

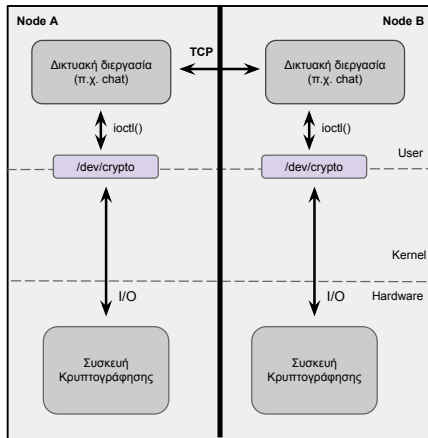
# Κρυπτογραφική συσκευή VirtIO για QEMU-KVM

Εργαστήριο Λειτουργικών Συστημάτων  
8ο εξάμηνο, ΣΗΜΜΥ

*Εργαστήριο Υπολογιστικών Συστημάτων (CSLab)*

Απρίλιος 2016

# Βασικό Πλαίσιο



- Βασικό σενάριο άσκησης
- Πραγματικό / εικονικό περιβάλλον;
- Λίγη θεωρία προτού γίνουμε πιο συγκεκριμένοι :)

# Άσκηση - Ζητούμενα

# Άσκηση - Ζητούμενα

## Z1: Εργαλείο chat πάνω από TCP/IP sockets

- Αμφίδρομη επικοινωνία πάνω από TCP/IP
- Με χρήση του BSD Sockets API

# Άσκηση - Ζητούμενα

## Z1: Εργαλείο chat πάνω από TCP/IP sockets

- Αμφίδρομη επικοινωνία πάνω από TCP/IP
- Με χρήση του BSD Sockets API

## Z2: Κρυπτογραφημένο chat πάνω από TCP/IP

- Με χρήση του cryptodev-linux από userspace
- Κλήσεις συστήματος `ioctl()` στο `/dev/crypto`



# Άσκηση - Ζητούμενα

## Z1: Εργαλείο chat πάνω από TCP/IP sockets

- Αμφίδρομη επικοινωνία πάνω από TCP/IP
- Με χρήση του BSD Sockets API

## Z2: Κρυπτογραφημένο chat πάνω από TCP/IP

- Με χρήση του cryptodev-linux από userspace
- Κλήσεις συστήματος ioctl() στο /dev/crypto

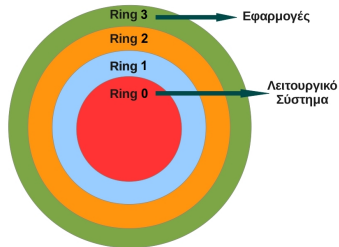
## Z3: Υλοποίηση συσκευής cryptodev με VirtIO

- Κρυπτογραφημένο chat μέσα σε VM
- με κλήσεις στο cryptodev του host
- Υλοποίηση οδηγού στο πλαίσιο VirtIO



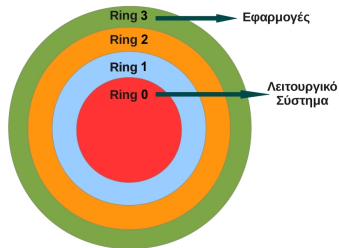
# ΑΣ και Privilege levels (Rings)

# ΛΣ και Privilege levels (Rings)



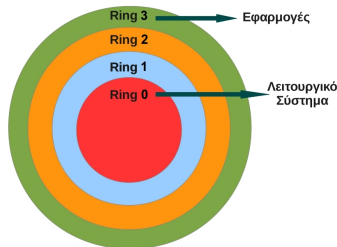


# ΛΣ και Privilege levels (Rings)



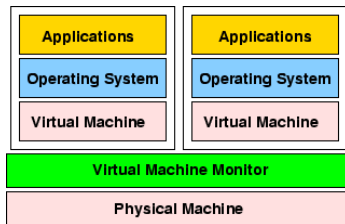
- Μηχανισμός ιεραρχίας του υλικού για εξασφάλιση προστασίας/διαχωρισμού εφαρμογών
- Κάθε ring επιτρέπει συγκεκριμένες λειτουργίες
- Πιο προνομιακό το χαμηλότερο ring
- Π.χ. Linux σε x86: εφαρμογές(user-mode) -> Ring 3 , ΛΣ(kernel-mode) -> Ring 0

