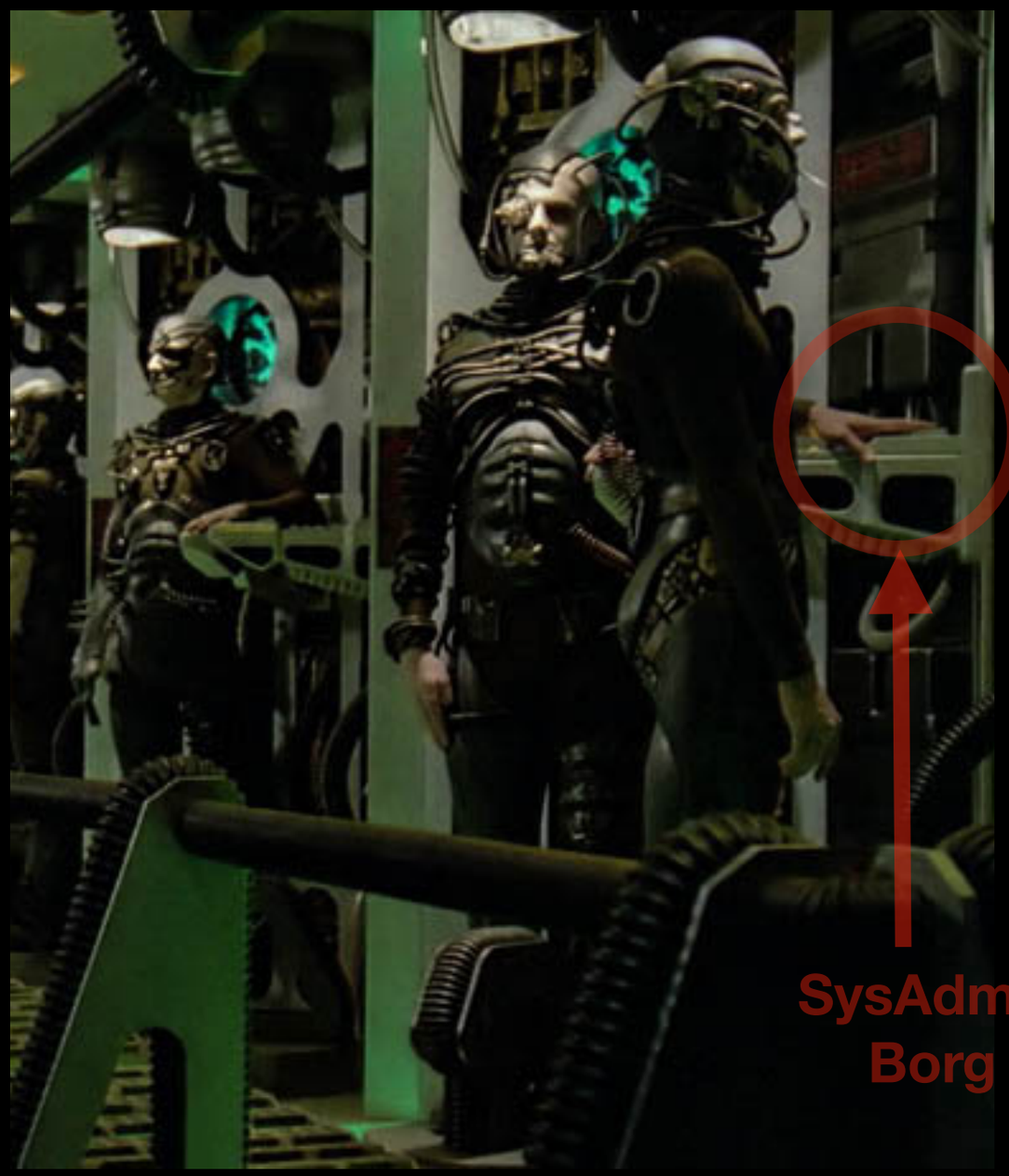
Kein Backup
Kein Mitleid

We are the borg, you will be assimilated!

BORG

- Borg?
- Anlegen
- Backup
- Restore
- Sync
- Automation



BORG?

