Bacula? Aber sicher!

Transport- und Backupverschlüsselung

Mathias Burger

http://www.mabu-logic.de

15. September 2010

v0.8, powered by LaTeX

Agenda

1. **Transportverschlüsselung**
   - Weshalb ist das so wichtig?
   - Was wird benötigt?
   - Wie wird es umgesetzt?
2. **Backupverschlüsselung**
   - Weshalb ist das so wichtig?
   - Was wird benötigt?
   - Wie wird es umgesetzt?
3. **Ausblick**

Begriffserklärung

Transportverschlüsselung

- Kryptographische Protokolle für sichere Kommunikation in unsicheren Netzwerken
- Bacula verwendet Transport Layer Security (TLS)

Weshalb überhaupt?

- Unverschlüsselter Netzwerk-Datenverkehr kann
  - mitgehört werden
  - manipuliert werden
- Böswilliger Kommunikationspartner könnte Schaden anrichten

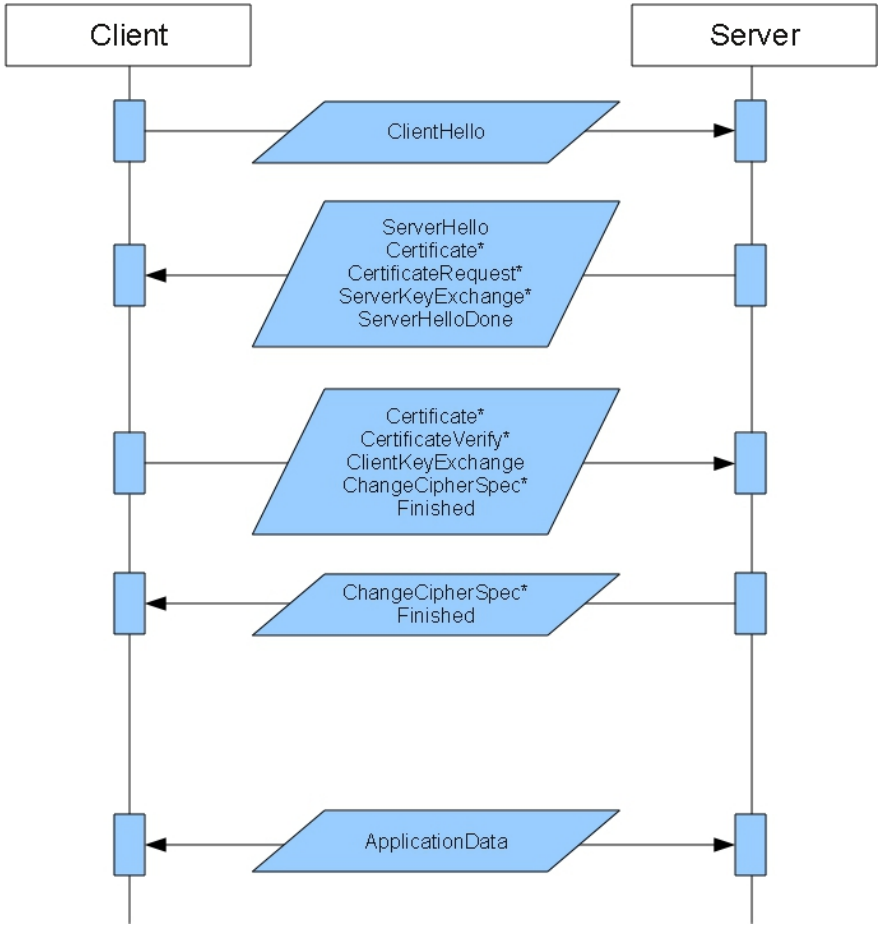
Was passiert?

- Authentifizierung
  - Ist mein Gegenüber wirklich derjenige, für den er sich ausgibt?
- Authorisierung
  - Darf derjenige überhaupt mit mir kommunizieren?
  - Bacula kann das Common Name Attribut (CN) des Zertifikats überprüfen
- Verschlüsselter Datenaustausch

Was passiert nicht?