# ΛΣ και Privilege levels (Rings)



- Μη επιτρεπτές εντολές π.χ. στο Ring 3 προκαρούν *trap* που χειρίζεται το ΛΣ
- Παράδειγμα 1: διαίρεση με το μηδέν
- Παράδειγμα 2: πρόσβαση σε μη έγκυρη περιοχή μνήμης
- Παράδειγμα 3: κλήση συστήματος σε x86: `int $0x80 => trap (software interrupt) + μετάβαση σε χώρο πυρήνα`

# Εικονικοποίηση



## Ορολογία

- VMM (Virtual Machine Monitor) ή *hypervisor*: δίνει την ψευδαίσθηση στα VMs ότι εκτελούνται σε φυσικό περιβάλλον
- **Host** -> ΛΣ διαχειρίζεται το φυσικό υλικό, **Guest** -> διαχειρίζεται το εικονικό υλικό
- Αναλογία: (φυσικό περιβάλλον) host ΛΣ - εφαρμογές => (εικονικό περιβάλλον) VMM - guest ΛΣ

Πηγή: [https://www.usenix.org/legacy/event/usenix01/sugerman/sugerman\\_html/node1.html](https://www.usenix.org/legacy/event/usenix01/sugerman/sugerman_html/node1.html)

# Ευρεία κατηγοριοποίηση συστημάτων εικονικοποίησης

- Paravirtualization
- Full virtualization (με χρήση binary translation)
- Hardware-assisted virtualization
- Hybrid virtualization



# Paravirtualization

- Απαιτούνται αλλαγές σε κομμάτια (συνήθως drivers) του πυρήνα ενός φυσικού μηχανήματος για να χρησιμοποιηθούν από ένα εικονικό μηχάνημα
- Αυτοί οι paravirtualized drivers “γνωρίζουν” ότι τρέχουν σε εικονικό περιβάλλον
- Έχοντας αυτή τη γνώση, πραγματοποιούν αιτήματα στον hypervisor (hypercalls - αναλογία με system calls).
- Παράδειγμα: Xen



# Paravirtualization

- Απαιτούνται αλλαγές σε κομμάτια (συνήθως drivers) του πυρήνα ενός φυσικού μηχανήματος για να χρησιμοποιηθούν από ένα εικονικό μηχάνημα
- Αυτοί οι paravirtualized drivers “γνωρίζουν” ότι τρέχουν σε εικονικό περιβάλλον
- Έχοντας αυτή τη γνώση, πραγματοποιούν αιτήματα στον hypervisor (hypercalls - αναλογία με system calls).
- Παράδειγμα: Xen

## Πλεονεκτήματα

- Συνήθως καλύτερη επίδοση (ειδικά στο I/O)

## Μειονεκτήματα

- Ειδικές αλλαγές στον κώδικα του ΛΣ μέσα στο VM

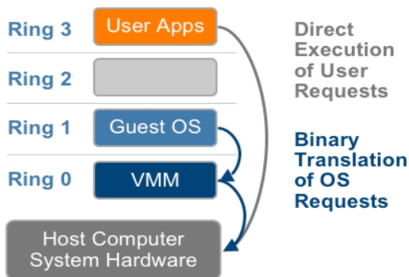


# Full virtualization (με χρήση binary translation)

- Το image του πυρήνα ενός φυσικού μηχανήματος χρησιμοποιείται χωρίς αλλαγές και στο εικονικό μηχάνημα
- Οι μη-προνομιούχες εντολές του guest ΛΣ εκτελούνται απευθείας στον επεξεργαστή
- Οι προνομιούχες εντολές του guest ΛΣ μεταφράζονται (binary translation) από τον VMM
- Παράδειγμα: QEMU (χωρίς χρήση KVM)

