. Sie werden ungefähr das hier sehen:

```
$ ls -l ~/backups
total 12
drwxr-xr-x 24 root root 4096 Feb 11 21:41 2
drwxr-xr-x 24 root root 4096 Feb 11 21:41 5
lrwxr-xr-x 24 root root 4096 Feb 11 21:41 latest -> 5
$
```

Jedes Verzeichnis unter `~/backups` ist eine Generation Ihres Backups, benannt nach dem "generation identifier" den Obnam vergibt. Der Symlink `latest` zeigt immer auf die neueste Generation.

[1] Anm. d. Ü.: Sprachkomik zu übersetzen ist schwer, der Witz bleibt dabei zu oft auf der Strecke. Daher versuche ich es garnicht erst.

Jetzt können Sie kinderleicht eine Datei wieder herstellen:

```
cp ~/backups/latest/home/tomjon/Documents/iloveyou.txt ~/restored.txt
```

Sie können beliebige Dateien aus dem Verzeichnis `~/backups` kopieren, aus jeder Generation, oder aus allen wenn Sie möchten. Sie können die Dateien auch erst ansehen, bevor Sie sie herauskopieren:

```
less ~/backups/2/home/tomjon/Documents/iloveyou.txt
```

So finden Sie leicht die Version die Sie suchen, nicht nur die neueste.

Etwas löschen können Sie in `~/backups` nicht. Das Verzeichnis ist read-only und Sie können weder absichtlich noch unabsichtlich etwas darin löschen oder verändern. Dieses Verhalten ist beabsichtigt: `obnam mount` soll eine sichere Möglichkeit bieten, Ihre Backups zu betrachten, ohne das Sie besondere Sorgfalt walten lassen müssen.

Wenn Sie Ihre Backups nicht mehr ansehen möchten, können Sie das Repository un-mounten:

```
fusermount -u ~/backups
```

Neben der Kommandozeile können Sie natürlich auch den Dateimanager Ihrer Wahl benutzen. Den mount und un-mount Vorgang müssen Sie (abhängig von der Konfiguration Ihres PCs) eventuell trotzdem auf der Kommandozeile durchführen.

6.2 Wiederherstellen ohne FUSE

Wenn `obnam mount` nicht verfügbar ist, können Sie direkt mit Obnam wiederherstellen. Verwenden Sie `obnam generations` und `obnam ls`, um die richtige Generation zur Wiederherstellung zu finden, und führen Sie dann einen Befehl wie diesen aus:

```
obnam restore --to /tmp/tomjon-restored /home/tomjon/Documents
```

So wird das angegebene Verzeichnis wieder hergestellt. Ohne Angabe, was wieder herzustellen ist wird die gesamte neueste Generation wieder hergestellt. Eine andere Generation währen Sie mit `--generation` aus:

```
obnam restore --to /tmp/tomjon-restored --generation 2
```

Hinweis: Sie können kein Verzeichnis wiederherstellen, das bereits existiert. So wird verhindert, das Sie ein bereits existierendes Verzeichnis mit den wieder hergestellten Daten überschreiben. Wenn Sie Ihre Live-Daten wirklich ersetzen möchten, sollten Sie zunächst in ein temporäres Verzeichnis zurücksichern und dann die Daten verschieben.

Die Datei bzw. das Verzeichnis das Sie wiederherstellen möchten muss mit absolutem Pfad angegeben werden, also beginnend mit /. Es darf nicht relativ zum aktuellen Verzeichnis sein.

6.3 Konkretes Beispiel einer Wiederherstellung

Um eine korrupte Datei meines News Readers wiederherzustellen ging ich wie folgt vor:

```
obnam --config=/home/foobar/cron/conf/obnam.conf generations>~/cron/upload/obgen.txt
```

Dies schreibt alle Generationen in eine Datei mit Namen “obgen.txt”, hier ein Auszug:

```
1207586    2014-08-25 08:00:43 .. 2014-08-25 08:08:24 (385163 files, 175029819657 bytes)
1208367    2014-08-25 12:00:42 .. 2014-08-25 12:08:31 (385965 files, 175057598863 bytes)
1209313    2014-08-25 16:00:12 .. 2014-08-25 16:07:33 (386537 files, 175076976590 bytes)
1210254    2014-08-25 20:00:15 .. 2014-08-25 20:09:41 (386896 files, 175086483254 bytes)
```

Ich wollte aus Generation 1208367 wiederherstellen. Das war der konkrete Befehl:

```
obnam --config=/home/benutzername/cron/conf/obnam.conf --generation=1208367 restore ~/News/rss/nnrss
```

Der Befehl holt die Datei ‘nnrss.el’ aus der angegebenen Generation und legt sie in ~/cron/upload/ ab. Von dort aus kopierte ich sie einfach wieder an ihren Platz.

6.4 Übung macht den Meister

Sie sollten das Zurücksichern üben. So bekommen Sie mehr Vertrauen und Ihre Backups und können ruhiger bleiben, wenn das Schlimmste passiert. Etwas hochgestocherner ausgedrückt: Sie sollten Ihren Disaster Recovery Plan testen.

Machen Sie testweise eine Wiederherstellung von ein paar Dateien oder sogar allen, bis Sie sicher wissen wie das geht. Von Zeit zu Zeit sollten Sie das wiederholen um sicher zu sein das Ihre Backups immer noch funktionieren. Es ist viel weniger beängstigend, nach Datenverlust eine echte Wiederherstellung zu machen, wenn man vorher geübt hat.

In extremen Fällen, insbesondere wenn Sie ein Obnam Entwickler sind, sollten Sie vielleicht mal Ihre Festplatte formatieren und dann die Wiederherstellung durchführen, nur um zu wissen, das Sie es können. Wenn Sie kein Obnam Entwickler sind, wäre das vielleicht ein bisschen extrem: Benutzen Sie einfach eine separate Festplatte statt der eingebauten.

Kapitel 7

Alte Backup-Generationen löschen

Jedes Mal wenn Sie ein Backup erstellen, wächst Ihr Backup-Repository in der Größe. Um ein Volllaufen des verfügbaren Speichers zu vermeiden müssen Sie ab und zu einige alte Backups los werden. Das ist natürlich ein bisschen ein Dilemma: Sie machen Backups um keine Daten zu verlieren, und jetzt müssen Sie genau das tun.

Obnam verwendet den Begriff “vergessen” für das Entfernen einer Sicherungsgeneration. Sie können mittels des Generation Identifier angeben, welche Generation manuell entfernt werden soll, oder Sie können einen Zeitplan anlegen, nach dem dann automatisch “vergessen” wird.

Eine bestimmte Generation löschen:

```
obnam forget 2
```

(Dieses Beispiel setzt voraus, dass Sie eine Konfigurationsdatei haben die Obnam automatisch findet, und dass Sie Dinge wie den Pfad zum Repository oder die Verschlüsselung nicht auf der Kommandozeile einzugeben brauchen).

Sie können jede Generation unabhängig von einander löschen. Obnam behandelt jede Generation als unabhängigen vollständigen Snapshot (in Wirklichkeit wird natürlich nicht jedes Mal eine vollständige Sicherung gemacht), Sie brauchen Sich also keine Sorgen um Unterschiede zwischen einer vollständigen und inkrementellen Sicherung machen.

Backups manuell zu löschen ist mühsam, wahrscheinlich werden Sie einen Plan verwenden wollen, nach dem Obnam die Generationen automatisch löscht.

Eine oft angewandter Zeitplan ist zum Beispiel dieser:

- Behalte ein Backup für jeden Tag der vorigen Woche
- Behalte ein Backup für jede Woche der vorigen 3 Monate
- Behalte ein Backup für jeden Monat der vergangenen 2 Jahre
- Behalte ein Backup für jedes Jahr der vergangenen 57 Jahre

Obnam verwendet die `--keep` Option um einen Zeitplan festzulegen. Die Einstellung für den oben Zeitplan sähe wie folgt aus:

```
--keep 7d,15w,24m,57y
```

Die Übereinstimmung ist etwas ungenau, weil ein Monat mehr oder weniger Wochen haben kann, aber sie sollte ausreichen. Die Einstellung "7d" wird als "die letzte Sicherung jedes Kalendertags der letzten sieben Tage, an denen Sicherungen gemacht wurden" interpretiert. Für den Rest des Zeitplans gilt das analog. Lesen Sie das Kapitel "Obnam Konfigurationsdateien und Einstellungen" für genauere Details.

