Effective Patch Management:

How to make the pain go away

Adam Shostack
adam@InformedSecurity.com

Overview

- *Why patch?*
- *Why is patching so painful?*
- *What can make it easier?*
- *Thinking about risk management*
- *How can we get out of the rat race?*

Why Patch?

- *Vendors issue patches to correct bugs*
  - *Performance/Reliability*
  - *Security is a subset of reliability*
- *End users apply patches to fix problems*
  - *Preventative/Reaction models*

Security Patches

- *Vendors release code*
  - *All code has bugs*
- *People find bugs*
  - *Sometimes they tell the vendor*
- *Vendor triages, and may release a fix*
- *Some want to install it to forestall problems*

Where we are

- *Why patch?*
- *Why is patching so painful?*
- *What makes it easier?*
- *Thinking about risk management*
- *How can we get out of the rat race?*

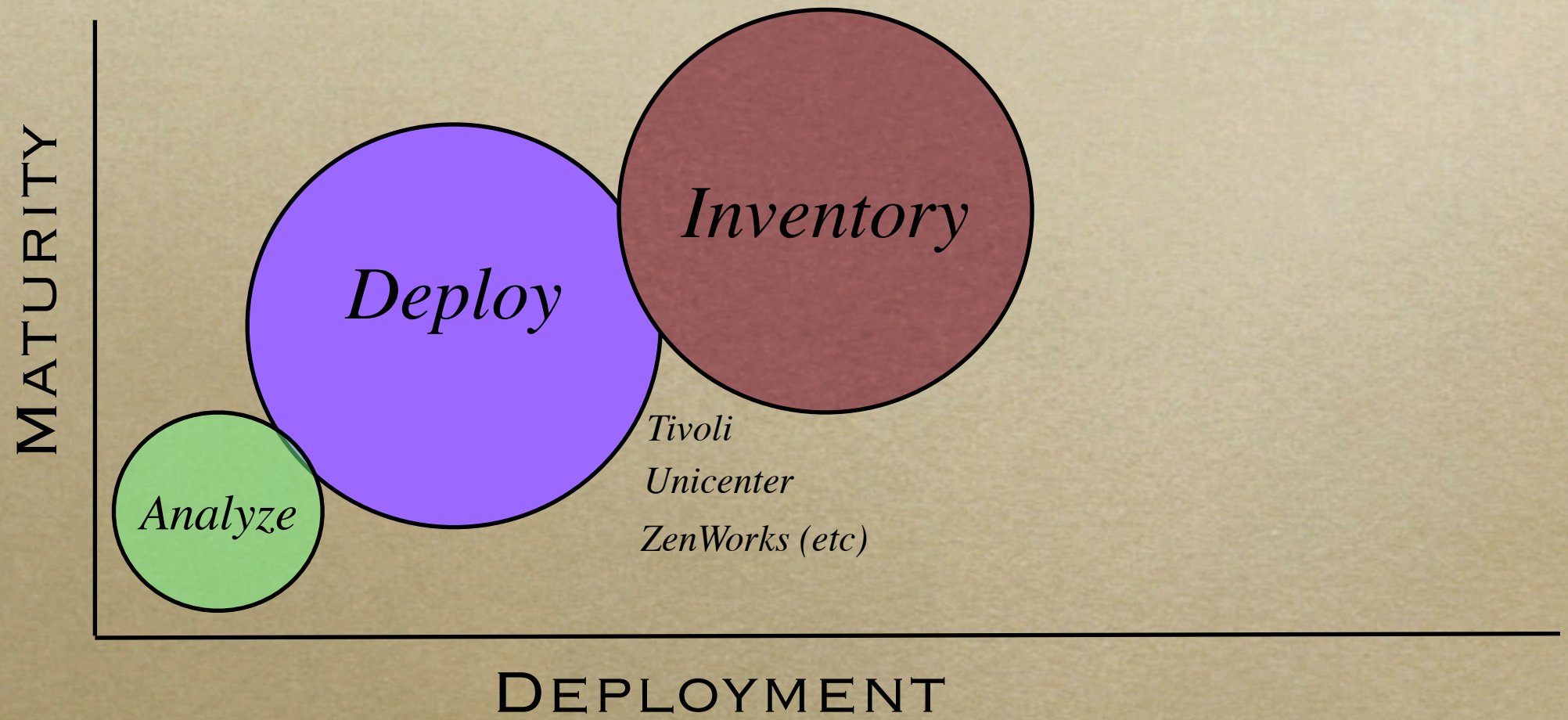
Why is patching painful?

