REORGANIZING UNIX FOR RELIABILITY

11th Asia-Pacific Computer Systems Architecture Conference

Shanghai, China
September 6-8, 2006

Jorrit N. Herder
Dept. of Computer Science
Vrije Universiteit Amsterdam

WHAT'S IN IT FOR YOU?

- **Straw poll:**
  - Raise your hand if your system has run perfectly with no crashes since you bought it

WHAT'S IN IT FOR YOU?

- **Straw poll:**
  - Raise your hand if your system has run perfectly with no crashes since you bought it

WHAT'S IN IT FOR YOU?

- **Straw poll:**
  - Raise your hand if your system has run perfectly with no crashes since you bought it

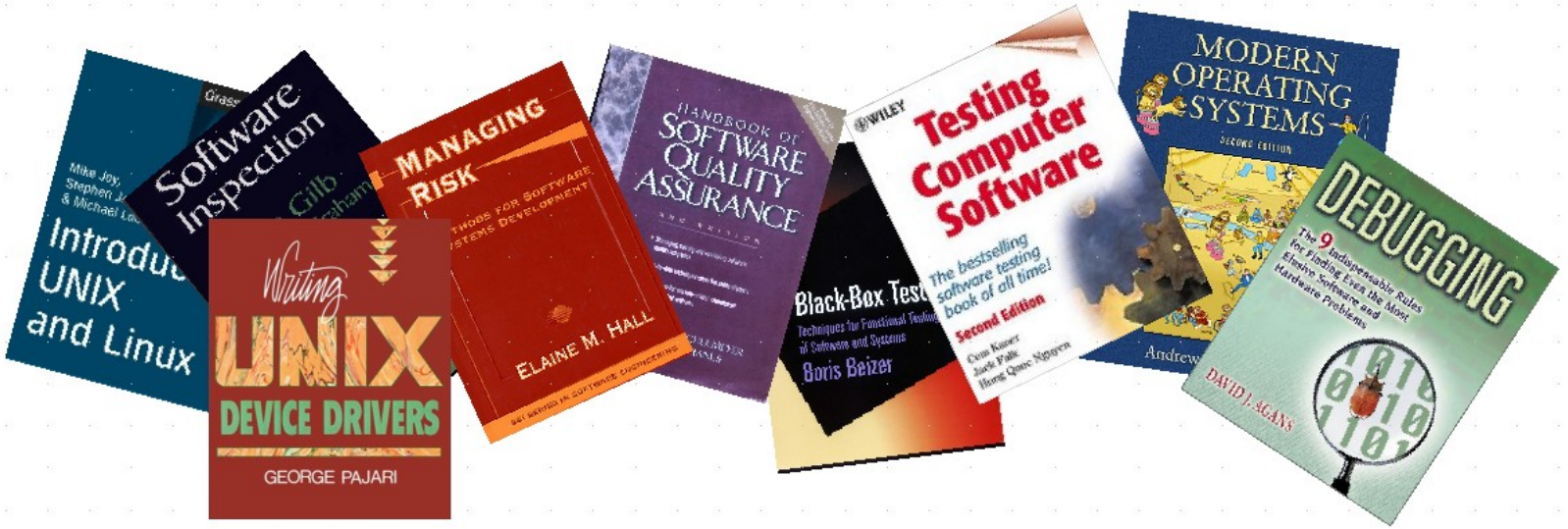
- **Contribution**
  - We have built a new, highly reliable UNIX-like OS: MINIX 3
  - OS is compartmentalized for fault isolation and containment
  - OS can automatically detect and repair certain defects
  - This talk discusses the current work and future directions

TALK OUTLINE

- Contribution (done)
- Introduction (next)
- Reorganizing UNIX
- Reliability features
- Performance
- Conclusion

