

# A MARKET-BASED APPROACH TO SOFTWARE EVOLUTION

DAVID F. BACON \*

YILING CHEN

DAVID PARKES

MALVIKA RAO

HARVARD UNIVERSITY

\* IBM RESEARCH



# BUGS ARE EVERYWHERE

## ANNOYING, COSTLY, DANGEROUS



“Software Crisis” (F. L. Bauer)  
First NATO Software Engineering Conference, 1968



# A TRADITION OF FAILURE

## FORMAL METHODS

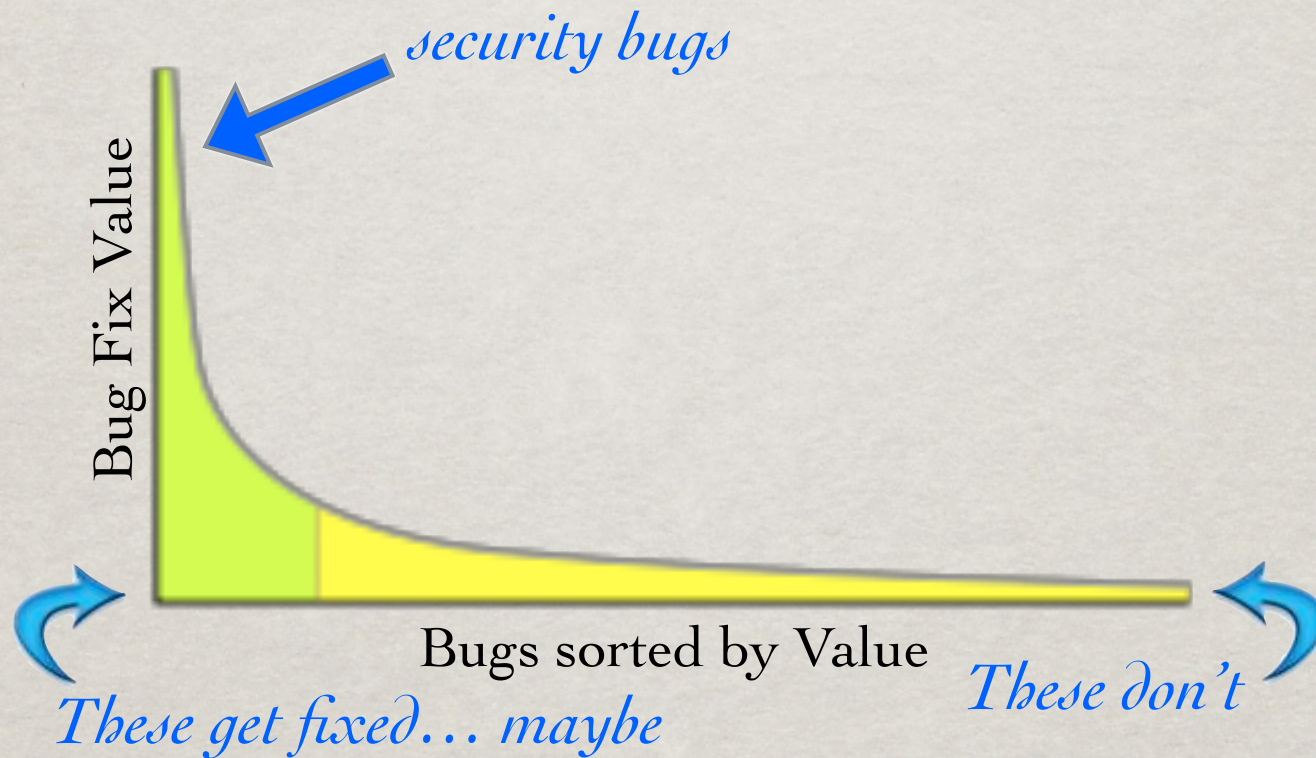
- ✱ Specs & Proofs
- ✱ Model Checking
- ✱ Fatal Flaws:
  - Rely on Spec
  - Don't Scale

## SOFTWARE ENGINEERING

- ✱ Methodology
- ✱ Process
- ✱ Fatal Flaws:
  - Not Quantitative
  - Degenerates to Religion



# BUGS HAVE A “LONG TAIL”



HOW DO BUGS GET SORTED??

HOW ARE COSTS DETERMINED??



# USERS AND DEVELOPERS ARE ISOLATED FROM EACH OTHER

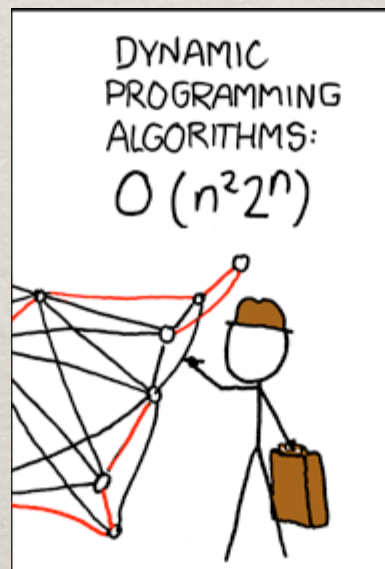
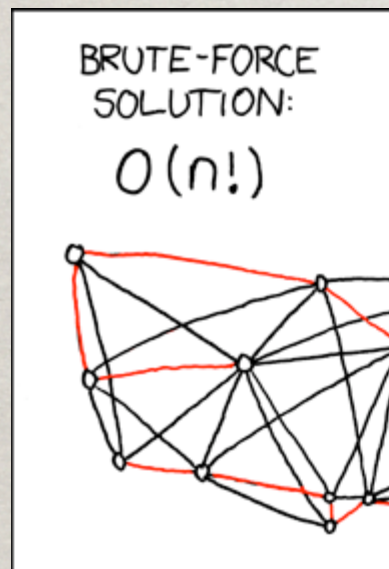


...deliberately  
because feedback can't be accumulated automatically



# CAN A MARKET HELP SOLVE THIS PROBLEM?

- ✱ Large supply of work
- ✱ Large supply of capable workers
- ✱ Real value for performing the work





# IMAGINE...



**The application Safari quit unexpectedly.**

Mac OS X and other applications are not affected.

Click Reopen to open the application again. Click Report to see details or send a report. Click Offer Bounty to contribute to a bounty for fixing this bug.

Close

Report...

Reopen

Offer Bounty





**Select an amount to offer as a bounty for fixing this bug.**

Your bounty will be held in escrow until the bug is fixed or the time limit expires. The default time limit is 6 months.

Currently, 875 users have offered a total of \$2298.45 for fixing this bug. You have been affected by this bug 7 times.

**\$0.99**

**Avg: \$2.63**

**Max: \$50**

**Other**



# CORRECTNESS DEMAND

- ✱ Sum of rewards for a bug is the demand to fix it
- ✱ Sum of all rewards is the *correctness demand*
- ✱ When correctness demand = 0 either
  - ✱ software is bug free or...
  - ✱ no one cares about it anymore.