- Handhabung zurückgezogener Zertifikate
  - Certificate Revocation Lists (CRL)
  - Online Certificate Status Responder (OCSP)
- Spezifikation der möglichen Verschlüsselungsprotokolle

TLS Ablaufschema



* = optionale Nachrichten

1. Verbindung aushandeln
2. Daten übertragen

Abbildung: TLS Ablaufschema

Wir benötigen

- gegenseitige Authentifizierung (sicherer)
  - Eine Zertifizierungsstelle
  - Serverkeys und -zertifikate
  - Clientkeys und -zertifikate
  - CA-Zertifikat auf Client und Server

Bacula Kommunikationsschema

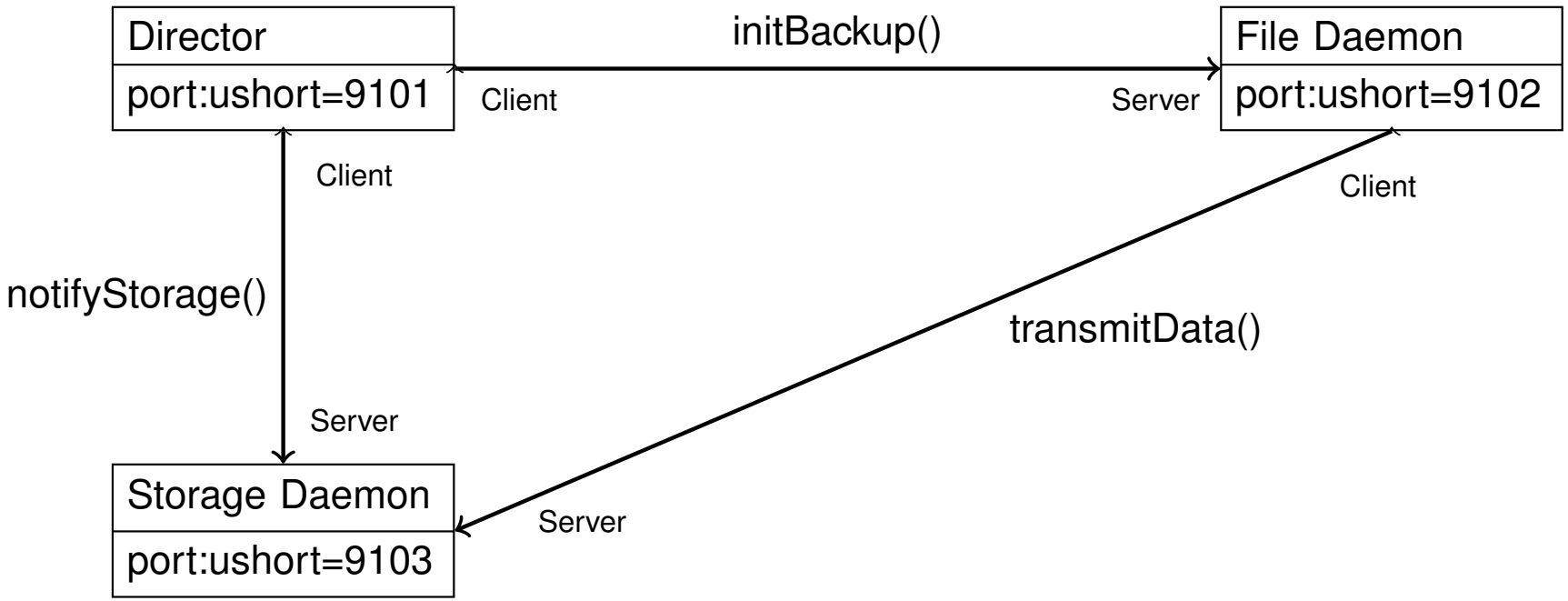


Abbildung: Bacula Kommunikationsschema

Bacula Schlüsselverteilung

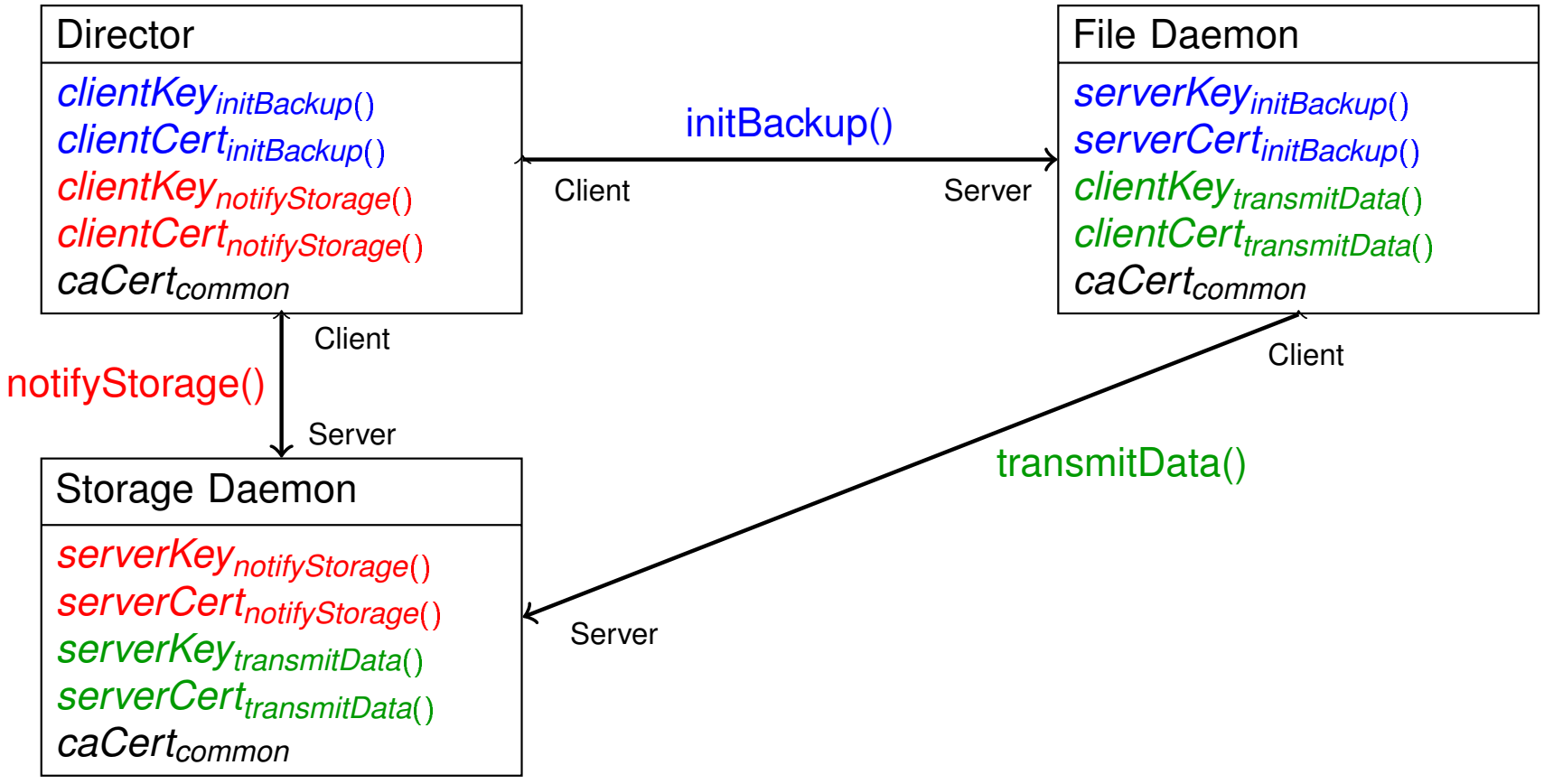
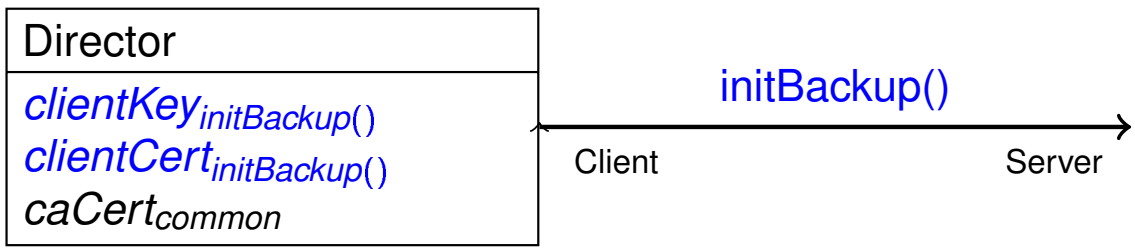


