OS Security

Malware (Part 2) & Intrusion Detection and Prevention

Radboud University Nijmegen, The Netherlands

Winter 2015/2016

A short recap

- Different categories of malware:
  - Virus (self-reproducing, needs host program)
  - Worm (spreading routine, no host program)
  - Trojan (no spreading routine, usually comes with useful masquerading functionality)

A short recap

- Different categories of malware:
  - Virus (self-reproducing, needs host program)
  - Worm (spreading routine, no host program)
  - Trojan (no spreading routine, usually comes with useful masquerading functionality)
- Rootkits are used to hide the results of an attack
- Particularly dangerous: kernel-space rootkits

A short recap

- Different categories of malware:
  - Virus (self-reproducing, needs host program)
  - Worm (spreading routine, no host program)
  - Trojan (no spreading routine, usually comes with useful masquerading functionality)
- Rootkits are used to hide the results of an attack
- Particularly dangerous: kernel-space rootkits
- Even worse: bootkits infect the MBR; hardware/firmware malware

A short recap

- Different categories of malware:
  - Virus (self-reproducing, needs host program)
  - Worm (spreading routine, no host program)
  - Trojan (no spreading routine, usually comes with useful masquerading functionality)
- Rootkits are used to hide the results of an attack
- Particularly dangerous: kernel-space rootkits
- Even worse: bootkits infect the MBR; hardware/firmware malware
- Malware detection either signature- or behavior-based (or a combination of both)
- Code polymorphism (and self-modifying code) to counter signature-based AV

A short recap

- Different categories of malware:
  - Virus (self-reproducing, needs host program)
  - Worm (spreading routine, no host program)
  - Trojan (no spreading routine, usually comes with useful masquerading functionality)
- Rootkits are used to hide the results of an attack
- Particularly dangerous: kernel-space rootkits
- Even worse: bootkits infect the MBR; hardware/firmware malware
- Malware detection either signature- or behavior-based (or a combination of both)
- Code polymorphism (and self-modifying code) to counter signature-based AV
- AV can hurt security: larger attack surface, false positives, user perception of security

Evolution of Malware: From PC to Smartphone

Evolution of Malware: From PC to Smartphone

- Led by an increase in number of smartphones used

Evolution of Malware: From PC to Smartphone

- Led by an increase in number of smartphones used
- Larger attack surface for malware authors; easy-to-deploy attacks; many forms of attack vectors

Evolution of Malware: From PC to Smartphone

- Led by an increase in number of smartphones used
- Larger attack surface for malware authors; easy-to-deploy attacks; many forms of attack vectors
- Motivation: 'low risk, high reward'

Evolution of Malware: From PC to Smartphone

- Led by an increase in number of smartphones used
- Larger attack surface for malware authors; easy-to-deploy attacks; many forms of attack vectors
- Motivation: 'low risk, high reward'
  - Various app markets: official (e.g. Google Play) and non-official (e.g. Pandaapp)
  - Decentralized: anyone can become an app developer; no proper vetting of new apps

Early days of smartphone malware

Early days of smartphone malware

- Back in 2004, a group known as **29A** released `Cabir` - a malware (worm) for Symbian
  - Propagate via Bluetooth
  - Bluetooth was the most used technology to transfer information between 2 devices at the time

Early days of smartphone malware

- Back in 2004, a group known as **29A** released `Cabir` - a malware (worm) for Symbian
  - Propagate via Bluetooth
  - Bluetooth was the most used technology to transfer information between 2 devices at the time
  - Countermeasure: simply turn Bluetooth off or switch it to the "invisible" mode

Early days of smartphone malware

- Back in 2004, a group known as **29A** released `Cabir` - a malware (worm) for Symbian
  - Propagate via Bluetooth
  - Bluetooth was the most used technology to transfer information between 2 devices at the time
  - Countermeasure: simply turn Bluetooth off or switch it to the "invisible" mode
- Trojan, `Qdial`, targeting Symbian users, was released in same year
  - Malware sent text messages to premium rate services, for which the handset owner would be charged, thus making an income for the malware author.

Early days of smartphone malware

- Back in 2004, a group known as **29A** released `Cabir` - a malware (worm) for Symbian
  - Propagate via Bluetooth
  - Bluetooth was the most used technology to transfer information between 2 devices at the time
  - Countermeasure: simply turn Bluetooth off or switch it to the "invisible" mode
- Trojan, `Qdial`, targeting Symbian users, was released in same year
  - Malware sent text messages to premium rate services, for which the handset owner would be charged, thus making an income for the malware author.
- In 2005, a variant of `Cabir` was released - `Pbstealer`
  - It copied all the information from an infected device's address book and attempted to transmit it to any Bluetooth-enabled device within range.