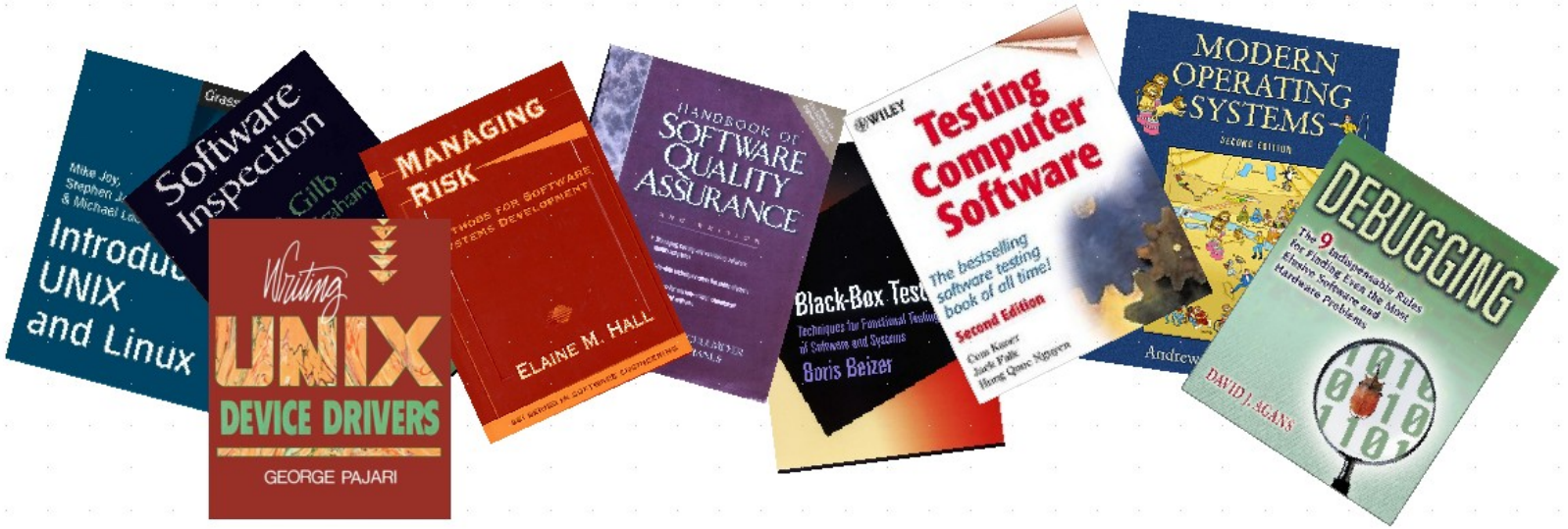
INTRODUCTION

SOFTWARE HAS BEEN STUDIED BEFORE ...



- **... but still we have security and reliability problems**
  - Application failures
  - Operating system failures
  - Digital pests (spyware, viruses, worms, etc.)

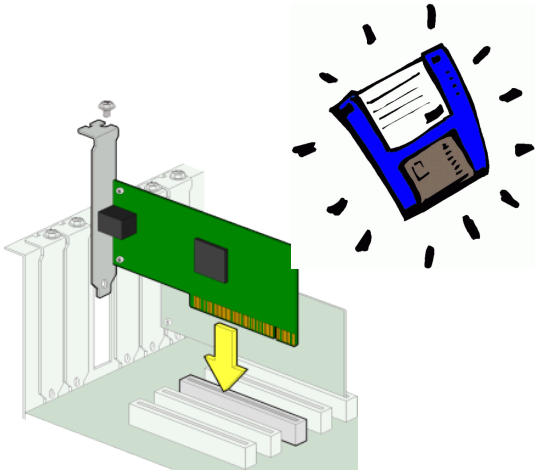
SOFTWARE HAS BEEN STUDIED BEFORE ...