Der Zeitplan wählt eine Reihe von Generationen aus, die behalten werden. Alles andere wird gelöscht.

7.1 Einen Zeitplan für das Löschen von Generationen auswählen

Der Zeitplan für das löschen von Backup-Generationen ist ein bisschen ein Ratespiel, genau wie Backups im Allgemeinen. Wenn Sie sicher die Zukunft vorhersagen könnten, wüssten Sie alle Katastrophen, die Ihre Daten gefährden können schon vorher und Sie könnten Ihr Backup auf die Dinge beschränken, die sonst verloren gehen würden.

In dieser Welt müssen Sie leider raten. Sie müssen darüber nachdenken mit welchen Risiken Sie (oder Ihre Daten) konfrontiert sind und wie viel Sie ausgeben wollen um sich (oder Ihre Daten) zu schützen.

- Haben Sie Angst das Ihre Festplatte plötzlich auf sehr spektakuläre Art und Weise, z.B. durch Brand oder Diebstahl abhanden kommt? Wenn ja, brauchen Sie eigentlich nur eine recht aktuelle Sicherung, um das Risiko abzusichern.
- Machen Sie sich Sorgen Ihre Festplatte, Ihr Dateisystem, Ihre Anwendungen oder Sie selbst könnten langsam Ihre Daten zerstören? Wie lange würde es dauern um das zu bemerken? Sie brauchen eine Backup-History die weiter zurückreicht als es dauert, das Problem zu erkennen.
- Ähnlich bei versehentlichem Löschen von Dateien. Wie lange wird es dauern bis Sie das bemerken? Mindestens so lang sollte Ihre Backup-History sein.

Natürlich gibt es auch noch andere Kriterien, zum Beispiel:

Möchten Sie in 50 Jahren sehen, wie Ihre Dateien werden heute abgelegt sind? Wenn ja, benötigen Sie ein fünfzig Jahre altes Backup, sowie vielleicht ein Backup von jedem Jahr, falls Sie vergleichen wollen, wie sich die Dateien jedes Jahr entwickelt haben. Mit wachsenden Speichermedien und guter Deduplizierung ist dies nicht ganz so teurer als es auf den ersten Blick scheint.

Es gibt keinen Zeitplan der jedermanns Bedürfnisse erfüllt. Sie müssen für sich selbst entscheiden, deshalb ist die Standard-Einstellung in Obnam alles für immer zu behalten. Es ist nicht Obnams Aufgabe zu entscheiden, ob Sie diese oder jene Sicherungsgeneration nicht vielleicht behalten sollten.

Kapitel 8

Backups überprüfen

Es ist 9 Uhr abends. Wissen Sie, ob Ihre Backups funktionieren? Wissen Sie, wann Sie das letzte mal eine erfolgreiche Sicherung aller Ihrer Daten gemacht haben? Wissen Sie, ob Sie aus dieser Sicherung alles wieder herstellen können? Wenn nicht, wie können Sie gut schlafen?

Sie sollten Ihre Backups überprüfen, und zwar regelmäßig – nicht nur das eine Mal, als Sie das Backup-System aufsetzten. Überprüfen bedeutet, das Sie tun was auch immer getan werden muss, um sicherzustellen, dass alle Ihre wertvollen Daten gesichert wurden und fehlerfrei aus den Backups wieder hergestellt werden können.

Der einfachste Weg dies zu tun ist alle Ihre Daten wieder herzustellen und dann mit den Live-Daten zu vergleichen um Unterschiede zu finden. Das erfordert entsprechend Speicherplatz um alles wieder herzustellen, aber es ist fast die einzige Möglichkeit sicher zu sein.

Gleichzeitig ist das auch ein guter Weg um sicherzustellen, das die Wiederherstellung tatsächlich funktioniert. Wenn Sie dies nicht testen, dürfen Sie – wenn es darauf ankommt – nicht erwarten, dass das Restore auch wirklich funktioniert.

Wenn Sie den Speicherplatz haben um eine komplette Wiederherstellung zu machen, sollten Sie dies auch tun. Es ist eine großartige Möglichkeit, Ihre Desaster-Recovery-Prozesse durchzuspielen.

Ein Weg das mal zu tun wäre dieser:

- Machen Sie ein Backup auf Ihrem Hauptcomputer
- Machen Sie eine komplette Wiederherstellung auf einem zweiten Computer (eventuell leihen Sie einfach einen), ohne den Hauptcomputer überhaupt zu benutzen.
- Beginnen Sie, mit den wiederhergestellten Daten als Live-Daten zu arbeiten. Machen Sie echte Arbeit und all die Dinge, die Sie normalerweise tun. Tun Sie so, als wäre Ihr Hauptcomputer von Ihrem Haustier gefressen worden.
- Wenn Sie bemerken das etwas fehlt oder korrupt ist, oder zu alt, holen Sie die Dateien von Ihrem Hauptcomputer und richten Sie Ihren Backup-Prozess damit Sie das nächste Mal nicht dieses Problem haben werden.

Wie oft man das tun sollte? Das wiederum hängt davon ab, wie viel Ihnen Ihre Daten bedeuten und wie viel Sie Ihren Backup-Tools und Prozessen vertrauen. Wenn es wirklich wichtig ist, dass Sie nach einer Katastrophe ein Restore erstellen können, benötigen Sie häufigere Überprüfungen.

Wenn der Datenverlust höchstens Umstände macht und Ihr Leben nicht katastrophal verändert, können Sie weniger häufig überprüfen.

Neben der Wiederherstellung von Daten bietet Obnam noch zwei andere Möglichkeiten, wie Sie Ihre Backups überprüfen können:

- `obnam verify` ist fast wie `obnam restore`, außer das es die gesicherten Daten mit Live-Daten vergleicht und meldet Unterschiede meldet. Das bedeutet natürlich, dass Sie darauf vertrauen, dass Obnam die Überprüfung richtig macht.
- Mit `obnam mount` können Sie auf Ihre gesicherten Daten zugreifen, als ob sie in einem normalen Verzeichnis lägen. Anschließend können Sie ein beliebiges Werkzeug Ihres Vertrauens benutzen, um die gesicherten Daten mit den Live-Daten zu vergleichen. Das ist fast genau so wie alles wiederherzustellen, da das Vergleichs-Tool alle Daten und Metadaten aus dem Backup extrahieren muss. Die Daten werden nur nicht weg geschrieben.

Beide Ansätze haben das Problem, dass sie eine Sicherung mit Live-Daten vergleichen und die Live-Daten sich nach der Sicherung geändert haben könnten. Sie müssen alle Unterschiede manuell überprüfen, was eine größere Aufgabe sein kann, wenn sich die Live-Daten häufig ändern.

Kapitel 9

Ein Repository mit mehreren Clients benutzen

Mit Obnam können Sie mehrere Computer in das gleiche Repository sichern. Jeder Client wird durch einen Namen identifiziert, der standardmäßig dem Hostnamen entspricht, also dem Namen den Sie erhalten, wenn Sie den Befehl `hostname` ausführen. Sie können den Name aber auch explizit mittels `--client-name` selbst angeben.

Alle Clients in einem gemeinsamen Repository benutzen sämtliche Daten (die chunks), die Deduplizierung funktioniert also über Client-Grenzen hinweg. Jeder Client hat seine eigenen Backup-Generationen, die völlig unabhängig von anderen Clients sind. Sie können beispielsweise alle Generationen eines Clients löschen (forget), ohne das dies einen Einfluss auf die Generationen oder gar die gesicherten Daten der anderen Clients hat.

Obnam kümmert sich automatisch um das Locking, so dass Sie auf jedem Client Obnam laufen lassen können, ohne sich darum kümmern zu müssen das zu jeder Zeit nur ein Client aktiv ist.

Einen Vorbehalt müssen Sie bei der gemeinsamen Nutzung von Repositories beachten: Jeder Client hat Zugriff auf alle chunks und kann jeden anderen Client aus dem Repository löschen. Das heißt, Sie sollten nur ein Repository mit Clients teilen, die in der gleichen Sicherheitsdomäne sind: Allen Clients sollte gleichermaßen vertraut werden. Wenn ein Client gehackt wird, dann erhält der Eindringling Zugriff auf alle Daten im Repository und kann die Backups aller Clients des Repository löschen.