Early days of smartphone malware

- Back in 2004, a group known as **29A** released `Cabir` - a malware (worm) for Symbian
  - Propagate via Bluetooth
  - Bluetooth was the most used technology to transfer information between 2 devices at the time
  - Countermeasure: simply turn Bluetooth off or switch it to the "invisible" mode
- Trojan, `Qdial`, targeting Symbian users, was released in same year
  - Malware sent text messages to premium rate services, for which the handset owner would be charged, thus making an income for the malware author.
- In 2005, a variant of `Cabir` was released - `Pbstealer`
  - It copied all the information from an infected device's address book and attempted to transmit it to any Bluetooth-enabled device within range.
  - Malware included the string: *".:: Good artist copy, Great artist steal ::."*

Current state of smartphone malware

- All major smartphone platforms have been infected

Current state of smartphone malware

- All major smartphone platforms have been infected
  - iOS: `WireLurker` (2014), can install malicious third-party applications to an iOS device through an infected Mac via a USB connection

Current state of smartphone malware

- All major smartphone platforms have been infected
  - iOS: `WireLurker` (2014), can install malicious third-party applications to an iOS device through an infected Mac via a USB connection
  - Windows Phone: Proof-of-concept for Windows Phone 8 presented at MalCon 2013; `FinSpy Mobile` spyware (2013)

Current state of smartphone malware

- All major smartphone platforms have been infected
  - iOS: `WireLurker` (2014), can install malicious third-party applications to an iOS device through an infected Mac via a USB connection
  - Windows Phone: Proof-of-concept for Windows Phone 8 presented at MalCon 2013; `FinSpy Mobile` spyware (2013)
  - Blackberry: Trojans use a technique referred to as 'BackStab'; steal unencrypted backups of phones from computers; does not require higher-level privileges or root access to the phone or computer

Current state of smartphone malware

- All major smartphone platforms have been infected
  - iOS: `WireLurker` (2014), can install malicious third-party applications to an iOS device through an infected Mac via a USB connection
  - Windows Phone: Proof-of-concept for Windows Phone 8 presented at MalCon 2013; `FinSpy Mobile` spyware (2013)
  - Blackberry: Trojans use a technique referred to as 'BackStab'; steal unencrypted backups of phones from computers; does not require higher-level privileges or root access to the phone or computer
- Android OS - most infected platform to date

Popular Android Malware

- First proof-of-concept malware released in 2008.
  - Causes the phone to accept all incoming calls
  - Turns off the radio, preventing outgoing/incoming calls
  - Causes the phone to end all calls
  - Gathers sensitive information and sends it to the attacker

Popular Android Malware

- First proof-of-concept malware released in 2008.
  - Causes the phone to accept all incoming calls
  - Turns off the radio, preventing outgoing/incoming calls
  - Causes the phone to end all calls
  - Gathers sensitive information and sends it to the attacker
- `Mobile Spy` spyware, 2009
  - Monitored infected device via web browser, phone calls, text messages, photos, videos, GPS locations

Popular Android Malware

- First proof-of-concept malware released in 2008.
  - Causes the phone to accept all incoming calls
  - Turns off the radio, preventing outgoing/incoming calls
  - Causes the phone to end all calls
  - Gathers sensitive information and sends it to the attacker
- `Mobile Spy` spyware, 2009
  - Monitored infected device via web browser, phone calls, text messages, photos, videos, GPS locations
  - Ran in 'stealth mode', no visible icon

Popular Android Malware

- First proof-of-concept malware released in 2008.
  - Causes the phone to accept all incoming calls
  - Turns off the radio, preventing outgoing/incoming calls
  - Causes the phone to end all calls
  - Gathers sensitive information and sends it to the attacker
- `Mobile Spy` spyware, 2009
  - Monitored infected device via web browser, phone calls, text messages, photos, videos, GPS locations
  - Ran in 'stealth mode', no visible icon
- `DroidKungFu`, capable of root-level access on vulnerable Android devices and evade the detection of security software by encrypting its exploits using AES.