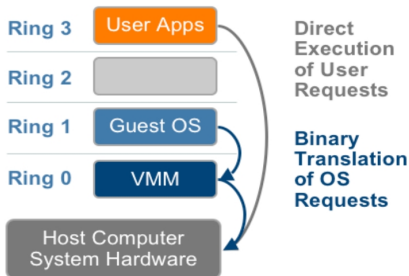


# Full virtualization (με χρήση binary translation)





# Full virtualization (με χρήση binary translation)



## Πλεονεκτήματα

- Ευκολότερη δημιουργία/διαχείριση εικονικών μηχανών
- Δεν απαιτεί αλλαγές στο υλικό

## Μειονεκτήματα

- Συνήθως χειρότερη επίδοση (λόγω binary translation)

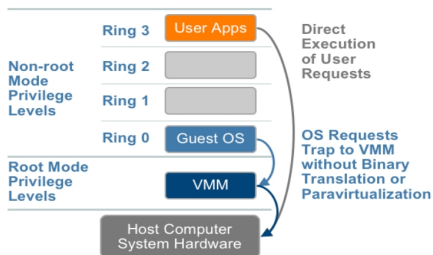
Πηγή εικόνας: [http://www.vmware.com/files/pdf/VMware\\_paravirtualization.pdf](http://www.vmware.com/files/pdf/VMware_paravirtualization.pdf)

# Hardware-assisted virtualization

- Υποκατηγορία του full virtualization - δεν απαιτούνται αλλαγές στον πυρήνα του guest
- Χρησιμοποιεί virtualization extensions στο υλικό (Intel VT-x, AMD-V)
- Παράδειγμα: KVM

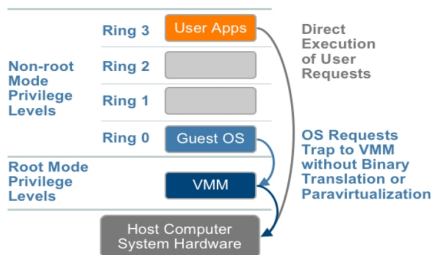


# Hardware-assisted virtualization



Πηγή εικόνας: [http://www.vmware.com/files/pdf/VMware\\_paravirtualization.pdf](http://www.vmware.com/files/pdf/VMware_paravirtualization.pdf)

# Hardware-assisted virtualization



## Πλεονεκτήματα

- Ευκολότερη δημιουργία/διαχείριση εικονικών μηχανών
- Καλύτερη επίδοση (λόγω υποστήριξης από το υλικό)

## Μειονεκτήματα

- Χειρότερη επίδοση σε I/O

Πηγή εικόνας: [http://www.vmware.com/files/pdf/VMware\\_paravirtualization.pdf](http://www.vmware.com/files/pdf/VMware_paravirtualization.pdf)

# Hybrid virtualization

- Συνδυασμός hardware-assisted virtualization + paravirtualization
- Άθικτος κώδικας του μεγαλύτερου μέρους του πυρήνα + αλλαγές στους drivers για γρηγορότερο I/O
- Παράδειγμα: KVM + VirtIO



# Hybrid virtualization

- Συνδυασμός hardware-assisted virtualization + paravirtualization
- Άθικτος κώδικας του μεγαλύτερου μέρους του πυρήνα + αλλαγές στους drivers για γρηγορότερο I/O
- Παράδειγμα: KVM + VirtIO

## Πλεονεκτήματα

- Γενικά καλύτερη επίδοση (και) στο I/O

## Μειονεκτήματα

- Αλλαγές στους drivers του VM



# Πλατφόρμα εικονικοποίησης KVM

## KVM (Kernel Virtual Machine)

- Hypervisor για Linux σε x86
- Hardware-assisted virtualization επιλογή
- Εκμεταλλεύεται τα virtualization extensions στο υλικό (Intel VT-x, AMD-V)
- Εντάσσεται σε υπάρχον σύστημα εξομοίωσης υλικού, συνήθως το QEMU



## QEMU

- Εξομοιώνει τη λειτουργία του υλικού
- Υλοποίηση υλικού σε λογισμικό (δίσκοι, κάρτες δικτύου, θύρες)
- Μπορεί να εκτελεστεί σε διάφορα modes (CPU emulation, system virtualization κλπ)
- Διεργασία χώρου χρήστη