Abbildung: Bacula Schlüsselverteilung

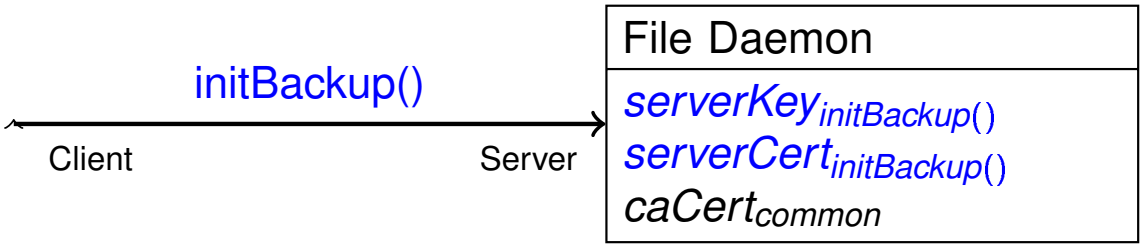
TLS zwischen DIR und FD: initBackup()



TLS zwischen DIR und FD: initBackup()

```
Client {
  Name = ganymed19-fd
  Address = ganymed19
  FDPort = 9102
  Catalog = MyCatalog
  Password = "1Xda/bXogmnuT7OoO9d8MR1kFU2DLbzE8Wq+RuXrwpIl"
  File Retention = 30 days
  Job Retention = 6 months
  AutoPrune = yes

  # TLS DIR-FD, client context
  # bacula-dir.conf, Client
  TLS Enable = yes
  TLS Require = yes
  TLS Certificate = /etc/bacula/tls/client@ganymed17-cert.pem
  TLS Key = /etc/bacula/tls/client@ganymed17-key.pem
  TLS CA Certificate File = /etc/bacula/tls/BaculaCA-cacert.pem
  # TLS DIR-FD end
}
```

TLS zwischen DIR und FD: initBackup()



TLS zwischen DIR und FD: initBackup()

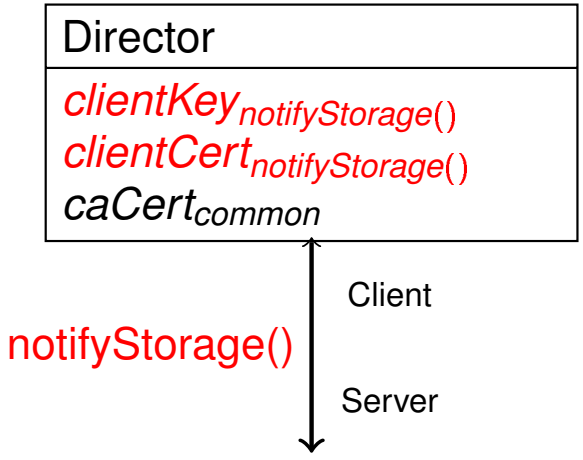
```
Director {
  Name = ganymed-dir
  Password = "1Xda/bXogmnuT7OoO9d8MR1kFU2DLbzE8Wq+RuXrwpIl"

  # TLS DIR-FD, server context
  # bacula-dir.conf, Director
  TLS Enable = yes
  TLS Require = yes
  TLS Certificate = /etc/bacula/tls/ganymed19-cert.pem
  TLS Key = /etc/bacula/tls/ganymed19-key.pem
  TLS CA Certificate File = /etc/bacula/tls/BaculaCA-cacert.pem
  TLS Verify Peer = yes
  TLS Allowed CN = "client@ganymed17"
  TLS DH File = /etc/bacula/tls/dh4096_fd.pem
  # TLS DIR-FD end
}
```

TLS zwischen DIR und FD: initBackup()