- *Patch notification*
- *Inventory & Roll-out*
- *Mobile*
- *Low bandwidth*

Lies and Excuses!

- *The problems of notification, inventory and roll-out, and mobile and low-bandwidth systems are roughly solved.*

State of Software Tools



The Real Problems

- *Patches are beta software*
- *Intense pressure to roll out beta software*
- *Poor data about patches*
- *Conflict between IT & IT Security*
- *Patches which can't roll back*

Uptime vs. Security

- *IT is rated on measured uptime*
  - *Every admin knows patching can break things, require reboots*
- *Security is rated on break-ins*
  - *Need to deploy patches to prevent*
- *Huge fights come from different priorities.*

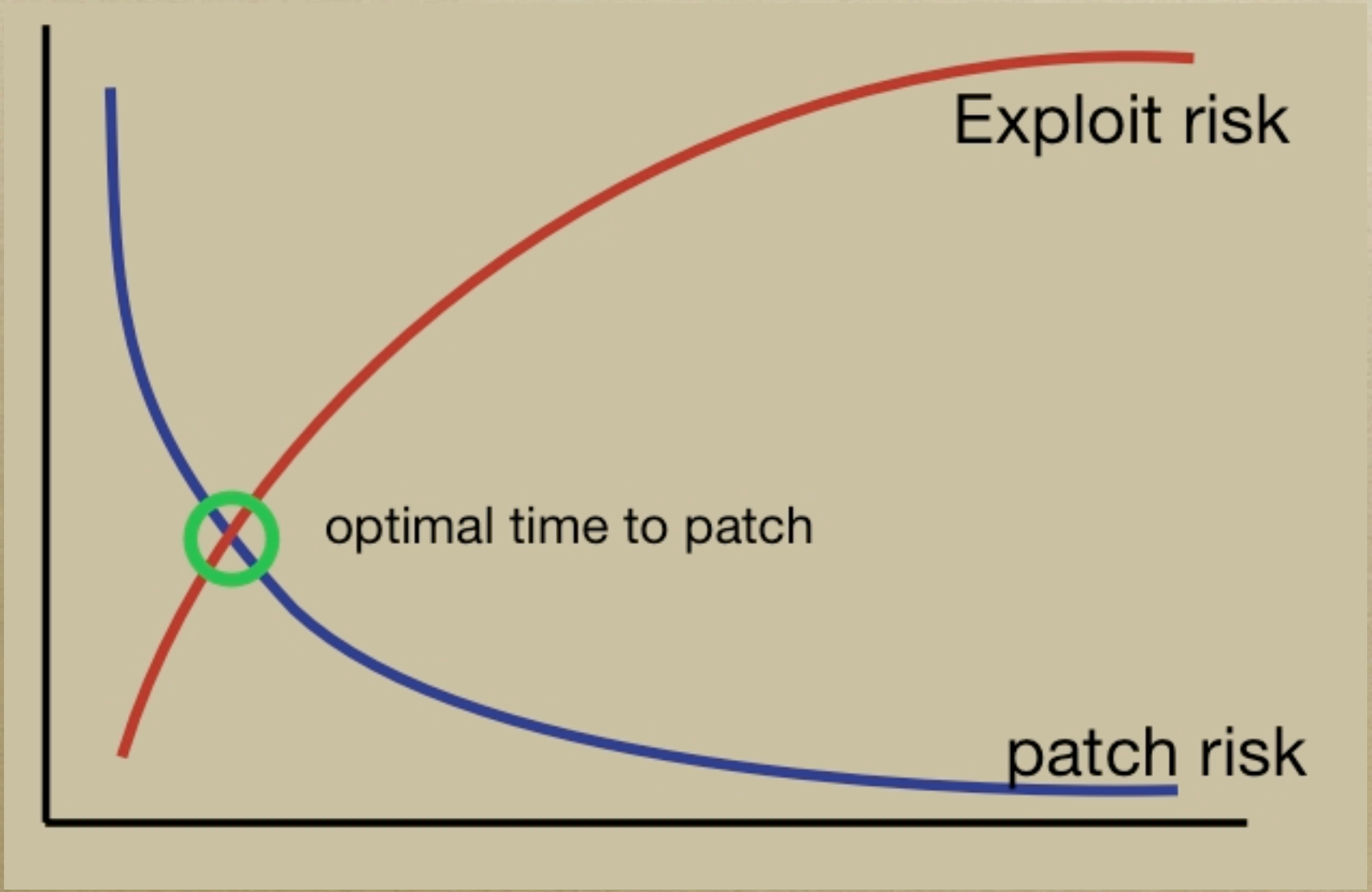
Where We Are

- *Why patch?*
- *Why is patching so painful?*
- *What makes it easier?*
- *Thinking about risk management*
- *How can we get out of the rat race?*

Reconciling The Views

- *Patch risk falls with time*
- *Exploit risk grows with time*
- *Can we put numbers on them?*
- *Can we engage in a risk trade-off?*

Timing the Application of Security Patches for Optimal Uptime



Timing the Application...

