Finding bugs and publishing advisories – the Core Security way

Carlos Sarraute

Core Security Technologies

and *Ph.D. program in Informatics Engineering, ITBA*

H2HC

Hackers to Hackers Conference 2010
São Paulo - Cancun

H2HC São Paulo, 27/28 Nov 2010

CORE LABS
RESEARCH & DEVELOPMENT

Brief presentation

- My company: Core Security Technologies
  - Boston (USA)
    - marketing and sales
  - Buenos Aires (Argentina)
    - research and development
- About me:
  - M.Sc. in Mathematics from UBA
  - I have worked as researcher in CoreLabs since 2000
  - One of my focus areas: applying Artificial Intelligence techniques to solve problems from the security field
    - OS detection using neural networks
    - Automated attack planning (see H2HC'09 presentation)

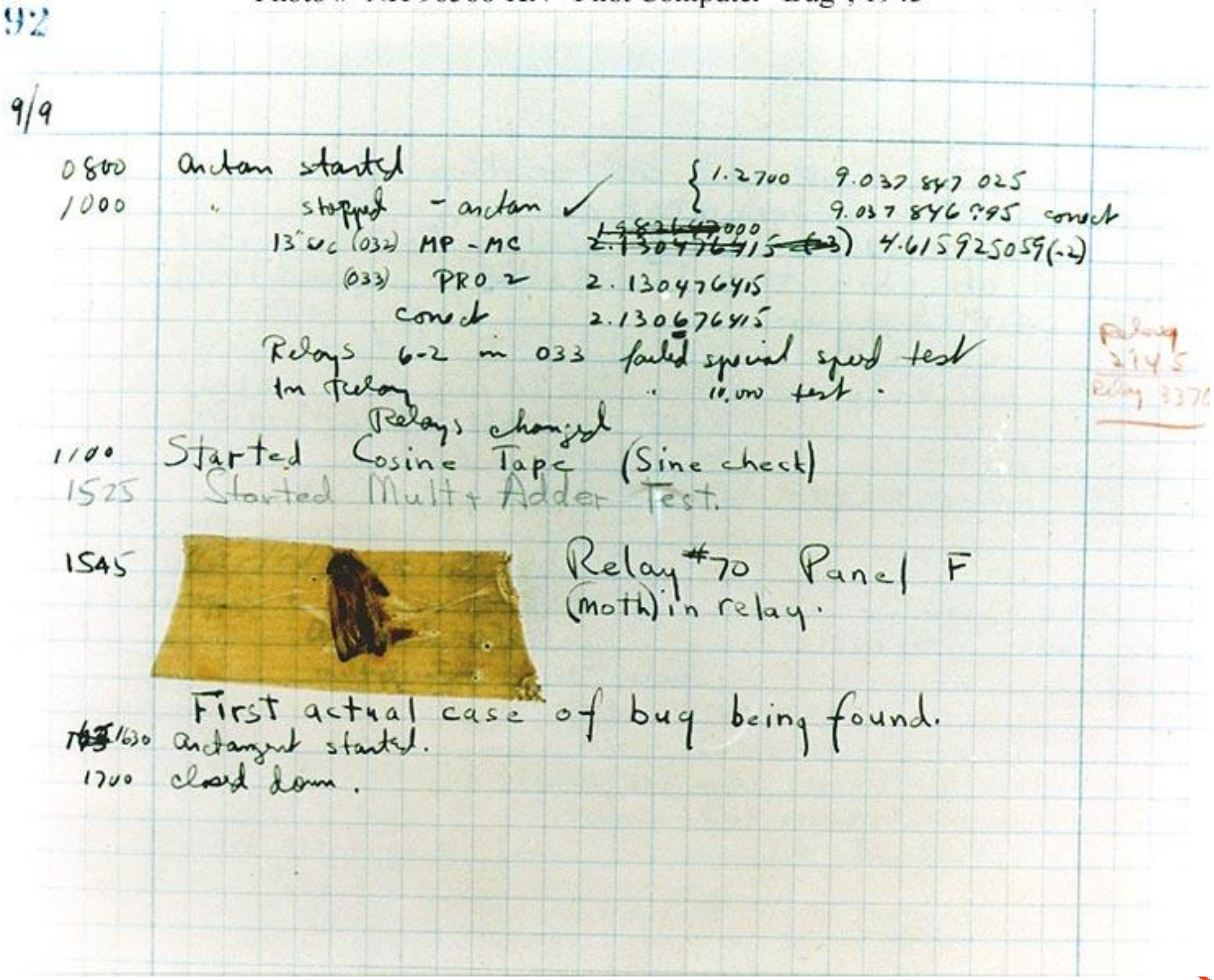
Outline

1. Bug fishing activities
2. The bug reporting and publication process
3. How we have improved our process

Bug fishing activities

First bug being found (9/9/1945)

Photo # NH 96566-KN First Computer "Bug", 1945



92

9/9

0800 Anctan started

1000 " stopped - anctan ✓ {1.2700 9.037 847 025

9.037 846 795 correct

13" sec (032) MP - MC ~~1.982147000~~ ~~2.130476415 (-3)~~ 4.615925059(-2)

(033) PRO 2 2.130476415

correct 2.130676415

Relays 6-2 in 033 failed special speed test

In relay " 11,000 test .

Relays changed

1100 Started Cosine Tape (Sine check)

1525 Started Mult + Adder Test.

1545 Relay #70 Panel F (moth) in relay.

First actual case of bug being found.

1630 Antangent started.

1700 closed down.