# CORRECTNESS POTENTIAL

- ✱ Set of possible workers
- ✱ For each bug, each worker has a cost to fix it
- ✱ If  $\text{cost} < \text{reward}$ , “worth fixing” for that worker
- ✱ Potential of bug: profit by most efficient worker
- ✱ *Correctness Potential* = the sum of bug potentials



# CORRECTNESS EQUILIBRIUM

- ✱ Market is in *correctness equilibrium* when correctness potential = 0
- ✱ In “living” software that never happens:
  - ✱ new bugs are found
  - ✱ bug bids change
  - ✱ workers come and go
- ✱ Goal: design a system that tends towards *dynamic equilibrium*



# IS IT A BUG OR A FEATURE? WHO CARES?!



**BUG**



**FEATURE**





# HOW DO WE DESIGN SUCH A MARKET?

## ✻ GUIDING PRINCIPLES:

- ✻ Autonomy: all actions are market-driven
  - ✻ Inclusiveness: all contributors are rewarded
  - ✻ Transparency: “financial disclosure”
  - ✻ Reliability: robustness to manipulation
- 
- ✻ Apply both market pressure and software tools



# WHAT ARE THE COMPONENTS?

- ✻ Funding
- ✻ Workflow Process
- ✻ Reputation System



# SHOW ME THE MONEY!

- ✱ Cash or scrip or votes?
- ✱ Sources of real cash:
  - ✱ direct user bids
  - ✱ escrow from sale (closed source)
  - ✱ escrow from contribution (shareware)
  - ✱ escrow from registration (open source)
- ✱ Time limit on bids - money reverts to source



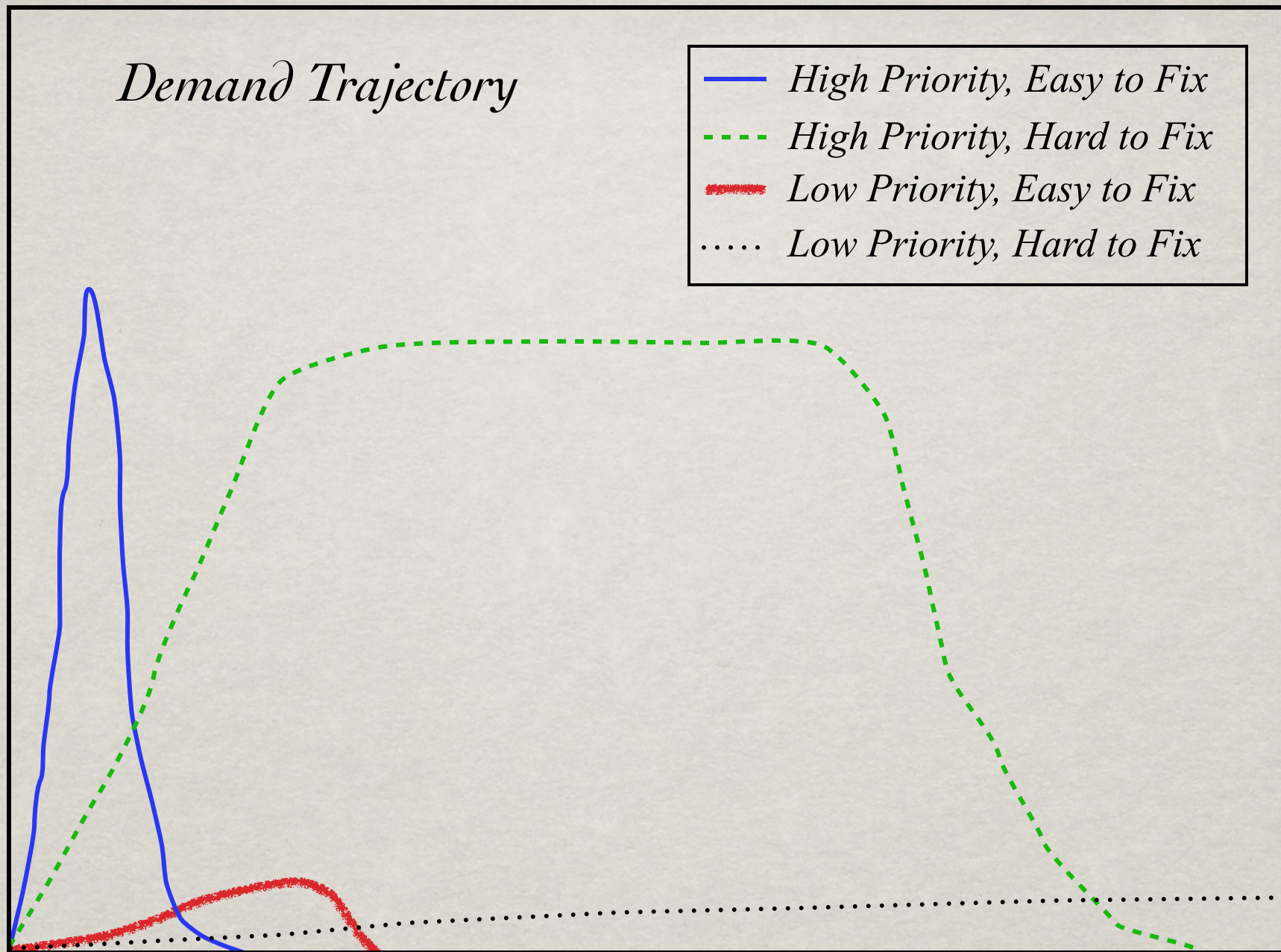
# *Demand Trajectory*

- *High Priority, Easy to Fix*
- - - *High Priority, Hard to Fix*
- ~~.....~~ *Low Priority, Easy to Fix*
- ..... *Low Priority, Hard to Fix*

*Bids - Payouts*

$t_0$

*Time*





# WORKFLOW: BUG

- ✻ Report
- ✻ Bid
- ✻ Categorize
- ✻ Reproduce
- ✻ Fix
- ✻ Test
- ✻ Commit
- ✻ Distribute

*Everyone Shares Reward*

*Humans vs Tools?*



# REPUTATION SYSTEM

- ✱ Ratings based on past performance
- ✱ Control certain activities (e.g. commits)
  - ✱ May also affect reward distribution
- ✱ Adjusted with information about software lifetime
- ✱ Can be seeded by central organization
  - ✱ useful when project is small
  - ✱ occasional escape hatch



# IT'S STARTED: APP STORE

## Player X



### Pang Mobile

Category: Games  
Released Oct 12, 2009  
Seller: Player X  
© 2009 Player X  
Version: 1.0.0 (iPhone OS 3.0 Tested)  
7.0 MB

\$2.99 [BUY APP](#)

Rated 4+



## CUSTOMER RATINGS

▼ Average rating for the current version: ★★★ 41 ratings



## APPLICATION DESCRIPTION

"Check it out for some throwback arcade goodness" – GEARDIARY.COM

A bubble blast from the past, Pang is the game you love and remember from the 80's, even better today on the iPhone and iPod touch.

Test your reflexes in this hugely addictive official remake, "Pang", also known as "Buster Bros" in the US. There's a reason this game was huge – its great!

Travel from Japan to the tropical shores of Hawaii as you take up the task of saving civilization from dangerously bouncing bubbles that threaten city landmarks across the planet.

Can you save Mount Fuji from being flattened by spherical dangers? Will your reactions be good enough to defend the Eiffel Tower or the Statue of Liberty? Or can you wear them down while saving your own skin?

Everyone's favourite from the arcades, Pang is easy to learn but difficult to master as you progress through 17 locations saving the world's attractions with an assortment of weapons from the grappling hook to the twin harpoon. The game is crammed with power-ups including a force field and time freeze; anything to give you the edge over those deadly bubbles!

