Microsoft® Research
Faculty Summit 2010

Debugging in the (Very) Large: Ten Years of Implementation and Experience

Galen C. Hunt
Principal Researcher
Microsoft Corporation

Acknowledgements

Paper originally appeared in *22nd ACM Symposium on Operating Systems Principles*

Co-authors (and system inventors):

- Kirk Glerum
- Kinshuman Kinshumann
- Steve Greenberg,
- Gabriel Aul
- Vince Orgovan
- Greg Nichols
- David Grant
- Gretchen Loihle

A Revelation

Software has bugs

Even **shipping** software

Even **Microsoft's** shipping software

Oh, and so does hardware

(but we'll come back to that point later...)

Two Definitions

Bug: a **flaw** in program logic

```
#define MYVAR *(int*)random()
...
MYVAR = 5;
```

Error: a **failure in execution** caused by a bug

- run it 5,000 times, you'll get ~ 5000 errors

One bug may cause <u>many</u> errors

The Challenge

Microsoft ships software to **1 billion** users,

- how do we find out when things go wrong?

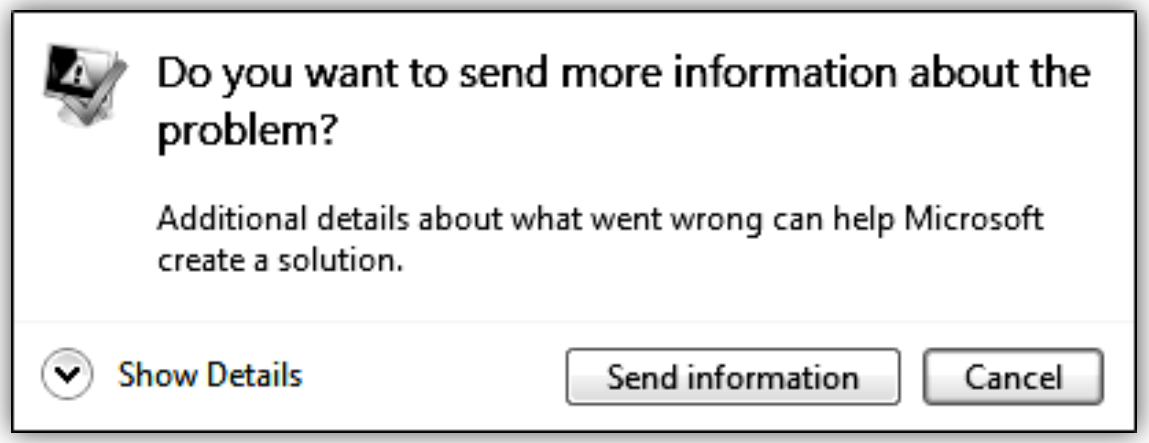
We want to

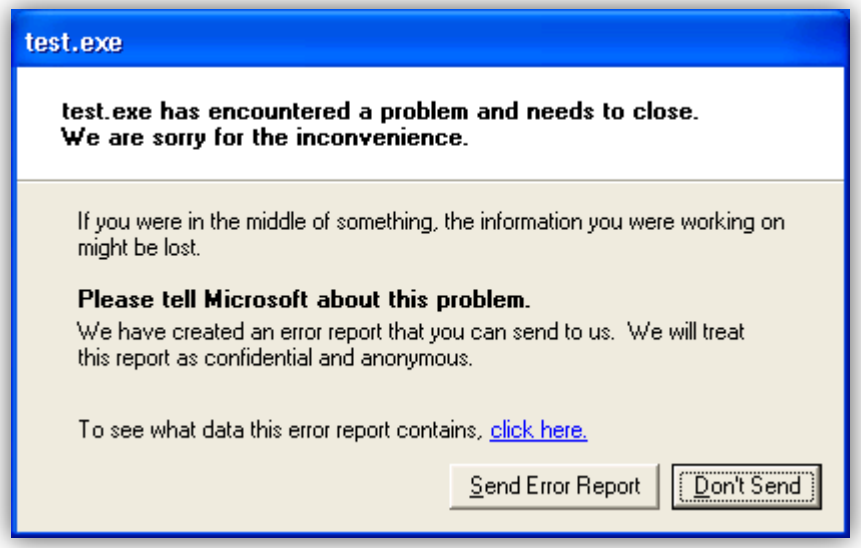
- fix bugs regardless of source
  - application or OS
  - software, hardware, or malware
- prioritize bugs that affect the most users
- generalize the solution to be used by any programmer