Testen

```
bconsole
*status client
```

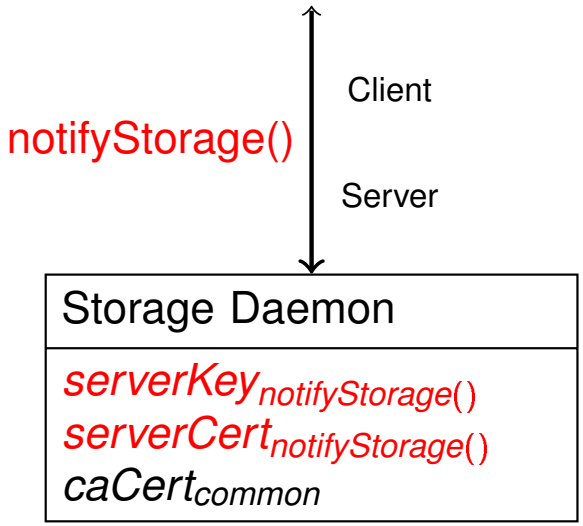
TLS zwischen DIR und SD: notifyStorage()



TLS zwischen DIR und SD: notifyStorage()

```
Storage {
  Name = File
  Address = ganymed18
  SDPort = 9103
  Password = "aVXCmbfrInE6y4s8nzgseS5ZzOxMX7LoSZXDBKU2P1SA"
  Device = FileStorage
  Media Type = File

  # TLS DIR-SD, client context
  # bacula-dir.conf, Storage
  TLS Enable = yes
  TLS Require = yes
  TLS Certificate = /etc/bacula/tls/client@ganymed17-cert.pem
  TLS Key = /etc/bacula/tls/client@ganymed17-key.pem
  TLS CA Certificate File = /etc/bacula/tls/BaculaCA-cacert.pem
  # TLS DIR-SD end
}
```

TLS zwischen DIR und SD: notifyStorage()



TLS zwischen DIR und SD: notifyStorage()

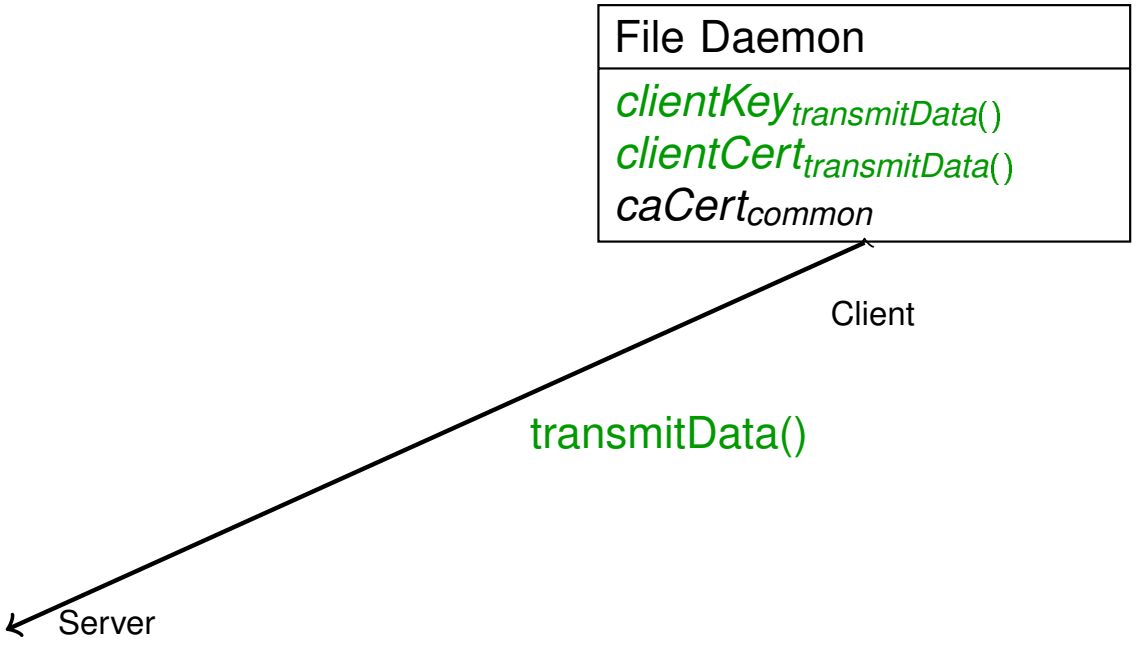
```
Director {
  Name = ganymed-dir
  Password = "aVXCmbfrInE6y4s8nzgseS5ZzOxMX7LoSZXDBKU2P1SA"

  # TLS DIR-SD, server context
  # bacula-sd.conf, Director
  TLS Enable = yes
  TLS Require = yes
  TLS Certificate = /etc/bacula/tls/ganymed18-cert.pem
  TLS Key = /etc/bacula/tls/ganymed18-key.pem
  TLS CA Certificate File = /etc/bacula/tls/BaculaCA-cacert.pem
  TLS Verify Peer = yes
  TLS Allowed CN = client@ganymed17
  TLS DH File = /etc/bacula/tls/dh4096_server.pem
  # TLS DIR-SD end
}
```