### FEATURES:

- 17 locations, covering 50 levels
- An assortment of weaponry
- Power-ups and prizes for points
- World Tour or Infinite Challenge to choose from
- Leaderboards – beat your friends for the highest score
- Customisable options suit your style

Download Pang NOW! Relive the great times of retro gaming.

Perfect for fans of Space Invaders, PacMan and Q\*Bert

Works only on iPhone / iPod touch 3.0 OS and above.

### Upcoming updates:

- 1.0.1- 2.2.1 OS and above support added.
- 1.0.2- OpenFeint online leaderboards supported.

Join our Facebook page here:

<http://tinyurl.com/PANG-Mob-FB>

<http://www.geardiary.com/2009/10/19/pang-for-iphonetouch-review/>

### LANGUAGES:

English



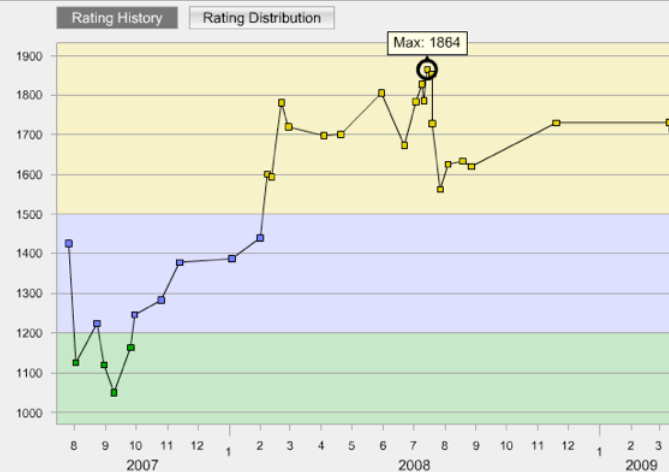
# TOPCODER



## Orange\_Cloud

**Algorithm Rating:** 2282  
**High School Rating:** not rated  
**Conceptualization Rating:** not rated  
**Specification Rating:** not rated  
**Architecture Rating:** not rated  
**Design Rating:** 756  
**Development Rating:** 1715  
**Assembly Rating:** 1216  
**Application Testing Rating:** not rated  
**Marathon Matches Rating:** 1204

**Total Earnings:** \$35,421.90  
**Member Since:** 07.24.2006  
**Country:** Russian Federation  
[\[Send a message\]](#)  
[\[Forum post history\]](#)  
[\[Achievements\]](#)



Algorithm

High School

Conceptualization

Specification

Architecture

Design

Development

Assembly

Application Testing

Marathon Matches

### Development Competitions

**Rating:**  
**1715**

[\[competition history\]](#)  
[\[current contests\]](#)  
[\[reliability detail\]](#)

**Percentile:** 87.5  
**Rank:** 12 of 96  
**Country Rank:** 1 of 2  
**Volatility:** 322  
**Competitions:** 58  
**Reliability:** 66.67%  
**Maximum Rating:** 1864  
**Minimum Rating:** 1050

### Submission Details

	Java	.NET	Total
Inquiries	58	0	58
Submissions	33	n/a	33
Submission Rate	56.90%	n/a	56.90%
Passed Screening	33	n/a	33
Screening Success Rate	100.00%	n/a	100.00%
Passed Review	33	n/a	33
Review Success Rate	100.00%	n/a	100.00%
Appeals*	241	n/a	241
Appeal Success Rate*	47.72%	n/a	47.72%
Maximum Score	99.43	n/a	99.43
Minimum Score	78.46	n/a	78.46
Average Score	94.06	n/a	94.06
Average Placement	1.85	n/a	1.85
Wins	19	n/a	19
Win Percentage	57.58%	n/a	57.58%

\* only includes appeals from projects posted on or after March 16, 2006 (TopCoder did not previously collect the relevant data)



# MARKET-BASED SOFTWARE

- ONLY POSSIBLE KIND OF SOLUTION
- EMPOWERS USERS AND PROGRAMMERS
- MAKES PROBLEM QUANTITATIVE



THANKS.

FEEDBACK?



# MECHANISM DESIGN PROBLEMS

- ✻ Avoiding Freeloading
- ✻ Preventing Fraudulent “Fixed” Claims by Providers
- ✻ Preventing Fraudulent “Not Fixed” Claims by Consumers
- ✻ Lag in fix verification by Consumers



# LOTS OF UNCERTAINTY

- ✻ When are two crashes the “same bug”?
  - ✻ Line number? Data set?
- ✻ When does a change “fix” a bug?
  - ✻ Partial fixes & incorrect fixes are not uncommon
  - ✻ One fix may improve or worsen another bug
- ✻ If multiple fixes submitted, which is best?
  - ✻ Band-aids versus Deep fixes
- ✻ *Program analysis can help reduce uncertainty, but will never eliminate it*



# NEXT STEPS

- ✻ Simplified market mechanism design with analytical equilibrium property
- ✻ Identify analysis and testing techniques that can be integrated into the system.
- ✻ Prototype market infrastructure
- ✻ Trial run (seed a market?)



# TOPCODER

- ✱ Handles “supply side” -- developers
- ✱ Highly differentiated stages of development
- ✱ Short, manageable tasks
- ✱ Competitive process
- ✱ Validation:
  - ✱ automated testing
  - ✱ competitive forces: challenges