Reported Bugs

Error	Reporting Trigger
Kernel Crashes	Crash dump found in page file on boot
Application Crashes	Unhandled process exception
Application Hangs	Failure to process user input for 5 seconds
Service Hangs	Service thread times out
Installation Failures	OS or application installation fails
App. Compat. Issues	Program calls deprecated API
Custom Errors	Program calls WER APIs to report error
UI Delays	Timer assert takes longer than expected
Invariant Violations	Ship assert in code fails

Windows Error Reporting (WER)





WER by the Numbers

Billions	Error reports collected per year (App,OS,HW)
1 billion	Clients
100 million	Reports /day processing capacity
17 million	Programs with error reports in WER
Many 1000s	Bugs fixed
Over 700	Companies using WER
200	TB of Storage
60	Servers
10	Years of use
2	Servers to record every error received
1	# of programmers needed to access WER data

Outline

Introduction

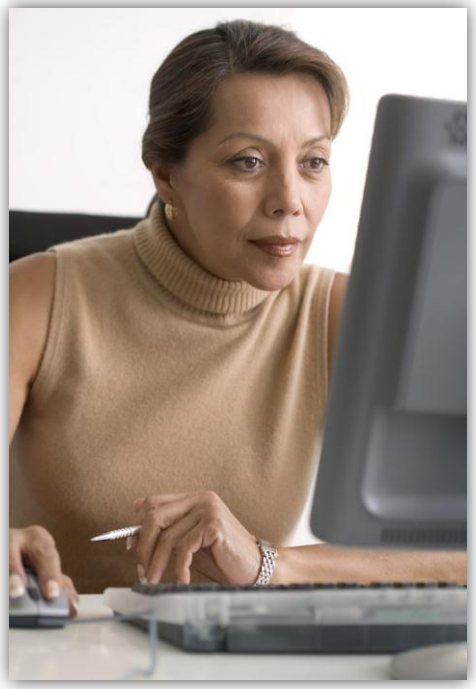
How do we process billions of error reports?