Um ein gemeinsames Repository zu erstellen müssen Sie folgendes tun:

- Legen Sie einen eindeutigen Namen für jeden Client fest. Der Name muss innerhalb des Repository eindeutig sein.
- Geben Sie jedem Client Zugriff auf das Repository.

Das ist schon alles.

Um zu sehen welche Clients ein Repository benutzen, führen Sie einfach dies aus:

```
obnam clients
```

Es gibt derzeit keine Möglichkeit, einen Client aus einem Repository zu entfernen, es sei denn, Sie benutzen die GnuPG-Verschlüsselung. Das ist als Bug in Obnam berücksichtigt und wird

irgendwann gefixt. Danach wird eine Zeitmaschine entwickelt, so dass dieser Absatz nie existiert haben wird.

Kapitel 10

Verschlüsselung nutzen

Mit Obnam können Sie Ihre Backups verschlüsseln. Dieses Kapitel beschreibt, warum und wie Sie das tun.

10.1 Sie sind kein Spion, ist Verschlüsselung da nicht überflüssig?

Sie sind nicht der einzige, der sich um Ihre Daten sorgt. Eine Vielzahl von Regierungen, Unternehmen, Kriminellen und allzu neugierigen Schnüfflern sind wahrscheinlich ebenfalls interessiert (... und es ist manchmal schwer, diese auseinander zu halten). Sie könnten Beweise gegen Sie (er)finden wollen, Sie erpressen wollen, oder einfach nur neugierig darauf sein, was Sie mit Freunden besprechen.

Sie könnten Ihre Daten aus statistischen Gründen interessant finden und überhaupt kein Interesse an Ihnen persönlich haben. Oder sie könnten ausschließlich an Ihnen interessiert sein.

Statt Ihre Dateien und eMails zu lesen oder Ihre Fotos und Videos anzusehen könnten sie Interesse daran haben, Ihnen den Zugang dazu zu versperren, oder einfach Ihre Daten ganz zerstören. Sie könnten sogar Ihre Daten korrumpieren, indem sie Kinderpornographie in Ihrem Foto-Archiv ablegen.

Sie schützen Ihren Computer so gut Sie können damit diese und andere schlimme Dinge nicht passieren. Ihre Sicherungen sollten Sie mit der gleichen Sorgfalt behandeln.

Wenn Sie auf ein USB-Laufwerk sichern, sollten Sie das Laufwerk verschlüsseln, genau wie auch Online-Speicher. Es gibt viele Arten von Verschlüsselung und ich bin nicht qualifiziert Ihnen Rat zu geben, aber jegliche gängige, moderne Verschlüsselung sollte ausreichen – außer für besonders entschlossene Angreifer.

Anstatt oder zusätzlich zur Verschlüsselung können Sie den physischen Zugang zu Ihren Backup-Medien absichern. Lagern Sie Ihr USB-Laufwerk z.B. in einem Safe oder Schließfach.

Die Vielzahl von Backups die Sie benötigen um sich gegen Erdbeben, Flutkatastrophen und marodierende Gangs dreirad-fahrender Clowns zu schützen, sind auch ein guter Schutz gegen Angreifer. Sie können Ihre Live-Daten und die Backups bei Ihnen zu Hause korrumpieren, aber

vermutlich könnten Sie nicht an das USB-Laufwerk herankommen, das in Beton gegossen an einem geheimen Ort vergraben ist, den nur Sie kennen.

Auf der anderen Seite möchten oder müssen Sie vielleicht Anderen Zugriff auf Ihre gesicherten Daten geben. Wenn Sie zum Beispiel von der Clown Gang entführt wurden, sollte Ihr Partner in der Lage sein, Ihren MI6-Verbindungsmann zu kontaktieren, damit der Geheimdienst Sie retten kann. Den sicheren Zugang zu (Teilen) Ihres Backups herzustellen ist ein interessantes Problem für sich, für das es verschiedene Lösungen gibt: Geben Sie Ihrem Partner das Verschlüsselungspasswort, oder geben Sie es einem Freund dem Sie vertrauen, oder einem Anwalt. Sie könnten auch so etwas wie libgfshare verwenden, um die Schlüssel auf sichere Weise treuhänderisch zu hinterlegen.

10.2 Wie Obnams Verschlüsselung funktioniert

Ein Obnam Repository enthält mehrere Verzeichnisse für verschiedene Arten von Daten.

- Ein Verzeichnis je Client, für Daten, die nur für diesen Client relevant sind, z.B. Generationen dieses Clients.
- Ein Verzeichnis für die Liste der Clients.
- Ein Verzeichnis für alle chunks, sowie zusätzliche Verzeichnisse, die für die Deduplizierung von chunks verwendet werden.

Das Verzeichnis je Client ist verschlüsselt, so dass nur der jeweilige Client zugreifen kann. Das bedeutet, dass nur der Client selbst seine Generationen, und die darin enthaltenen Dateien sehen kann.

Die gemeinsam genutzten Verzeichnisse (Client-Liste, chunks) sind so verschlüsselt, dass alle Clients sie benutzen können. Dies ermöglicht es den Clients, chunks gemeinsam zu nutzen, so dass die Deduplizierung über alle Clients laufen kann.

Dieses Verschlüsselungsverfahren geht davon aus das alle Clients die sich ein Repository teilen einander vertrauen und dass es in Ordnung ist, sämtliche chunks zu lesen, die sie wollen. Zum Beispiel verhindert die Verschlüsselung nicht, das Geschwister die eMails des anderen aus dem Repository lesen, aber die Eltern können das nicht, weil ihnen der geeignete Schlüssel fehlt.

Zusätzlich zu den für die Client-Verschlüsselungen können Sie zusätzliche Schlüssel hinzufügen. Diese Schlüssel haben dann ebenfalls Zugang zum Backup-Repository. Beispielsweise könnte der Schlüssel der Eltern dem Repository hinzugefügt werden, so dass, wenn es sein muss, die Eltern Daten der Kinder wiederherstellen können, auch wenn das Kind seinen eigenen Schlüssel verloren hat.

In einer Unternehmensumgebung könnte der Schlüssel des Backup- Administrators hinzugefügt werden. So kann dieser zum Beispiel die Integrität des Repository prüfen, oder auf Daten eines Mitarbeiters zugreifen, der im Lotto gewonnen hat und wegen der schlechten Internet-Verbindung zum Mond nicht verfügbar ist.

Solche zusätzlichen Schlüssel können entweder für jeden einzelnen Client oder alle gleichzeitig hinzugefügt werden.

10.3 Verschlüsselung in Obnam einrichten

Obnam benutzt PGP-Schlüssel, genauer gesagt deren GNU Privacy Guard (GnuPG, gpg) Implementierung. Um verschlüsselte Backups zu erstellen müssen Sie erst ein PGP-Schlüsselpaar erzeugen. Wie das geht steht in der GnuPG Dokumentation (englisch).

Sobald Sie ein funktionierendes GnuPG-Setup und ein Schlüsselpaar (bestehend aus einem öffentlichen Schlüssel und einem geheimen Schlüssel) haben, müssen Sie die Schlüssel-ID finden. Führen Sie den folgenden Befehl aus und wählen Sie Ihren Schlüssel aus der Liste.

```
gpg --list-keys
```

Die Ausgabe sieht ungefähr so aus:

```
pub   4096R/5E8511F9 2009-07-22
uid                  Lars Wirzenius <liw@liw.fi>
sub   2048R/9BE35AE6 2011-08-05
```

Das ist die Ausgabe für einen einzelnen Schlüssel, es kann auch mehrere geben. Die Schlüssel-ID steht in der Zeile die mit `pub` beginnt in zweiten Spalte nach dem Schrägstrich. Im obigen Beispiel wäre das die 5E8511F9.

Für die restlichen Beispielen dieses Kapitels gehen wir davon aus das Ihre Key-ID CAFEFACE ist.

Um von der Verschlüsselung zu profitieren benutzten Sie den `--encrypt-with` Schalter:

```
[config]
encrypt-with = CAFEFACE
```

Das ist alles.