- Steve Beattie, Seth Arnold, Crispin Cowan, Perry Wagle, Chris Wright, and Adam Shostack.
- Presented at the USENIX 16th Systems Administration Conference (LISA 2002)
- http://www.homeport.org/~adam/time-to-patch-usenix-lisa02.pdf
- (Don’t copy down the URL: Google finds my homepage, that’s bullet #7)

Where We Are

- *Why patch?*
- *Why is patching so painful?*
- *What makes it easier?*
- *Thinking about risk management*
- *How can we get out of the rat race?*

Risk Management

- *You can do this at home!*
- *Easy math leads to useful results*
- *Cost to deploy, cost to fix problems (security or broken patch)*
- *Goal is to move away from argument and worry*
- *Consider Security Risk, Patch Risk, Business Impact*

Security Issues

- *Patch criticality*
  - *Software Vendor*
  - *CERT metrics (ADDED: CVSS)*
  - *CNN*
- *Mitigating controls*
  - *Firewalls*
  - *Configurations*

Patch Issues

- *How big is the patch?*
- *How many issues does it fix?*
- *Can it be backed out?*
- *Does it require a reboot?*
- *Testing (internal, external, web & lists)*

Business Issues

- *What's the business function of the system?*
- *Is there an impending deadline?*
- *What's your MTTR?*
  - *(Mean Time To Repair)*

Making it concrete

- *Know your cost to deploy a patch*
- *Know your cost of downtime*
- *Estimate the risk of attack*

Some sample numbers

- *1,000 node network with manual patching by $100 techies, at 1 hour/node:*
- *$100,000 to deploy a patch*
- *So what do you do if:*
  - *Attack that would cost you $1,000,000*
  - *Attack that would cost you $105,000*
  - *Attack that would cost you $25,000*

The $105,000 question

- *Expected 5% ROI on cash*
  - *Didn't specify time*
- *Alternate activities?*
- *Cost of capital/ROI?*

- *Why patch?*
- *Why is patching so painful?*
- *What makes it easier?*
- *Thinking about risk management*
- *How can we get out of the rat race?*

Better Patch Mgmt SW

- *Research and risk data*
- *Workflow*
- *Testing support*
- *Risk Management support*