TLS zwischen DIR und SD: notifyStorage()

Testen

```
bconsole
*status storage
```

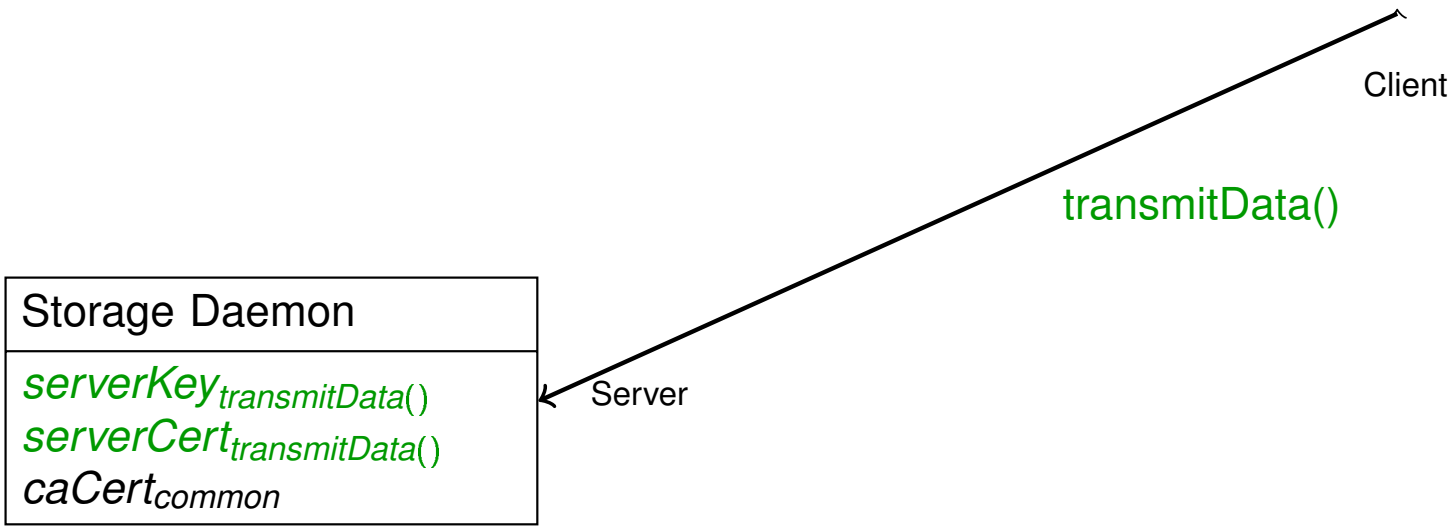
TLS zwischen FD und SD: transmitData()



TLS zwischen FD und SD: transmitData()

```
FileDaemon {
  Name = ganymed19-fd
  FDport = 9102
  WorkingDirectory = /var/lib/bacula
  Pid Directory = /var/run
  Maximum Concurrent Jobs = 20

  # TLS FD-SD, client context
  # bacula-fd.conf, Storage
  TLS Enable = yes
  TLS Require = yes
  TLS Certificate = /etc/bacula/tls/client@ganymed19-cert.pem
  TLS Key = /etc/bacula/tls/client@ganymed19-key.pem
  TLS CA Certificate File = /etc/bacula/tls/BaculaCA-cacert.pem
  # TLS FD-SD end
  # PKI-DATAENC
  PKI Signatures = Yes
  PKI Encryption = Yes
  PKI Keypair = "/etc/bacula/encryption/ganymed19-dataenc.keypair"
  PKI Master Key = "/etc/bacula/encryption/master.cert"
  # PKI-DATAENC end
}
```