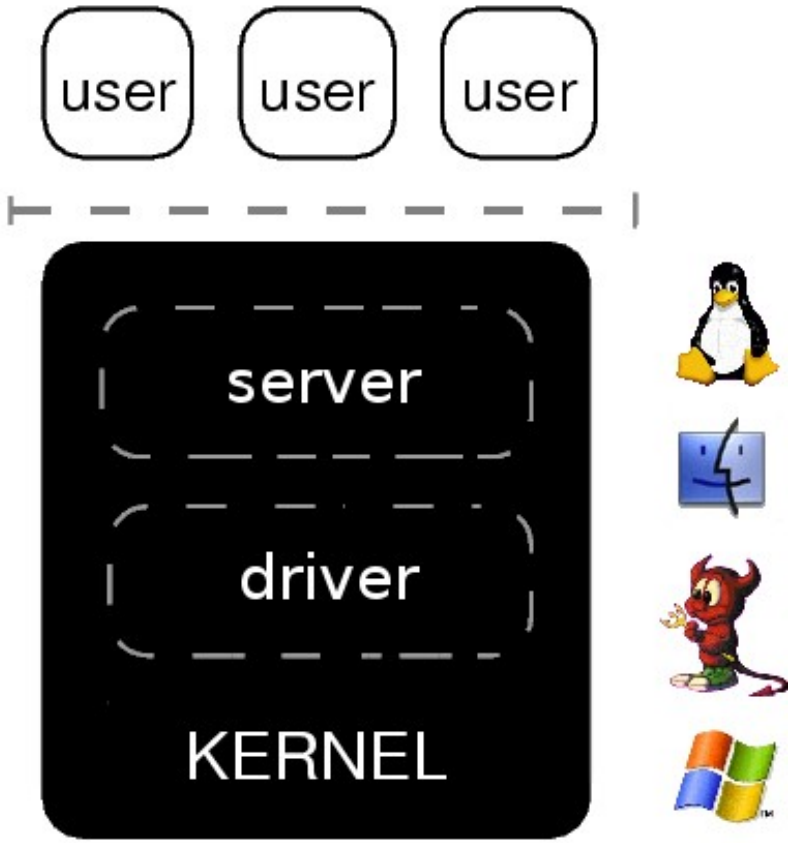


- **... but still we have security and reliability problems**
  - Application failures
  - Operating system failures **<-- this talk's focus**
  - Digital pests (spyware, viruses, worms, etc.)

DRIVERS IN A MONOLITHIC OPERATING SYSTEM

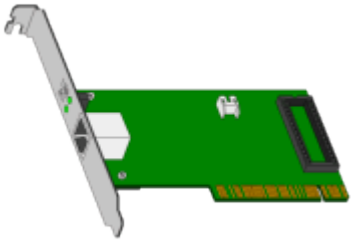
- Device drivers control hardware
- Driver is installed in the kernel

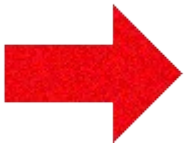




INHERENT PROBLEMS OF MONOLITHIC DESIGNS

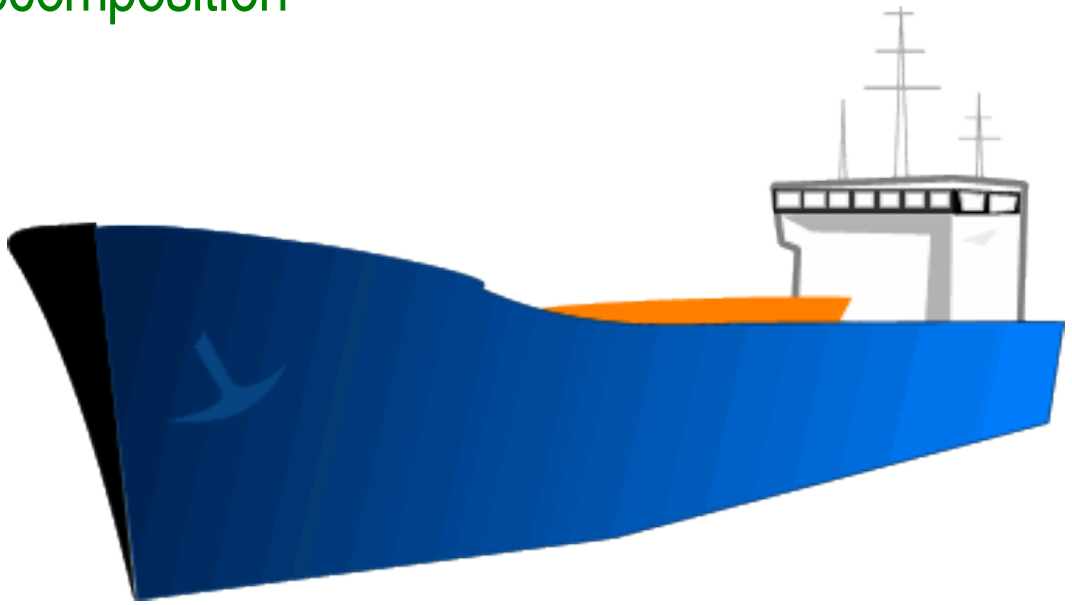
- **Fundamental design flaws in monolithic kernels**
  - All code runs at highest privilege level (breaches POLA)
  - No proper fault isolation (any bug can be fatal)
  - Huge amount of code *in* kernel (6-16 bugs per 1000 LOC)
  - Untrusted, 3rd party code in kernel (70% of code, more bugs)
  - Entangled code increases complexity (hard to maintain)





