Fixing the Fixing

Mitja Kolsek
CEO and Co-Founder
0patch and ACROS Security
@mkolsek, @0patch

Stanka Salamun
COO and Co-Founder
0patch and ACROS Security
@0patch
16 Years of Breaking in...

FIND PUBLIC EXPLOIT for a known vulnerability

TAILOR EXPLOIT to work with your RAT

MUTATE EXPLOIT until VirusTotal doesn't recognize it

PHISH THE TARGET until you're in
„But… We have all this cool technology“
Beating Around the Bush
Your Knee Hurts?

Doctors:

„No problem, we’ll cut off your leg and replace it with a new one.“
Security Update Gap
Are 0-Days a Real Problem?

“\textit{We don't need zero-days to get inside your network.}”

Rob Joyce,
NSA Hacker-In-Chief
Updates: Days from release to install

176

* US banks; source: NopSec, 2015 State of Vulnerability Risk Management
Updates: Days from release to exploit

3

* Source: FireEye, Angler EK Exploiting Adobe Flash CVE-2015-0359 with CFG Bypass
111,000,000,000

* Cybersecurity Ventures, 0 day report Q1 2017 prediction: 111 billions lines of new code
Patching is Still a Hard Problem
But it’s someone else’s problem

END OF LIFE PRODUCTS
• Win Srv 2003, Win XP
• Java JRE 7, IE9, IE10

UNPATCHED VULNERABILITIES
• 0days
• known vulnerabilities

INTER-OPERABILITY REQUIREMENTS

LEGACY SYSTEMS
• SCADA
• Mainframes

3rd PARTY LIBRARIES
• OpenSSL

IoT
• botnets
• massive attacks against and from IoT

OLD VERSIONS
• Java
• Flash
• QuickTime
Users

- Hate **downtime**
- Expensive patch deployment
- **Complex** patches – no control of new code
- Uninstalling patches
- Big official updates change functionalities
- Anti-malware protections **bypassable**
- Updating = risk of breakage
- Not updating = risk of ownage

Software Vendors

- Direct and opportunity costs
- Patch development „traditional“ and long
- Testing and distributing fixes is costly
- Have **better** things to do
We couldn’t complete the updates
Undoing changes
Don’t turn off your computer
Emerging Alternatives in Patching
Evolution of Patching

- No patching
- „Fat“ patching
- „Live“ patching
- Micro patching
(Re-)Emerging Patching Trends

- Live ("hot") patching
- Runtime Application Self-Protection (RASP)
- Virtual patching
Live Patching

App patching (Jspatch)

Hot patching (discon.)

Opatch

Adaptive kernel live (Baidu)

Patch Droid

LIVE PATCHING

Linux/UX

Cloud

Opatch

Ksplice

Kpatch

Kexec

kGraft

Live Update

Cloud Linux

Kernel Care

XEN Project

RSA Conference 2017
Linux Live (or „hot“) Patching

Key Characteristics

- No system/application rebooting
- „unpatch“ feature
- Focused on kernel patching
- From source code, decently automated
- Replacing entire functions (problem if the function is executing)
Linux Live Patching: Before

![Diagram showing the original function call with NOP bytes and return path](image.png)
Linux Live Patching: After

Original Function

CALL/JMP

Replacement Function
Linux Live Patching Today

Shortcomings

• Source code needed to replace entire function
  • No patching of closed-source applications
• Original function must be prepared to be patchable (NOP prolog)
• Patching and unpatching functions on call stack is risky and complex
• Vendor still has monopoly on patches
Micropatching: Next-Generation Live Patching

Fundamentally changing the security game!
1. Patching closed-source code
2. Minimal risk of defects
3. Enable 3rd-party review of patches
4. Enable anyone to contribute patches
Micropatching: Before

call

Function

Some instructions

return
Micropatching: After

- Function
  - JMP
  - Micropatch
    - Some instructions

call

return
Micropatching Advantages

- **MINIMAL CODE CHANGES**
  minimal risk, easy to review

- **3RD PARTY „CROWDPATCHING“**
  even for closed source

- **LOW BANDWIDTH**
  smart grid, satellite, HF radio, SMS

- **NO DELAYS**
  for functions currently on call stack

- **IOT: REMOTE PATCHING AND UNPATCHING**
  automatic and safe

- **POTENTIAL FOR FORMAL PROOFS**
  and code-change impact analysis
Demo: Micropatching WebEx
Chrome Extensions

Extensions

Cisco WebEx Extension 1.0.1
Join WebEx meetings using Google Chrome™
Details
- Allow in incognito
- Enabled

Google Docs 0.9
Create and edit documents
Details
- Allow in incognito
- Enabled

Google Docs Offline 1.4
Get things done offline with the Google Docs family of products.
Details
- Allow in incognito
- Enabled
What Can be Micropatched?

Any „reasonably static“ code

- Native binary files (executables, drivers, libraries)
- Compiled bytecode (Java, C#)
- Just-in-time compiled code
- „Installable“ web applications (WordPress, Magento, Bugzilla, etc)
- IoT devices
- Medical devices
- Mobile devices – OS and apps
Not Ideal for Micropatching

- Code that is often manually modified
  - Administrative scripts
  - PHP, Perl scripts

- Code that is not deployed to users
  - In-house web applications (easy to manually modify)
Goal: Decoupling Security Patches From (Mostly Functional) Updates

|--------------------|------------|--------------------------|--------------------------|--------------------------|--------------------------|--------------------------|------------|--------------------------|-----|

...
What Can You Do?

Getting micropatching off the ground
Organizations and Users

Tomorrow

- Measure your Security Update Gap
- Find main reasons for your delays in applying security patches

Next six months

- Consider using existing live patching for updating your Linux servers
- Set up a test process for applying micropatches wherever possible
Software Vendors

Tomorrow
• Calculate your users’ costs because of „fat“ (conventional) patching
• Analyze your total production, testing, deployment and PR costs for in-house security patch production

Next six months
• Launch a micropatching pilot with one product
• IoT vendors: consider automatic micropatching of your devices
Researchers

**Tomorrow**
- Arm yourself with powerful tools (WinDbg, IDA, binary editors)
- Download your copy of free 0patch Agent for Developers and play with it

**Next six months**
- Brush up on your low level programming, reverse engineering skills
- When preparing an exploit PoC, also write a micropatch
Malicious Use of Live Patching

SWIFT - Bank of Bangladesh

- BAE Systems: „Two bytes to $951m“
- SWIFT Alliance Access Software „micropatched“
- 2 bytes of liboradb.dll replaced with NOP
Software Patching Sci-Fi

It's 2025.

People are using 3rd party patches for "dumbing down" their smart devices, blocking vendors from peeking in their fridge and collecting data.
200 micropatches walk into a bar.

...Nobody notices.

**Thumbs up** if you think that’s how patching should look like in the future.
Let’s Fix the Fixing!

We can make attackers’ job much, much harder.