Calculating security debt

Meaningful Software Security Metrics Panel

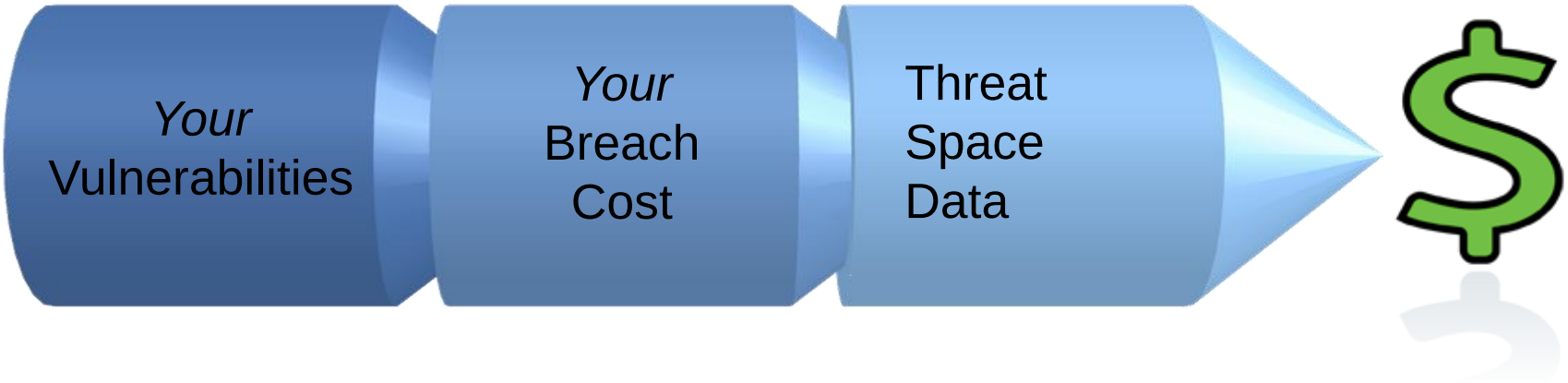
Chris Wysopal
CTO & co-founder

VERACODE

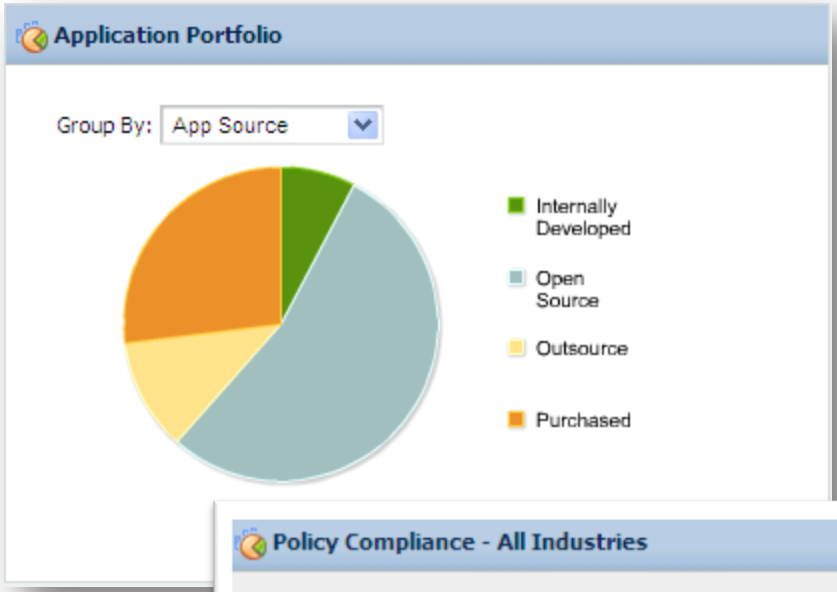
SANS AppSec – March 8, 2011

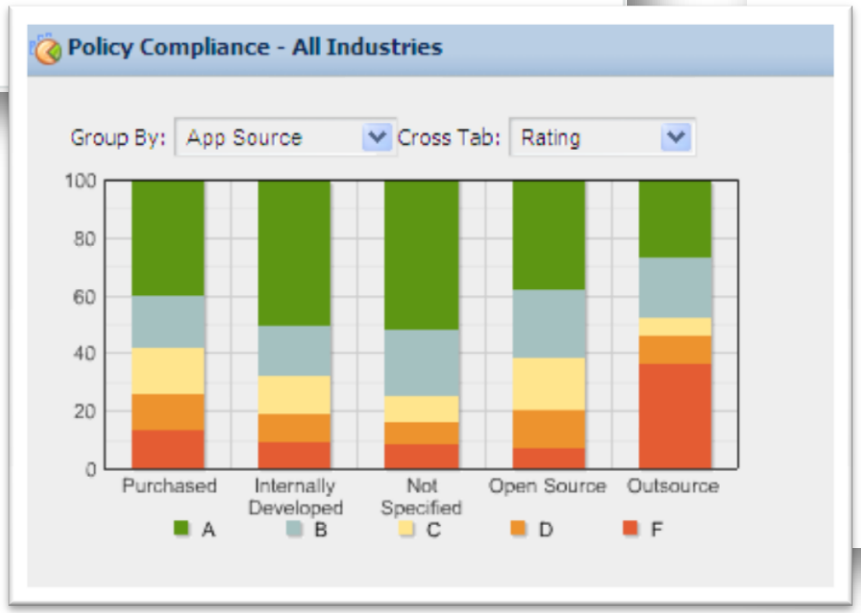
Monetary Risk of Application Portfolio

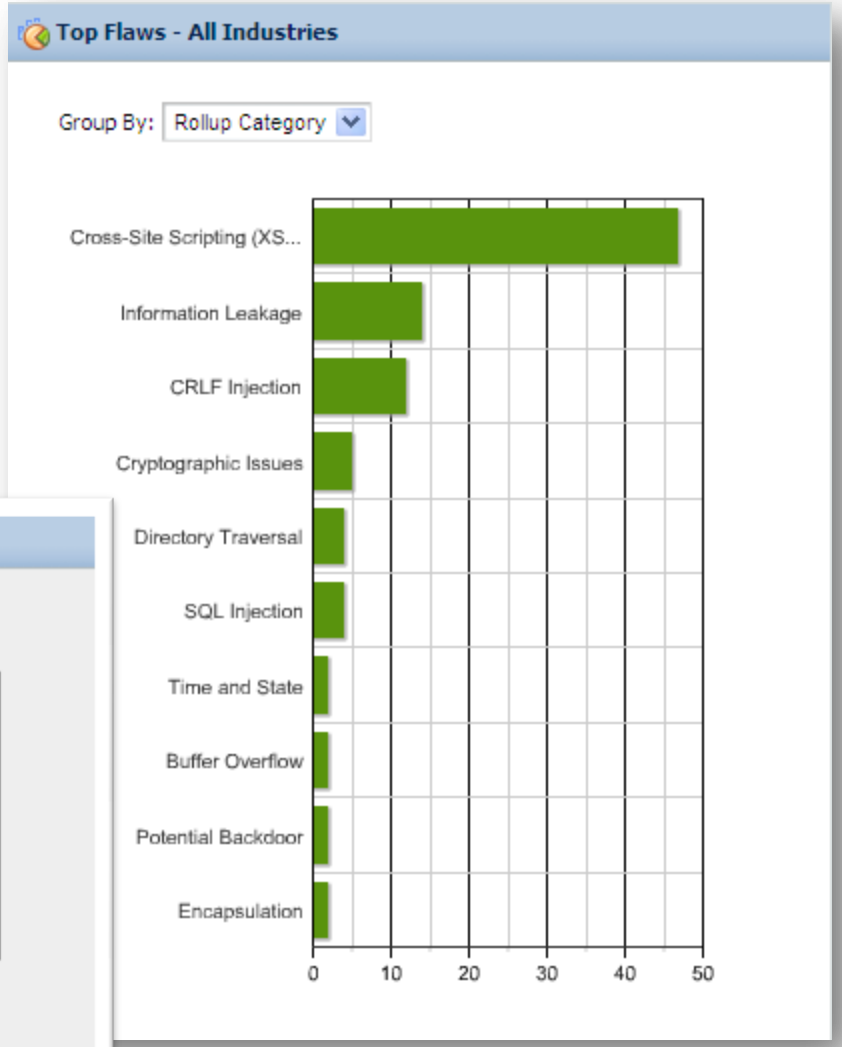
- **Security debt it the cost of the vulnerabilities in your application portfolio.**
- **Question: What is the monetary risk from vulnerabilities in your application portfolio?**