Relay 2145

Relay 3370

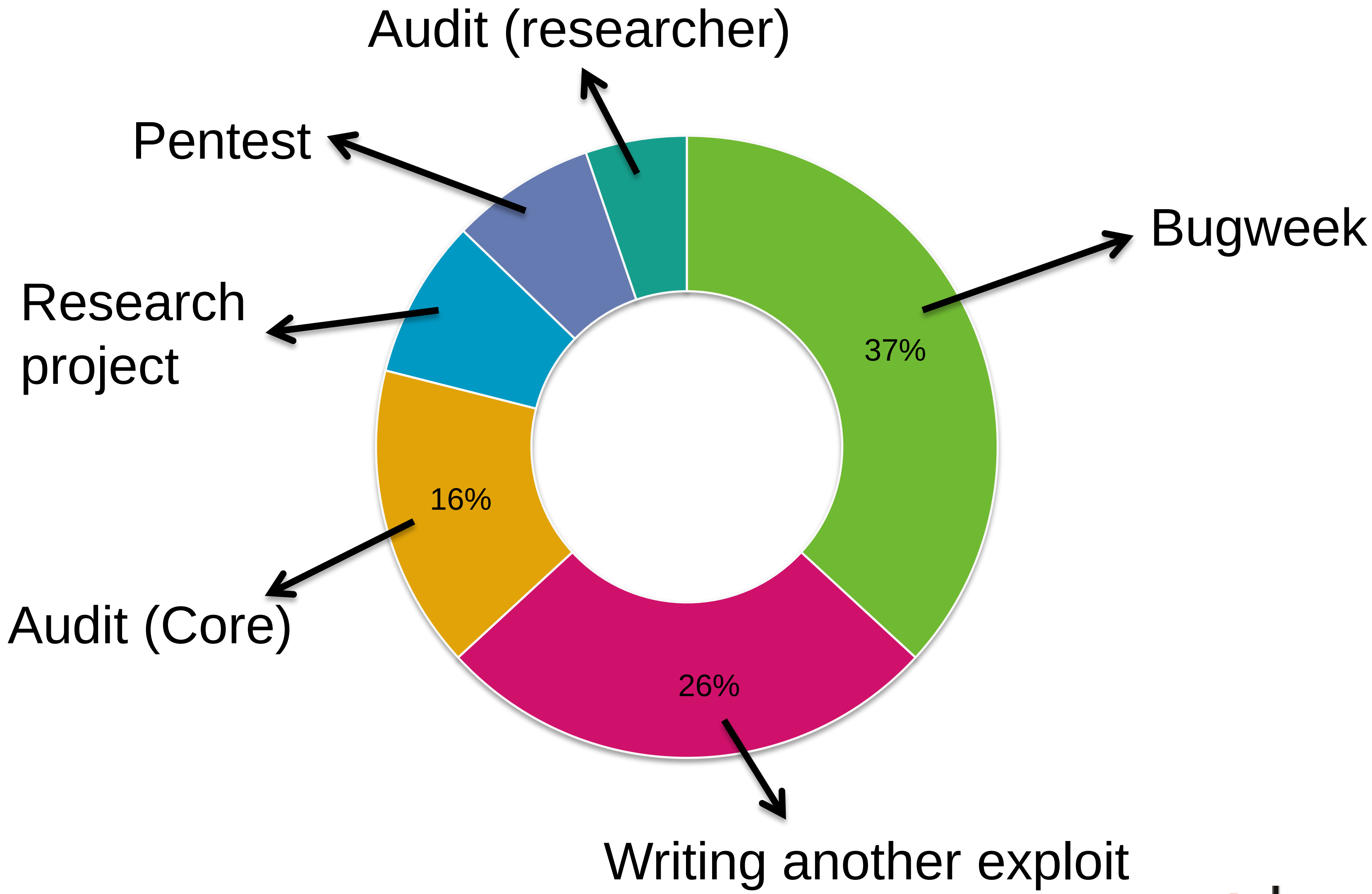
Core's vulnerability research

- Core founded in 1996 in Buenos Aires, Argentina
  - involved in security research and vulnerability discovery ever since
- Early adopters of the public disclosure process of software bugs (mid 1990s)
- 146 advisories published (stats based on this sample)
  - plus papers and technical articles
- Several hundredths of bugs reported.
- Coordinated bug reports with Microsoft, Cisco, Sun, SGI, IBM, Digital, HP, all Linux vendors, BSD, etc.
- CVE Numbering Authority (CNA)

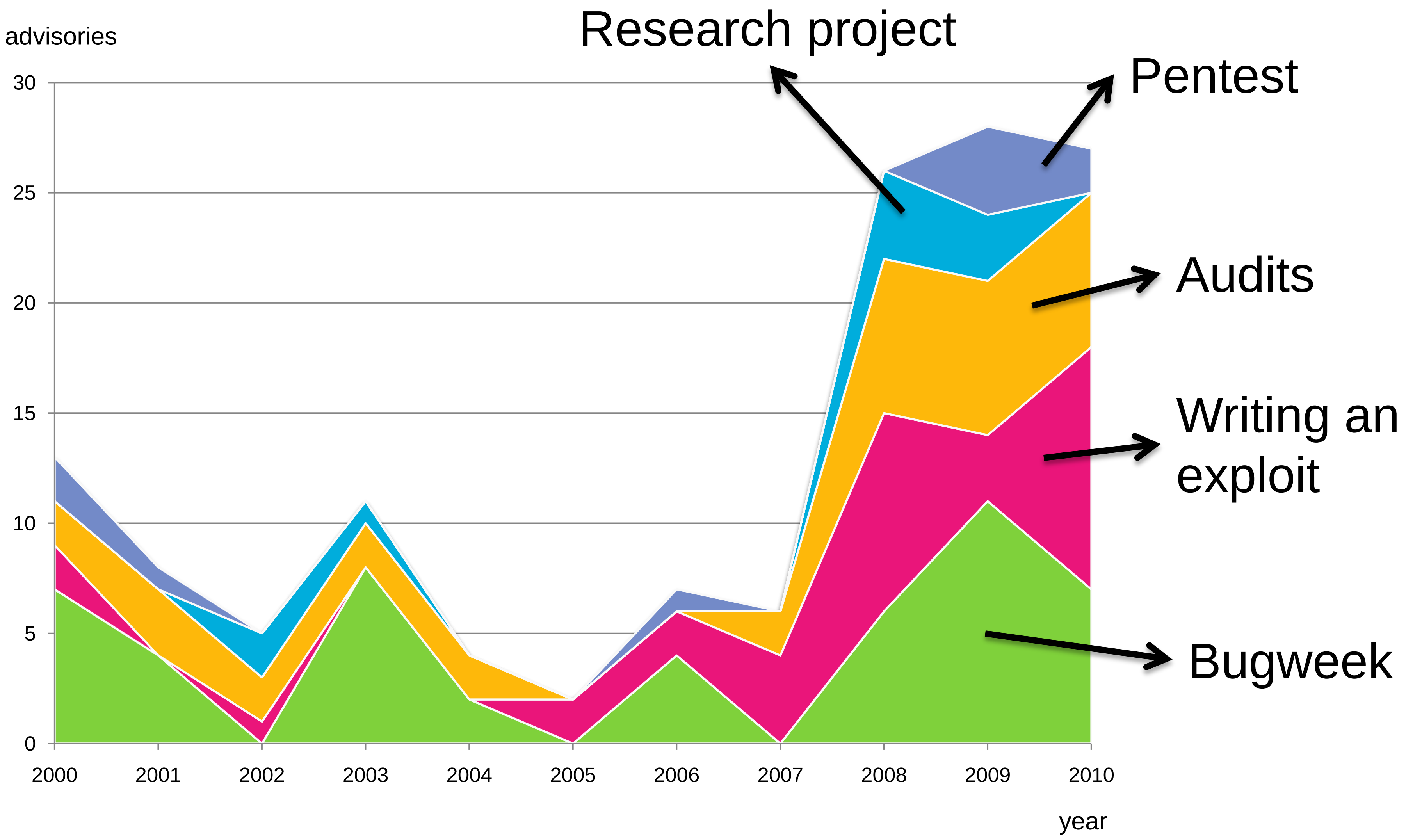
Why do we look for bugs?

- The end goal is to help vulnerable users & organizations understand and mitigate risk
- Not a revenue generating activity
  - Brand and technical recognition
- Knowledge acquisition and transfer
  - Good way to learn about information security
- Research activity
  - Advancement of the discipline
- Sometimes bugs are found without looking for them

When do we find bugs?