Beachten Sie, dass ein Repository vollständig oder gar nicht verschlüsselt sein sollte, und dass man nicht hin und her wechseln kann. Wenn Sie Ihre Meinung ob sie Verschlüsselung benutzen möchten ändern, müssen Sie mit einem neuen Repository von vorn beginnen. Alle Clients, die ein Repository teilen müssen verschlüsseln, oder aber keiner von ihnen. Wenn Sie beides vermischen können verwirrende Fehlermeldungen erscheinen.

Obnam verschlüsselt automatisch alle Daten, die es ins Repository schreibt und entschlüsselt sie wenn nötig. Solange Sie nur ein Schlüssel für jeden Client benutzen, kümmert sich Obnam darum die richtigen Schlüssel an den richtigen Stellen hinzu zu fügen.

10.4 Prüfen ob ein Repository Verschlüsselung nutzt

Es gibt keinen direkten Weg um mit Obnam zu überprüfen, ob ein Repository Verschlüsselung benutzt. Sie können das jedoch auch manuell überprüfen: Wenn Ihr Repository die Datei `clientlist/key` enthält, wird das Repository verschlüsselt.

10.5 FIXME: Verwalten der Schlüssel in einem Repository

In diesem Abschnitt wird erläutert, wie Sie die Schlüssel in einem Repository verwalten: Wie Sie zusätzliche Schlüssel für jedes Toplevel hinzufügen und wie Sie die Schlüssel eines Client ändern. Es zeigt auch wie Sie prüfen können, welche Schlüssel verwendet werden, und welchen Zugriff jeder Schlüssel hat.

Kapitel 11

Verschiedenes

Dieses Kapitel behandelt Themen die kein eigenes Kapitel füllen, z.B. das Komprimieren von Backups und wie Obnam von cron ausgeführt wird.

11.1 k4dirstat cache files

k4dirstat ist ein Programm das die Festplattenbelegung. Obnams `kdirstat` Befehl kann verwendet werden, um den Inhalt einer Generation in einem Format auszugeben, das von k4dirstat mittels `File`, `Read  Cache File` eingelesen werden kann.

Beispiel: $ obnam kdirstat –client CLIENT –generation GENID > CLIENT.kdirstat.cache $ gzip -v9 CLIENT.kdirstat.cache # OPTIONAL

`CLIENT.kdirstat.cache[.gz]` kann nun in `k4dirstat` eingelesen werden.

Kapitel 12

Fallbeispiele

In diesem Kapitel gehen wir auf einige typische Anwendungsfälle für Backups ein. Wir besprechen welche Daten gesichert werden und erklären wie Sie eine passende Backup-Strategie, ein passendes Medium usw. auswählen.

Einige Fallbeispiele:

- Einzelner Laptop-Benutzer, typische Daten wie Dokumente, Fotos, Musik, Backup auf eine USB-Festplatte.
- Ein VPS oder ähnlicher Server mit Web-Seiten, eMails und vielleicht anderen Daten, Backup auf einen anderen Server.
- Ein kleines Unternehmen mit einer Reihe von Laptops und Desktop-PCs, einem lokalen Fileserver, ein gemieteter Server bei einem Provider, Backup auf einem gemieteten Server beim Provider und auf mehrere große USB-Laufwerke die durchgetauscht werden.
- Wiederherstellung eines Offsite-Backup nachdem alle lokalen Computern und Speichermedien zerstört wurden.

Kapitel 13

Fehlersuche

Dieses Kapitel beschreibt, wie Sie Probleme mit Obnam debuggen. Es umfasst Dinge wie Log-Dateien, verschiedene Ebenen der Protokollierung und

Ablaufverfolgung und häufige Probleme bei der Benutzung von Obnam. Es erklärt auch, welche Dinge wo in einem Obnam Backup-Repository landen.

13.1 Logging einschalten

Obnam kann ein Logfile erstellen, was durch mehrere Optionen gesteuert wird. Diese zu kennen kann helfen, die wichtigsten Informationen zu sammeln, wenn ein Problem untersucht werden muss.

- `--log=obnam.log` weist Obnam an, wohin geloggt werden soll. Das Logfile ist eine einfache Textdatei.
- `--log-level=debug` veranlasst Obnam, mit maximalen Details zu loggen. Der Standard-Loglevel `info` lässt die meisten Debug-Informationen weg.
- `--trace=obnamlib --trace=larch` lässt Obnam zusätzliche Debug-Informationen loggen. Beide Argumente passen auf sämtliche Obnam-Dateinamen und auf die der Larch-Bibliothek, die Obnam benutzt. Diese zusätzlichen Informationen sind für diejenigen interessant, die den Programmcode lesen und verstehen können

Bitte beachten Sie, das diese Einstellungen die Logfiles stark vergrößern können (mehrere 10 Megabytes). Die Größe hängt von der Anzahl und Größe der Dateien der Live-Daten ab.

13.2 Fehlerbericht schreiben

Wenn Sie ein Problem bei der Benutzung von Obnam feststellen und Sie einen Fehlerbericht schreiben möchten (bitte tun Sie das), dann helfen folgende Informationen dabei, das Problem einzugrenzen.

- Senden Sie Fehlerberichte an die Mailingliste `obnam-support@obnam.org`. Auf dieser öffentlich archivierten Mailingliste helfen Benutzer anderen Benutzern.

- Beim Antworten an `obnam-support`, bitte **immer** ein CC an die Mailingliste schicken. So können auch andere die Antwort lesen, was die Chance erhöht, das jemand anders (der evtl. das Problem besser versteht), besser helfen kann. Außerdem kann die archivierte Diskussion anderen Lesern helfen, manchmal sogar Jahre später.
- Beschreiben Sie den Fehler. Was wollten Sie tun, was ist stattdessen passiert?
- Die Versionen von Obnam und Larch, die Sie benutzen und wie Sie sie installiert haben.
  - Unter Debian führen Sie einfach `dpkg -l obnam python-larch` in einem Terminal aus und schicken die Ausgabe mit.
- Die genaue Befehlszeile, die Sie benutzt haben. Bitte benutzen Sie die "Kopieren-und-Einfügen"-Funktion, anstatt abzuschreiben: Das Problem könnte verdeckt sein, wenn Sie nicht die exakte Befehlszeile wiedergeben. Außerdem ist Abschreiben langweilig und wir sollten in unserem Leben Langeweile vermeiden.
- Wenn Sie eine Fehlermeldung erhalten, bitte ebenfalls mittels Kopieren-und-Einfügen der Mail hinzufügen.
- Die komplette Konfiguration. Fügen Sie die Ausgabe von `obnam --dump-config` zur eMail hinzu. Sie können die Ausgabe als Anhang an Ihre eMail an `obnam-support` senden. Bitte bedenken Sie, vertrauliche Informationen wie z.B. Datei- oder Maschinen-Namen durch XXXX zu ersetzen.
- Sollten Sie das Problem mit `--log-level=debug`, `--log=obnam.log`, `--trace=obnamlib` und `--trace=larch` reproduzieren können, senden Sie bitte einen passenden Abschnitt vom Ende des Logfiles. "Passend" kommt in diesem Fall auf die Situation an; sollten die letzten ca. 200 Zeilen nicht ausreichen, werden wir uns schon melden. Bitte bedenken Sie, vertrauliche Informationen wie z.B. Datei- oder Maschinen-Namen durch XXXX zu ersetzen.
- Die Ausgabe des `env`-Befehls, ausgeführt im gleichen Terminal in dem Sie Obnam ausgeführt haben. (Bitte auch als Anhang).
- Sollte Ihr Fehlerbericht die Geschwindigkeit betreffen, starten Sie Obnam bitte mit profiling und hängen Sie den Profiling-Bericht an. Um Obnam mit profiling zu starten, installieren Sie bitte Python profile (das `python-profiler`-Paket in Debian / Ubuntu), und setzen Sie die Umgebungsvariable `OBNAM_PROFILE` auf den Dateinamen der Ausgabedatei (die Sie dann bitte per eMail senden).

Beispiel: `OBNAM_PROFILE=obnam.prof obnam backup` startet das Backup mit profiling und schreibt das Ergebnis nach `obnam.prof`.

