# Praktikum angewandte Systemsoftwaretechnik

## Aufgabe 2

Alexander Würstlein, Moritz Strübe, Rainer Müller

Lehrstuhl Informatik 4

8. Mai 2014

# SSH-Authentifizierung mit Schlüsseln (Host)

- SSH-Authentifizierung mit einem Schlüsselpaar **ohne** Passwort
  - **Privaten** und **öffentlichen** Schlüssel mit `ssh-keygen` erzeugen

```
$ ssh-keygen -f <gruppen_name> -N ""
Generating public/private rsa key pair.
Your identification has been saved in <gruppen_name>.
Your public key has been saved in <gruppen_name>.pub.
(...)
```

  - Erzeugte Schlüssel (`gruppen_name = gruppe0`)

```
$ ls -l
-rw------- 1 thoenig users  1675  8. Mai 11:29 gruppe0
-rw-r--r-- 1 thoenig users   394  8. Mai 11:29 gruppe0.pub
```

- Port-Weiterleitung für SSH-Verbindungen (QEMU)

```
$ qemu <...> -net user,hostfwd=tcp:127.0.0.1:5022-:22
```

- Hinweis: In der Betriebsumgebung des Host-Rechners ausführen.

# SSH-Authentifizierung mit Schlüsseln (VM)

- Installation des SSH-Servers in der virtuellen Maschine

```
# apt-get install ssh openssh-server
```

- Zugriffe auf die virtuelle Maschine unter Zuhilfenahme des generierten **öffentlichen** Schlüssels
- Hinterlegen des **öffentlichen** Schlüssels

```
$ su - <vm_user>
$ mkdir .ssh
$ scp <user>@<host_ip>:~/<gruppen_name>.pub \
  /home/<vm_user>/.ssh/authorized_keys
```

- Alternative: `ssh-copy-id(1)`

- Hinweis: In der Betriebsumgebung der virtuellen Machine ausführen

# Verbindungsaufbau vom Host zur virtuellen Maschine

- Kontrollverbindung zur virtuellen Maschine aufbauen

```
$ ssh -p 5022                \
      -i <gruppen_name>      \
      <vm_user>@localhost
```

- Datenverbindung

```
$ scp -P 5022                          \
      <datei1> (...)                   \
      <vm_user>@localhost:/<vm_path>
```

- Alternative: SSHFS (benötigt root-Rechte!)

```
$ sshfs -p 5022                                          \
      -o IdentityFile=<absolute_path>/<gruppe_name>      \
      <vm_user>@localhost:/<vm_path>                     \
      <mount_point>
```

- Hinweis bei Verwendung von `sshfs(1)`: Absoluter Pfad zum Schlüssel zwingend notwendig

# Zugriff auf freigegebene Verzeichnisse des Hosts

- *Plan 9 Folder Sharing* erlaubt transparentes Einhängen freigegebener Verzeichnisse des Hosts
- Konfiguration des Gast-Kernels

```
CONFIG_NET_9P=y
CONFIG_NET_9P_VIRTIO=y
#CONFIG_NET_9P_DEBUG=y
CONFIG_9P_FS=y
CONFIG_9P_FS_POSIX_ACL=y
```

# Zugriff auf freigegebene Verzeichnisse des Hosts (Forts.)

- Freigabe von Verzeichnissen mit QEMU

```
(host) $ mkdir shared
(host) $ qemu <...> \
-virtfs local,path=shared,security_model=none,mount_tag=shared
```

- Optional: `,readonly`
- Einhängen der freigegebenen Verzeichnisse mit `mount`

```
(guest) # mkdir /mnt/shared
(guest) # mount -t 9p -o trans=virtio,version=9p2000.L \
    shared /mnt/shared
```

- Automatisches Einhängen über Konfiguration in `/etc/fstab` im Gast-System:

```
shared  /mnt/shared  9p  trans=virtio,version=9p2000.L  0 0
```