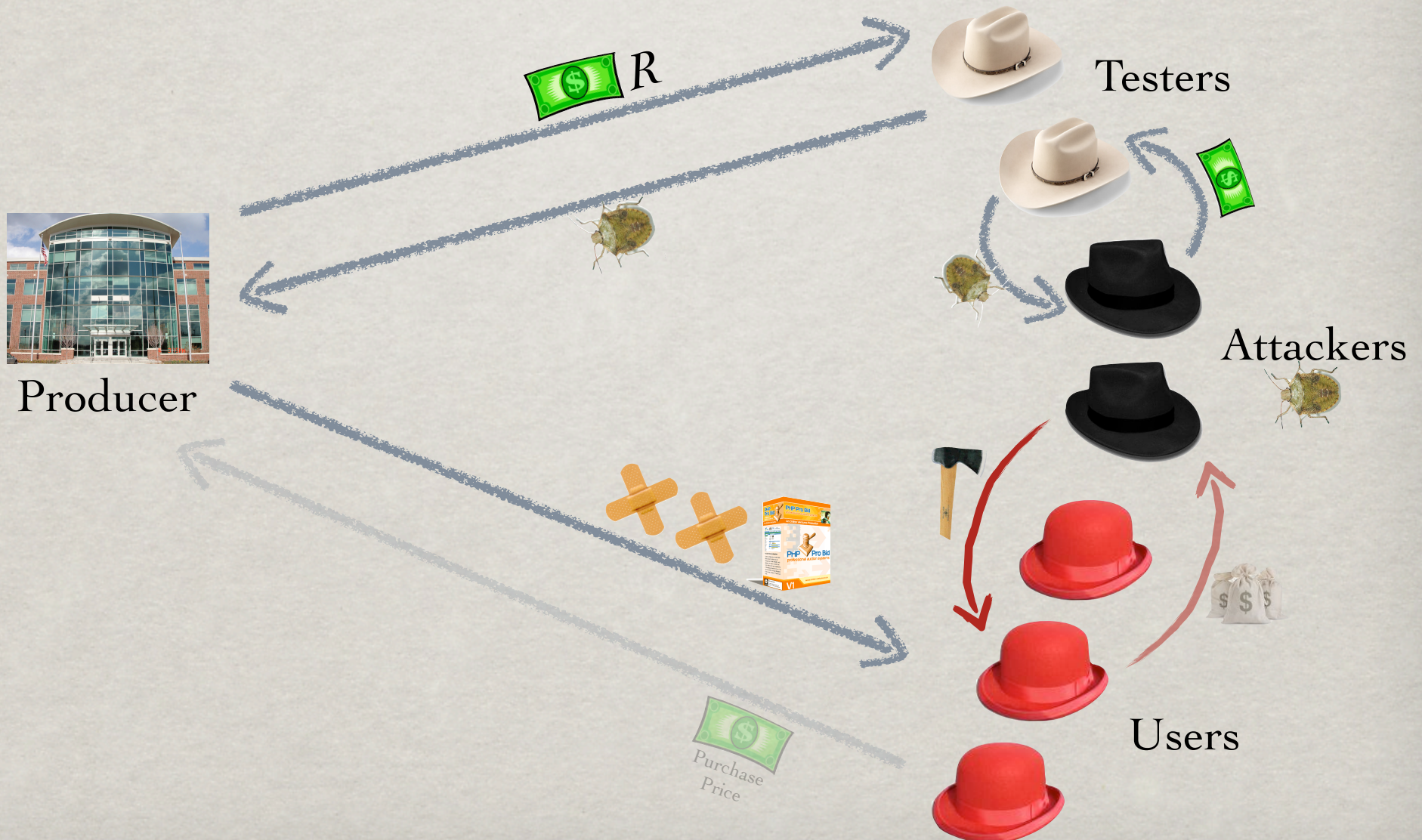


# ITUNES APP STORE

- ✱ Micropayment system with broad acceptance
- ✱ Primarily supply side
  - ✱ but often compete for users on similar apps
- ✱ Monolithic -- but apps are fine-grained
  - ✱ Developers responsive to user feedback
- ✱ Software Distribution Mechanism



# BUG AUCTIONS FOR VULNERABILITY MARKETS





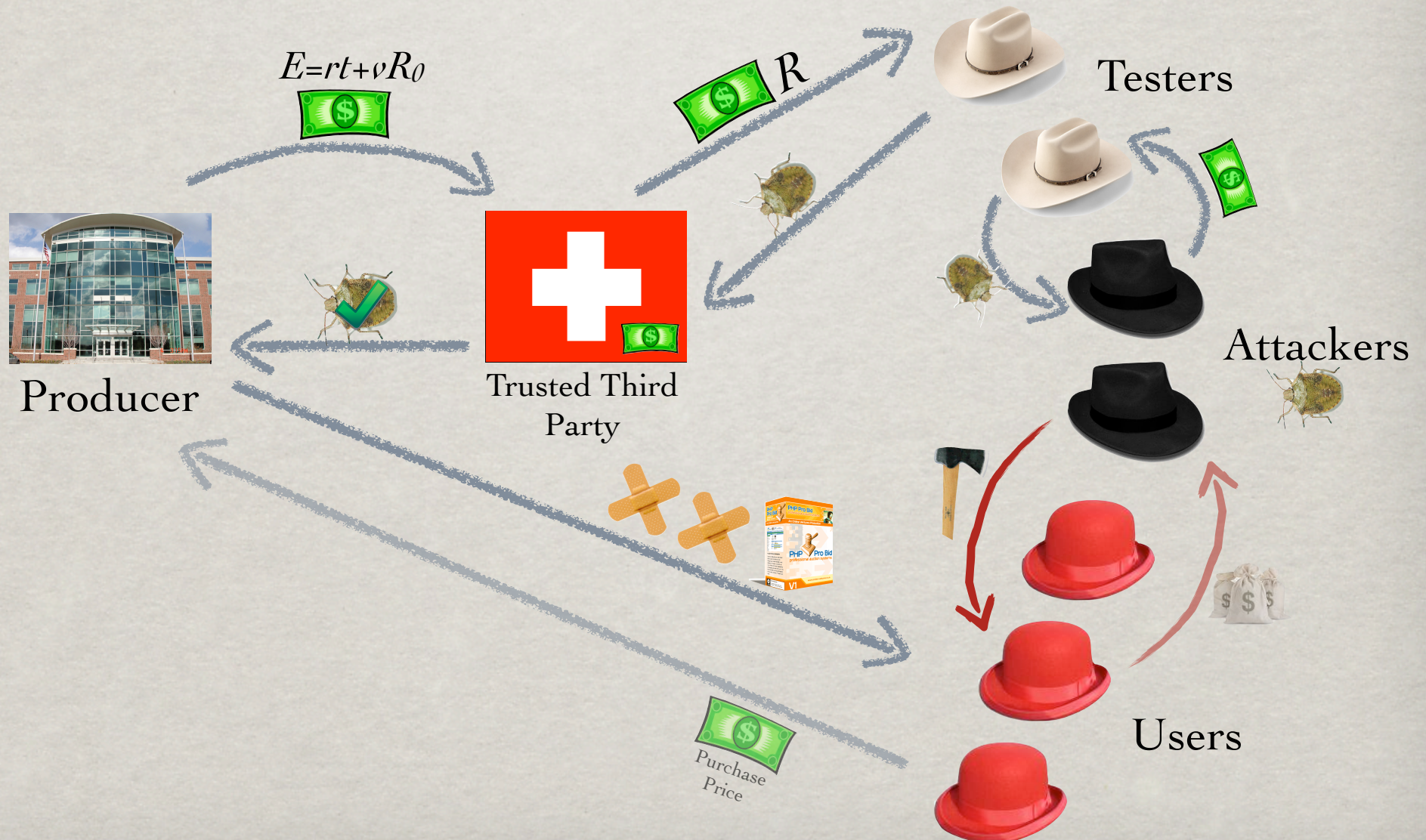
# BUG AUCTIONS FOR VULNERABILITY MARKETS

(OZMENT'S REDEFINITION OF SCHECHTER)

- ✻ Note: paying for bug reports (“user” activity)
- ✻ Bounty  $R$  starts at  $R_0$  increasing by  $\partial/\text{day}$
- ✻ Open first-price ascending (reverse Dutch) auction
  - ✻ Open auction speeds discovery
- ✻ Non-security bugs receive  $fR$ , where  $f \ll 1$
- ✻  $R$  acts as a “measure of security”