HOW ABOUT MODULAR DESIGNS?

- **Modularity is commonly used in other engineering disciplines**
  - Ship's hull is compartmentalized to improve it's 'reliability'
  - Aircraft carrier is build out of many, well-isolated parts
- **Use modularity to improve OS reliability**
  - We propose an extreme decomposition



OTHER APPROACHES

- Software-based isolation and recovery of in-kernel drivers
- Device drivers in dedicated user-mode virtual machines
- Minimal kernel designs running drivers in single-server OS
- MMU-protected user-mode drivers without fault resilience
- Language-based protection and formal code verification
- Recovery-oriented computing with system-wide undo

REORGANIZING UNIX

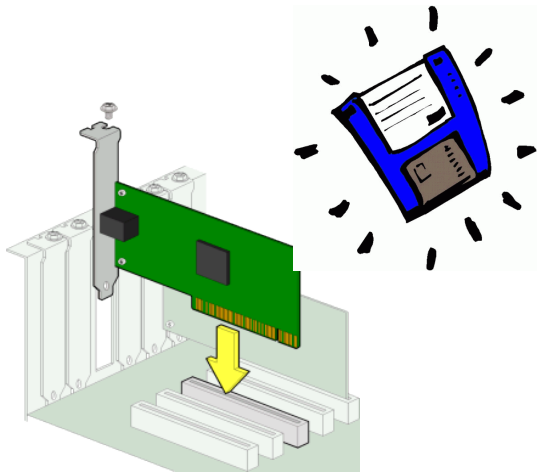
Jorrit N. Herder <jnherder@cs.vu.nl>

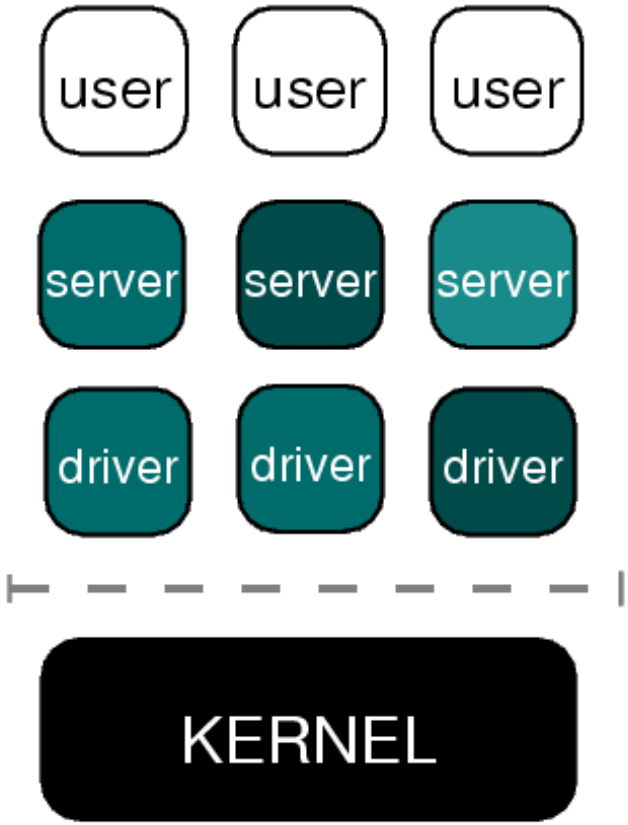
MINIX 3: A HIGHLY RELIABLE OPERATING SYSTEM