Bug finding context – evolution



Research project

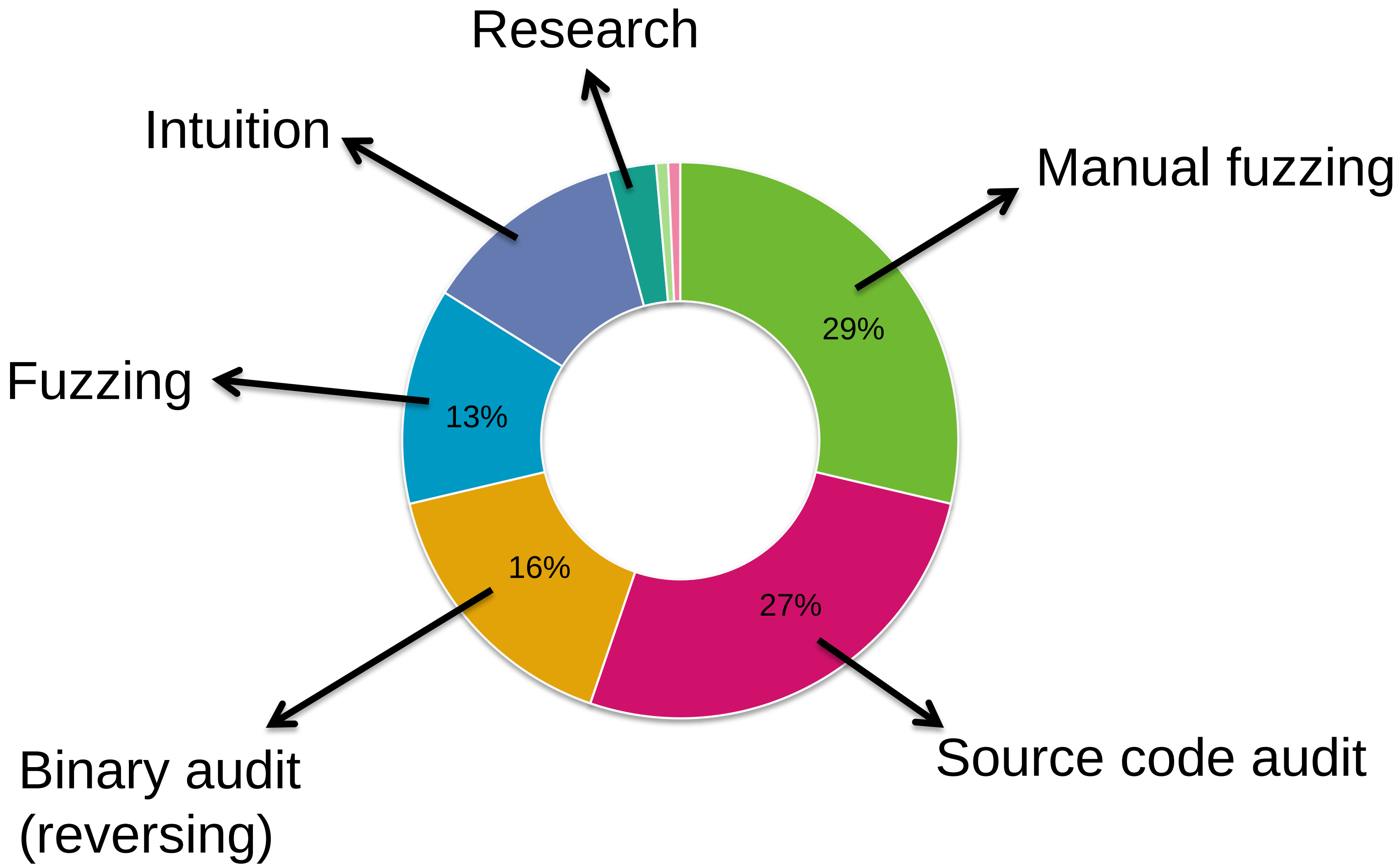
Pentest

Audits

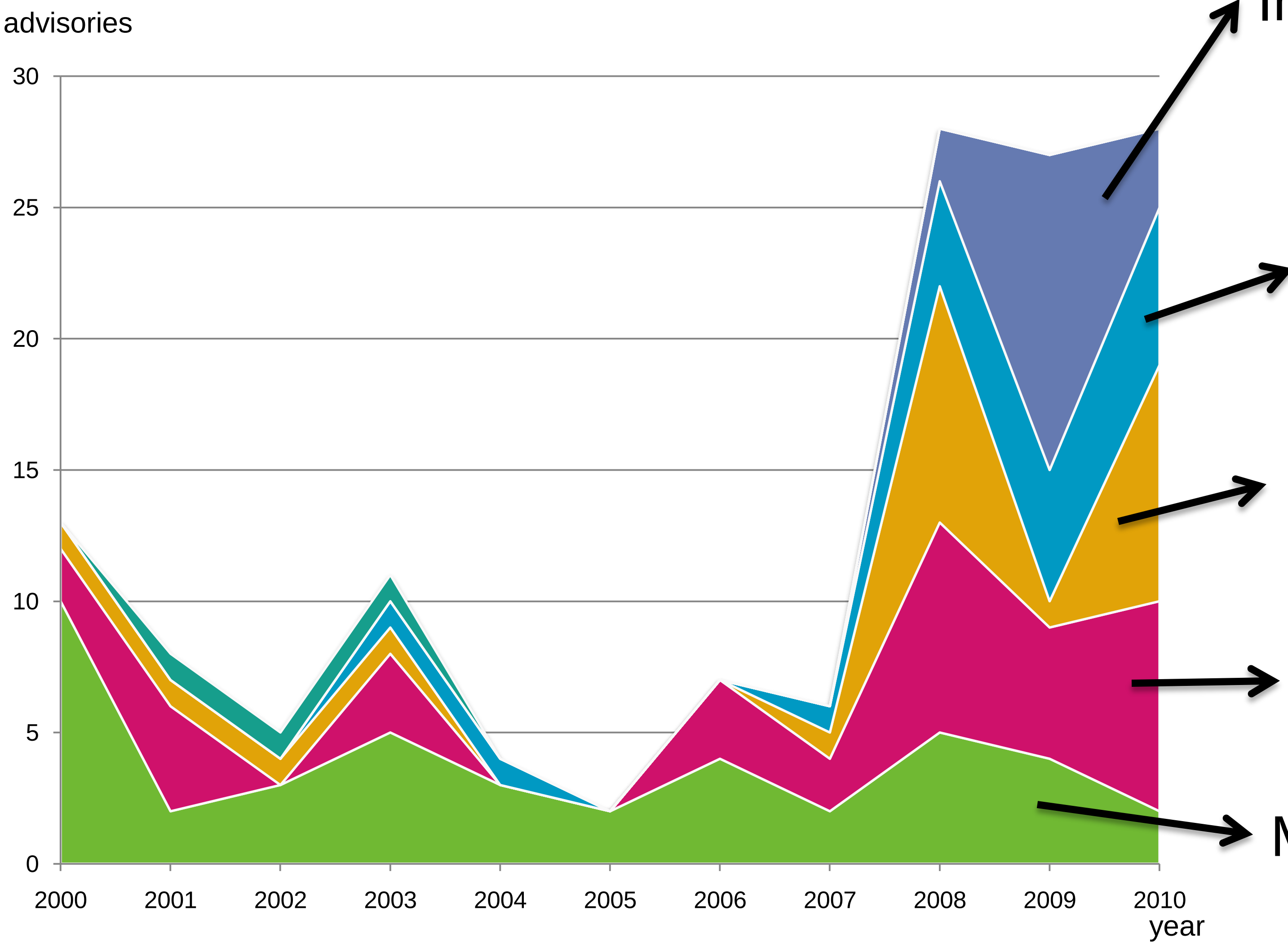
Writing an exploit

Bugweek

How do we find bugs?



Methodologies – evolution



Intuition

Fuzzing

Binary audit

Source code audit

Manual fuzzing

The Bugweek

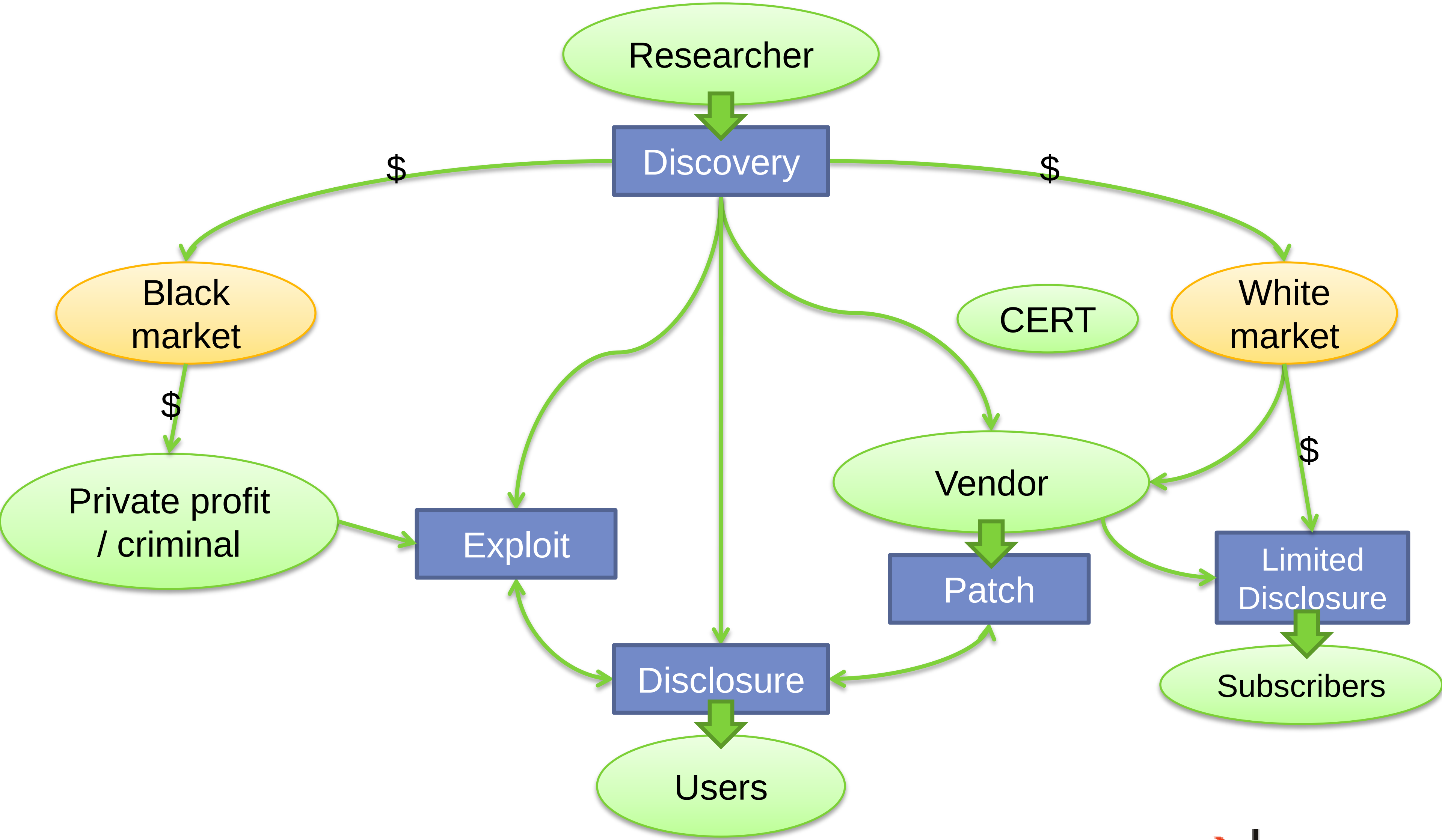
- Main vulnerability research activity
- All the security professionals of the company dedicate one week to bughunting
  - From developers to exploit writers & QA analysts
- Prior to the Bugweek, employees are invited to workshops
  - Source code audit, fuzzing, webapps security, etc.
- More learning and working material
  - Bug Fisher Manual
  - Documentation of previous Bugweeks
  - Tools and fuzzers written in previous years
  - Repository of Degenerated Files