# BUG AUCTIONS FOR VULNERABILITY MARKETS (OZMENT'S ENHANCEMENTS)





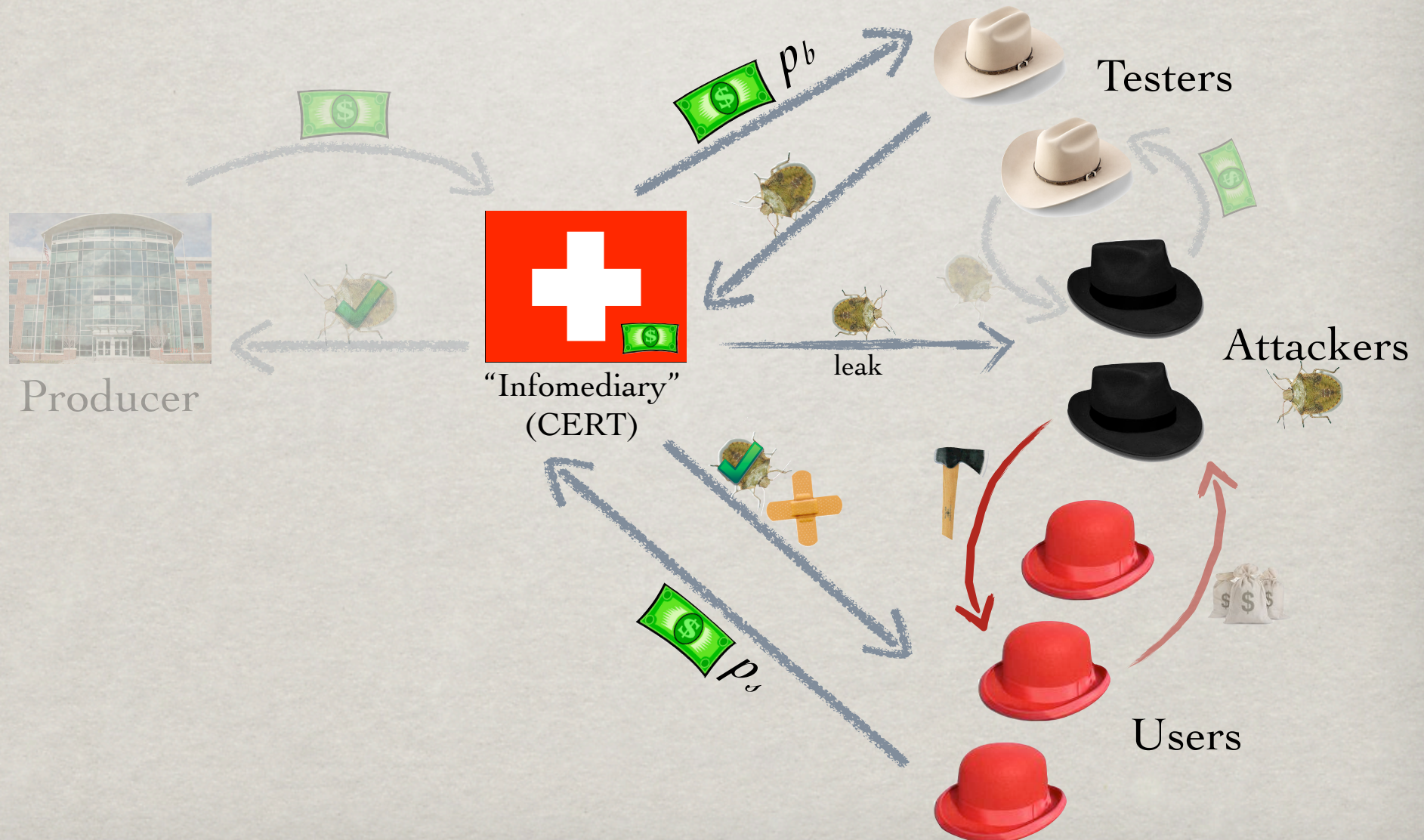
# BUG AUCTIONS FOR VULNERABILITY MARKETS (OZMENT'S ENHANCEMENTS)

- ✻ Set initial reward (first  $R$ ) high
- ✻ Include reputation reward
- ✻ Commit/escrow minimum payout  $E=rt+\nu R_0$
- ✻ Reduce  $R$  to  $Rx$  ( $x < 1$ ) if exploit precedes fix
- ✻ Don't expose number of testers (unless small)
- ✻ Give reward for registered testers
- ✻ Use trusted third party to escrow reward fund



# VULNERABILITY MARKETS

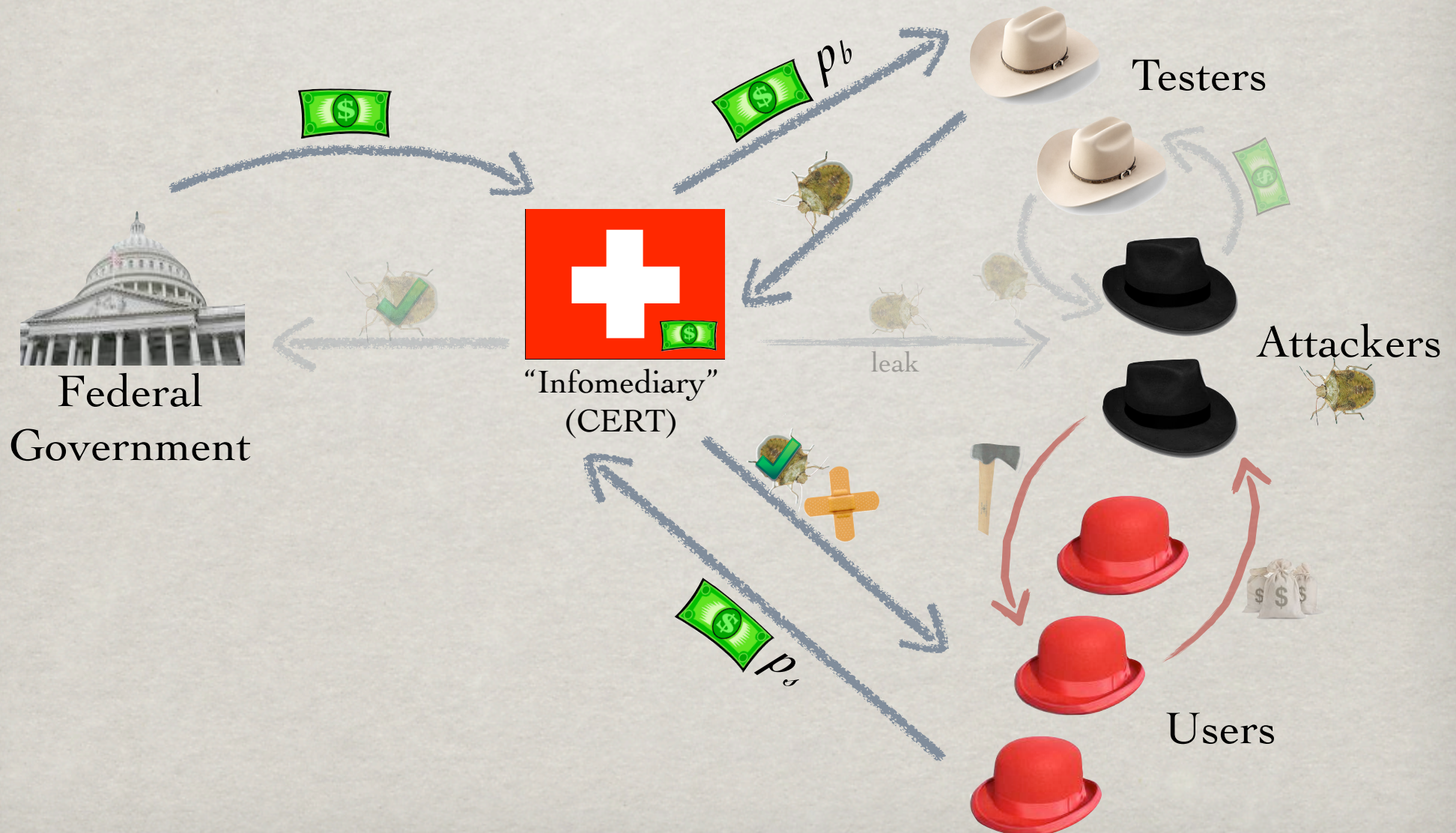
(KANNAN & TELANG)





# “FEDERAL FUNDING”

(KANNAN & TELANG)