- Borg ist eine Backup/Restore Anwendung
- Borg ist ein fork von Attic
- Attic wurde von Jonas Borgström gegründet

- Borg kann lokale Backups anlegen oder auf entfernten Servern
- Backups sind komprimiert und dedupliziert
- Backups können verschlüsselt werden
- Schlüssel können lokal oder im Repository gelagert werden
- Borg kann pruning

Installieren

Distribution	Source	Kommando
ArchLinux	[community]	pacman -S borg
Debian/Raspbian/Ubuntu	Debian packages	apt install borgbackup
Gentoo	eBuild	emerge borgbackup
GNU Guix	GNU Guix	guix package —install borg
RHEL/Fedora	Fedora official repository	dnf install borgbackup
FreeBSD	FreeBSD ports	cd /usr/ports/archivers/py-borgbackup && make install clean
MacOS	Brew cask	brew cask install borgbackup
Mageia	cauldron	urpmi borgbackup
NetBSD	pkgsrc	pkg_add py-borgbackup
OpenBSD	OpenBSD ports	pkg_add borgbackup
OpenIndiana	OpenIndiana hipster repository	pkg install borgbackup
openSuSe	openSUSE official repository	zypper in borgbackup

Abhängigkeiten

- **Python 3** >= 3.4.0, plus development headers. Even though Python 3 is not the default Python version on most systems, it is usually available as an optional install.
- **OpenSSL** >= 1.0.0, plus development headers.
- **libacl** (which depends on **libattr**), both plus development headers.
- We have bundled code of the following packages, but borg by default (see setup.py if you want to change that) prefers a shared library if it can be found on the system (lib + dev headers) at build time:
  - **liblz4** >= 1.7.0 (r129)
  - **libzstd** >= 1.3.0
  - **libb2**
- some Python dependencies, pip will automatically install them for you
- optionally, the **llfuse** Python package is required if you wish to mount an archive as a FUSE filesystem. See setup.py about the version requirements.

Backup initialisieren

- Angreifer muss passphrase und keyfile kennen:
  - borg init -e keyfile user@server:pfad/{hostname}
- Angreifer muss passphrase kennen:
  - borg init -e repokey user@server:pfad/{hostname}
- Angreifer hat alles:
  - borg init -e none user@server:pfad/{hostname}

borg init -e keyfile test-{hostname}

Enter new passphrase:
Enter same passphrase again:
Do you want your passphrase to be displayed for verification? [yN]: N

By default repositories initialized with this version will produce security errors if written to with an older version (up to and including Borg 1.0.8). If you want to use these older versions, you can disable the check by running:

borg upgrade --disable-tam test-DarkMatter

See https://borgbackup.readthedocs.io/en/stable/changes.html#pre-1-0-9-manifest-spoofing-vulnerability for details about the security implications.

IMPORTANT: you will need both KEY AND PASSPHRASE to access this repo!
Use "borg key export" to export the key, optionally in printable format.
Write down the passphrase. Store both at safe place(s).

borg key export test-{hostname} keyfile.txt
cat keyfile.txt

BORG_KEY
8e1def5ee32d397d89ba838f3e5bd4b2b4a9712615c602ed91a7ae3cf5d
0c7dchqlhbGdvcml0aG2mc2hhMjU2pGRhdGHaAN6w+6+xlgQkNDW8O
46TMt357qDcNlq+A2WMtcasLUu3XBGpF9n6dL3Mt8elHLBn3x43bW0n
AYpo9c86yKJM2nOK8ujjCqakdm7tx+mFbf3KxRFVSmKHlx7kav+t41Q3E
3SVaIDEBF1p6yr2jCCZg/
NmOdjtaHkIAA55vgvXCt+8F7ji3aM2f8zFAdGacOdgPobQh1Wqp6+oKd
A8w3ACUJ6yxw9Kl2O3gO1PptOseg1iTxA4FghkYazlyDOqoWKCoIOZZo
A8wNrz5yPNFDzfeFqL9wbh4gUz48omrHOs1kmkaGFzaNoAIDruNQV+A
h82anAf3EomPfkOyfLzTd/
y8R5aFFgo+HwqmI0ZXJhdGIvbnPOAAGGoKRzYWx02gAgCMgZOygtR
8c/rEUgUiLp3D5QomnArx3hI2PwG5KSc2qndmVyc2IvbgE=