Popular Android Malware

- First proof-of-concept malware released in 2008.
  - Causes the phone to accept all incoming calls
  - Turns off the radio, preventing outgoing/incoming calls
  - Causes the phone to end all calls
  - Gathers sensitive information and sends it to the attacker
- `Mobile Spy` spyware, 2009
  - Monitored infected device via web browser, phone calls, text messages, photos, videos, GPS locations
  - Ran in 'stealth mode', no visible icon
- `DroidKungFu`, capable of root-level access on vulnerable Android devices and evade the detection of security software by encrypting its exploits using AES.
  - One of the exploits used was the `RageAgainstTheCage` (RATC) exploit.
  - Also known as *adb setuid exhaustion attack*
  - A race condition between RATC and adb-server

Popular Android Malware

- First proof-of-concept malware released in 2008.
  - Causes the phone to accept all incoming calls
  - Turns off the radio, preventing outgoing/incoming calls
  - Causes the phone to end all calls
  - Gathers sensitive information and sends it to the attacker
- `Mobile Spy` spyware, 2009
  - Monitored infected device via web browser, phone calls, text messages, photos, videos, GPS locations
  - Ran in 'stealth mode', no visible icon
- `DroidKungFu`, capable of root-level access on vulnerable Android devices and evade the detection of security software by encrypting its exploits using AES.
  - One of the exploits used was the `RageAgainstTheCage` (RATC) exploit.
  - Also known as *adb setuid exhaustion attack*
  - A race condition between RATC and adb-server
  - See https://thesnkchrmr.wordpress.com/2011/03/24/rageagainstthecage/ for more details about the exploit and its source code

Rootkits & Bootkits

- One of the first Android rootkit was presented at DEF CON 18 (2010)

Rootkits & Bootkits

- One of the first Android rootkit was presented at DEF CON 18 (2010)
- Rootkit was used to track location of smartphone's owner, read SMS and redirect calls

Rootkits & Bootkits

- One of the first Android rootkit was presented at DEF CON 18 (2010)
- Rootkit was used to track location of smartphone's owner, read SMS and redirect calls
- A demo of a clickjacking rootkit targeting Android 4.0, https://www.youtube.com/watch?v=RxpMPrqnxC0

Rootkits & Bootkits

- One of the first Android rootkit was presented at DEF CON 18 (2010)
- Rootkit was used to track location of smartphone's owner, read SMS and redirect calls
- A demo of a clickjacking rootkit targeting Android 4.0, https://www.youtube.com/watch?v=RxpMPrqnxC0
- Bootkit, `Android.Oldboot` (2014) has the capability of reinstalling itself even after all of its working components have been deleted. Primary targets were rooted Android devices.

Bitcoin Mining malware

- In 2014, several malicious apps found on Google Play store were used in a large-scale crypto currency mining operation

Bitcoin Mining malware

- In 2014, several malicious apps found on Google Play store were used in a large-scale crypto currency mining operation
- Contained a hidden crypto miner that stealthily exploit users' device for computational resources

Bitcoin Mining malware

- In 2014, several malicious apps found on Google Play store were used in a large-scale crypto currency mining operation
- Contained a hidden crypto miner that stealthily exploit users' device for computational resources
- Malware was deployed through Wallpaper apps, with more than 500 downloads

Tools to analyze Android Malware

- Mobile Malware can be analyzed in 2 ways: *statically* and *dynamically*

Tools to analyze Android Malware

- Mobile Malware can be analyzed in 2 ways: *statically* and *dynamically*
- Static Analysis: Analyze suspicious app through reverse-engineering

Tools to analyze Android Malware

- Mobile Malware can be analyzed in 2 ways: *statically* and *dynamically*
- Static Analysis: Analyze suspicious app through reverse-engineering
- Dynamic Analysis: Execute the suspicious app in a controlled environment and monitor its behaviors

Tools to analyze Android Malware

- Mobile Malware can be analyzed in 2 ways: *statically* and *dynamically*
- Static Analysis: Analyze suspicious app through reverse-engineering
- Dynamic Analysis: Execute the suspicious app in a controlled environment and monitor its behaviors
- Tools: IDA Pro, JD-Gui, Dex2Jar, Android SDK

Tools to analyze Android Malware

- Mobile Malware can be analyzed in 2 ways: *statically* and *dynamically*
- Static Analysis: Analyze suspicious app through reverse-engineering
- Dynamic Analysis: Execute the suspicious app in a controlled environment and monitor its behaviors
- Tools: IDA Pro, JD-Gui, Dex2Jar, Android SDK
- Countermeasures against Android malware:

Tools to analyze Android Malware

- Mobile Malware can be analyzed in 2 ways: *statically* and *dynamically*
- Static Analysis: Analyze suspicious app through reverse-engineering
- Dynamic Analysis: Execute the suspicious app in a controlled environment and monitor its behaviors
- Tools: IDA Pro, JD-Gui, Dex2Jar, Android SDK
- Countermeasures against Android malware:
  - There is no single solution!
  - Download apps from official markets only
  - Read permissions carefully before downloading and installing an app

Intrusion Detection & Prevention

Intrusion Detection & Prevention

- Two kinds of intrusion detection systems (IDS):
  - Network-based IDS (NIDS)
  - Host-based IDS (HIDS)

Intrusion Detection & Prevention

- Two kinds of intrusion detection systems (IDS):
  - Network-based IDS (NIDS)
  - Host-based IDS (HIDS)
    - Special kind of HIDS: antivirus software (AV)
    - AV is typically more generally anti-malware software (aka virus scanners, malware scanners)

Intrusion Detection & Prevention

- Two kinds of intrusion detection systems (IDS):
  - Network-based IDS (NIDS)
  - Host-based IDS (HIDS)
    - Special kind of HIDS: antivirus software (AV)
    - AV is typically more generally anti-malware software (aka virus scanners, malware scanners)
- Some systems have additional capabilities to *prevent* intrusion
- Those systems are called intrusion prevention systems (IPS), again:
  - Network-based IPS (NIPS)
  - Host-based IPS (HIPS)

Intrusion Detection & Prevention

- Two kinds of intrusion detection systems (IDS):
  - Network-based IDS (NIDS)
  - Host-based IDS (HIDS)
    - Special kind of HIDS: antivirus software (AV)
    - AV is typically more generally anti-malware software (aka virus scanners, malware scanners)
- Some systems have additional capabilities to *prevent* intrusion
- Those systems are called intrusion prevention systems (IPS), again:
  - Network-based IPS (NIPS)
  - Host-based IPS (HIPS)
- IDS/IPS tool: SNORT (more on this later)

NIDS

- Network-based intrusion detection system

NIDS

- Network-based intrusion detection system
- NIDS monitors traffic on its network segment as a data source
- This is achieved by placing the network interface in *promiscuous mode* to capture all traffic that crosses its network segment

NIDS

- Network-based intrusion detection system
- NIDS monitors traffic on its network segment as a data source
- This is achieved by placing the network interface in *promiscuous mode* to capture all traffic that crosses its network segment
- Packets matching these 3 types of signatures are considered of interest:

NIDS

- Network-based intrusion detection system
- NIDS monitors traffic on its network segment as a data source
- This is achieved by placing the network interface in *promiscuous mode* to capture all traffic that crosses its network segment
- Packets matching these 3 types of signatures are considered of interest:
  - String signatures
  - Port signatures
  - Header condition signatures

NIDS

- Network-based intrusion detection system
- NIDS monitors traffic on its network segment as a data source
- This is achieved by placing the network interface in *promiscuous mode* to capture all traffic that crosses its network segment
- Packets matching these 3 types of signatures are considered of interest:
  - String signatures
  - Port signatures
  - Header condition signatures
- String signatures look for a text string that indicates a possible attack. For example: `"cat "+ +" > /.rhosts"` might cause a UNIX system to become extremely vulnerable to network attack

NIDS

- Network-based intrusion detection system
- NIDS monitors traffic on its network segment as a data source
- This is achieved by placing the network interface in *promiscuous mode* to capture all traffic that crosses its network segment
- Packets matching these 3 types of signatures are considered of interest:
  - String signatures
  - Port signatures
  - Header condition signatures
- String signatures look for a text string that indicates a possible attack. For example: `"cat "+ +" > /.rhosts"` might cause a UNIX system to become extremely vulnerable to network attack
- Port signatures monitor connection attempts to well-known, frequently attacked ports. Examples of these ports include telnet (TCP port 23), FTP (TCP port 21/20), SUNRPC (TCP/UDP port 111), and IMAP (TCP port 143)

NIDS

- Network-based intrusion detection system
- NIDS monitors traffic on its network segment as a data source
- This is achieved by placing the network interface in *promiscuous mode* to capture all traffic that crosses its network segment
- Packets matching these 3 types of signatures are considered of interest:
  - String signatures
  - Port signatures
  - Header condition signatures
- String signatures look for a text string that indicates a possible attack. For example: `"cat "+ +" > /.rhosts"` might cause a UNIX system to become extremely vulnerable to network attack
- Port signatures monitor connection attempts to well-known, frequently attacked ports. Examples of these ports include telnet (TCP port 23), FTP (TCP port 21/20), SUNRPC (TCP/UDP port 111), and IMAP (TCP port 143)
- Header signatures watch for suspicious combinations in packet headers. For example: a TCP packet with both the SYN and FIN flags set, signifying that the requester wishes to start and stop a connection at the same time.