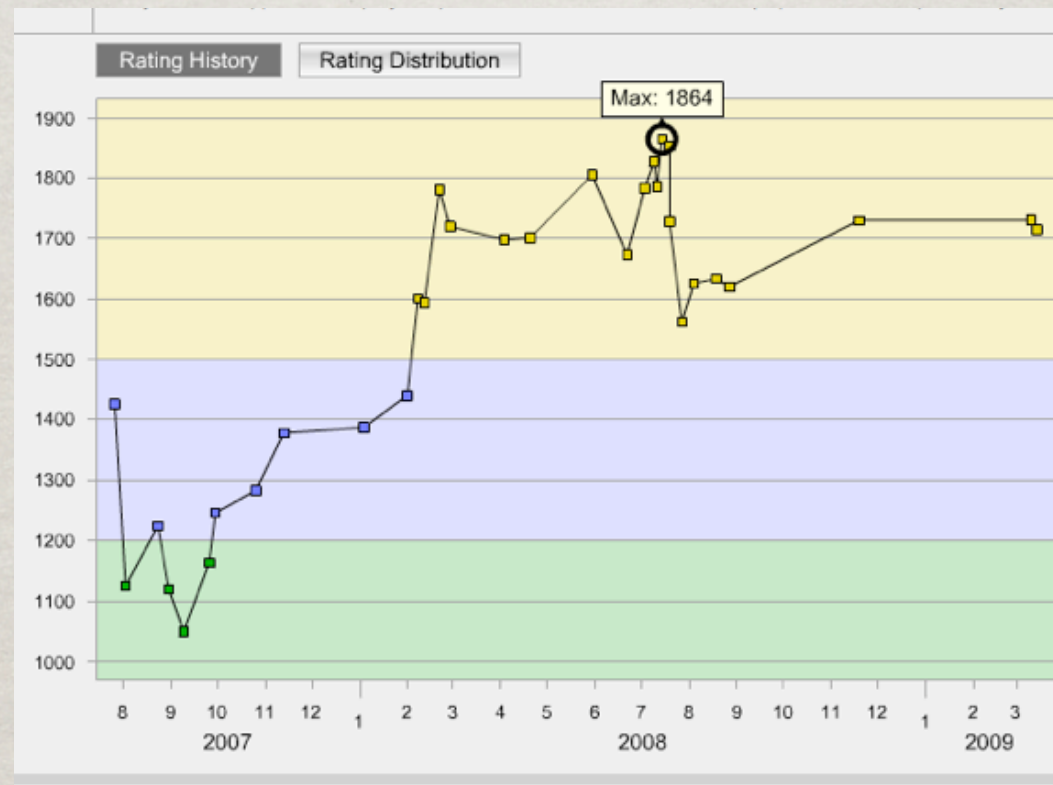
A COMPREHENSIVE  
MARKET FOR  
SOFTWARE  
EVOLUTION



# FORMAL TECHNIQUES WON'T NEVER BE A SCALE

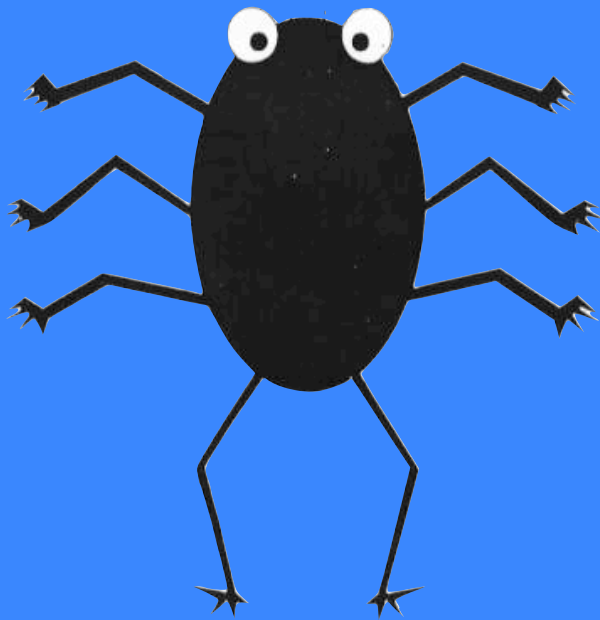
- ✱ Specifications and Proofs of Correctness
  - ✱ Limited to ~1000 line programs
- ✱ Model Checking
  - ✱ Limited to problems with small state spaces
- ✱ Big, real-world programs often have no precise “spec”
  - ✱ ...or it's too complex to verify or test exhaustively
- ✱ Dijkstra Turing Award prediction failed to happen



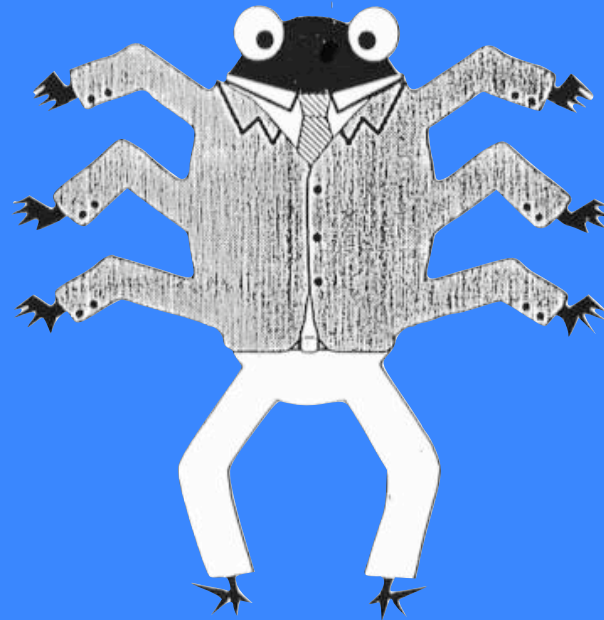




# BUT WHY DIFFERENTIATE?



**BUG**



**FEATURE**





# ASIDE: MECHANISM DESIGN

- ✻ What information is revealed has a big impact





# BugBounty.Com



## Top 3 Fatal Bugs Mozilla Firefox



COMPONENT	DESCRIPTION	BOUNTY HUNTERS	USERS	PER-USER BOUNTY	TOTAL BOUNTY
Widget: Cocoa	firefox hangs if cookie ask permission to set whilst save target as dialog is open (image)	3	1521	\$2.27	\$3457.98
Places	Live bookmarks load way too aggressively (lock up/hang/freeze browser)	1	162	\$9.12	\$1477.44
XUL	UI freezes if alert/dialog comes up while dragging (Modal dialog during drag causes hang)	0	3818	\$0.34	\$1298.12



# SINCE SPECS ARE FALLIBLE...

- ✻ Forget formal specification
- ✻ *The spec is what the market says it ought to be*