The Bugweek teams

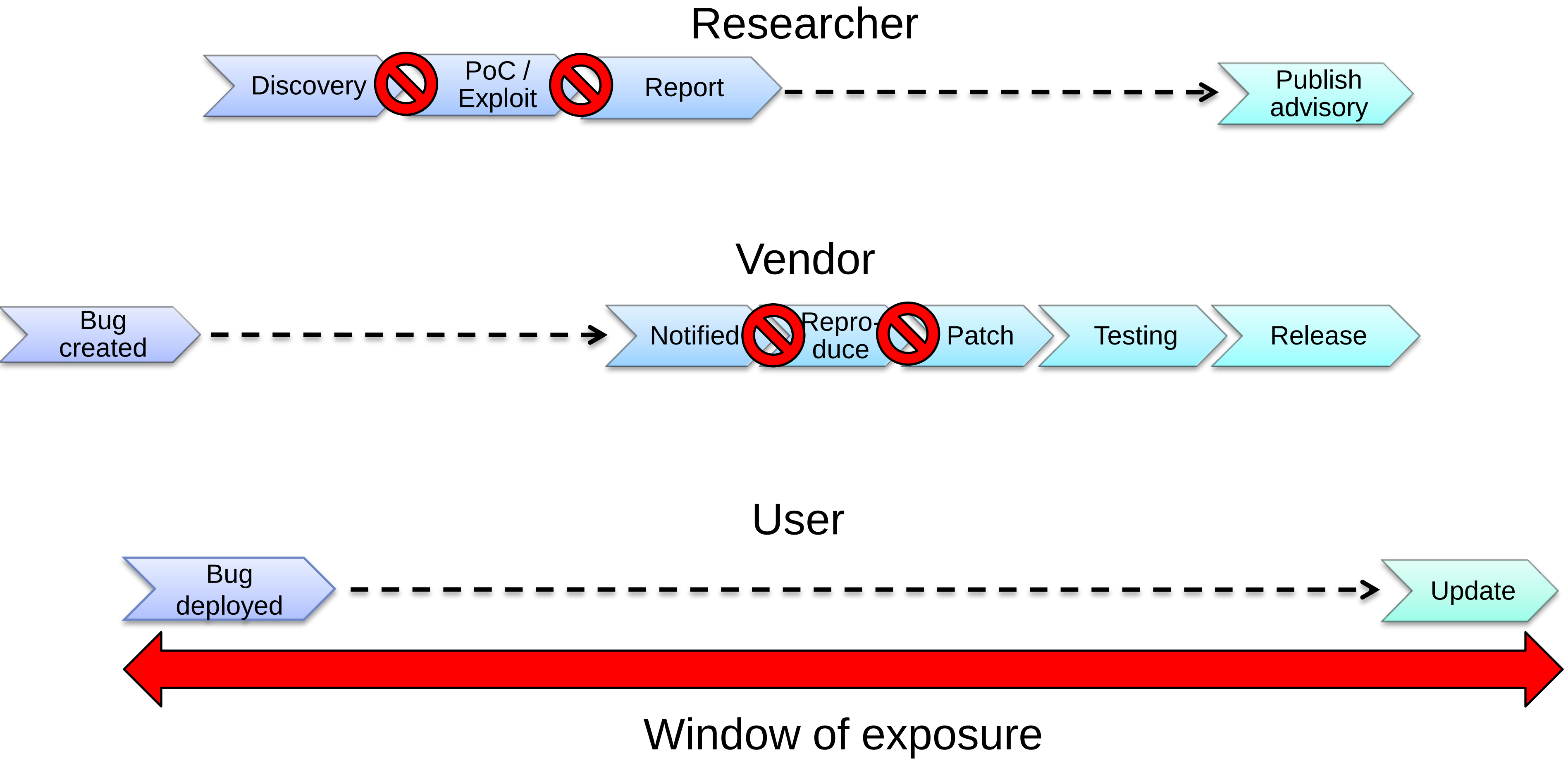
- Employes are organized into teams
  - ~20 teams of ~5 persons
  - The captain has technical skills
  - We used Integer Linear Programming to define the teams
    - Input: each captain “bids” on who he wants in his team
- Result: a set of teams that mix skill sets from different departments
  - Team building experience
  - Knowledge transfer
  - For example, a GUI developer with little security background gets the chance to work with an expert exploit writer
  - Each team decides its targets and methodologies

The Bug reporting and publication process

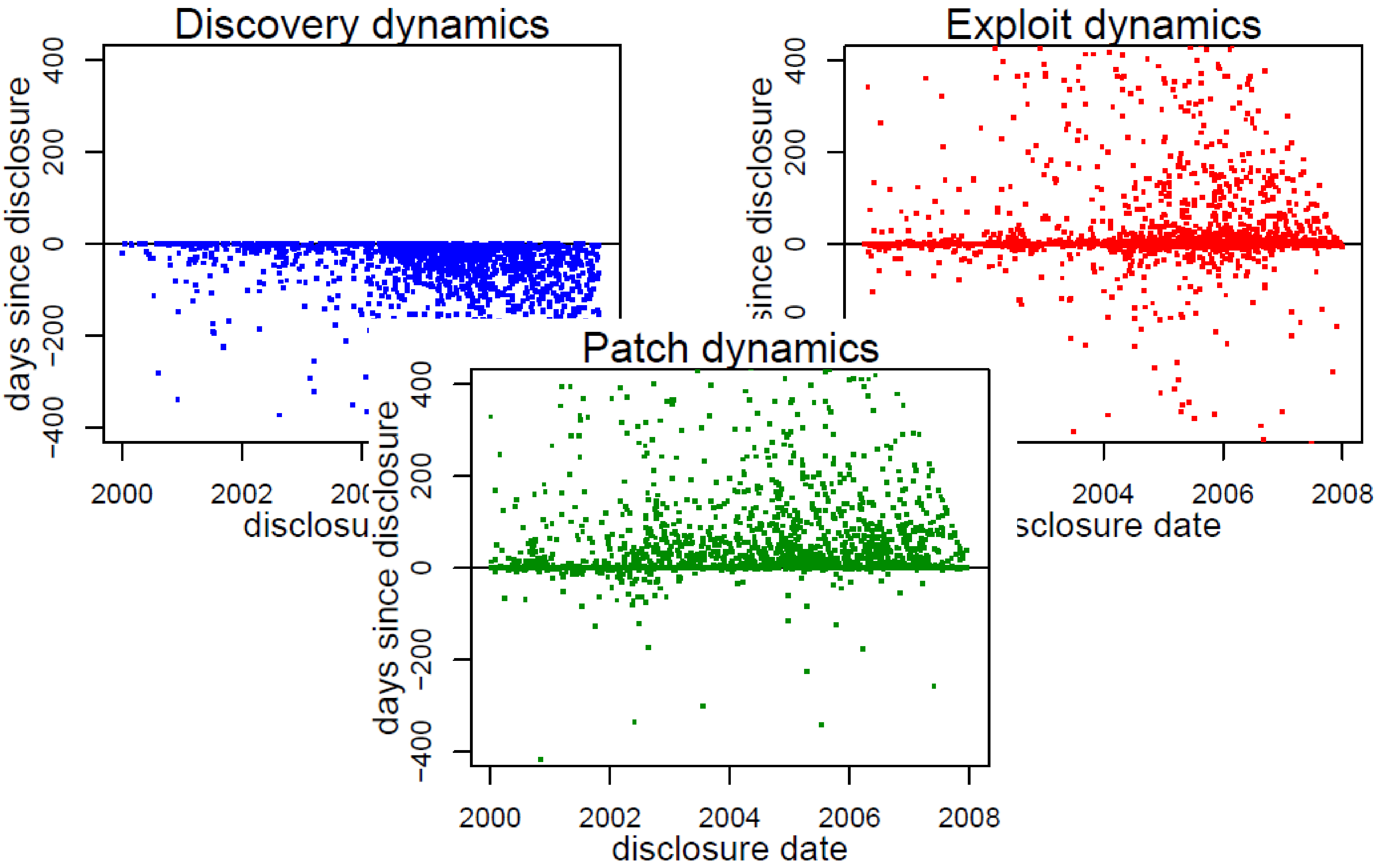
Actors in the Security Ecosystem



Bugs in the Bug reporting process



Vulnerability Lifecycle dynamics



Discovery dynamics

Exploit dynamics

Patch dynamics

Stefan Frei et al. *Modelling the Security Ecosystem - The Dynamics of (In)Security*