Backup anlegen

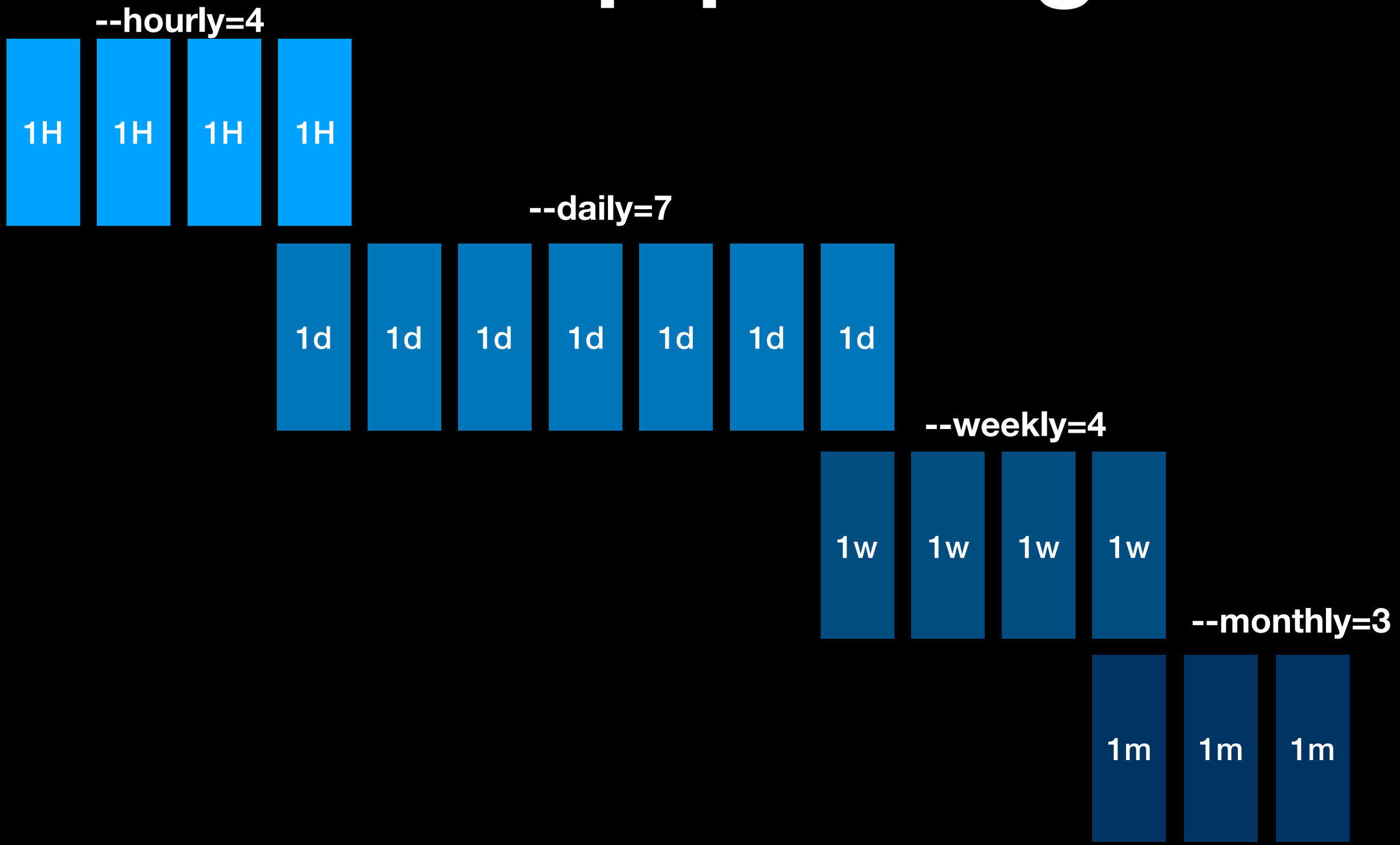
- Nun müssen wir nur noch etwas reintun:
  (-x bleib in den genannten Filesystemen)

  - borg create -xC zlib,9 user@server:pfad/{hostname}::{now:%Y%m%d-%H%M}' / /boot

- Und da wir eh schon dabei sind, räumen wir auch noch auf:

  - borg prune -H 6 -d 7 -w 4 -m 6 user@server:pfad/{hostname}

Backup pruning



Backup nachgucken

- Was ist nun drin?
  borg list user@server:pfad/{hostname}
  20190321-1741 Thu, 2019-03-21 17:41:49 [a47a1d6deb46413aac98af0311a894eb00458be09108c73dc2aa60b81095407c]

```
borg info user@server:/pfad/{hostname}::20190321-1755
Archive name: 20190321-1755
Archive fingerprint: 3f0a8dc1d65787137986319f907b1e6ed6cadc9dd470353f515ca4affd5721b1
Comment:
Hostname: DarkMatter
Username: brandy
Time (start): Thu, 2019-03-21 17:55:29
Time (end): Thu, 2019-03-21 17:55:29
Duration: 0.02 seconds
Number of files: 34
Command line: borg create -xC zlib,9 'user@server:/pfad/{hostname}::{now:%Y%m%d-%H%M}' Downloads/P/
Utilization of maximum supported archive size: 0%
------------------------------------------------------------------------------
                       Original size      Compressed size    Deduplicated size
This archive:               28.66 MB             28.01 MB              5.13 kB
All archives:               57.35 MB             56.02 MB             28.02 MB

                       Unique chunks         Total chunks
Chunk index:                      45                   86
```