# Unterschiede zwischen Quelltexten ermitteln

- Grundlegendes Werkzeug um Unterschiede zweier Programm-Code-Revisionen zu erhalten: `diff(1)`
  - zeilenorientiert
  - bevorzugter Modus: ,,unified diff"

- Beispiel: ,,Normaler" Diff

```
$ diff -r a b
diff -r a/linux/drivers/staging/crystalhd/crystalhd_hw.c
        b/linux/drivers/staging/crystalhd/crystalhd_hw.c
87c87
<     * (63MHz * 40us = 0x9D8)
---
>     * (63MHz * 20us = 0x4EC)
89c89
<    crystalhd_reg_wr(adp, GISB_ARBITER_TIMER, 0x9D8);
---
>    crystalhd_reg_wr(adp, GISB_ARBITER_TIMER, 0x4EC);
```

# Unterschiede zwischen Source-Code

- Beispiel: ,,Vereinheitlichter" Diff (unified diff)

```
$ diff -ur a b
--- a/linux/drivers/staging/crystalhd/crystalhd_hw.c 16:55
+++ b/linux/drivers/staging/crystalhd/crystalhd_hw.c 16:57
@@ -84,9 +84,9 @@
  /*
   * Bus Arbiter Timeout: GISB_ARBITER_TIMER
-  * (63MHz * 40us = 0x9D8)
+  * (63MHz * 20us = 0x4EC)
   */
- crystalhd_reg_wr(adp, GISB_ARBITER_TIMER, 0x9D8);
+ crystalhd_reg_wr(adp, GISB_ARBITER_TIMER, 0x4EC);

  /*
   * Decoder clocks: MISC_PERST_DECODER_CTRL
Only in b/linux/drivers/staging/crystalhd: crystalhd-fix.msg
```

# Unterschiede zwischen Source-Code

- Beispiel: ,,Vereinheitlichter Diff (unified diff), zusätzlich: `-N` und `-p`

```
$ diff -urNp a b
--- a/linux/drivers/staging/crystalhd/crystalhd_hw.c  16:55
+++ b/linux/drivers/staging/crystalhd/crystalhd_hw.c  16:57
@@ -84,9 +84,9 @@ static bool crystalhd_bring_out_of_rst(s
  /*
   * Bus Arbiter Timeout: GISB_ARBITER_TIMER
-  * (63MHz * 40us = 0x9D8)
+  * (63MHz * 20us = 0x4EC)
   */
- crystalhd_reg_wr(adp, GISB_ARBITER_TIMER, 0x9D8);
+ crystalhd_reg_wr(adp, GISB_ARBITER_TIMER, 0x4EC);

  /*
   * Decoder clocks: MISC_PERST_DECODER_CTRL
diff -urNp a/linux/drivers/staging/crystalhd/crystalhd-fix.msg
           b/linux/drivers/staging/crystalhd/crystalhd-fix.msg
--- a/linux/drivers/staging/crystalhd/crystalhd-fix.msg 01:00
+++ b/linux/drivers/staging/crystalhd/crystalhd-fix.msg 17:10
@@ -0,0 +1 @@
+Initial patch description.
```

# Diffs gezielt durchsuchen

- Funktioniert auch mit gepackten Diffs
- Kann führende Unterverzeichnisse überspringen

```
$ lsdiff -z --strip=1 big-diff.gz | grep ^tools
tools/perf/.gitignore
tools/perf/CREDITS
tools/perf/Documentation/Makefile
tools/perf/design.txt
tools/perf/perf.c
tools/perf/perf.h
[...]
```

# Diffs gezielt durchsuchen

- Funktioniert auch mit gepackten Diffs
- Kann führende Unterverzeichnisse überspringen

```
$ lsdiff -z --strip=1 big-diff.gz | grep ^tools
tools/perf/.gitignore
tools/perf/CREDITS
tools/perf/Documentation/Makefile
tools/perf/design.txt
tools/perf/perf.c
tools/perf/perf.h
[...]
```