Disclosure guidelines

- Keep in mind the objectives of the advisory
  - *Final objective:* Inform users of the vulnerability
  - *Short term objective:* Inform vendor of the bug
    - With enough info to reproduce the bug
  - *Broader objective:* Inform the security community
    - Understand root cause of the bug
    - Analyze variants of the bug
    - Discuss exploitation techniques
- Keep it simple
  - The process is resource-consuming (mostly time)
  - Always have clear deadlines
- Minimize harm / protect users

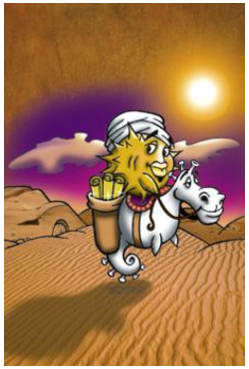
Communication is key

- Vendor learns about the vulnerability
- Researcher learns about the vendor’s analysis of the vulnerability and the patch development process
  - Continued communications between vendor and researcher are fundamental
- Users learn about the flaw and evaluate countermeasures

In Core’s case, communications are handled by a dedicated Advisories Team (6 persons)

- Working part-time on advisories
- De-coupled from discoverers / researchers

When a bug is a security bug?



- The *OpenBSD* story (CVE-2007-1365)
  - Alfredo Ortega found a vulnerability that results in a memory corruption in OpenBSD's kernel
    - In the code that handles IPv6 packets
    - By sending ICMPv6 fragmented packets, an attacker can overflow mbuf structures (in kernel memory) that could allow remote arbitrary code execution.
  - OpenBSD team did not consider it a security problem
  - OpenBSD team quickly developed a fix
    - Fix commited without warning
    - Labelled as a "reliability fix"
  - Discussions with Theo de Raadt
    - Theo: "Pablumfication" of the term "security vulnerability"

The *OpenBSD* story (cont.)

- One week later... Core developed a PoC that demonstrated remote code execution in kernel, by exploiting the mbuf overflow.
- OpenBSD had to change the homepage:

Only **two remote holes** in the default install, in a heck of a long time!

- Conclusion: be conservative
  - Exploitable = there **exists one** way to exploit the bug
  - Not exploitable = **all** the exploitation techniques will fail

How much technical information?

- Debate that has been going on for the last 10 years.
- Publish enough technical details to facilitate accurate and precise assessment of risk.
- Research and publish potential workarounds and alternative mitigation strategies.
  - Patching is not the only possible way to address software security bugs
  - The official vendor is not the only possible solution provider.
- A fully working exploit is not necessary
  - A simple PoC is enough to reproduce the exploitable condition

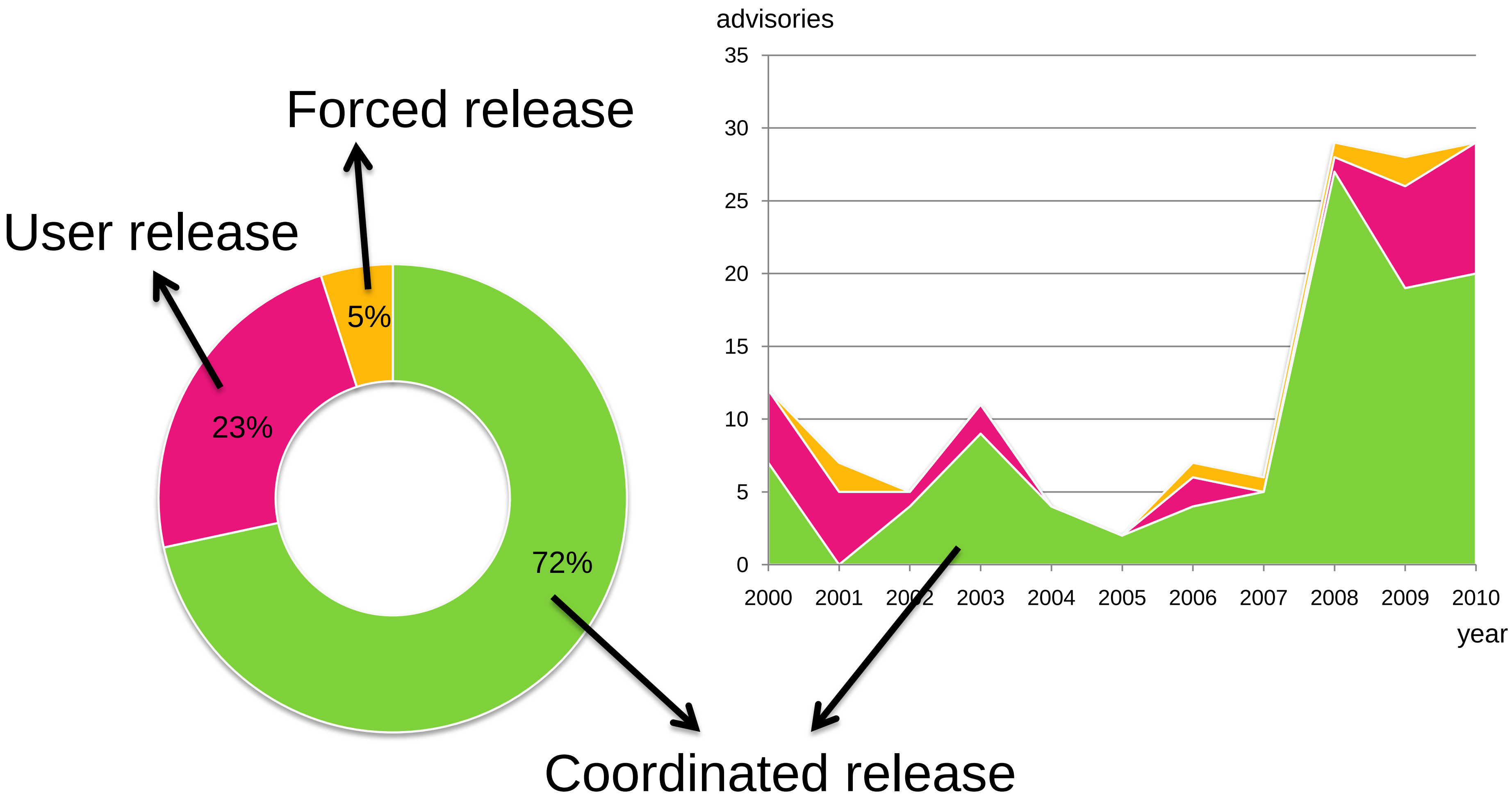
What bug are we talking about?