- **Transformation into a microkernel design (< 4000 LOC)**
  - Low-level operations to support user-space OS
- **All servers and drivers run as user-mode processes**
  - MMU protection and various other encapsulation properties
- **We added mechanisms to detect and repair failures**
  - Privileged server can replace failed components

ARCHITECTURE OF MINIX 3

- Device drivers are fully isolated in user space
- Local failures cannot spread



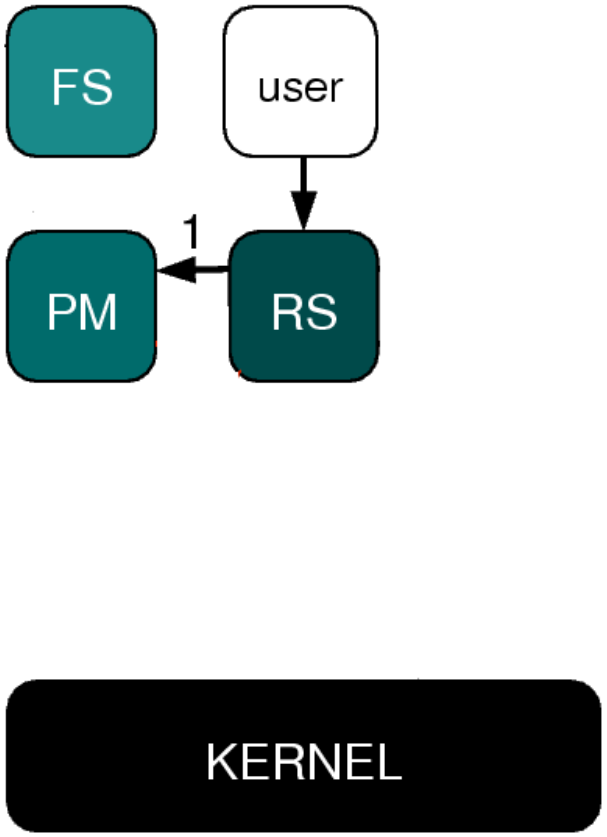


Jorrit N. Herder <jnherder@cs.vu.nl>

THE MINIX 3 USER-MODE SERVERS AND DRIVERS

- **Core servers**
  - File Server (FS)
  - Process Manager (PM)
  - Reincarnation Server (RS)
  - Data Store (DS)
- **Device drivers, e.g.:**
  - S-ATA, Floppy, RAM disk
  - TTY, PTY, RS232 lines
  - Video, Audio, Printer
  - (Fast) Ethernet drivers
- **Other services**
  - Network Server
  - Information Server
  - X Window System