TLS zwischen FD und SD: transmitData()



TLS zwischen FD und SD: transmitData()

```
Storage {
  Name = ganymed-sd
  SDPort = 9103
  WorkingDirectory = "/var/lib/bacula"
  Pid Directory = "/var/run"
  Maximum Concurrent Jobs = 20

  # TLS SD-FD, server context
  # bacula-sd.conf, Storage
  TLS Enable = yes
  TLS Require = yes
  TLS Certificate = /etc/bacula/tls/ganymed18-cert.pem
  TLS Key = /etc/bacula/tls/ganymed18-key.pem
  TLS CA Certificate File = /etc/bacula/tls/BaculaCA-cacert.pem
  TLS Verify Peer = yes
  TLS Allowed CN = "client@ganymed17" "client@ganymed19" "client@ganymed20"
  TLS DH File = /etc/bacula/tls/dh4096_server.pem
  # TLS SD-FD end
}
```

TLS zwischen FD und SD: transmitData()

Testen

```
bconsole
*run
```

Und was ist mit bconsole?

Analog.
Director-Abschnitt in bconsole.conf für Client.
Director-Abschnitt in bacula-dir.conf für Server.

Begriffserklärung

Backupverschlüsselung

- Verschlüsselung der Backupdaten mit AES-128-CBC
- Signieren der Daten, um Datenintegrität zu gewährleisten
- Masterkeys zur Entschlüsselung bei Schlüsselverlust

Weshalb überhaupt?

- Datendiebstahl findet oftmals innerhalb des eigenen Unternehmens statt
- Kritische Unternehmensdaten sollten nur befugten Personen zugänglich sein
- Geschäftsgeheimnisse müssen sicher verwahrt werden

Was passiert?

- Dateien werden AES-128-CBC verschlüsselt
  - Ähnliche Klartext-Daten haben verschlüsselt sehr unterschiedliches Aussehen
- Daten können signiert werden
  - Eine Signatur pro Datei
  - Bei vielen kleinen Dateien steigt die Last enorm an
  - Besonders bei Schlüsseln > 512 Bit enorm bemerkbar
  - Hardwarebeschleunigung sinnvoll

Was passiert nicht?

- Meta-Daten unverschlüsselt
- Datenintegrität bei Wiederherstellung mit Masterkey