- Änderungen auf Unterverzeichnisse beschränken:

```
$ filterdiff -z  --strip=1 -i '*/tools/perf/*' big-diff.gz | lsdiff
tools/perf/.gitignore
tools/perf/CREDITS
tools/perf/Documentation/Makefile
tools/perf/Documentation/asciidoc.conf
[...]
```

# Größe von Diffs ermitteln

- Ausgabe aus dem Kommando `diff` kann direkt weiterverwendet werden:

```
$ diff -Nurp linux-2.6.18/ linux-2.6.32/ | diffstat
 .gitignore                                         |   36
 .mailmap                                           |  107
 CREDITS                                            |  367
 Documentation/00-INDEX                             |  240
 [...]
 virt/kvm/ioapic.c                                  |  362
 virt/kvm/ioapic.h                                  |   76
 virt/kvm/iodev.h                                   |   70
 virt/kvm/iommu.c                                   |  233
 virt/kvm/irq_comm.c                                |  422
 virt/kvm/kvm_main.c                                | 2830 +
 36994 files changed, 7819725 insertions(+), 3043036 deletions(-)
```

# Heartbleed

# Was ist passiert?

- Fehler in OpenSSL 1.0.1: CVE-2014-0160, -0346

# Was ist passiert?

- Fehler in OpenSSL 1.0.1: CVE-2014-0160, -0346
- Heartbeat-Funktionalitaet: `ping` in TLS
  - analog `ping`: Nutzdaten hin und zurück
  - Länge der Nutzdaten

# Was ist passiert?

- Fehler in OpenSSL 1.0.1: CVE-2014-0160, -0346
- Heartbeat-Funktionalitaet: `ping` in TLS
  - analog `ping`: Nutzdaten hin und zurück
  - Länge der Nutzdaten
    - im Paketkopf: /1

# Was ist passiert?

- Fehler in OpenSSL 1.0.1: CVE-2014-0160, -0346
- Heartbeat-Funktionalitaet: `ping` in TLS
  - analog `ping`: Nutzdaten hin und zurück
  - Länge der Nutzdaten
    - im Paketkopf: /1
    - aus Paketgröße: /2

# Was ist passiert?

- Fehler in OpenSSL 1.0.1: CVE-2014-0160, -0346
- Heartbeat-Funktionalitaet: `ping` in TLS
  - analog `ping`: Nutzdaten hin und zurück
  - Länge der Nutzdaten
    - im Paketkopf: /1
    - aus Paketgröße: /2
- Problem bei Beantwortung von heartbeats:
  - `response = malloc(/1)`

# Was ist passiert?

- Fehler in OpenSSL 1.0.1: CVE-2014-0160, -0346
- Heartbeat-Funktionalitaet: `ping` in TLS
  - analog `ping`: Nutzdaten hin und zurück
  - Länge der Nutzdaten
    - im Paketkopf: /1
    - aus Paketgröße: /2
- Problem bei Beantwortung von heartbeats:
  - `response = malloc(/1)`
  - `memcpy(response, request, /2)`

# Was ist passiert?

- Fehler in OpenSSL 1.0.1: CVE-2014-0160, -0346
- Heartbeat-Funktionalitaet: `ping` in TLS
  - analog `ping`: Nutzdaten hin und zurück
  - Länge der Nutzdaten
    - im Paketkopf: /1
    - aus Paketgröße: /2
- Problem bei Beantwortung von heartbeats:
  - `response = malloc(/1)`
  - `memcpy(response, request, /2)`
  - `send(response, /1)`

# Auswirkungen

- bis 65kB uninitialisierten Speicher an Angreifer
- „uninitialisiert“: evtl. schonmal benutzt, dann `free()`
- z.B. bei Webserver: häufiges `malloc()/free()` auf