# AND WHILE WE'RE AT IT...

## BROADEN THE MARKET

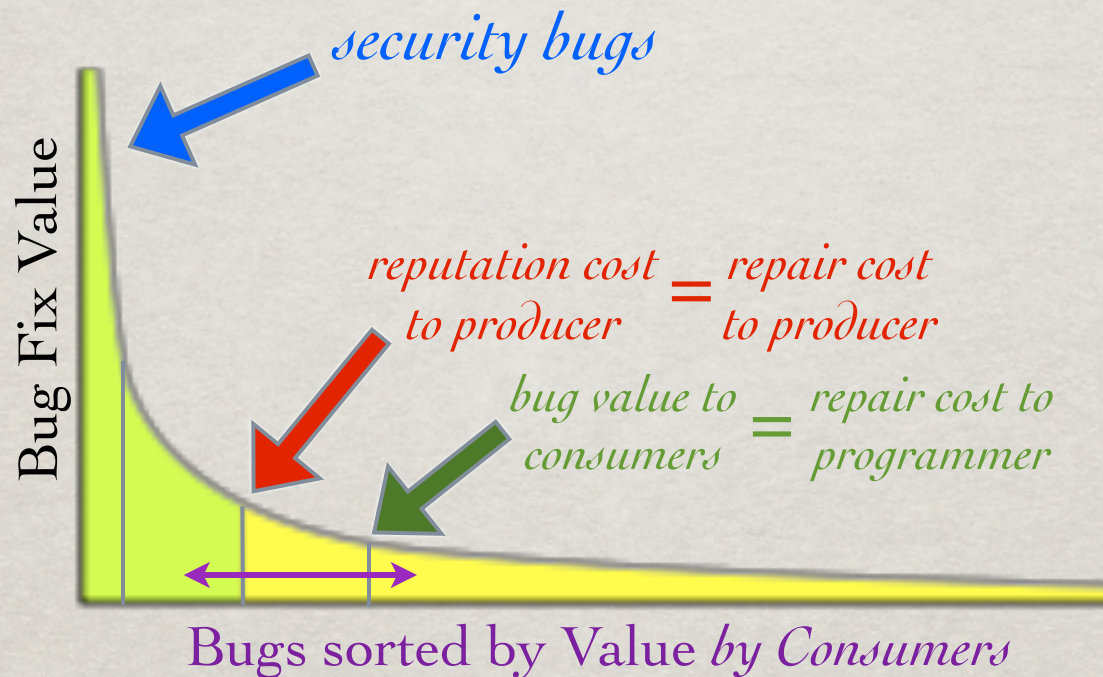
- ✻ Documentation

- ✻ “Help Desk” Support (0-line aka RTFM fixes)

- ✻ Installation



# EMPOWERING THE TAIL: CONSUMER BUG BOUNTIES



Select an amount to offer as a bounty for fixing this bug.

Your bounty will be held in escrow until the bug is fixed or the time limit expires. The default time limit is 6 months.

Currently, 875 users have offered a total of \$2298.45 for fixing this bug. You have been affected by this bug 7 times.