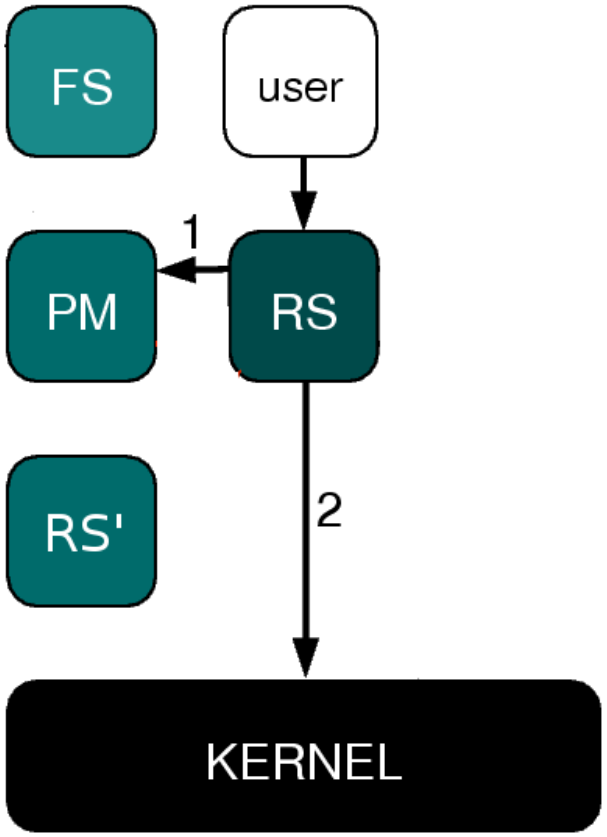
DEVICE DRIVER MANAGEMENT



- **Starting a new driver**

  (1) Fork new process

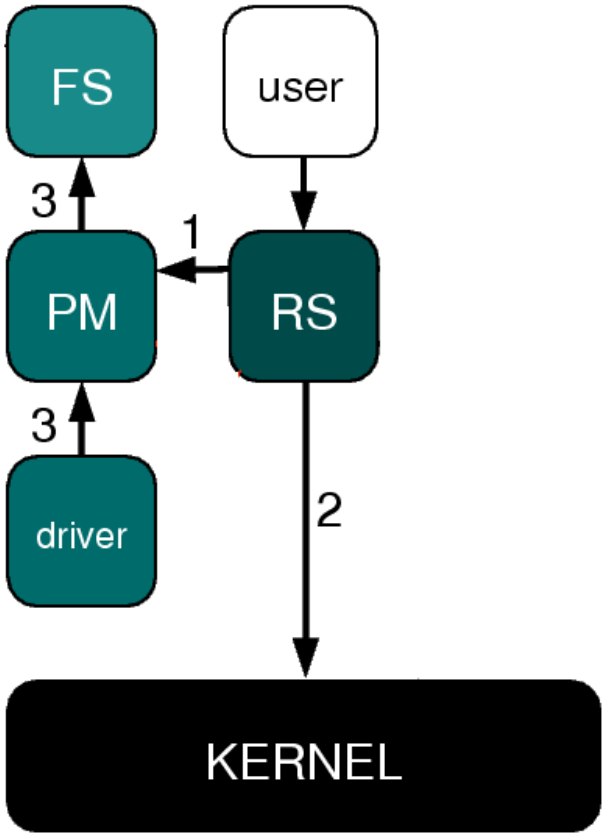
DEVICE DRIVER MANAGEMENT



- **Starting a new driver**
  - (1) Fork new process
  - (2) Assign privileges

Jorrit N. Herder <jnherder@cs.vu.nl>

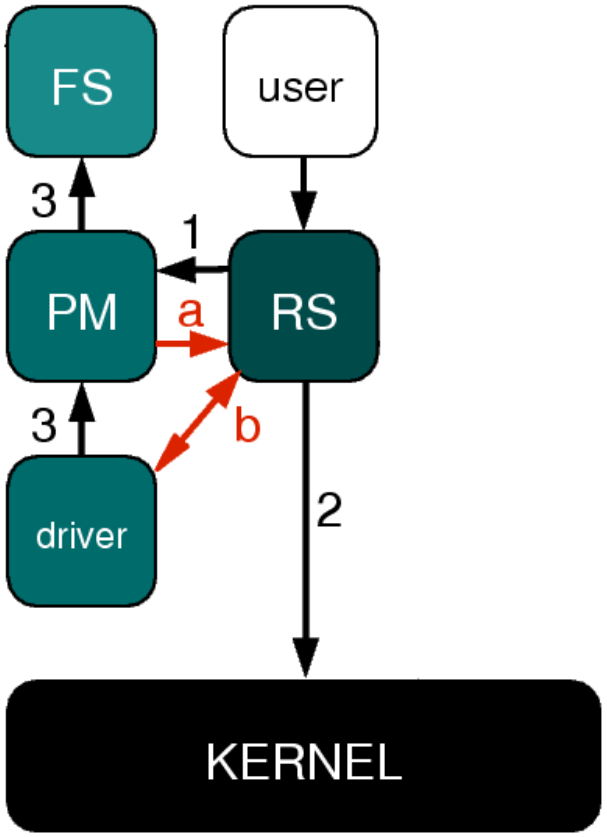
DEVICE DRIVER MANAGEMENT



- **Starting a new driver**
  - (1) Fork new process
  - (2) Assign privileges
  - (3) Execute binary

Jorrit N. Herder <jnherder@cs.vu.nl>

DEVICE DRIVER MANAGEMENT



- **Starting a new driver**
  - (1) Fork new process
  - (2) Assign privileges
  - (3) Execute binary
- **Monitor drivers**
  - (a) Exit notification
  - (b) Heartbeat message