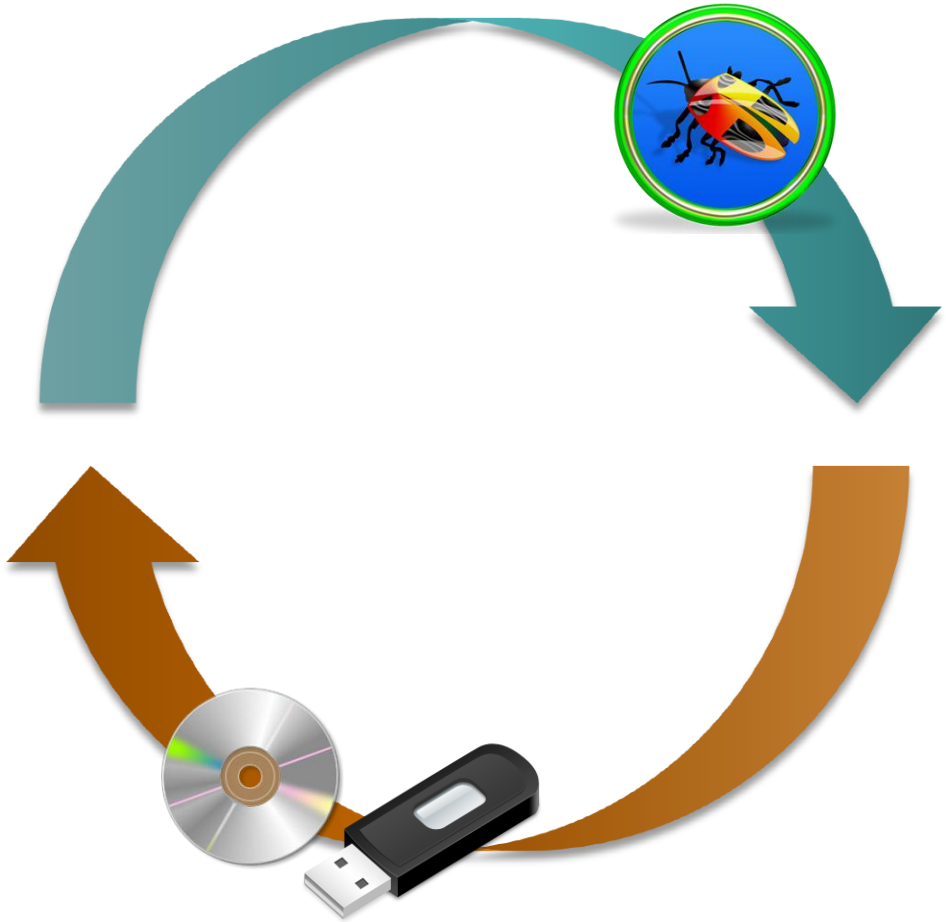
Experiences fixing bugs from

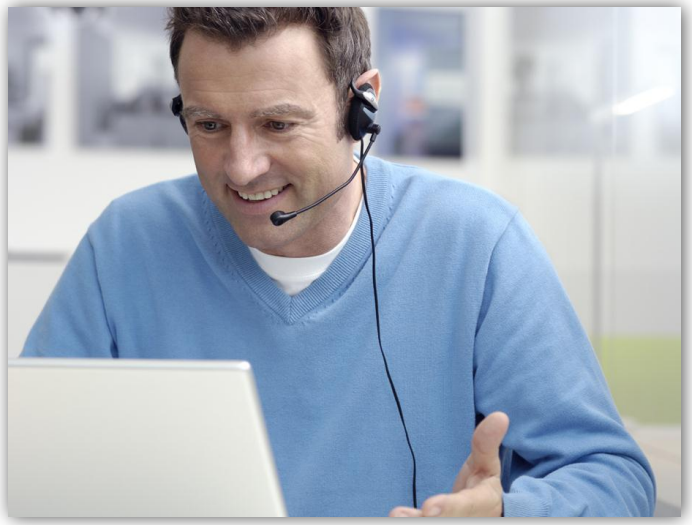
- software
- hardware
- malware

Conclusion

Debugging in the Small…

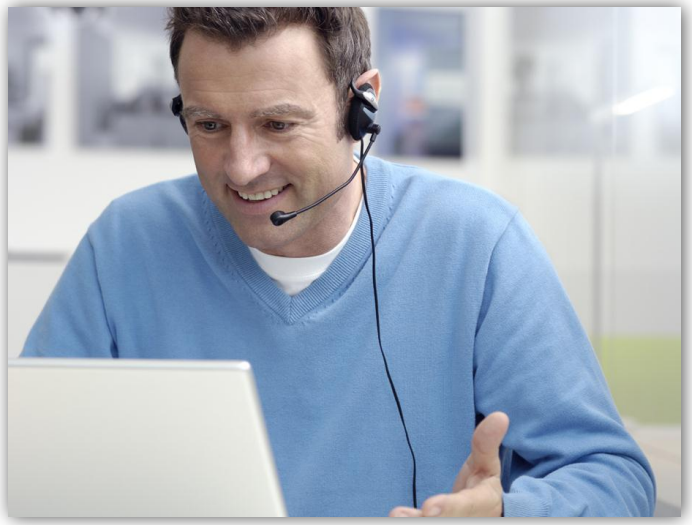






In the Large without WER…

Support **technician** tries to diagnose error



User calls technical support

Technicians report "top ten" issues to **programmers**

The Human Bottleneck

Can’t hire enough technicians

Data is inaccurate

Hard to get additional data

No “global” baseline

Useless for heisenbugs

Need to remove humans

Goal: Fix the Data Collection Problem

Allow one service to record

- **every error** (application, OS, and hardware)
- on every **Windows system**
- **Worldwide**

Corollary:

- **that which we can measure, we can fix…**

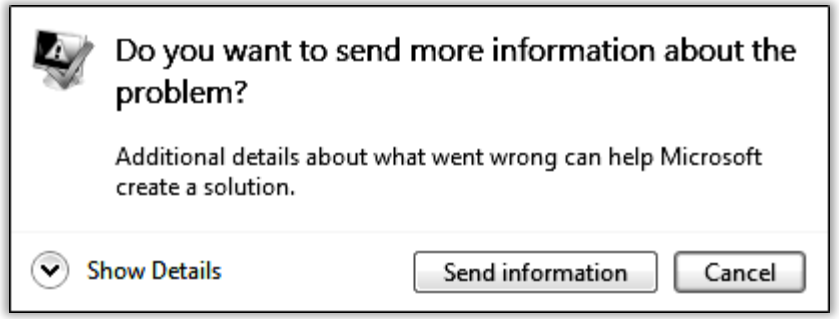
An Outlook Plug-in Example

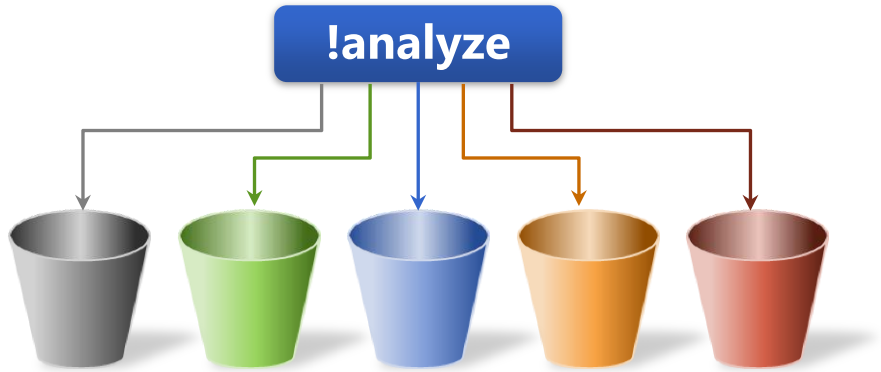
plugin.dll:

```
#define MYVAR *(int*)random()
...
void foo(int i, int j)
{
    if (i & 1)
        memcpy(&MYVAR, j, 4);
    else
        …
}
```

Debugging in the Large with WER…

Minidump





!analyze

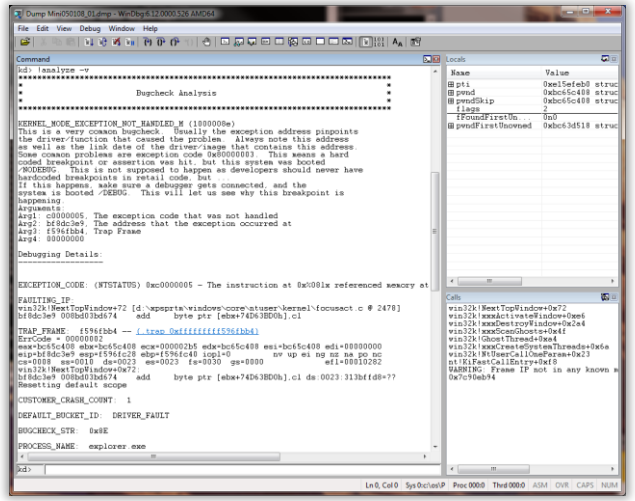
Engine for WER bucketing heuristics

Extension to the *Debugging Tools for Windows*

- input is a minidump, output is bucket ID
- runs on WER servers (and programmers desktops)
- http://www.microsoft.com/whdc/devtools

500 heuristics

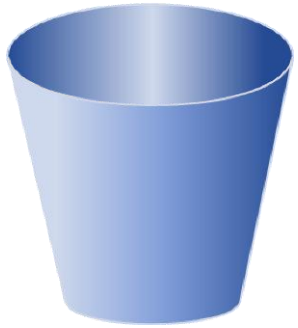
- grows ~ 1 heuristic/week



To Recap and Elaborate…

What I told you:

- client automatically collects a minidump
- sends minidump to servers
- !analyze **buckets** the error with similar reports
- increments the bucket count
- programmers **prioritize** buckets with highest count



Actually…

- only upload first few hits on a bucket, others just inc.
- programmers request additional data as needed

2-Phase Bucketing Strategy

Labeling (on client): bucketed by failure point

```
outlook.exe,plugin.dll,v1.0.2305,0x23f5
{program name},{binary},{version},{pc offset}
```

Classifying (on servers):
re-bucketed toward root cause by **!analyze**

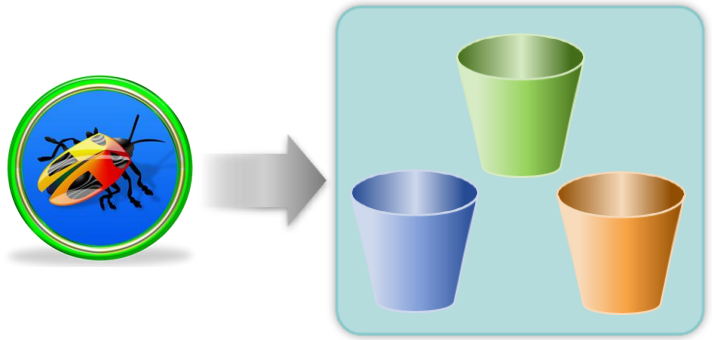
- consolidate version and replace offset with symbol
  `outlook.exe,plugin.dll,`**`memcpy`**
- find caller of **memcpy** (because it isn't buggy)
  `outlook.exe,plugin.dll,`**`foo`**
- etc.

