Automated Web Patrol with Strider HoneyMonkeys:
Finding Web Sites That Exploit Browser Vulnerabilities

Yi-Min Wang, Doug Beck, Xuxian Jiang, Roussii Roussev, Chad Verbowski, Shuo Chen, and Sam King
Microsoft Research, Redmond

Lisa Johansen
The softer side of Microsoft

• They develop and maintain the most widely distributed operating system and web browser

• They must deal with the implications of doing so
  – Popular attack target
  – Large distribution of fixes
Patch Tuesday

• “Patch Tuesday is the second Tuesday of each month, the day on which Microsoft releases security patches.”

• Exploit Wednesday: “Many exploits are seen shortly after the release of a patch. By analyzing the patch, exploit developers can more easily figure out how to exploit the underlying vulnerability.”
... and the other side

- They have a ridiculous amount of money, resources, and talent
- The research that they are able to perform is (largely) only able to be performed by them
- Another example: Google
  - What can we learn from their research?
The problem

• Malicious or hacked web sites can install malcode by exploiting browser and OS vulnerabilities
  – Visitation only - no interaction

• Whose fault is this?
2 step process

1) Finding the bad websites
2) Stop them from infecting systems
1) Finding the bad websites

- Choose URLs to check
- Use Strider HoneyMonkeys to find out if the site installs malicious code
- Find out if any other sites or URLs are involved
- Determine to what level of patches the exploit works
Choosing URLs

• Suspicious URLs
  – Those known to host malware, phishing links, porn, typos of popular websites, etc.

• Popular websites
  – Google, Amazon, CNN, etc.

• Specific use websites
  – Make sure my website has not been compromised or that I am not visiting compromised websites
Strider HoneyMonkeys

- VMs with different patch levels (Windows) and versions of IE run “monkey” programs
- The “Strider Tracer” catches illegal actions outside of the sandbox indicating an exploit
- The first step is to examine large sets of sites and, if an exploit is found, look at each site individually
Who is involved?

- The system can determine through recursive redirection what other sites are involved
  - Identify relationships
- Allows for the creation of relationship graphs
  - May find some interesting things
How bad is it?

• The final stage of the process increases the patch level to determine how “strong” the exploit is
  – Allows for identification of known exploits
  – Allows for discovery of zero-day exploits
2) Stop them from infecting systems

- Patch it (and then release it on Tuesday)
- Be Microsoft and carry a big stick (make it stop)
Methodology

• Implement and execute this system over a large period of time
  – Windows XP at different patch levels
• Examine characteristics of findings
  – This is very useful and interesting
• QED
Results

• Topology graphs led to identification of exploit sites
  – Know what kind of sites to be aware of (porn, song lyrics, game cheats, celebrities, wallpapers, wrestling)
  – Watch out for major sites with every new exploit
• Popular sites are hit too
• They found a zero-day exploit
• Others
Further Problems

• Elude the HoneyMonkeys
  – The time tradeoff
  – Make sure a human is present
  – Blacklist the machines
  – Detect VMs
  – Randomizing the attacks

• VSED
  – Insert breakpoints to stop execution of potentially malicious code
  – Not complete
How is this research useful?

• For Microsoft
• For the rest of the research community