Danke für Ihre Hilfe beim Verbessern von Obnam.

Kapitel 14

Obnam Konfigurationsdateien und Einstellungen

In diesem Kapitel geht es um Obnams Konfigurationsdateien: Wo sie sind, was sie enthalten, und wie sie verwendet werden.

14.1 Wo ist meine Konfiguration?

Obnam sucht seine Konfigurationsdatei an folgenden Orten:

- `/etc/obnam.conf`
- `/etc/obnam/*.conf`
- `~/.obnam.conf`
- `~/.config/obnam/*.conf`

In `/etc/obnam` und `~/.config/obnam` werden alle Dateien mit dem Suffix `.conf` in "asciibetischer" Reihenfolge geladen. Das ist ähnlich wie alphabetisch, basiert aber auf dem Zeichencode und nicht auf dem, was die Leute denken. Im Gegensatz zu alphabetisch ist das sprachunabhängig.

Alle Dateien der obigen Liste können existieren (oder auch nicht). Wenn eine Datei vorhanden ist wird sie gelesen, dann die nächste Datei, und so weiter. Eine Einstellung in einer Datei wird durch eine spätere Datei überschrieben, wenn auch dort die Option eingestellt ist. Zum Beispiel könnte `/etc /obnam.conf` den `log-level` auf `INFO` setzen, aber `~ /.obnam.conf` setzt ihn dann auf `DEBUG`, weil der Benutzer detailiertere Log-Dateien wünscht.

Die Obnam Konfigurationsdateien in `/ etc` gelten für alle, die Obnam auf dieser Maschine benutzen. Das ist wichtig: Sie gelten nicht nur für `root`.

Wenn Sie mehrere Konfigurationen für Obnam haben möchten, um zum Beispiel verschiedene Backup-Repositories zu nutzen, müssen Sie die Dateien so ablegen oder benennen, das sie nicht zur Liste oben passen. Zum Beispiel:

- `/etc/obnam/system-backup.profile`
- `~/.config/obnam/online.profile`
- `~/.config/obnam/usbdrive.profile`

Bei der Ausführung von Obnam müssen Sie dann nur noch das File angeben, mit dessen Konfiguration gearbeitet werden soll:

```
obnam --config ~/.config/obnam/usbdrive.profile`
```

Sollten Sie außerdem wünschen das Obnam sämtliche Standard-Konfigurationsdateien ignoriert, müssen Sie die Option `--no-default-config` mitgeben:

```
obnam --no-default-config --config ~/.obnam-is-fun.conf
```

Optionen die auf der Kommandozeile geben werden, überschreiben Werte die aus Konfigurationsdateien geladen wurden.

14.2 Syntax der Konfigurationsdateien

Obnam Konfigurationsdateien verwenden die INI-Datei Syntax, genauer gesagt die Variante, die von der Python ConfigParser Bibliothek implementiert wird.

Sie sehen so aus:

```
[config]
log-level = debug
log = /var/log/obnam.log
encrypt-with = CAFEBEEF
root = /
one-file-system = yes
```

Die Namen der Konfigurationsvariablen sind die gleichen wie die entsprechenden Befehlszeilenoptionen. Wenn `--foo` die Befehlszeilenoption ist, dann ist die Variable in der Datei `foo`. Jede Kommandozeilen-Option `- foo = bar` kann in einer Konfigurationsdatei als `foo = bar` verwendet werden. Es gibt einige Ausnahmen (`-- no-default-config`, `--config`, `--help` und ein paar andere), aber die würden Sie sowieso nicht in einer Konfigurationsdatei setzen.

Jede Option oder Einstellung hat einen Typ. Meist ist dies nicht relevant, es sei denn, Sie geben ihr einen Wert der ungeeignet ist. Die beiden wichtigsten Ausnahmen sind:

- Boolean bzw. ja/nein oder an/aus Zum Beispiel ist `--exclude-caches` eine Option, die entweder an ist (wenn die Option benutzt wird) oder aus ist (wenn sie nicht benutzt wird). Für jede Option `--foo` gibt es auch eine Option `--no-foo`. In Konfigurationsdateien wird `foo` durch setzen auf `yes` oder `true` eingeschaltet, und durch `no` oder `false` abgeschaltet.
- Einige Optionen können eine Werteliste aufnehmen, zum Beispiel `--exclude`. Sie können `--exclude` verwenden so oft Sie wollen, jedes Mal wird ein neuer Ausschluss hinzugefügt, statt den vorherigen zu ersetzen. In einer Konfigurationsdatei trennen Sie die Werte mit Komma und schreiben Sie hintereinander, z.B.: `exclude = foo, bar, baz`. Durch später geladenen Konfigurationsdateien wird die gesamte Werteliste ersetzt, anstatt hinzugefügt.

Eine genauere Erklärung der Syntax finden Sie in der **cliapp**(5) manpage Ihres Systems oder im WWW cliapp man page (englisch).

14.3 Meine Konfiguration prüfen

Weil Obnam seine Konfigurationsdaten von mehreren Stellen bezieht, kann es schwierig sein herauszufinden welche Optionen nun wirklich Anwendung finden. Die Option `--dump-config` hilft dabei.

```
obnam --config ~/.obnam.fun --exclude-caches --dump-config
```

Diese Option liest alle Konfigurationsdateien und gibt eine Zusammenfassung auf stdout aus, die jede Einstellung enthält, als wäre `--dump-config` nicht benutzt worden.

So können Sie schnell die Einstellungen überprüfen. Außerdem ist das ein guter Anfang wenn Sie mal eine Konfigurationsdatei von Hand neu erstellen möchten.

14.4 Alle Konfigurationseinstellungen herausfinden

Diese Anleitung beinhaltet und erklärt noch nicht alle Einstellungen. Obnam bietet aber eine integrierte Hilfe (`obnam - help`) und eine manpage, die automatisch aus der integrierten Hilfe erzeugt wird (`man obnam` oder siehe obnam man page (englisch) oder deutsch). Eines Tages wird dieses Kapitel einen automatisch generierten Abschnitt enthalten der jede Einstellung erklärt. Bis dahin dürfen Sie gern mit dem Finger auf Obnams Autor zeigen und über seine Faulheit kichern.

Kapitel 15

Interna eines Repository

Dieses Kapitel beschreibt die Interna eines Obnam Repository. Überspringen Sie dieses Kapitel, außer wenn Sie sind daran interessiert sind.

Im Moment sehen Sie bitte die Website unter http://obnam.org/development/ an.

15.1 Dateiberechtigungen im Repository

Obnam setzt die Berechtigungen aller Dateien, die es im Repository anlegt so, dass nur der Eigentümer der Dateien sie lesen und schreiben kann. (Technisch bedeutet dies: Dateien: 0600, Verzeichnisse 0700.)

Dies dient dazu, Informations-Abfluss zu vermeiden, falls jemand Lesezugriff auf das Repository hätte. Dieses Verhalten ist nicht konfigurierbar.

Kapitel 16

Performance-Tuning

In diesem Kapitel werden verschiedene Möglichkeiten zur Optimierung der Geschwindigeit in verschiedenen Situationen aufgezeigt. Es behandelt verschiedene Optionen welche den Speicherverbrauch und die CPU-Belastung betreffen sowie Hinweise wie selbst die "richtigen" Einstellungen gefunden werden können.

Für den Anfang lesen Sie bitte http://obnam.org/faq/tuning/.

16.1 Running Obnam under the Python profiler

Ein **profiler** ist ein Programm mit dem gemessen werden kann, wie andere Programme ihre Zeit verbringen. Das kann nützlich sein, wenn herauszufinden ist, warum ein Programm langsam arbeitet.