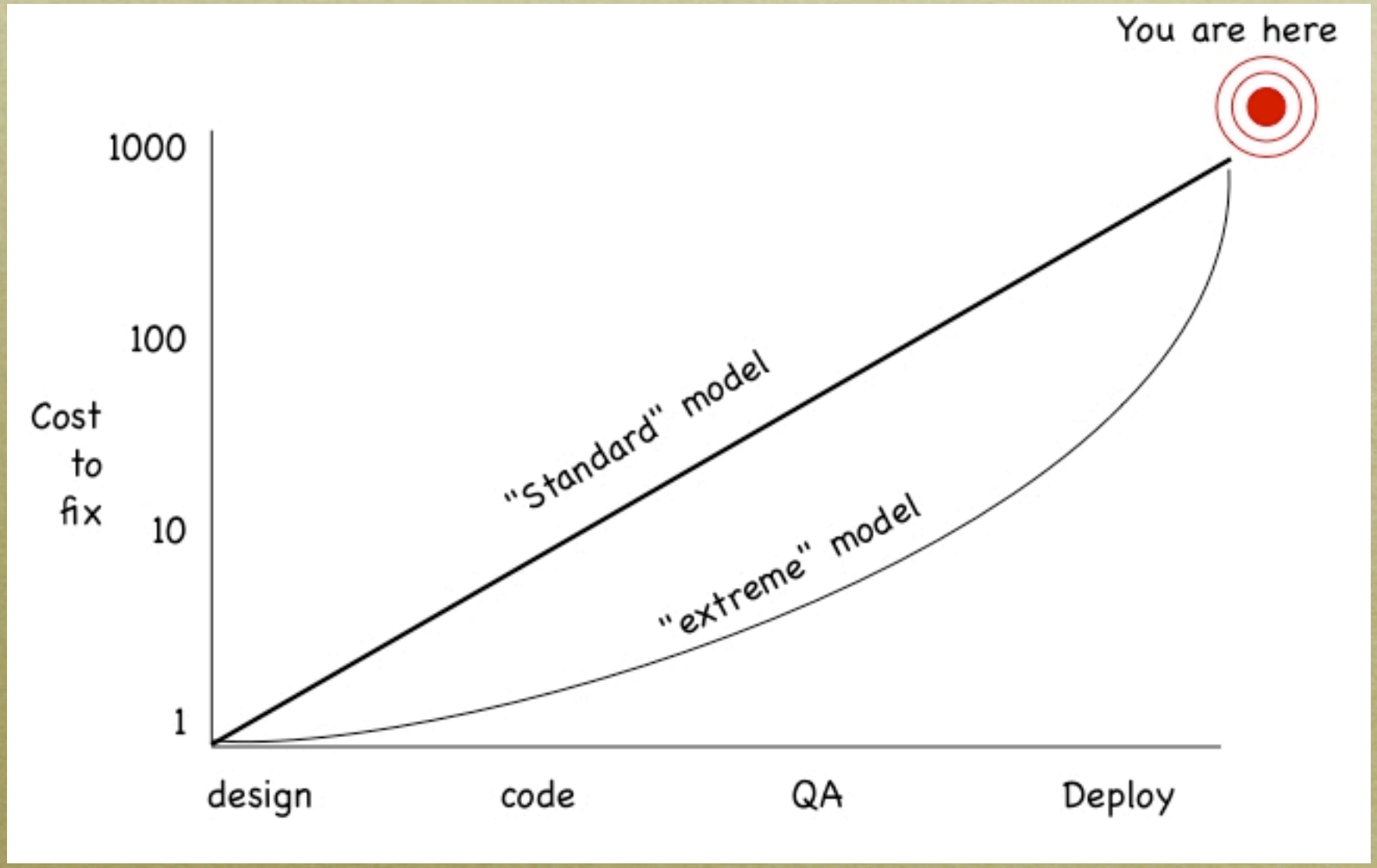
More Managable Deployments

- *Use security software (Okena, Immunix, Sana, etc) to stop classes of attack*
- *Use software to deploy and manage systems*
- *Work to increase MTBF, decrease MTTR*

More Secure Software

- *The core problem is that security is not a buying criteria*
- *Make it one*
- *Push your vendor to discuss and then improve their software processes: Design, Development, Testing, Deploy*