- Advisories should have enough technical details to uniquely identify the bug
- The *Windows Creation vulnerability* story (CVE-2010-1897):
  - June 2010, typical Patch Tuesday… the exploit writer Nicolas Economou investigates MS10-032 to reproduce the vulnerability
    - Problem: the patch doesn't patch!
    - Several mails with MSRC later, we come to the conclusion that we are speaking about a different bug
    - The bug is in a different function than the original issue and occurs due to a different, previously unknown, issue with the window handle

The Release modes

- Coordinated release
  - Advisory and fixes are released simultaneously
  - We try to publish all advisories in a coordinated way
- User release
  - When the vendor doesn’t respond
  - Or the vendor won’t fix the bug
  - Or researcher and vendor don’t agree on the timeframe
- Forced release
  - When a third party releases info about the bug
  - Or one of the stakeholders leaks info about the bug
  - Or the bug is exploited *in the wild*

Proportion of release modes



Forced release

User release

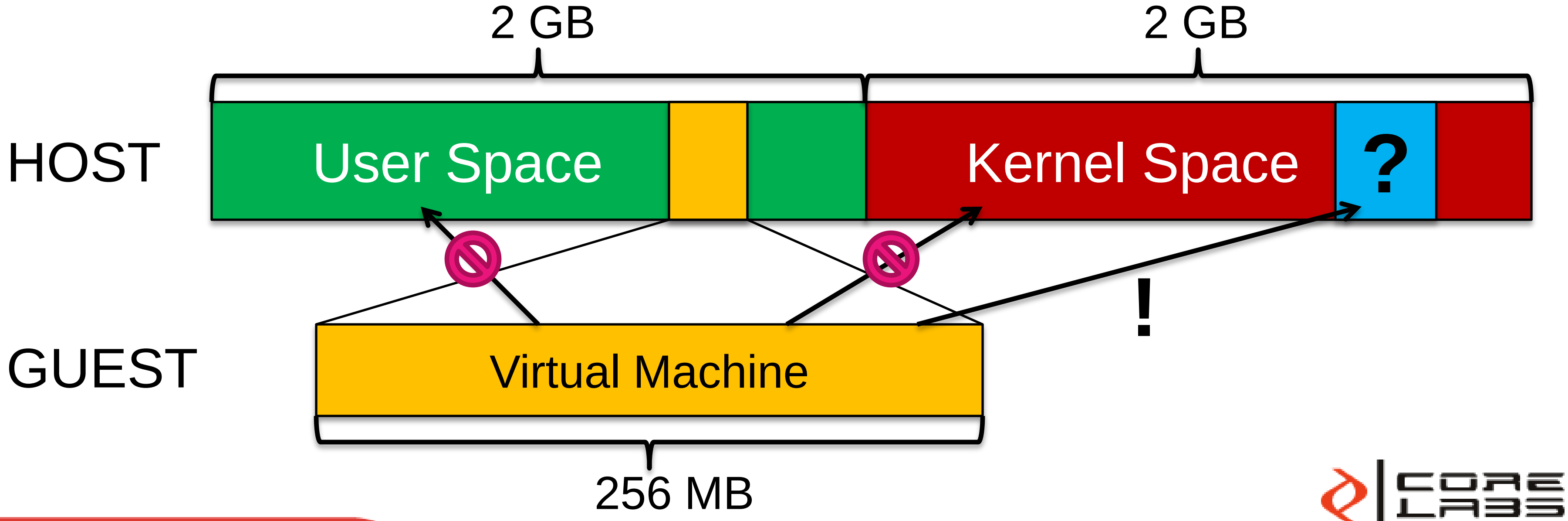
Coordinated release

Some “user release” cases

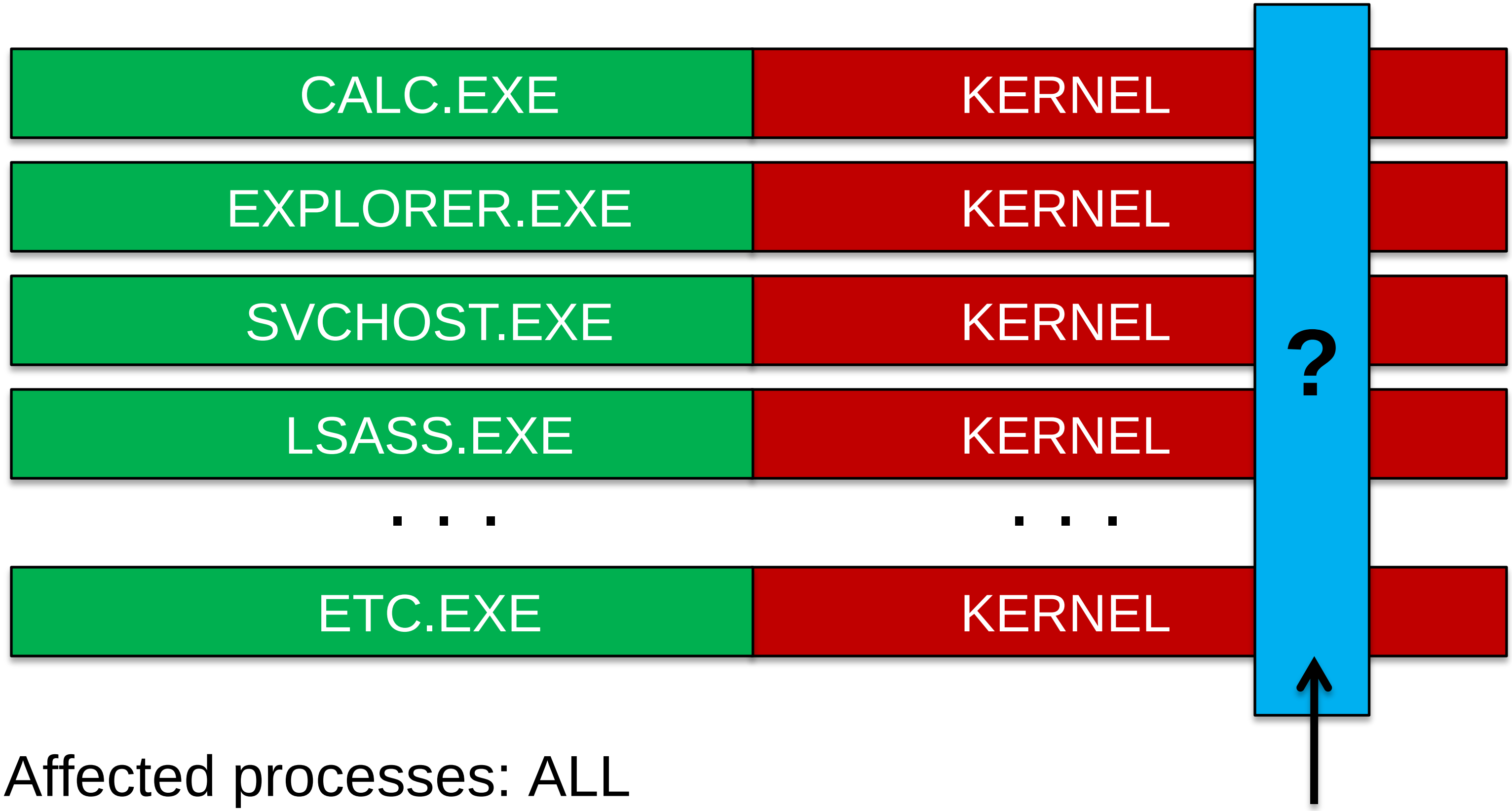
- When the vendor is unresponsive
  - Autodesk 3D Studio, Corel Paint Shop, AOL ICQ, etc...
- The *“Movie Maker and Producer”* story (CVE-2010-0265)
  - Damian Frizza found a bug in the function IsValidWMToolsStream() of Movie Maker that leads to remote code execution.
  - Also present in Producer (add-on for Office)
  - After 6 months and 18 interactions...
    - Patches were ready for Movie Maker
    - MS wanted to match the release of fixes for Producer with the release of new product version (Office 2010)
    - And postpone release of patches and advisory publication to an undetermined date
    - Core respectfully disagrees → “user release”

The *VirtualPC Hyper-hole* story (1/3)

- The *VirtualPC Hyper-hole* story
  - In Virtual PC, the Virtual Machine Monitor (VMM) is responsible for mediating access to hardware resources
  - The bug found by Nico Economou: VMM allows the Guest OS to read/write few memory areas above 2GB limit
  - The Guest OS kernel DOESN'T know this memory area



The *VirtualPC Hyper-hole* story (2/3)



The Hyper-hole

- Affected processes: ALL
- Affected guests: ALL
- Vulnerable Versions: - Virtual PC 2004, 2007, Virtual Server
  - XP Mode in Windows 7

The *VirtualPC Hyper-hole* story (3/3)

- Lots of interactions with MSRC (40 mails in total!) to discuss if this is a security issue. Conclusion:
  - It allows an attacker to bypass DEP and SafeSEH.
    - MSRC: These are defense-in-depth mechanisms
  - In specific conditions it causes vulnerabilities that were deemed not exploitable to become exploitable.
    - Example: gera’s *abo2* is indeed exploitable when running in Windows XP Mode on Windows 7
  - Design problem, very difficult to fix.
  - MS will not issue a security bulletin.
  - Advisory published as “user release” on March 16, 2010.
- STILL UNPATCHED!