Obnam kann einfach mittels des Python Profilers ausgeführt werden, der natürlich installiert sein muss. Prüfen Sie die Dokumentation Ihres Betriebssystems oder Ihrer Python-Installation und herauszufinden, wie der Profiler installiert wird.
So prüfen Sie, ob der Profiler bereits auf Ihrem System installiert ist:

```
python -c 'import cProfile'
```

Wenn bei diesem Aufruf nichts ausgegeben wird, ist alles OK. Sollten Sie eine Fehlermeldung (siehe unten) erhalten, so ist der Profiler nicht installiert:

```
Traceback (most recent call last):
  File "<string>", line 1, in <module>
ImportError: No module named cProfiler
```

Wenn Sie den Profiler installiert haben, können Sie Obnam wie folgt starten:

```
OBNAM_PROFILE=backup.prof obnam backup
```

So werden die Profiling-Daten in die Daten `backup.prof` geschrieben. Sie können das natürlich mit jedem Obnamm so tun und in ein beliebiges Feil schreiben

Die Profiling-Daten werden binär gespeichert. Obnam bringt ein kleines Hilfsprogramm mit, welches die Daten in menschen-lesbare Form umwandelt:

```
obnam-viewprof backup.prof | less
```

Wenn Sie dies ausführen, werden Sie sehen: Die Ausgabe nützt nur Programmierern und Zirkusclowns. Wenn Sie die Ausgaben verstehen: Toll! Falls nicht, ist es immer noch hilfreich die Daten an die Obnam- Entwickler zu senden, wenn Sie ein Problem mit der Geschwindigkeit von Obnam haben.

Kapitel 17

Bei der Entwicklung von Obnam mithelfen

Das Obnam Projekt ist, verglichen mit anderen Software-Projekten, recht klein. Es gibt einen Hauptentwickler und ein paar Andere, die manchmal helfen. Es wäre schön wenn sich mehr Menschen beteiligten, dieses Kapitel ist als Einführung dafür gedacht.

Es ist ein verbreitetes Missverständnis, dass in einem Software Projekt ausschließlich Code etwas zählt. Im Gegenteil, ohne eine Anzahl ander Dinge ist Code nutzlos, insbesondere in einem freien Software-Projekt wie Obnam.

Beispiele für notwendige Dinge in fast jedem ernsthaften Software-Projekt:

- Schreiben und aktualisieren der Dokumentation, inkl. Handbüchern und Webseiten
- Übersetzung von Dokumentation und Benutzeroberfläche
- Unterstützung anderer Benutzer bei Fragen oder Problemen
- reporting actionable bugs
- Bearbeitung von Ferhlerberichten: Nachfragen und Klärung, den gemeldete Fehler reproduzieren, die Ursache für den Fehler herausarbeiten und das Entwickeln einer passenden Lösung
- die Software auf verschiedene Plattformen portieren, z.B. verschiedene Betriebssysteme, verschiedene Versionen dieser Betriebssysteme, verschiedene Versionen der Sprachen und Bibliotheken welche die Software verwendet, unterschiedliche Hardware, usw.
- Qualitätssicherung: Entwicklung und Durchführung von manuellen und automatisierten Tests und Benchmarks, Analyse der Ergebnisse
- Hosting und Betrieb von Webseiten, Mailinglisten, IRC-Kanälen und anderen Kommunikationskanälen
- Projektsteuerung, inkl. Umgang mit Konflikten zwischen Projektmitgliedern
- Projektmanagement im Allgemeinen, inkl. Sicherung des Projektfortschritts

- Letzlich, das Schreiben von Code. Sicherlich ein notwendiger, aber nicht ausreichender Teil eines Projekts, das auch Personen abseits der Entwicklung verwenden können.

Diese Liste ist unvollständig, Hinweise zur Erweiterung werden gern angenommen. Um zu erfahren wie Sie diese Liste vervollständigen können, lesen Sie bitte den Rest dieses Kapitels.

17.1 Hilfe beim User-Support

Vielleicht ist der einfachste Weg, sich am Projekt zu beteiligen die Mithilfe bei der Unterstützung anderer Benutzer der Software. Das ist einfach und Sie müssen nicht notwendigerweise mehr können, als die Software selbst zu benutzen. Dennoch ist es eien sehr wertvolle Tätigkeit, da sie dadurch andere entlasten. Sogar bei Softare höchster Qualität und einfachster Bedienung gibt es immer Bedarf für die Benutzerunterstützung:

- Code kann fehlerhaft sein und Benutzer könnten dies bemerken. Die Analyse der Situation und das Isolieren des Fehlers sind ein wichtiger Bestandteil des Software-Entwicklungsprozesses.
- Die Dokumentation kann unzureichend oder veraltet sein, oder auch eine Funktoin beschreiben, die es noch garnicht gibt.
- Manche Menschen haben - aus welchem Grund auch immer - Missverständnisse, die zu Problemen im Umgang mit der Software führen. Das eigentliche Problem und die Ursache herauszufinden, kann eine zeitaufwendige Sache sein. Oft erfordert diese Fleißaufgabe keine besonderen Fähigkeiten (mit Ausnahme von Geduld und der Bereitschaft, eine Menge Fragen zu stellen.

Der beste Weg dem Obnam-Projekt zu helfen? Abonnieren Sie die Mailingliste `obnam-support@obnam.org` oder verbinden Sie Sich mit dem IRC Kanal `#obnam` (irc.oftc.net) und beginnen Sie, Fragen zu beantworten.

Es ist OK kein Experte zu sein. Anderen zu helfen ist eine gute Möglichkeit, selbst etwas zu lernen. Wenn Sie deutlich machen das Sie zwar kein Experte sind, aber trotzdem versuchen zu helfen, wird Ihre Hilfe in der Regel noch mehr geschätzt.

Eine Vorschläge zur Unterstützung dieses Projekts:

- Versuchen Sie zu verstehen was die Personen die Hilfe brauchen tatsächlich erreichen möchten. Meist ist das besser als die Frage wörtlich zu beantworten. Noch besser wäre es, Sie tun beides.
- Sie brauchen keine Problemlösung, um sich zu beteiligen. Eine schnelle aber unvollständige Antwort, welche die Diskussion voranbringt, ist ebenfalls hilfreich. Auch wenn Sie die korrekte Antwort nicht kennen, macht es Sinn Rückfragen zu stellen, Hilfe suchende geben dann z. B. weiterführende Informationen oder finden die Lösung sogar selbst. Eventuell führt Ihre Rückfrage auch dazu, dass jemand anders die Lösung findet.
- Seien sie bitte hilfsbereit und höflich. Immer. Antworten Sie nie mit “read the fine manual” (RTFM for short). Es ist OK darauf hin zu weisen, dass die Antwort im Handbuch steht, aber bitte zitieren Sie dann aus dem Handbuch und geben Sie einen Link zum Nachlesen.
- Wer Hilfe sucht ist oft frustriert, manchmal sogar verzweifelt. Er hat vermutlich wieder und wieder versucht, das Problem selbst zu lösen und ist daran gescheitert. Dies kann sich in

den Anfragen wieder spiegeln. Ignorieren Sie diesen Subtext, es sei denn der Fragende wird tatsächlich konkret unhöflich, dann sollten Sie die Situation eskalieren. Vermeiden Sie einen Streit, bei dem es darum geht wer Recht hat und wer nicht, und wer wann was gesagt hat und wie es gemeint war.

- Es ist besser gar nicht zu antworten, als eine Antwort zu verfassen während man noch gereizt, verärgert oder wütend ist. Für das Projekt ist eine höfliche und hilfreiche Arbeitsatmosphäre auf lange Sicht wichtiger als die konkrete Lösung eines technischen Problems.

Kurz gesagt: Wenn Sie Ihr Bestes geben um höflich, freundlich, und hilfsbereit zu sein, dann antworten Sie ruhig.

17.2 Schreiben und Aktualisieren der Dokumentation

Das Projekt hat mehrere Arten von Dokumentation:

- Die `obnam.org` Website.
- Die manpage.
- Das Handbuch (das Sie gerade lesen).
- Verschiedene Blog posts im Internet.

Dokumentation schreiben ist recht einfach. Sie aktuell zu halten ist schon ein wenig mehr Aufwand, denn dazu muss die bereits bestehende Dokumentation auf Aktualität geprüft werden. Hauptziele der Obnam-Dokumentation sind:

- Genauigkeit.
- Klarheit.
- Vollständigkeit.
- Eine Prise trockener Humor hier und da.

Jede Hilfe die sie hier geben können ist sehr willkommen.

- Lesen Sie die bestehende Dokumentation.
- Wenn Sie etwas finden das falsch, ungenau, unvollständig oder unklar ist oder wenn etwas fehlt, schreiben Sie eine Mail an die Entwickler-Mailingliste.
- Wenn Sie eine bessere Formulierung beisteuern können, wäre das klasse. Es ist aber nicht zwingend erforderlich.
- Perfekt wäre es, wenn Sie sogar einen konkreten Patch erstellen können, weil es dann am einfachsten ist, Ihren Vorschlag umzusetzen. Aber auch dies ist nicht zwingend erforderlich.