Bug (and software) Development



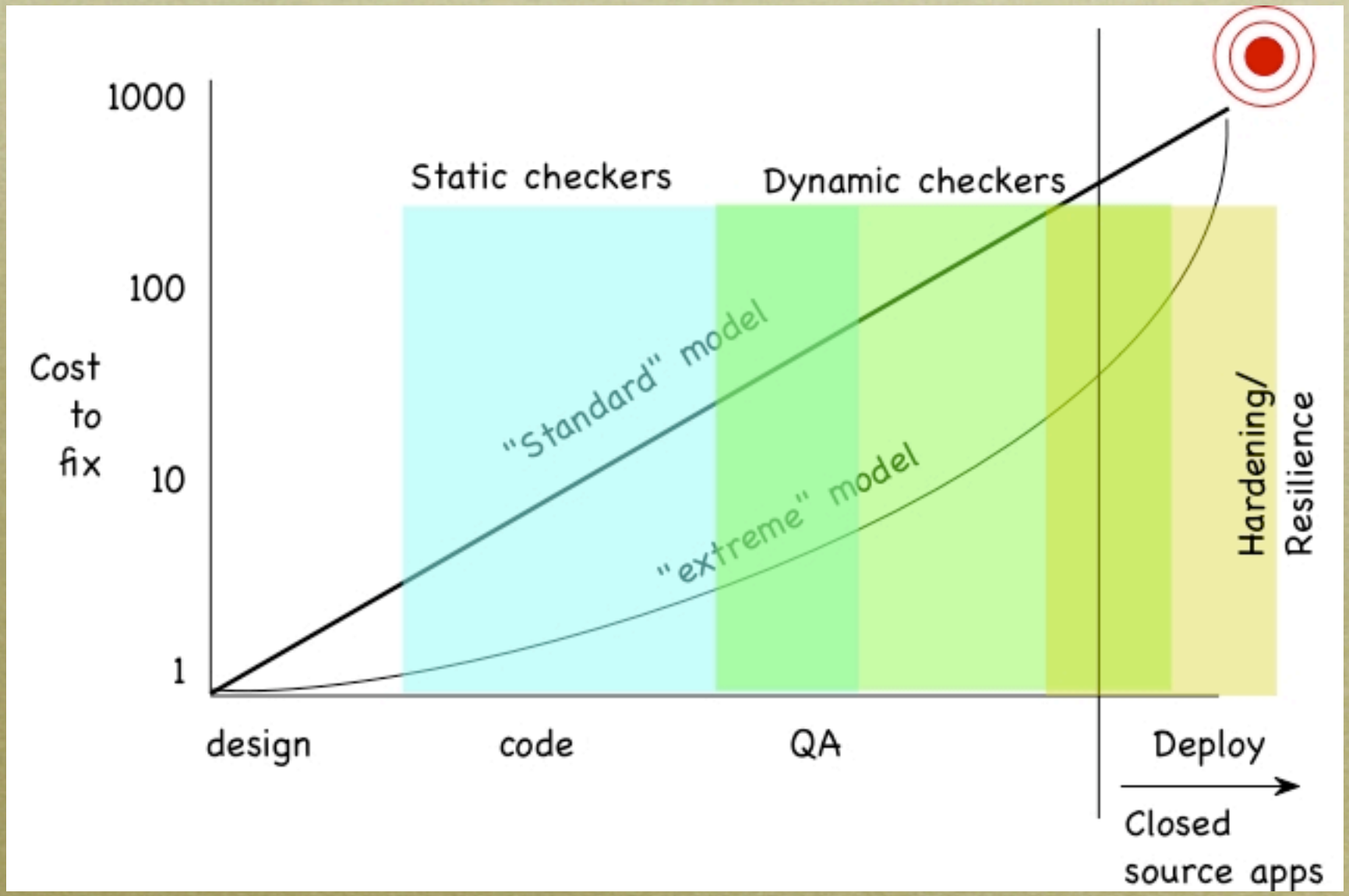
How To Move?

- It’s actually worse than that
- That’s a graph for a single program
- You deploy lots of programs

How To Get There

- Better software tools
  - Internal, external
- Better Deployment tools
  - Security
  - Operations

Where The Tools Fit



Static Checkers

- Work with source code
- Lots of different languages
- Results generally easier to fix
  - They're associated with lines of code
- High false positive rates
- Find "sins of commission" like strcat()
- Fast

Dynamic Checkers

- Work on binary code
  - Never wonder if the optimizer was too clever
- Find “Sins of Omission” like SQL injection
- Slow! (Can be hours or days)

Language Selection

- Some languages seem to be more prone to security flaws
  - C, PHP
- We may not have found the classes of flaws in Java, C#
- New classes keep showing up (integer underflows, etc)

Adding Resilience to Code

- How to
  - deploy
  - operate
- Buggy code *more* securely

Free UNIX techniques

- chroot/jail
- Unprivileged daemon accounts
  - Painful if you need fast code on port 80
- Free security enhanced OSes:
  - OpenBSD, SELinux

More advanced tools

- OS hardening tools
  - Immunix subdomain
  - Sana kernel enhancements
- Application hardening
  - Stackguard & company
  - (Recompile vs kernel modules)

Issues with Hardening Tools

- How to measure their effectiveness
- Configuration effort
- Costs (percieved and real)
  - Cash up front
  - Speed
  - Supportability + Vendor finger pointing

Selling Your Boss

- Or, Security folks are from Mars, businesspeople are from Wheaton

How You Buy Software

- Functionality, supportability, price
- Can you get security in there?
- Probably requires being able to get lots of complexity into a 1-5 score (or somesuch)
- The above can be used for that

Sample Scoring

- 0-1 point for a good language
- 0-1 point for documented use of tools to check code
- 0-1 point for unprivileged, chroot install
- 0-1 point for logging
- 0-1 point for local analysis

Deployment Budgets

- Cash for wires, hubs, power, air
- Where does security fit?
- What's the real cost of a failure?
  - (Hint, its not $1m, unless you're a large bank)

Deployment Business Cases

- Cost of operations with and without tool X
- Cost of special events:
  - Patching
  - Breakins
  - Worms
- Frequency of special events

Summary

- *We'll always have patches to deploy*
- *We can build rational decision processes*
- *We can use better tools*
- *We can push vendors to sell better SW*