SOSP paper contains much more detail on bucketing…

Bucketing Mostly Works

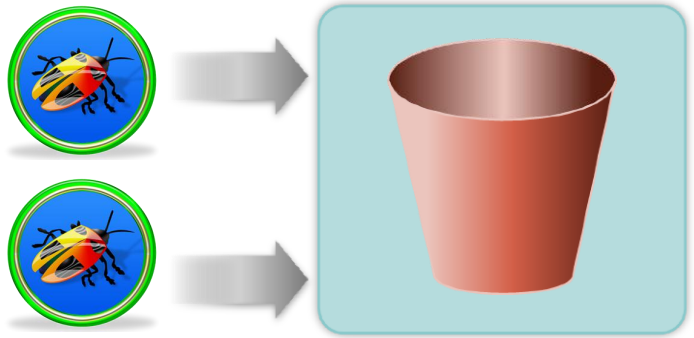
One bug can hit multiple buckets

- up to 40% of error reports
  `memcpy(&MYVAR, j, 4);`
  - one bucket when &MYVAR is illegal address
  - many others when &MYVAR is in a data section
- extra server load
- duplicate buckets must be hand triaged



Multiple bugs can hit one bucket

- up to **4%** of error reports
- harder to isolate each bug



Solution: scale is our friend

Outline

Introduction

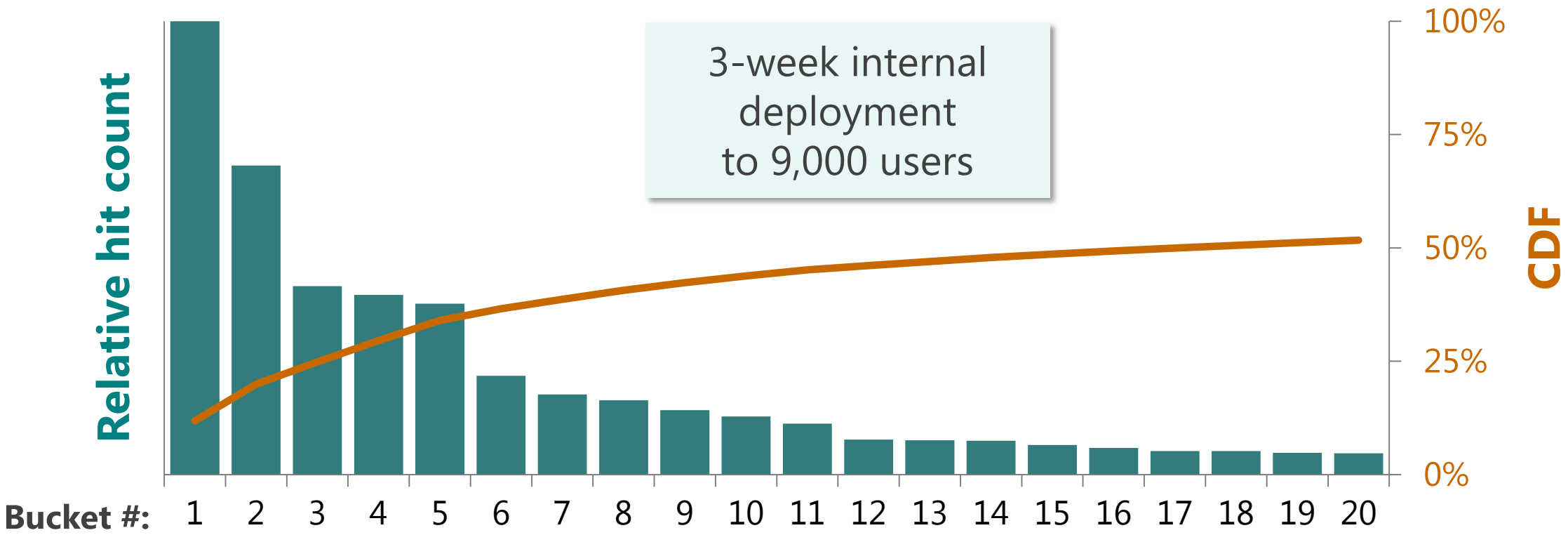
How do we process billions of error reports?