Improving our process

Open XML advisory format

- Format used internally by Core Advisories team, developed by Fernando Miranda
- We are releasing it for the community at http://corelabs.coresecurity.com/index.php?module=Wiki&action=view&type=tool&name=Open_XML_Advisory_Format
- Easily convertible to text, HTML, wiki format, ...
- Files included:
  - advisory-schema-OXAF-v22.xsd
  - advisory-template-OXAF-v22.xml
  - common-OXAF-v22.xsl
  - xml2html-OXAF-v22.xsl
  - xml2txt-OXAF-v22.xsl
  - xml2wiki-OXAF-v22.xsl

Some XML fields

- <title>Virtual PC Hypervisor Memory Protection Vulnerability</title>
- <author fullname="Nicolás Economou" nick="nico"/>
- <created year="2009" month="08" day="19"/>
- <advisory id="CORE-2009-0803" local="Yes" remote="No">
- <discovered-during>writing-exploit</discovered-during>
- <metodology>binary-code-audit</metodology>
- <release-mode>user-release</release-mode>
- <published-date year="2010" month="03" day="16"/>

Use industry standards

- CVE = Common Vulnerabilities and Exposures

  -
    ```
    <track-ids>
        <id from="cve">2010-1002</id>
        <id from="bugtraq">38764</id>
    </track-ids>
    ```
- CWE = Common Weakness Enumeration

  -
    ```
    <vulnerability-class><cwe id="285">Improper
    Access Control</cwe></vulnerability-class>
    ```
- To be added: CPE = Common Platform Enumeration
- More at “Making security measurable”: http://measurablesecurity.mitre.org/

Simple references (LaTex style)

- In the text:

```
As an example, the abo2 exercise from gera's Insecure
Programming page <xref target='abos'/> is shown below.
```

- In the references section:

```
<reference label='abos'>
gera's Insecure Programming by Example<break-line/>
<eref target='http://community.corest.com/~gera/
InsecureProgramming/'/>
</reference>
```

- The references are numbered and cross-linked automatically when rendering the output as text or HTML
- Encourage the writer to add references!
  - Write the advisory as a technical report

Detailed timelines – motivation

- In the last 10 years we have seen a lot of debate around disclosure policies
  - Full disclosure, responsible disclosure, limited disclosure, no disclosure
- One size doesn’t fit all
  - Correct procedure determined on a case by case basis
- We need to understand better the disclosure process
  - Enforce process transparency.
  - Document and publish communications between stakeholders.

The timeline in the XML

```
<timeline>
<event year="2009" month="08" day="19" what="team-
interaction">
   <core/> notifies the Microsoft team of the
   vulnerability and sends a brief technical report.
</event>
<event year="2009" month="08" day="19" what="vendor-
interaction">
   The Microsoft team acknowledges the vulnerability
   report.
</event>
...
<event year="2010" month="03" day="16" what="advisory-
published">
   Advisory <advisory-id/> is published.
</event>
</timeline>
```

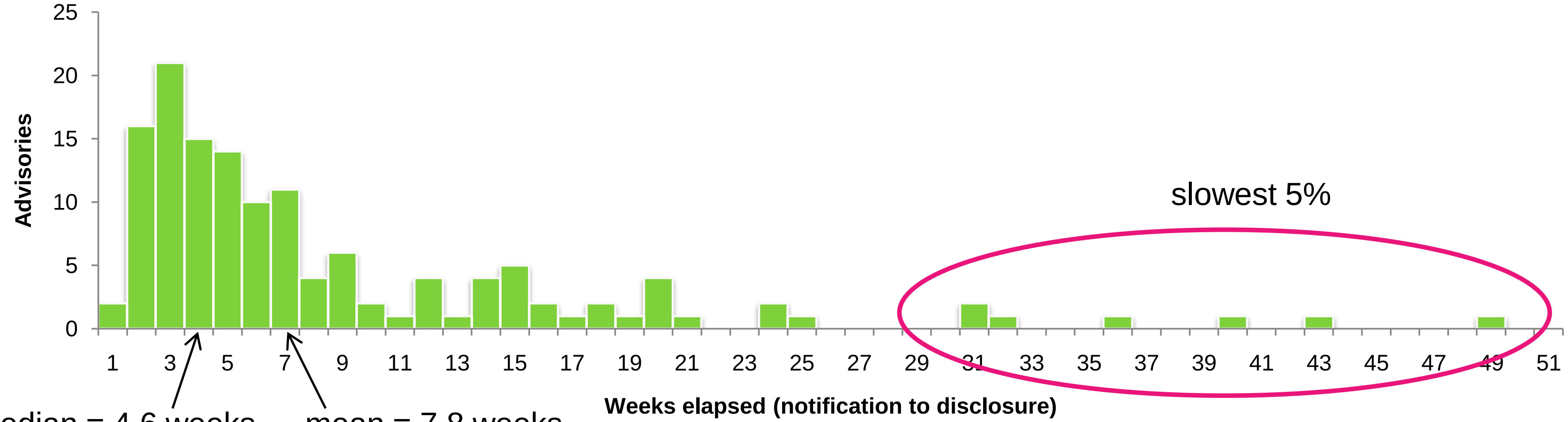
More event types in the timeline

- advisory-started
- advisory-finished
- advisory-published
- advisory-cancelled
- conference-call
- exploit-in-the-wild
- id-requested
- id-assigned
- team-interaction
- team-research-started
- team-research-finished
- vendor-interaction
- vendor-research-started
- vendor-research-finished
- patch-available
- wont-patch

Values extracted from the Advisory timeline

- Elapsed time (from notification to publication)
- Release mode
- Number of interactions = mails and phone calls exchanged with the vendor (and other stakeholders)
- Number of times the publication date was rescheduled
- From the vendor side
  - time to reproduce the vulnerability
  - time to assess exploitability
  - time to develop fixes
  - time to test fixes