HIDS

- Host-based intrusion detection system
- HIDS goes beyond malware scanning (although there may be some overlap)
- Typically register certain resources with the IDS, those resources are monitored
- Examples of resources: system files, Windows registry entries, network ports

HIDS

- Host-based intrusion detection system
- HIDS goes beyond malware scanning (although there may be some overlap)
- Typically register certain resources with the IDS, those resources are monitored
- Examples of resources: system files, Windows registry entries, network ports
- Idea: remember state of resource, detect modifications
- Typically store hash values of resources
- Crucial to protect the table of hashes!

HIDS

- Host-based intrusion detection system
- HIDS goes beyond malware scanning (although there may be some overlap)
- Typically register certain resources with the IDS, those resources are monitored
- Examples of resources: system files, Windows registry entries, network ports
- Idea: remember state of resource, detect modifications
- Typically store hash values of resources
- Crucial to protect the table of hashes!
- Additionally, analyze log files (e.g., `/var/log/syslog`)
- For log-file analysis, two possibilities:
  - Signature-based intrusion detection
  - Behavior-based intrusion detection

HIDS

- Host-based intrusion detection system
- HIDS goes beyond malware scanning (although there may be some overlap)
- Typically register certain resources with the IDS, those resources are monitored
- Examples of resources: system files, Windows registry entries, network ports
- Idea: remember state of resource, detect modifications
- Typically store hash values of resources
- Crucial to protect the table of hashes!
- Additionally, analyze log files (e.g., `/var/log/syslog`)
- For log-file analysis, two possibilities:
  - Signature-based intrusion detection
  - Behavior-based intrusion detection
- Problem of signature-based IDS: same as with AV

HIDS

- Host-based intrusion detection system
- HIDS goes beyond malware scanning (although there may be some overlap)
- Typically register certain resources with the IDS, those resources are monitored
- Examples of resources: system files, Windows registry entries, network ports
- Idea: remember state of resource, detect modifications
- Typically store hash values of resources
- Crucial to protect the table of hashes!
- Additionally, analyze log files (e.g., `/var/log/syslog`)
- For log-file analysis, two possibilities:
  - Signature-based intrusion detection
  - Behavior-based intrusion detection
- Problem of signature-based IDS: same as with AV
- Problem of behavior-based IDS: hard to obtain good detection rate at low false-positive rate in highly dynamic systems

NIPS

- Network-based intrusion prevention system
- NIPS is a system that monitors a network as well as protect the confidentiality, integrity and availability of network.

NIPS

- Network-based intrusion prevention system
- NIPS is a system that monitors a network as well as protect the confidentiality, integrity and availability of network.
- NIPS utilizes one of the 3 detection methods:

NIPS

- Network-based intrusion prevention system
- NIPS is a system that monitors a network as well as protect the confidentiality, integrity and availability of network.
- NIPS utilizes one of the 3 detection methods:
  - Signature-based detection: Signatures are attack patterns predetermined and pre-configured. This detection method monitors the network traffic and compares it with the pre-configured signatures so as to find a match.

NIPS

- Network-based intrusion prevention system
- NIPS is a system that monitors a network as well as protect the confidentiality, integrity and availability of network.
- NIPS utilizes one of the 3 detection methods:
  - Signature-based detection: Signatures are attack patterns predetermined and pre-configured. This detection method monitors the network traffic and compares it with the pre-configured signatures so as to find a match.
  - Anomaly-based detection: This method of detection creates a baseline on average network conditions. Once a baseline has been created, the system intermittently samples network traffic on the basis of statistical analysis and compares the sample to the created baseline.