Experiences fixing bugs from

- software
- hardware
- malware

Conclusion

Top 20 Buckets for *MS Word 2010*



Just 20 buckets account for **50% of all errors**

Fixing a small # of bugs will help **many users**

Fixing bugs in software

First use found >=**5-year old** heisenbugs in Windows

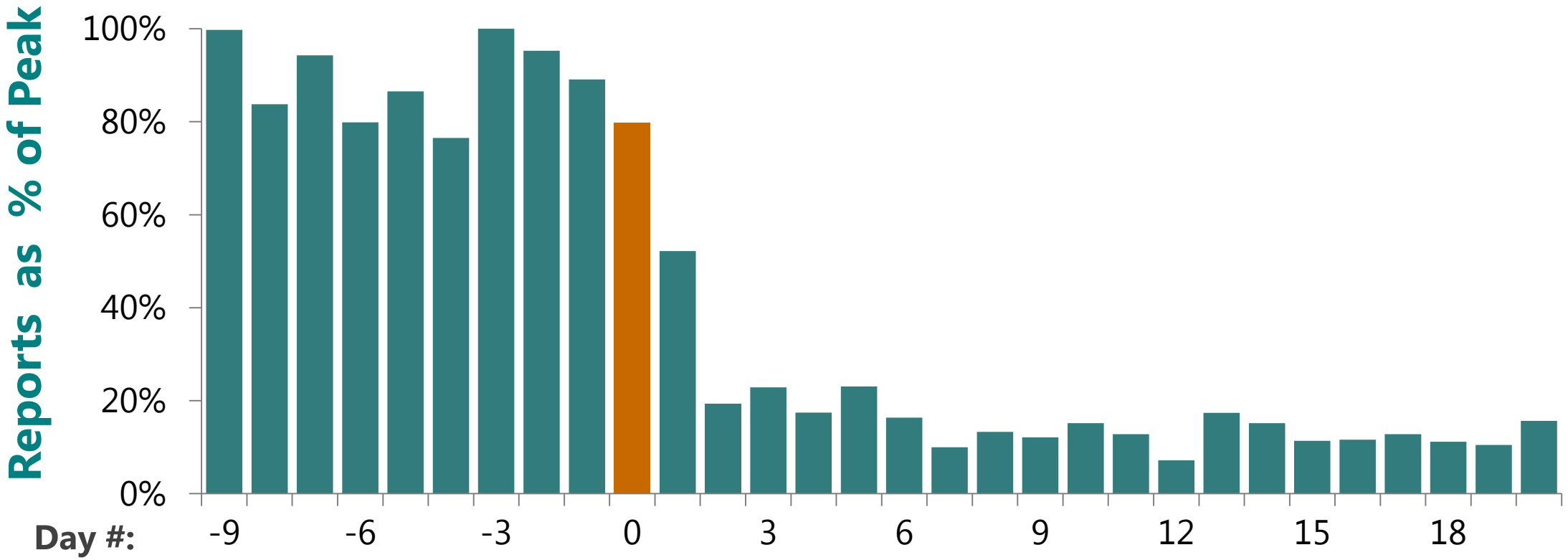
Windows Vista team **fixed 5,000 bugs** in beta

Anti-Virus vendor fixed top 20 buckets and
dropped from 7.6% to 3.6% of all kernel crashes