# Auswirkungen

- bis 65kB uninitialisierten Speicher an Angreifer
- „uninitialisiert“: evtl. schonmal benutzt, dann `free()`
- z.B. bei Webserver: häufiges `malloc()/free()` auf
  - Verbindungsinhalte

# Auswirkungen

- bis 65kB uninitialisierten Speicher an Angreifer
- „uninitialisiert“: evtl. schonmal benutzt, dann `free()`
- z.B. bei Webserver: häufiges `malloc()/free()` auf
  - Verbindungsinhalte
  - Verbindungsmetadaten

# Auswirkungen

- bis 65kB uninitialisierten Speicher an Angreifer
- „uninitialisiert“: evtl. schonmal benutzt, dann `free()`
- z.B. bei Webserver: häufiges `malloc()/free()` auf
  - Verbindungsinhalte
  - Verbindungsmetadaten
  - Sitzungsschlüssel

# Auswirkungen

- bis 65kB uninitialisierten Speicher an Angreifer
- „uninitialisiert“: evtl. schonmal benutzt, dann `free()`
- z.B. bei Webserver: häufiges `malloc()/free()` auf
  - Verbindungsinhalte
  - Verbindungsmetadaten
  - Sitzungsschlüssel
  - privater Schlüssel zum Serverzertifikat

# Auswirkungen

- bis 65kB uninitialisierten Speicher an Angreifer
- „uninitialisiert“: evtl. schonmal benutzt, dann `free()`
- z.B. bei Webserver: häufiges `malloc()/free()` auf
  - Verbindungsinhalte
  - Verbindungsmetadaten
  - Sitzungsschlüssel
  - privater Schlüssel zum Serverzertifikat
- notwendige Behebungsmassnahmen
  - Session-Cookies neu erzeugen
  - Passwörter neu setzen
  - kryptographische Schlüssel neu erzeugen
  - Zertifikate invalidieren

# Wie kann sowas passieren?

- selten benutzte Funktionalität
  - daher kaum getestet
  - auch selten angeschaut
- OpenSSL-eigene Speicherverwaltung?
- Code-Review?
- „unsichere“ Sprache C?

# Aufgabe 2

- Auswahl eines Vortragsthemas
- Recherche zu den vorgestellten Themen
- Präsentation der Ergebnisse als Vortrag am **15. Mai**
  - Folien vorher an die Liste schicken
  - Aufbereitung mit diffstat, Grafiken, ...
  - Pro Gruppe etwa 15 Minuten

# Thema 1: OpenBSD/LibreSSL

Die jüngsten Änderungen des OpenBSD-Teams an OpenSSL und LibreSSL

- was wurde geändert?
- in welchem Umfang und wo?
- wie und nach welchen Kriterien?
- wie ist das organisiert?

# Thema 2: SSL-Vergleich

Die Softwareentwicklung von OpenSSL, GnuTLS, PolarSSL und NSS im Vergleich

- welchen Weg nimmt mein Patch?
- wie unterscheidet sich die Arbeitweise der Projekte?
- aufgrund eurer Recherche: was wuerdet Ihr waehlen, und warum?
- Thema ist auch teilbar in 2 Themen

# Thema 3: OpenSSL in Debian

Die Paketierung von OpenSSL in Debian

- Linux-Distributionen verteilen Software als Pakete
- oft mit eigenen Modifikationen, Bugfixes, Erweiterungen
- wie/was/wieviel/wozu/woher sind diese in Debian?

# Thema 4: malloc

Speicherverwaltung in OpenSSL vs. „wie man's richtig macht“

- Vergleich der Speicherverwaltung in OpenSSL
- mit der Speicherverwaltung ausgewählter Systembibliotheken
  - eure Wahl, z.B. GNU libc, OpenBSD
- mit spezialisierten Bibliotheken für Debugging oder sicherere Speicherverwaltung
  - ebenfalls eure Wahl, z.B. eFence, valgrind, smalloc
- Wirkmechanismen
- Aufwand der Implementierung oder Benutzung