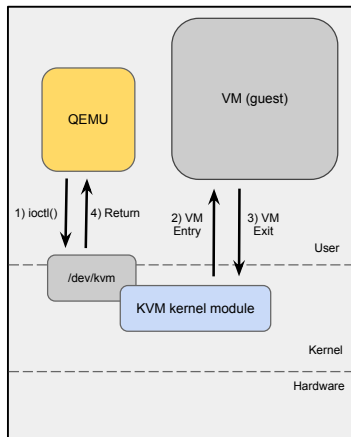


# VirtIO

- Είναι ένα πρότυπο για εικονικοποίηση συσκευών E/E
- Αποτελεί **paravirtualized** επιλογή
- Split-driver μοντέλο: **1)** frontend (guest kernel) - **2)** backend (qemu userspace) drivers
- Κυκλικοί buffers για επικοινωνία μεταξύ frontend/backend
- Αντί για εντολές I/O (πχ IN, OUT), εγγραφή σε αντίστοιχη θέση μνήμης ενός buffer → doorbell
- **Υπάρχουσες υλοποιήσεις:** virtio\_net, virtio\_blk, virtio\_console
- hybrid: **KVM + VirtIO** = full virtualization με paravirtualized I/O

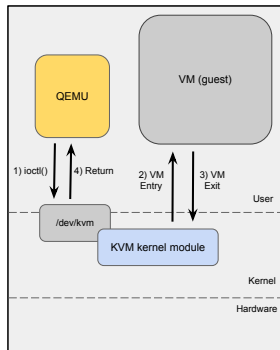


# KVM - QEMU (1)



- Kernelspace driver: υποστηρίζει συσκευή χαρακτήρων `/dev/kvm`
- Η διεργασία QEMU εκτελεί `ioctl()` στο `/dev/kvm` (1)
- Δημιουργείται το VM και το module κάνει VM Entry (2)

## KVM - QEMU (2)



- Οι μη-προνομιούχες εντολές του guest εκτελούνται απευθείας στον επεξεργαστή
- Οι προνομιούχες εντολές (π.χ. I/O) προκαλούν αρχικά trap (VM Exit (3)) και περνάνε στη διεργασία του QEMU(4)
- Το QEMU εξυπηρετεί την εντολή I/O, αν χρειαστεί με κλήση συστήματος

# Κρυπτογραφία και υλοποιήσεις

## Software

- Από τον προγραμματιστή
  - ▶ Απαιτεί εξειδικευμένες γνώσεις και δεν είναι πρακτικό
- Μέσω βιβλιοθήκης υψηλού επιπέδου
  - ▶ Ο συνήθης τρόπος (π.χ. OpenSSL), αλλά...
  - ▶ οι υλοποιήσεις σε λογισμικό είναι αργές

## Hardware

- Ήδη διαθέσιμο σε πολλές οικογένειες επεξεργαστών/boards (ακόμα και οικιακές, π.χ. AES σε Intel i7 Westmere)
- Πολύ ταχύτερο, αλλά μπορεί να απαιτεί ειδικούς drivers.



# Cryptodev

- Kernel driver που υποστηρίζει εξειδικευμένους επιταχυντές σε υλικό. Αρχική υλοποίηση: OpenBSD cryptodev API.
- Πραγματοποιεί εξομοίωση σε χώρο πυρήνα για μηχανήματα που δεν διαθέτουν hardware accelerator.
- Υποστηρίζεται από την OpenSSL (ως engine).
- Εξάγει το αρχείο συσκευής χαρακτήρων /dev/crypto



# Χρήση cryptodev API

- Τέσσερις κλήσεις `ioctl()` προς τη συσκευή μας δίνουν όλες τις δυνατότητες!

## Παράδειγμα

```
fd = open("/dev/crypto");
ioctl(fd, CIOCGSESSION); /* get session */
crypt.src = clr; crypt.dst = enc; crypt.op = COP_ENCRYPT;
ioctl(fd, CIOCRYPT, &crypt); /* encrypt data */
crypt.dst = clr; crypt.src = enc; crypt.op = COP_DECRYPT;
ioctl(fd, CIOCRYPT, &crypt); /* decrypt data */
ioctl(fd, CIOCFSESSION); /* close session */
close(fd);
```

## Τι μπορούμε να κάνουμε μαζί με TCP/IP sockets;

- Encrypted chat :)

