



PERCONA  
Performance Consulting Experts

# How to Really Prevent Downtime in MySQL

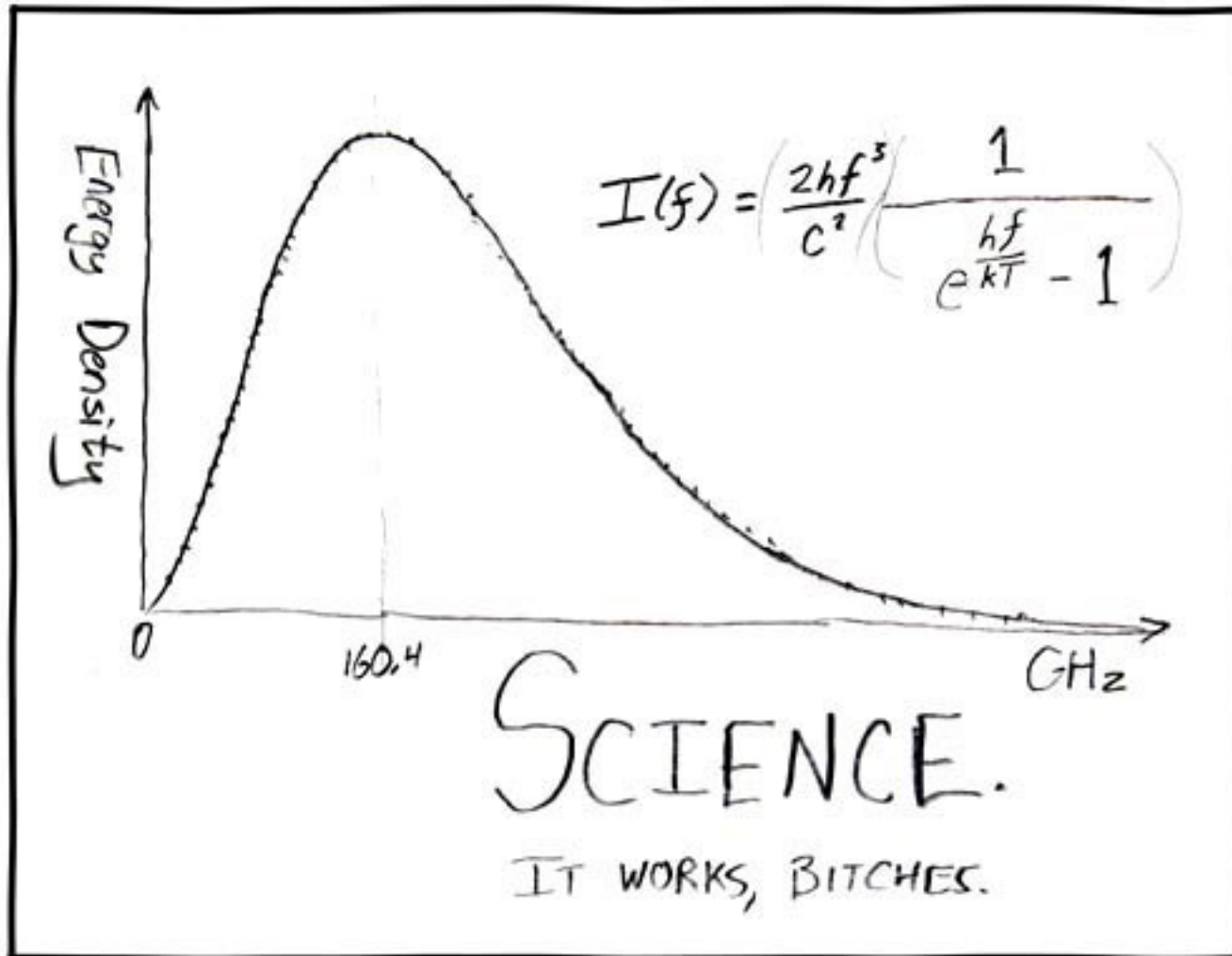
Baron Schwartz  
RubyNation 2011

# Myths About Downtime

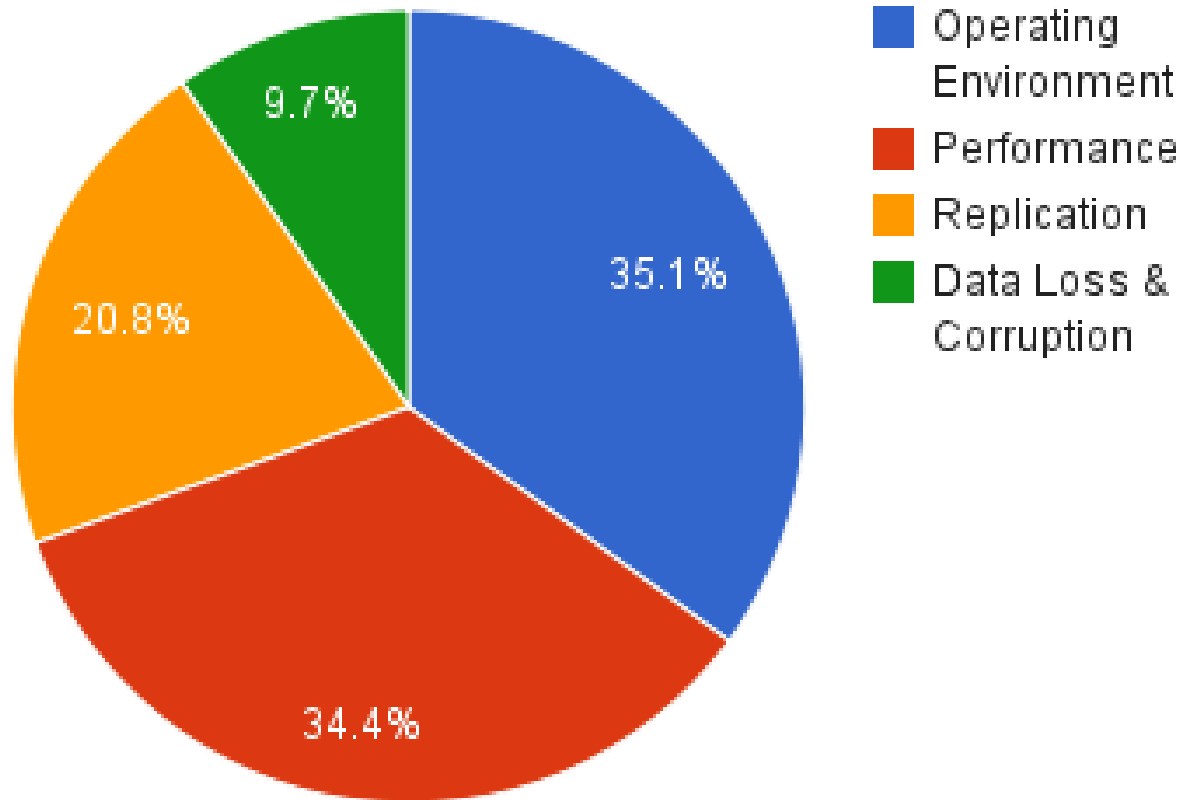
---

- A SAN is a high availability solution
- Bad SQL causes most problems
- You should monitor 116 things for warning signs
  - see <http://tinyurl.com/over-monitoring>

# Let's Stop Guessing.



# Where Downtime Happens



# Operating Environment Problems

---

- Most problems are storage problems.
- Others: network, init scripts, power failure...

# Storage Failures

---

- Leading failure: disk full.
- Runner-up: SAN failure.

# Performance Problems

---

- Top causes: bad SQL and bad indexing.
- Surprise: idle transactions.

# Replication Problems

---

- Top problem: data out of sync.
- Top root cause: humans and bad HA tools.

# Data Loss & Corruption

---

- Top cause: DROP TABLE and similar.  
(Again with the humans!)
  - Note: replication doesn't save you from this.
- InnoDB does not corrupt easily.

# Root Cause Analysis

---

- There Really Is No Root Cause.
- <http://tinyurl.com/complex-systems-fail>

# #1 Prevention: Change Control

---

- Upgrades are risky.
  - Risk 1: not upgrading
  - Risk 2: upgrading without testing
- See mk-upgrade in Maatkit.

# Operating Environment

---

- MySQL is vulnerable to its environment
- Anything it relies on that fails...
  - DNS
  - Error logging and log rotating
  - Init scripts
  - Storage system

# Monitoring, Graphing

---

- Monitor selectively
- Graph promiscuously

# What Does Proactive Look Like?

---

- Reboot your servers every now and then.

# Proactivity, Continued

---

- Measure statistical variations in workload
- Watch [mysqlperformanceblog.com](http://mysqlperformanceblog.com) for more research on this

# Proactivity, Continued

- Use good-quality, formally-tested tools
- Use Maatkit!



# Proactivity, Continued

---

- Don't trust automatic startup processes
- Example: set `skip_slave_start` and `read_only`

# Further Reading

---

- 2 white papers on downtime causes/prevention
- <http://www.percona.com/about-us/mysql-white-papers/>

# Make it Someone's Job

---

- Decide what to do (read the white papers)
- Then assign follow-through to someone

# What Can We Learn?

---

- Untested scripts are dangerous.
- “Tuning the server” is a waste of time.
- Humans are probably the biggest risk.