\$0.99

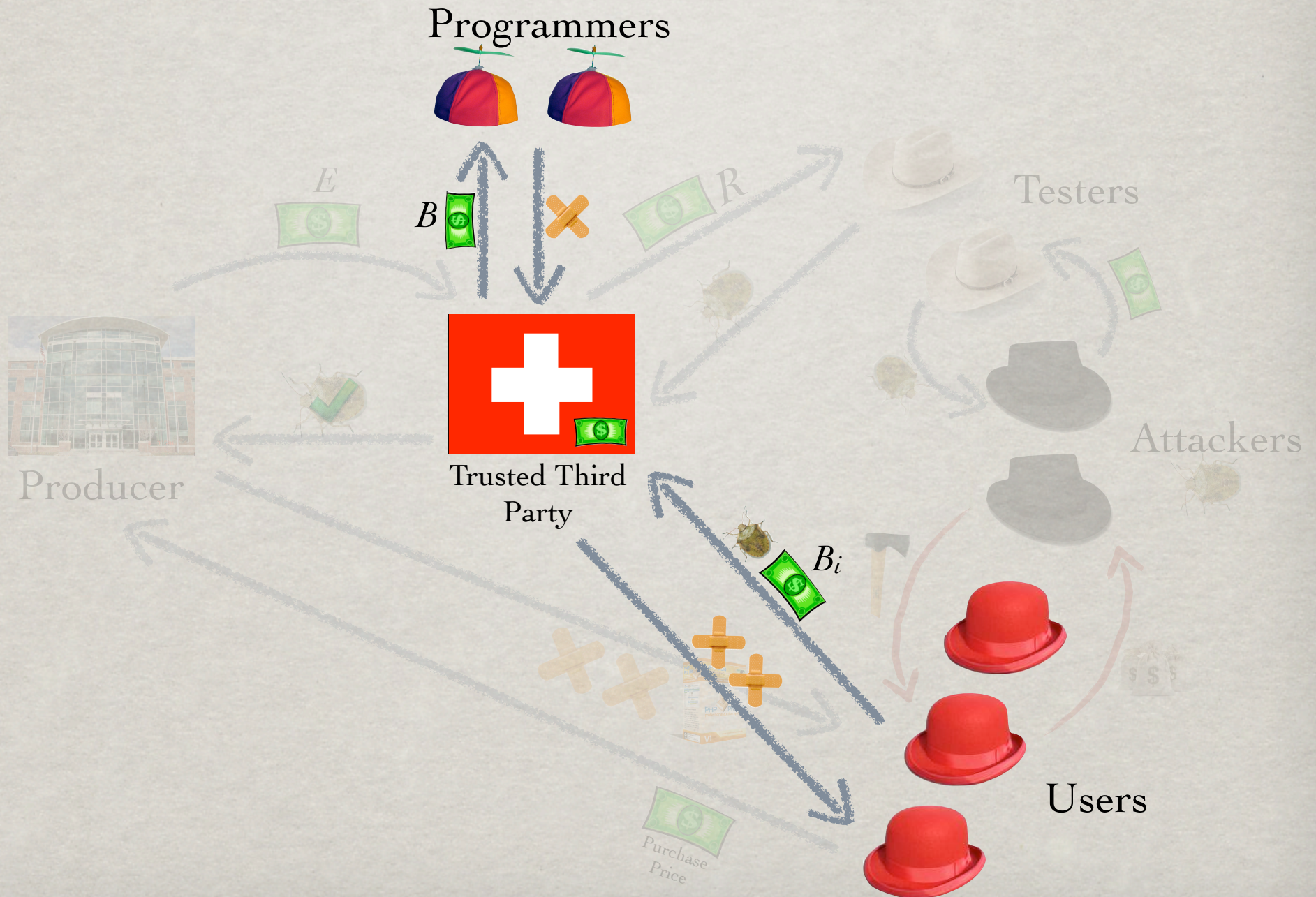
Avg: \$2.63

Max: \$50

Other



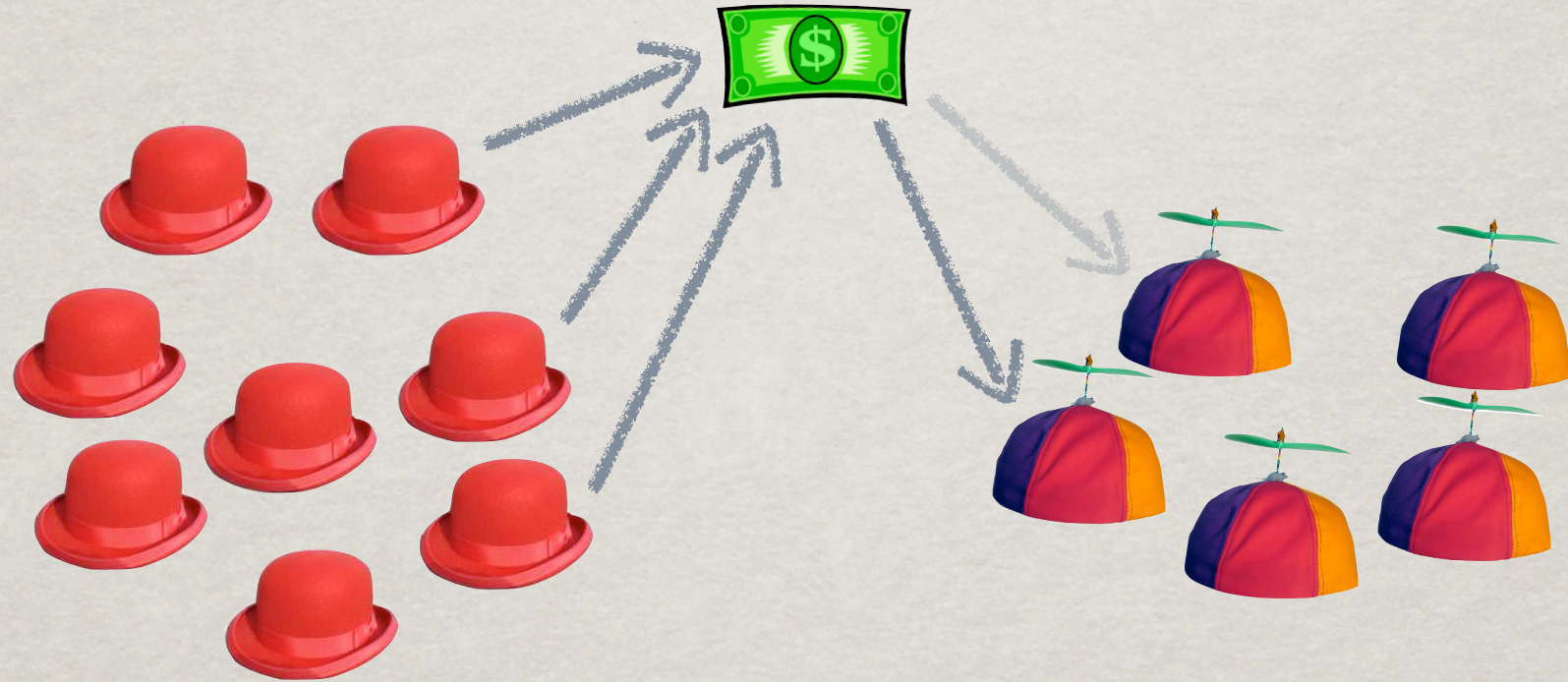
# SOFTWARE IMPROVEMENT





# COMPLEX STRUCTURE

## PROBLEM ONLY WITH UNCERTAINTY(?)



Multiple Aggregated  
“Consumers”

Multiple Competing  
“Providers”



# SOCIAL UTILITY ISSUES



Open source: avoid “crowding out” altruistic providers

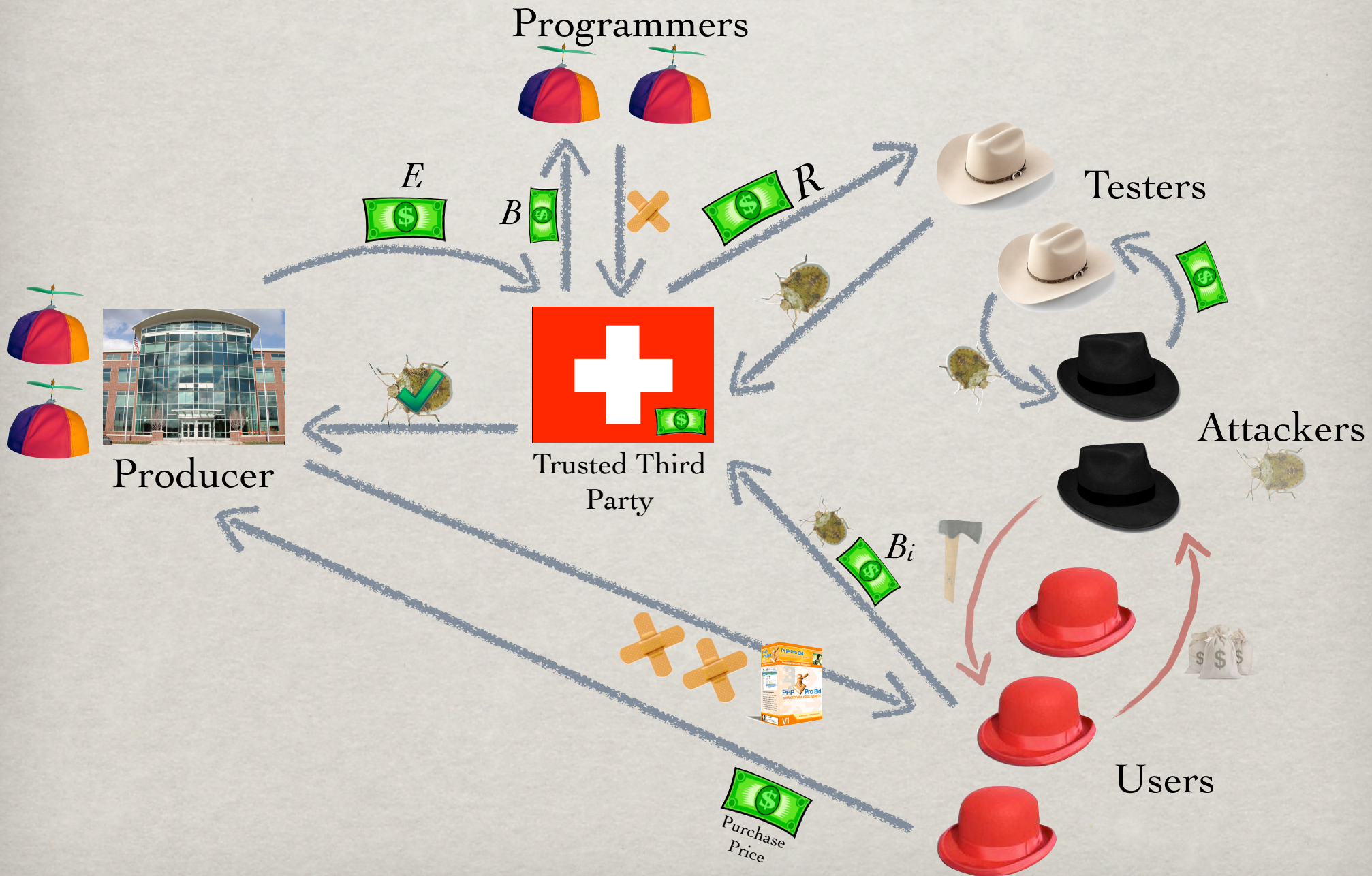


Closed source: drive collaboration and profit-sharing

- Would companies allow their programmers to collect bounties?




# GENERALIZED MARKET





# GENERALIZED APPLICATION

- ✱ Security bugs
- ✱ Functional bugs
- ✱ Non-fatal bugs
- ✱ Feature requests



How are these  
reported and  
aggregated??



# ASSUME AWAY UNCERTAINTY?

- ✱ Design market assuming we can
  - ✱ precisely classify bugs
  - ✱ precisely identify fixes



# ATTACK UNCERTAINTY SEPARATELY

- ✱ Program analysis
  - ✱ Program slicing
  - ✱ Statistical clustering techniques
- ✱ User Observation
  - ✱ Change in bug frequency
  - ✱ Rating of Producers (for fixes) and Consumers (for acceptance tests)



# “APP STORE” MODEL

✻  $N$  Consumers, but only 1 Producer