## Z1, Z2: Κρυπτογραφημένο chat πάνω από TCP/IP

- Command line εργαλείο για μεταφορά μηνυμάτων πάνω από TCP/IP sockets
- Επέκτασή του ώστε να χρησιμοποιεί το cryptodev-linux
- Σας δίνονται παραδείγματα χρήσης των sockets και του cryptodev

# Άσκηση

## Z1, Z2: Κρυπτογραφημένο chat πάνω από TCP/IP

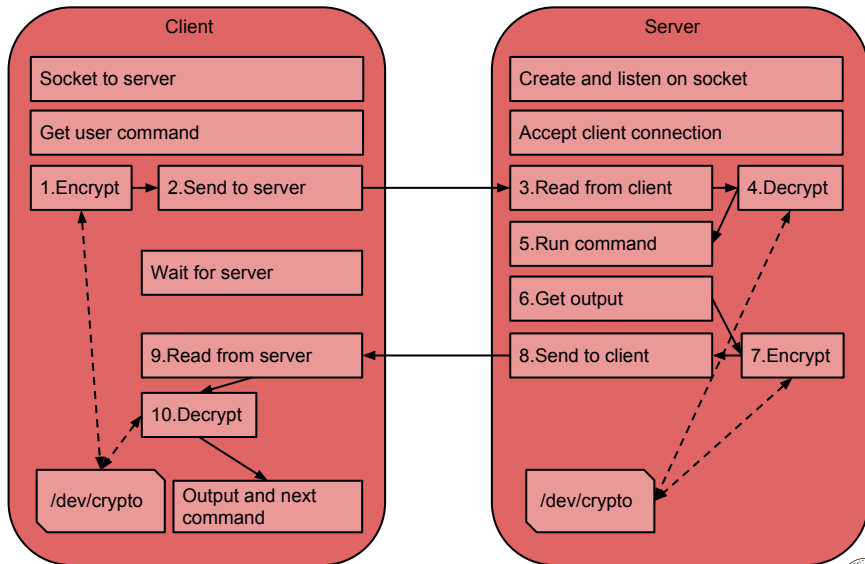
- Command line εργαλείο για μεταφορά μηνυμάτων πάνω από TCP/IP sockets
- Επέκτασή του ώστε να χρησιμοποιεί το cryptodev-linux
- Σας δίνονται παραδείγματα χρήσης των sockets και του cryptodev

## Προαιρετικά

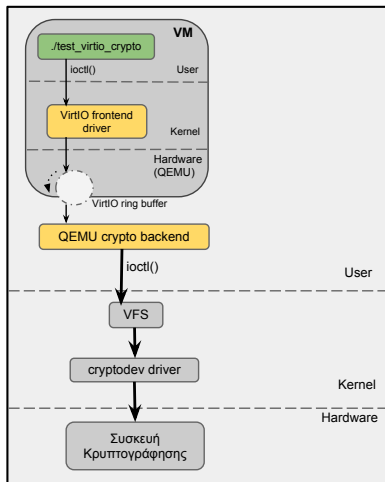
- Πολλοί clients ταυτόχρονα (IRC!)



# Παράδειγμα χρήσης cryptodev



## Z3: Υλοποίηση συσκευής cryptodev με VirtIO



- Υλοποίηση frontend+backend **virtio\_crypto** driver
- Εκτέλεση εργαλείου `chat` με TCP/IP πάνω από αυτό

# Αναφορές

- A. Kivity, Y. Kamay, D. Laor, U. Lublin, and A. Liguori. kvm: the linux virtual machine monitor. In Linux Symposium, pages 225-230, Ottawa, Ontario, Canada, 2007.
- F. Bellard. Qemu, a fast and portable dynamic translator. In ATEC '05 Proceedings of the annual conference on USENIX Annual Technical Conference, 2005.
- R. Russel. virtio: Towards a de-facto standard for virtual i/o devices. In ACM SIGOPS Operating Systems, 2008.
- Qumranet. Kernel-based Virtualization Driver (White Paper), 2006.
- Yasunori Goto. The kernel-based virtual machine Technology. Fujitsu Global Book, 2011.



# Ερωτήσεις;

Λίστα μαθήματος:  
`os-lab@lists.cslab.ece.ntua.gr`