Backup nachgucken

- Das geht eleganter und spuckt die Infos des letzten Backuplaufs aus:
  - borg list user@server:pfad/{hostname}
  - borg info user@server:pfad/{hostname}::`borg list user@server:pfad/{hostname}|tail -n1|cut -d" " -f1`

Backup überprüfen

- Man kann die Integrität des Backups überprüfen und ggf. reparieren:

- borg check user@server:pfad/{hostname}

- borg check --repair user@server:pfad/{hostname}

Backup ist gut

Restore ist besser!

Restore

- Es gibt mehrere Möglichkeiten
  - extract
    Holt eine Datei oder Dateiverzeichnisse wieder
  - export-tar
    man bekommt das ganze als tarball
  - mount
    einfach als fuse filesystem mountain und drin spazieren gehen

Restore

- Extract am Beispiel
  - borg extract --stdout user@server:pfad-{hostname}::20190321-1755 | less
- Export-tar am Beispiel
  - borg export-tar —tar-filter=“gzip -9” user@server:pfad-{hostname}::20190321-1755 yesterday.tar.gz

Restore

- Mount am Beispiel
  - borg mount user@server:pfad/{hostname} mnt
    ls mnt
    20190321-1731 20190321-1755
  - mount | grep mnt
    borgfs on /data/backup/mnt type fuse
    (ro,nosuid,nodev,relatime,user_id=0,group_id=0)

Lock picking

Mit borg ;)

break-lock

- Wenn eine Maschine bootet während eines Backup-Laufs kann es zu einem gelocktem Zustand kommen.

- break-lock
  Damit kann man diesen Schutz aufheben. ABER immer vorher nachsehen, dass wirklich KEIN borg Prozess darauf zugreift.

- borg break-lock user@server:pfad/{hostname}

Aufräumen

Die üblichen Hausmeistereien…

Delete

- Man kann damit einzelne Zeitpunkt löschen, oder auch das ganze Archiv, dabei wird der Speicherplatz freigegeben

- borg delete user@server:pfad/{hostname}::20190321-1755
  löscht nur diesen Zeitpunkt

- borg delete user@server:pfad/{hostname}
  löscht das ganze Repository

Sync?

- Um mehrere Repositorien zu unterstützen kann man folgendes tun:
  - Direkt mit Borg mehrere Repositorien anlegen
  - Oder per rsync zwischen Standorten verteilen

Ansible

```
- hosts: backup01.srv.local
  vars:
    user: backup
    group: backup
    home: /home/backup
    pool: "{{ home }}/repos"
    auth_users:
      - host: johndoe.clnt.local
        key: "{{ lookup('file', '/path/to/keys/
johndoe.clnt.local.pub') }}"
      - host: web01.clnt.local
        key: "{{ lookup('file', '/path/to/keys/
web01.clnt.local.pub') }}"
      - host: app01.clnt.local
        key: "{{ lookup('file', '/path/to/keys/
app01.clnt.local.pub') }}"
  tasks:
  - pacman: name=borg state=latest
update_cache=yes
  - group: name="{{ group }}" state=present
  - user: name="{{ user }}" shell=/bin/bash
home="{{ home }}" createhome=yes
group="{{ group }}" groups= state=present
  - file: path="{{ home }}" owner="{{ user }}"
group="{{ group }}" mode=0700 state=directory
  - file: path="{{ home }}/.ssh" owner="{{ user }}"
group="{{ group }}" mode=0700 state=directory
  - file: path="{{ pool }}" owner="{{ user }}"
group="{{ group }}" mode=0700 state=directory
  - authorized_key: user="{{ user }}"
                    key="{{ item.key }}"
                    key_options='command="cd
{{ pool }}/{{ item.host }};borg serve --restrict-to-
path {{ pool }}/{{ item.host }}",restrict'
    with_items: "{{ auth_users }}"
  - file: path="{{ home }}/.ssh/authorized_keys"
owner="{{ user }}" group="{{ group }}" mode=0600
state=file
  - file: path="{{ pool }}/{{ item.host }}"
owner="{{ user }}" group="{{ group }}" mode=0700
state=directory
    with_items: "{{ auth_users }}"
```