Our sample – duration of the process



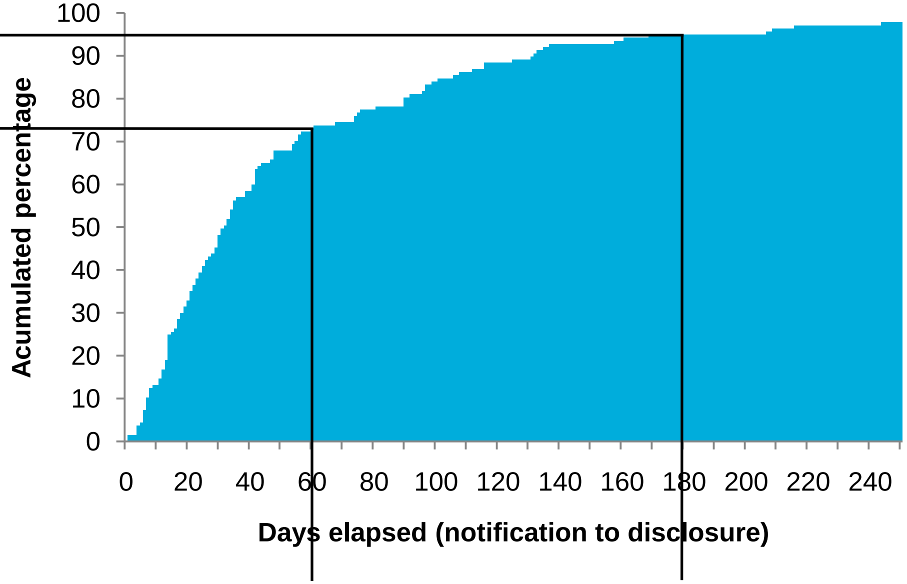
slowest 5%

median = 4,6 weeks

mean = 7,8 weeks

95 %

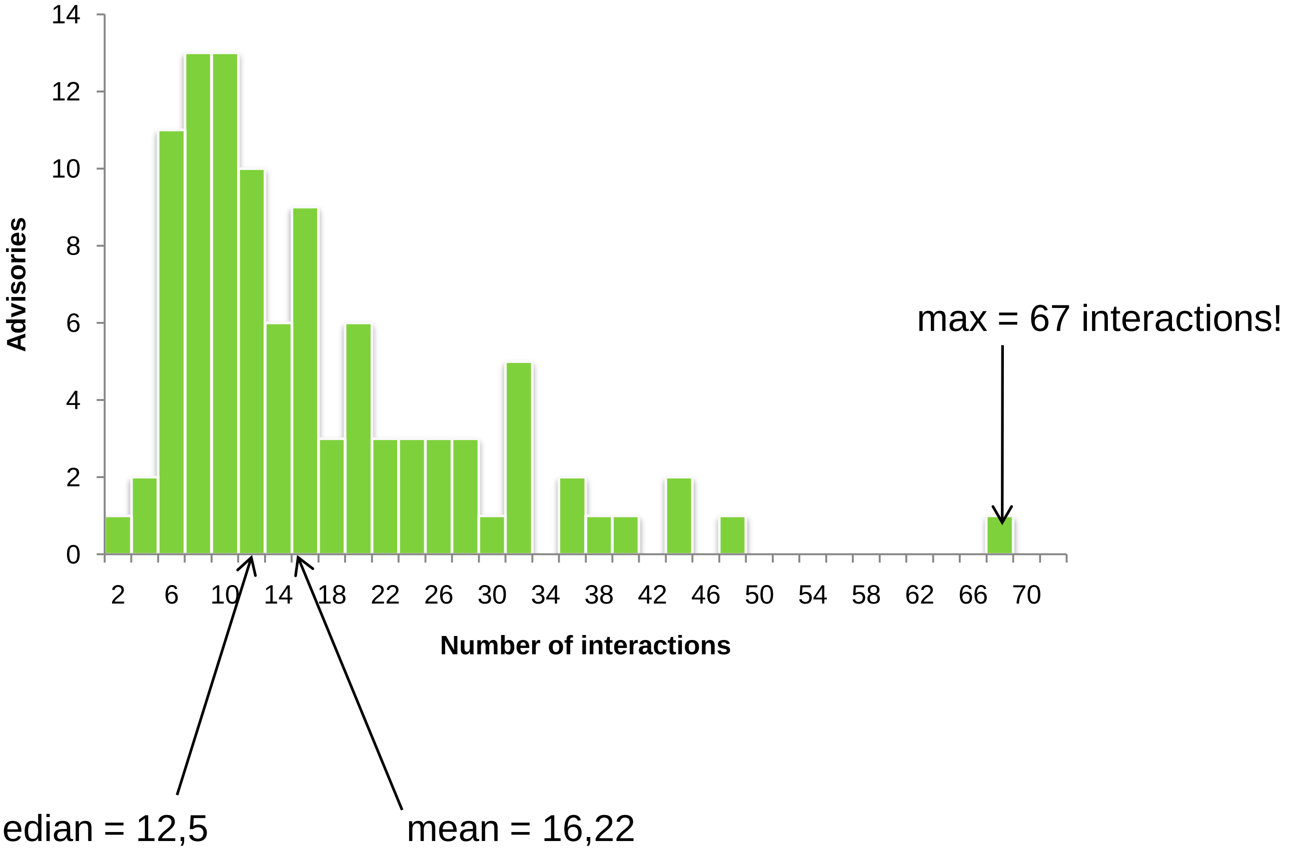
73 %



2 months

6 months

Our sample - number of interactions



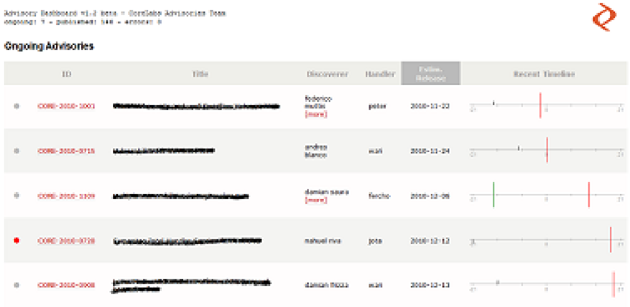
max = 67 interactions!

median = 12,5

mean = 16,22

Benefits of having a standard format

- Easier parsing of advisories information
- Easier tracking of ongoing advisories
  - Advisories dashboard (trac plugin)



Advisory Dashboard v1.2 beta - CoreLabs Advisories Team
ongoing: 7 - published: [illegible] - aborted: 0

Ongoing Advisories

	ID	Title	Discoverer	Handler	[illegible]	Recent Timeline
●	CORE-2010-1001	[redacted]	federico muttis [more]	peter	2010-11-22	
●	CORE-2010-0715	[redacted]	andres blanco	wari	2010-11-24	
●	CORE-2010-1109	[redacted]	damian saura [more]	fercho	2010-12-06	
●	CORE-2010-0728	[redacted]	nahuel riva	jota	2010-12-12	
●	CORE-2010-0908	[redacted]	damian frizza	wari	2010-12-13	

- Automate publication workflow
- Encourage researchers to share information in a consistent way
- Facilitate the scientific study of the lifecycle of the bugs

Summary

- Coordinated release is desirable
  - Not always possible (forced and user release)
- Include precise technical information of the bug
- Document the disclosure process
  - Detailed and structured timelines
  - Statistical study of the process
  - Put the discussion around disclosure policies on technical ground
- Use the Open XML advisory format!

The Bibliography

References 1/3

- Rain Forest Puppy. (2000). *Full Disclosure Policy (RFPolicy) v2.0.* http://www.wiretrip.net/rfp/policy.html
- @stake. (2000). *@stake Security Advisory Disclosure Policy.*
- Ivan Arce. (2001). *Vulnerability Reporting: Bugs in the bug reporting process*. TISC Insight, Volume 3, Issue 3
- Steve Christey, Chris Wysopal. (2002). *Responsible Vulnerability Disclosure Process.* http://tools.ietf.org/html/draft-christey-wysopal-vuln-disclosure-00
- Stefan Frei, Bernhard Tellenbach, Bernhard Plattner. (2008). *0-Day Patch - Exposing Vendors (In)security Performance.* Black Hat 2008.
- Stefan Frei, Dominik Schatzmann, Bernhard Plattner, Brian Trammel. (2009). *Modelling the Security Ecosystem - The Dynamics of (In)Security*. Workshop on the Economics of Information Security (WEIS), London, June 2009.
- Oulu University Secure Programming Group (OUSPG). *Vulnerability disclosure publications and discussion tracking.* https://www.ee.oulu.fi/research/ouspg/Disclosure_tracking
- Making security measurable http://measurablesecurity.mitre.org/

References 2/3

- Bruce Schneier. (Sept 15, 2000). *Full Disclosure and the Window of Exposure*. http://www.schneier.com/crypto-gram-0009.html#1
- Bruce Schneier. (Nov 15, 2001). *Full Disclosure.* http://www.schneier.com/crypto-gram-0111.html#1
- Michal Zalewski. (April 27, 2010). *Responsibilities in vulnerability disclosure.* http://lcamtuf.blogspot.com/2010/04/responsibilities-of-disclosure.html
- Michal Zalewski. (July 21, 2010). *"Testing takes time".* http://lcamtuf.blogspot.com/2010/07/testing-takes-time.html
- Chris Evans *et al.* (July 20, 2010). *Rebooting Responsible Disclosure: a focus on protecting end users.* http://googleonlinesecurity.blogspot.com/2010/07/rebooting-responsible-disclosure-focus.html
- Matt Thomlinson. (July 22, 2010). *Announcing Coordinated Vulnerability Disclosure.* http://blogs.technet.com/b/msrc/archive/2010/07/22/announcing-coordinated-vulnerability-disclosure.aspx

References 3/3

- CVE Numbering Authorities
  http://cve.mitre.org/cve/cna.html

- Alfredo Ortega. (13 March 2007). OpenBSD's IPv6 mbufs remote kernel buffer overflow (CVE-2007-1365)
  http://www.coresecurity.com/content/open-bsd-advisorie

- Damian Frizza. (9 March 2010). Windows Movie Maker and Microsoft Producer IsValidWMToolsStream() Heap Overflow
  http://www.coresecurity.com/content/movie-maker-heap-overflow

- Nicolas Economou. (16 March 2010). Virtual PC Hypervisor Memory Protection Vulnerability
  http://www.coresecurity.com/content/virtual-pc-2007-hypervisor-memory-protection-bug

- Nicolas Economou. (10 August 2010). Microsoft Windows CreateWindow function callback vulnerability (CVE-2010-1897)
  http://www.coresecurity.com/content/microsoft-windows-createwindow-function-callback-bug

- Nicolas Economou. (16 Sept 2010). 2x1 Microsoft Bugs: 'Virtual PC hyper-hole-visor' + 'Windows Creation Vulnerability (MS10-048)'. In Ekoparty 2010.
  http://corelabs.coresecurity.com/index.php?module=Wiki&action=view&type=publication&name=2x1_Microsoft_Bugs_Virtual_PC_hyper-hole-visor_Windows_Creation_Vulnerability_MS10-048

Thank you!

Carlos Sarraute → carlos@coresecurity.com

http://corelabs.coresecurity.com