Sie müssen kein toller Autor sein. Andere werden im Rahmen es Prozesses prüfen was Sie erstellen und ggf. darauf hinweisen, was ihrer Meinung nach verbessert werden könnte. Zum Beispiel könnte Ihnen auffallen, das ein Absatz dieses Handbuchs unklar ist, auch wenn Sie nicht genau wissen wie er besser formuliert wäre. Wenn Sie dies jetzt in einer eMail erwähnen, können anschließend andere eine bessere Formulierung finden.

17.3 Übersetzungen

Das Obnam Handbuch und die Manpage sind in Englisch geschrieben und wurden auf Deutsch übersetzt. Andere Sprachen sind herzlich willkommen.

Der Autor dieses Handbuchs ist nicht besonders vertraut mit dem Prozess der Übersetzung und wünscht sich, das jemand anders dieses Kapitel füllt.

Die Obnam Benutzerschnittstelle kann momentan nicht übersetzt werden, um dies zu erreichen sind Änderungen am Code notwendig. Hilfe bei diesen Änderungen wären nett...

17.4 Entwicklung des Codes

Vorausgesetzt, Sie wissen bereits wie man programmiert, ist es ziemlich einfach mit der Obnam Code-Base zu arbeiten. Zumindest sollte es so sein: Wenn Sie Schwierigkeiten haben, fragen Sie und weisen darauf hin, was unklar oder falsch ist.

Ziehen Sie Sich die Quellen vom git Server und lesen Sie das `README` um Einzelheiten darüber zu erfahren, wie Sie loslegen könnnen, die automatisierten Tests ausführen, und wie Sie Patches senden können. Auf der Webseite finden Sie einige Entwickler-Dokumenation, inkl. Erklärungen zu den on-disk Datenstrukturen.

Nicht-triviale Änderungen am Code sollten in einer Form gesendet werden, mit der git umgehen kann. Das könnten ein Patch sein, den Sie an die Mailingliste schicken, oder auch eine URL von der die Änderungen gemerged werden können.

17.5 Projektführung

Das Obnam-Projekt hat eine eher informelle Führungsstruktur: Der Gründer, Lars Wirzenius, hat alle Macht, jeder andere keine. Mit zunehmendem Wachstum des Projekts wird sich das ändern.

Wenn es irgendwo ein Zwischenmenschliches Problem gibt, zum Beispiel wenn sich jemand daneben benimmt, ist es am besten, das direkt an Lars zu melden. Sollte Lars das Problem sein ist es am besten ihn direkt zur Rede zu stellen.

Kapitel 18

Anhang: Fehlermeldungen

Dieser Anhang listet sämtliche Obnam-Fehlermeldungen auf. Möglicherweise sehen Sie andere noch Fehlermeldungen während Sie Obnam benutzen. Diese sind hier nicht aufgeführt, weil sie nicht von Obnam stammen.

Die Meldungen sind zwei Mal aufgeführt: Einmal kurz in Reihenfolge der Fehlernummern und dann nochmal ausführlich in alphabetischer Reihenfolge.

18.1 Nach Fehlercode

- `R018FCX ToplevelIsFileError`
- `R01F56X RepositorySettingMissingError`
- `R02C17X HardlinkError`
- `R0B15DX RepositoryGenerationDoesNotExist`
- `R0BE94X RepositoryClientNotLocked`
- `R0C79EX GpgError`
- `R0F22CX URLSchemeAlreadyRegisteredError`
- `R0FC21X SetMetadataError`
- `R169C6X MissingFilterError`
- `R173AEX NoFilterTagError`
- `R1A025X RepositoryClientKeyNotAllowed`
- `R1CA00X ClientDoesNotExistError`
- `R22E66X SizeSyntaxError`
- `R24424X RepositoryClientDoesNotExist`
- `R283A6X UnitNameError`
- `R2FA37X WrongNumberOfGenerationSettingsError`
- `R338F2X BackupRootMissingError`
- `R3B42AX WrongNumberOfGenerationsForVerify`
- `R3E151X RepositoryFileDoesNotExistInGeneration`
- `R3E1C1X RestoreTargetNotEmpty`
- `R41CE6X RepositoryClientAlreadyExists`
- `R43272X RepositoryChunkDoesNotExist`
- `R45B50X DuplicatePeriodError`

- R47416X WrongHostKeyError
- R4C3BCX BackupErrors
- R57207X RepositoryClientGenerationUnfinished
- R5914DX InvalidPortError
- R5F98AX NoHostKeyError
- R681AEX LockFail
- R6A098X RepositoryGenerationKeyNotAllowed
- R6C1C8X RepositoryClientListNotLocked
- R6EAF2X RepositoryClientLockingFailed
- R7137EX BagIdNotSetError
- R79699X RepositoryFileKeyNotAllowed
- R79ED6X BackupRootDoesNotExist
- R7B8D0X FileNotFoundError
- R826A1X UnknownVFSError
- R8AAC1X NoHostKeyOfWantedTypeError
- R8F974X RepositoryChunkIndexesLockingFailed
- R91CA1X ShowFirstGenerationError
- R9808DX ForgetPolicySyntaxError
- RA4F35X RootIsNotADirectory
- RA5942X WrongNumberOfGenerationsForDiffError
- RA7D64X UnknownRepositoryFormatWanted
- RA881CX RepositoryChunkContentNotInIndexes
- RA920EX NotARepository
- RABC26X FuseModuleNotFoundError
- RB1048X RepositoryClientListLockingFailed
- RB4324X GAImmutableError
- RB8E98X WrongURLSchemeError
- RB927BX SeparatorError
- RBF6DDX RepositoryAccessError
- RCB0CAX KeyAuthenticationError
- RCE08AX ObnamIOError
- RCEF5CX MallocError
- RD5FA4X ObnamSystemError
- RD6259X RestoreErrors
- RDF30DX Fail
- RE187FX RepositoryChunkIndexesNotLocked
- REFB32X RepositoryClientHasNoGenerations
- RF4EFDX UnknownRepositoryFormat

18.2 Nach Name

BackupErrors (R4C3BCX) Fehler während des Backups

BackupRootDoesNotExist (R79ED6X) Backup root existiert nicht oder ist kein Verzeichnis: {root}

BackupRootMissingError (R338F2X) Kein Backup root angegeben

BagIdNotSetError (R7137EX) Bag id not set: Kann keinen Blob anfügen (Programmierfehler)

ClientDoesNotExistError (R1CA00X) Client {client} existiert nicht im Repository {repo}

DuplicatePeriodError (R45B50X) Forget policy darf Zeitraum nicht duplizieren ({period}): {policy}

Fail (RDF30DX) {filename}: {reason}

FileNotFoundError (R7B8D0X) FUSE: Datei nicht gefunden: {filename}

ForgetPolicySyntaxError (R9808DX) Syntaxfehler in der forget policy: {policy}

FuseModuleNotFoundError (RABC26X) Konnte das Modul "fuse" nicht laden, installieren Sie ggf. python-fuse

GAImmutableError (RB4324X) Attempt to modify an immutable GADirectory

GpgError (R0C79EX) gpg failed with exit code {returncode}: {stderr}

HardlinkError (R02C17X) Kann keine Hardlinks mittels SFTP erstellen

Dies beruht auf einer Einschränkung der Python paramiko Bibliothek welche Obnam für SSH / SFTP benutzt.