Etwas Übersicht

bitte

borg commands (create, extract, …)

Data access (Archive.iter_items, item …)

Compression

Encryption

Repository

Remote repository

ssh

sshd

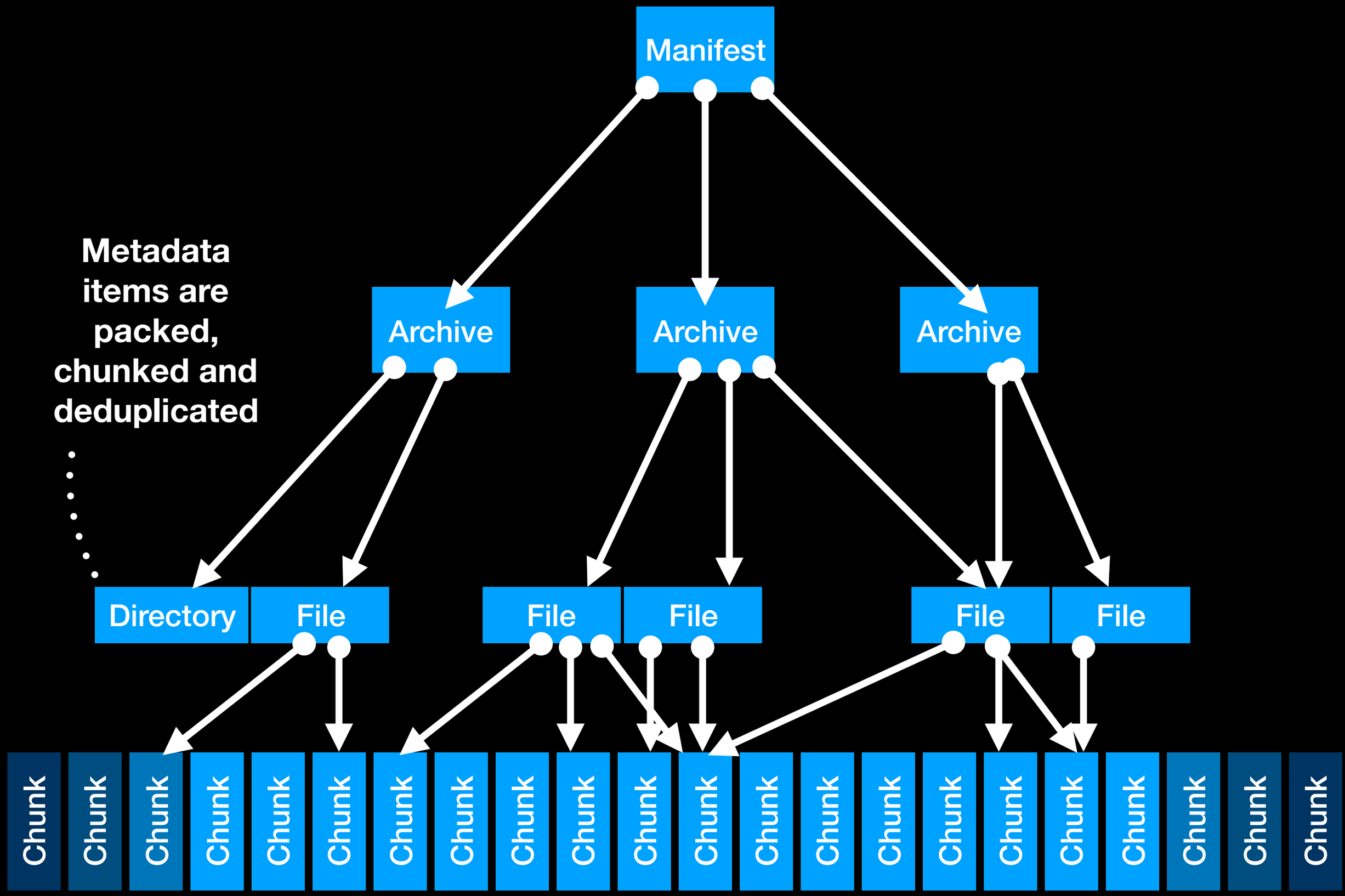
borg serve

Client

OS provided

Server / Repository

Metadata items are packed, chunked and deduplicated



borg backup

https://borgbackup.readthedocs.io/

Q&A

Mathias Brandstetter
info@linuxpinguin.de

@filid