Office 2010 team fixed 22% of reports in **3 weeks**

And you can fix yours…

Hardware: Processor Bug



WER helped fix hardware error

Manufacturer **could** have caught this earlier w/ WER

Other Hardware Bugs

SMBIOS

- memory overrun in resume-from-sleep

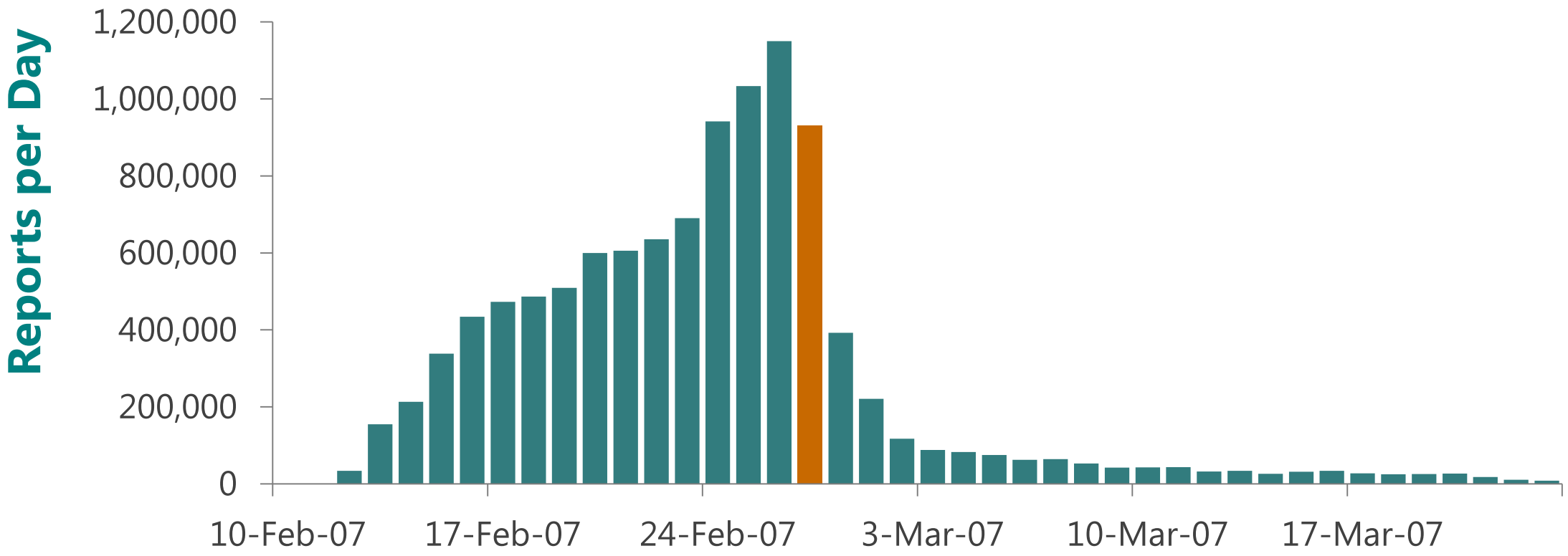
Motherboard USB controller

- only implemented 31 of 32 DMA address bits

Lots of information about failures due to

- overclocking
- hard disk controller resets
- substandard memory

Renos Malware



Early detection w/o user action (renos, blaster, slammer, etc.)

WER **scales** to handle global events

Other Things in the SOSP Paper

Bucketing details (Sec. 3)

Statistics-based debugging (Sec. 4)

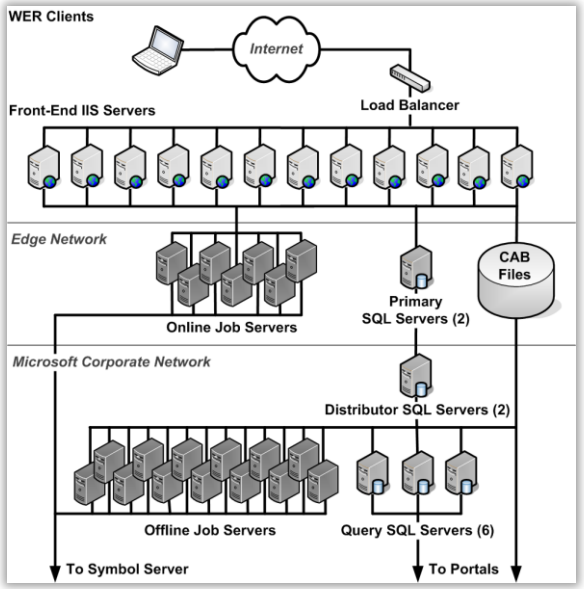
Progressive data collection (Secs. 2.2 & 5.4)

Service implementation (Sec. 5)

WER experiences (Sec. 6)

OS Changes (Sec. 7)

Related work (Sec. 8)



Conclusion

Windows Error Reporting (WER)

- the **first** post-mortem reporting system with automatic diagnosis
- the **largest** client-server system in the world (by installs)
- helped 700 companies fix **1000s** of bugs and **billions** of errors
- **fundamentally changed SW development at MS**

"WER forced us to stop making [things] up."

WER works because bucketing *mostly* works

http://winqual.microsoft.com

Microsoft® Research

Faculty Summit 2010