InvalidPortError (R5914DX) Ungültige Port-Nummer {port} in {url}: {error}

KeyAuthenticationError (RCB0CAX) Kann nicht mittels Schlüssel am SSH server authentifizieren

LockFail (R681AEX) Konnte lock nicht erzeugen {lock_name}: {reason}

MallocError (RCEF5CX) malloc out of memory bein Aufruf von {function}

MissingFilterError (R169C6X) Unknown filter tag: {tagname}

NoFilterTagError (R173AEX) Kein filter tag gefunden

NoHostKeyError (R5F98AX) Kein host key für {hostname} gefunden

NoHostKeyOfWantedTypeError (R8AAC1X) No known type {key_type} host key for {hostname}

NotARepository (RA920EX) {url} scheint kein Obnam repository zu sein

ObnamIOError (RCE08AX) I/O error: {filename}: {errno}: {strerror}

ObnamSystemError (RD5FA4X) Systemfehler: {filename}: {errno}: {strerror}

RepositoryAccessError (RBF6DDX) Repository existiert nicht oder es kann nicht drarauf zugegriffen werden: {error}

RepositoryChunkContentNotInIndexes (RA881CX) Repository chunk index hat keinen Inhalt

RepositoryChunkDoesNotExist (R43272X) Repository enthält chunk {chunk_id} nicht, der unter {filename} sein sollte.

RepositoryChunkIndexesLockingFailed (R8F974X) Repository chunk indexes are already locked

RepositoryChunkIndexesNotLocked (RE187FX) Repository chunk indexes are not locked

RepositoryClientAlreadyExists (R41CE6X) Repository client {client_name} existiert schon

RepositoryClientDoesNotExist (R24424X) Repository client {client_name} existiert noch nicht

RepositoryClientGenerationUnfinished (R57207X) Kann keine neue Generation für {client_name} anlegen: Vorherige ist noch nicht beendet (Programmierfehler)

RepositoryClientHasNoGenerations (REFB32X) Client {client_name} hat keine Generationen

RepositoryClientKeyNotAllowed (R1A025X) Client {client_name} nutzt das Repository-Format {format}, das keinen Schlüssel {key_name} für die Clients erlaubt.

RepositoryClientListLockingFailed (RB1048X) Repository client list could not be locked

RepositoryClientListNotLocked (R6C1C8X) Repository client list is not locked

RepositoryClientLockingFailed (R6EAF2X) Repository client {client_name} could not be locked

RepositoryClientNotLocked (R0BE94X) Repository client {client_name} is not locked

RepositoryFileDoesNotExistInGeneration (R3E151X) Client {client_name}, Generation {genspec} enthält keine Datei {filename}

RepositoryFileKeyNotAllowed (R79699X) Client {client_name} uses repository format {format} which does not allow the key {key_name} to be use for files

RepositoryGenerationDoesNotExist (R0B15DX) Kann angegebene Generation {gen_id!r} für Client {client_name} nicht finden

RepositoryGenerationKeyNotAllowed (R6A098X) Client {client_name} uses repository format {format} which does not allow the key {key_name} to be used for generations

RepositorySettingMissingError (R01F56X) Keine –repository-Einstellung gefunden. Setzen Sie die Option auf der Kommandozeile oder in einer Konfigurationsdatei.

RestoreErrors (RD6259X) Es traten Fehler bei der Wiederherstllung auf.

Die vorhergehende Fehlermeldung enthält Details.

RestoreTargetNotEmpty (R3E1C1X) The restore –to directory ({to}) is not empty.

RootIsNotADirectory (RA4F35X) {baseurl} ist kein Verzeichnis, aber ein VFS root muss ein Verzeichnis sein

SeparatorError (RB927BX) Forget policy muss durch Kommata getrennt sein, siehe: {position}: {policy}

SetMetadataError (R0FC21X) {filename}: Couldn't set metadata {metadata}: {errno}: {strerror}

ShowFirstGenerationError (R91CA1X) Can't show first generation. Use 'obnam ls' instead

SizeSyntaxError (R22E66X) "{size}" is not a valid size

ToplevelIsFileError (R018FCX) File at repository root: {filename}

URLSchemeAlreadyRegisteredError (R0F22CX) VFS URL scheme {scheme} already registered

UnitNameError (R283A6X) "{unit}" is not a valid unit

UnknownRepositoryFormat (RF4EFDX) Unknown format {format} at {url}

`UnknownRepositoryFormatWanted (RA7D64X)` Unknown format {format} requested

`UnknownVFSError (R826A1X)` Unknown VFS type: {url}

`WrongHostKeyError (R47416X)` Der SSH server {hostname} zeigte uns einen ungültigen öffentlichen Schlüssel.

Dies kann an einem veralteten host key in Ihrer "known hosts"-Datei liegen. Sollte das so sein, benutzen Sie `ssh-key -R` um ihn zu entfernen. Es könnte aber auch ein Zeichen dafür sein, daß Sie Opfer eines Man-in-the-middle Angriffs werden. Seinen Sie vorsichtig.

`WrongNumberOfGenerationSettingsError (R2FA37X)` Der `restore`-Befehl benötigt exakt eine Generation

`WrongNumberOfGenerationsForDiffError (RA5942X)` Benötige eine oder zwei Generationen

`WrongNumberOfGenerationsForVerify (R3B42AX)` verify must be given exactly one generation

`WrongURLSchemeError (RB8E98X)` SftpFS used with non-sftp URL: {url}

Kapitel 19

Siehe auch...

Dieses Kapitel gibt Hinweise auf weitere Informationen über Obnam, Backups, und verwandte Dinge. Zur Zeit ist dies eine sehr kurze Liste, aber Vorschläge zu Dingen, die hinzugefügt werden können akzeptieren wir gern.

- Obnam Homepage (englisch): http://obnam.org.
  - Kurze Tutorials, Download-Links, eine FAQ, Kontaktinformationen usw.
- Lars Wirzenius, interessante Blog-Tags.
  - http://blog.liw.fi/tag/backups/
  - http://blog.liw.fi/tag/obnam/
- Cache directory tag Standard: http://www.bford.info/cachedir/
  - http://liw.fi/cachedir/ Ein Programm, um die Tag-Dateien zu verwalten

Kapitel 20

Rechtliches

Dieses Werk steht unter der GNU General Public License Version 3 oder,je nach Vorliebe, einer beliebigen neueren Version.

> Copyright 2010-2013 Lars Wirzenius
>
> This program is free software: you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation, either version 3 of the License, or (at your option) any later version.
>
> This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.
>
> You should have received a copy of the GNU General Public License along with this program. If not, see http://www.gnu.org/licenses/.

Eine Kopie der GPL liegt in Form der Datei `COPYING` dem Quellcode bei und kann zusätzlich unter der oben angegebenen URL abgerufen werden.

Dieses Handbuch (alle Inhalte des Verzeichnisses `manual` in den Quellen) steht zusätzlich unter der Creative Commons Attribution 4.0 International Lizenz. Sie können sich aussuchen ob Sie die GPL or die CC-Lizenz für das Handbuch anwenden.

Eine Kopie der Creative Commons Lizenz liegt in Form der Datei `CC-BY-SA-4.0.txt` den Quellen bei und kann zusätzlich online unter folgender URL eingesehen werden (Englisch): http://creativecommons.org/licenses/by-sa/4.0/legalcode

Kapitel 21

Die Entwicklung von Obnam unterstützen

Obnam ist freie Software: Sie erhalten vollen Zugriff auf den Quellcode, können die Software so modifizieren wie Sie wollen, und Sie können Kopien der Software in ihrer ursprünglichen oder geänderten Form verteilen. Obnam ist auch komplett kostenlos.

Eines der Ziele von Obnam ist es sicherzustellen, dass jeder Zugang zu einer vernünftigen Backup-Software hat, ohne jemand anderem dafür verpflichtet zu sein. Sie können Obnam verwenden und Ihre Backups überall dort speichern wo es Ihnen passt, die Obnam Entwickler haben dazu nichts zu sagen.

Allerdings erfordert die Entwicklung von Obnam einige Ressourcen. Obnam wird in erster Linie von Lars Wirzenius, seinem ursprünglichen Autor, in seiner Freizeit entwickelt (Hi!). Wenn Sie helfen möchten die Entwicklung zu unterstützen, finden Sie hier eine Liste der Dinge, die Sie tun können:

- Senden Sie Korrekturen und Verbesserungen, entweder am Code oder der Dokumentation.
- Spenden Sie etwas an den Autor. Anregungen finden Sie unter http://obnam.org/donate/
- Beauftragen Sie den Autor mit der Obnam-Weiterentwicklung. Treten Sie dazu mit ihm privat in Kontakt (liw@liw.fi).

Beachten Sie, dass all dies optional ist. Wenn Sie Obnam einfach benutzen und glücklich dabei sind, ist das völlig OK.