NIPS

- Network-based intrusion prevention system
- NIPS is a system that monitors a network as well as protect the confidentiality, integrity and availability of network.
- NIPS utilizes one of the 3 detection methods:
  - Signature-based detection: Signatures are attack patterns predetermined and pre-configured. This detection method monitors the network traffic and compares it with the pre-configured signatures so as to find a match.
  - Anomaly-based detection: This method of detection creates a baseline on average network conditions. Once a baseline has been created, the system intermittently samples network traffic on the basis of statistical analysis and compares the sample to the created baseline.
  - Protocol state analysis detection: This type of detection method identifies deviations of protocol states by comparing observed events with predefined profiles

SNORT

- Can be used for IDS, IPS
- Free and open source

SNORT

- Can be used for IDS, IPS
- Free and open source
- Uses a simple rules description language to create rules
- Snort rules are divided into 2 logical sections: rule header and rule options

SNORT

- Can be used for IDS, IPS
- Free and open source
- Uses a simple rules description language to create rules
- Snort rules are divided into 2 logical sections: rule header and rule options
- The rule header contains the rule's action, protocol, source and destination IP addresses and netmasks, and the source and destination ports information

SNORT

- Can be used for IDS, IPS
- Free and open source
- Uses a simple rules description language to create rules
- Snort rules are divided into 2 logical sections: rule header and rule options
- The rule header contains the rule's action, protocol, source and destination IP addresses and netmasks, and the source and destination ports information
- The rule option section contains alert messages and information on which parts of the packet should be inspected to determine if the rule action should be taken.

SNORT

- Can be used for IDS, IPS
- Free and open source
- Uses a simple rules description language to create rules
- Snort rules are divided into 2 logical sections: rule header and rule options
- The rule header contains the rule's action, protocol, source and destination IP addresses and netmasks, and the source and destination ports information
- The rule option section contains alert messages and information on which parts of the packet should be inspected to determine if the rule action should be taken.
- `action`proto src_ip src_port direction dst_ip dst_port (options)

SNORT

- Can be used for IDS, IPS
- Free and open source
- Uses a simple rules description language to create rules
- Snort rules are divided into 2 logical sections: rule header and rule options
- The rule header contains the rule's action, protocol, source and destination IP addresses and netmasks, and the source and destination ports information
- The rule option section contains alert messages and information on which parts of the packet should be inspected to determine if the rule action should be taken.
- `action`proto src_ip src_port direction dst_ip dst_port (options)
- Example: `log tcp any :1024 -> 192.168.1.0/24 500:`

SNORT

- Can be used for IDS, IPS
- Free and open source
- Uses a simple rules description language to create rules
- Snort rules are divided into 2 logical sections: rule header and rule options
- The rule header contains the rule's action, protocol, source and destination IP addresses and netmasks, and the source and destination ports information
- The rule option section contains alert messages and information on which parts of the packet should be inspected to determine if the rule action should be taken.
- `action`proto src_ip src_port direction dst_ip dst_port (options)
- Example: `log tcp any :1024 -> 192.168.1.0/24 500:`
- Log tcp traffic from priveleged ports less than or equal to 1024 going to ports greater than or equal to 500

Recover after intrusion

- Easy situation: download a file from the Internet, AV complains.
  $\Rightarrow$ Don't run/open file, but stop download (or delete file).

Recover after intrusion

- Easy situation: download a file from the Internet, AV complains.
  $\Rightarrow$ Don't run/open file, but stop download (or delete file).
- Hard situation: AV complains about *old* files (or IDS reports intrusion)

Recover after intrusion

- Easy situation: download a file from the Internet, AV complains.
  $\Rightarrow$ Don't run/open file, but stop download (or delete file).
- Hard situation: AV complains about *old* files (or IDS reports intrusion)
- AV software typically offers to "remove the virus/worm/trojan"
- Question: Is that enough?

Recover after intrusion

- Easy situation: download a file from the Internet, AV complains.
  $\Rightarrow$ Don't run/open file, but stop download (or delete file).
- Hard situation: AV complains about *old* files (or IDS reports intrusion)
- AV software typically offers to "remove the virus/worm/trojan"
- Question: Is that enough?
- There is only one responsible answer: **No.**

Recover after intrusion

- Easy situation: download a file from the Internet, AV complains.
  $\Rightarrow$ Don't run/open file, but stop download (or delete file).
- Hard situation: AV complains about *old* files (or IDS reports intrusion)
- AV software typically offers to "remove the virus/worm/trojan"
- Question: Is that enough?
- There is only one responsible answer: **No.**
- Once a system has been compromised, you don't know what else is broken
- Only reasonable recovery from intrusion:
  - Isolate the system (to prevent further damage)
  - Analyze what was compromised and how (forensics)
  - Restore to a clean state (reinstall, restore clean data backup)