Bacula Verschlüsselungsschema

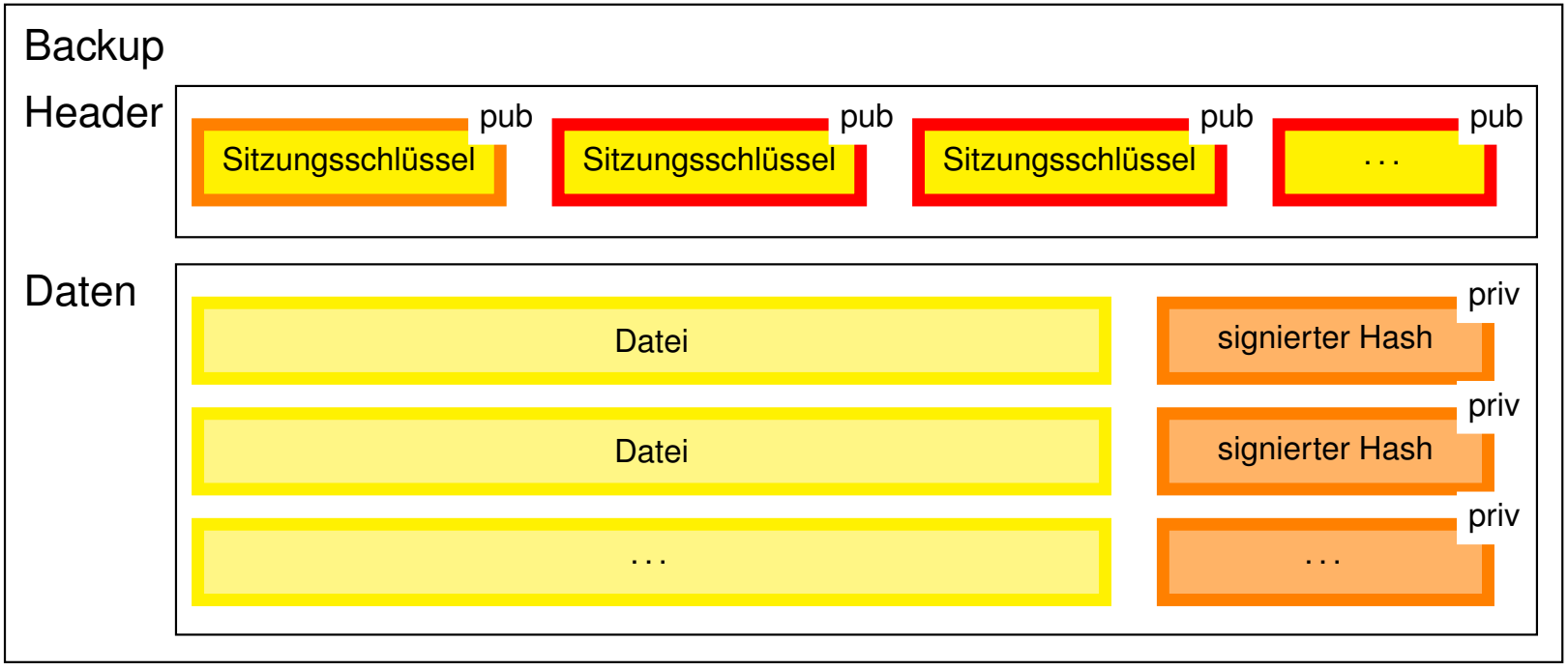


Abbildung: Bacula Backupverschlüsselungsschema

Wir benötigen

- Einen öffentlichen und privaten Schlüssel
- Masterkeys
  - Nur den öffentlichen Schlüssel speichern
  - Privaten Schlüssel sicher verwahren

Filedaemon Konfiguration

```
FileDaemon {
  Name = ganymed19-fd
  FDport = 9102
  WorkingDirectory = /var/lib/bacula
  Pid Directory = /var/run
  Maximum Concurrent Jobs = 20

  # TLS FD-SD, client context
  # bacula-fd.conf, Storage
  TLS Enable = yes
  TLS Require = yes
  TLS Certificate = /etc/bacula/tls/client@ganymed19-cert.pem
  TLS Key = /etc/bacula/tls/client@ganymed19-key.pem
  TLS CA Certificate File = /etc/bacula/tls/BaculaCA-cacert.pem
  # TLS FD-SD end
  # PKI-DATAENC
  PKI Signatures = Yes
  PKI Encryption = Yes
  PKI Keypair = "/etc/bacula/encryption/ganymed19-dataenc.keypair"
  PKI Master Key = "/etc/bacula/encryption/master.cert"
  # PKI-DATAENC end
}
```

Filedaemon Konfiguration: Restore mit Masterkey

```
FileDaemon {
  Name = ganymed17-fd
  FDport = 9102
  WorkingDirectory = /var/lib/bacula
  Pid Directory = /var/run
  Maximum Concurrent Jobs = 20

  # TLS FD-SD, client context
  # bacula-fd.conf, FileDaemon
  TLS Enable = yes
  TLS Require = yes
  TLS Certificate = /etc/bacula/tls/client@ganymed17-cert.pem
  TLS Key = /etc/bacula/tls/client@ganymed17-key.pem
  TLS CA Certificate File = /etc/bacula/tls/BaculaCA-cacert.pem
  # TLS FD-SD end
  # PKI-DATAENC
  PKI Signatures = Yes
  PKI Encryption = Yes
  PKI Keypair = "/etc/bacula/encryption/master.keypair"
  PKI Master Key = "/etc/bacula/encryption/master.cert"
  # PKI-DATAENC end
}
```

Weitere Sicherheitsthemen

- Verify Jobs
- Firewallkonfiguration
- Passwortübergabe für Katalogbackup

Fazit

- Transportverschlüsselung
- Backupverschlüsselung
- Verhindern
  - (Mit)Lesen
  - Manipulation
- Einfache Integration in Firewallumgebung
- Verify Jobs
- Wünschenswert wäre noch
  - Wahl der Verschlüsselungsverfahren (Transport & Backup)
  - Schnellere Methode für Datenintegritätscheck