RELIABILITY FEATURES

Jorrit N. Herder <jnherder@cs.vu.nl>

FAULT ISOLATION

- **Limit consequences of faults to enable recovery**
- **All servers and drivers can fail independently**
  - Servers and drivers fully compartmentalized in user space
  - Private address spaces protected by kernel and MMUs
  - Privileges of each process reduced according to POLA

FAULT RESILIENCE

- **Fault-tolerant systems use redundancy to overcome failures**
- **Our fault-resilient design tries to automatically *repair* defects**
  - (1) Identify malfunctioning component
  - (2) Execute associated recovery script
  - (3) Replace component with a fresh copy
- **Assumes restart enables recovery**
  - Cannot recover if hardware fails

DEFECT DETECTION

- **Human user observes failure because of malfunctioning**
  - System crashes or becomes unresponsive
- **OS defect detection requires constant monitoring**
  - Reincarnation server is parent of all servers and drivers
  - Reincarnation server periodically checks drivers status

RECOVERY PROCESS

- **Policy scripts**
  - Shell script controls recovery steps taken
  - Full flexibility: write to log, send e-mail, restart component
- **Restarting dead drivers**
  - Either from disk or copy of binary in memory
- **Reintegrating the component**
  - Restarted component can retrieve lost state from data store
  - Dependent components are informed through data store

PERFORMANCE

PERFORMANCE MEASUREMENTS

- **Overhead of user-mode drivers (without optimizations)**
  - Run times for typical applications: 6% overhead
  - File system and disk I/O performance: 9% overhead
  - Disk throughput (with fast disk and DMA) up to 70 MB/s
  - Networking performance: Fast Ethernet at full speed
    - Initial experiments show gigabit ethernet is possible
- **System feels fast and responsive**
  - Time from multiboot monitor to login is under 5 sec.
  - The system can do a full build of itself within 4 sec.

SOURCE CODE STATISTICS

- **Kernel (including kernel tasks): < 4000 LOC**
- **Most important servers and drivers: ~2500 LOC**
- **Minimal POSIX-conformant system: ~20,000 LOC**
  - Critical source code reduced by >2 orders of magnitude
  - Sources are small enough to read and understand

RELIABILITY EVALUTION

- **Fault-injection experiments are work in progress**
- **Measurements of the recovery overhead:**

CONCLUSION

CONCLUSIONS 1/2

- **We have reorganized UNIX for reliability**
  - Full compartmentalization of the OS in user space
  - Additional fault isolation and fault containment
  - Explicit mechanisms to make system fault resilient

- **Improvements over other operating systems**
  - Number of fatal (kernel) bugs is reduced
  - Isolation properties limits bug damage
  - Recovery from common failures is possible

CONCLUSIONS 2/2

- **Evaluation of MINIX 3**
  - Critical source code reduced by >2 orders of magnitude
  - Performance overhead of 5-10% compared to base system
  - Crash simulation experiments prove viability of approach

- **Practicality of our approach**
  - Trend towards user-mode drivers in Linux and Windows
  - Our techniques can be applied to other operating systems
  - Limited costs make real-world adoption attractive

MORE SERVER DETAILS AND BACKGROUNDS

- **Jorrit N. Herder, Herbert Bos, Ben Gras, Philip Homburg, Andrew S. Tanenbaum,**

  Reorganizing UNIX for Reliability,

  Proc. 11th Asia-Pacific Computer Systems Architecture Conference, Shanghai, China, Sep. 2006.

- **Jorrit N. Herder, Herbert Bos, Ben Gras, Philip Homburg, Andrew S. Tanenbaum,**

  Construction of a Highly Dependable Operating System,

  To appear: Proc. 6th European Dependable Computing Conference, Coimbra, Portugal, Oct. 2006 .

TIME FOR QUESTIONS

- **Try it yourself!**
  - MINIX 3 Live CD-ROM
  - Current version: see website
- **More information**
  - Web: www.minix3.org
  - News: comp.os.minix
  - E-mail: jnherder@cs.vu.nl
- **The MINIX 3 team:**
  - Jorrit Herder
  - Mischa Geldermans
  - Ben Gras
  - Philip Homburg
  - Herbert Bos
  - Andy Tanenbaum

ANSWERS