- **Useful Answer: Monetary risk is *your* expected loss; derived from *your* vulnerabilities, *your* breach cost,, threat space data**



Vulnerabilities in Your Application Portfolio







Your Breach Cost

- **Use cost analysis from your earlier breaches**
- **Use breach cost from public sources**
  - Example: April 2010 Ponemon Institute Report

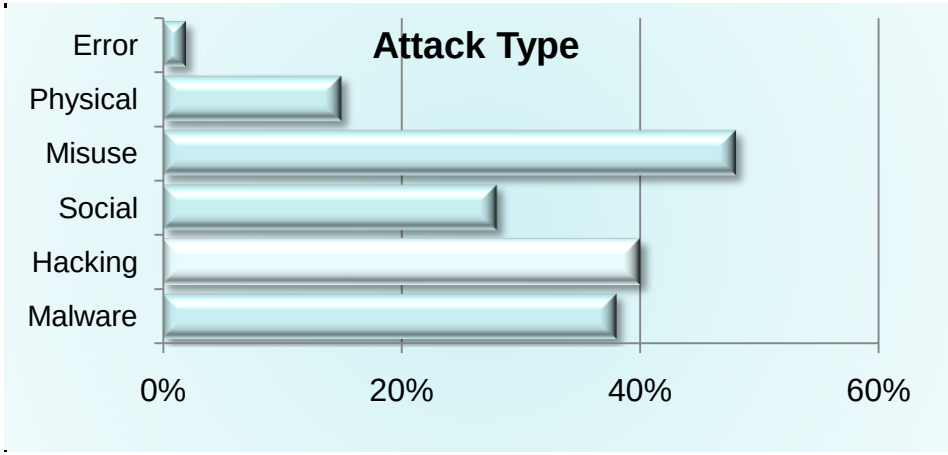
(US Dollars)	Detection & Escalation	Notification	Ex-Post Response	Lost Business	Total
Average	264,208	500,321	1,514,819	4,472,030	6,751,451
Per-capita	8	15	46	135	**204**

Ponemon average and per-capita US breach cost (US Dollars)

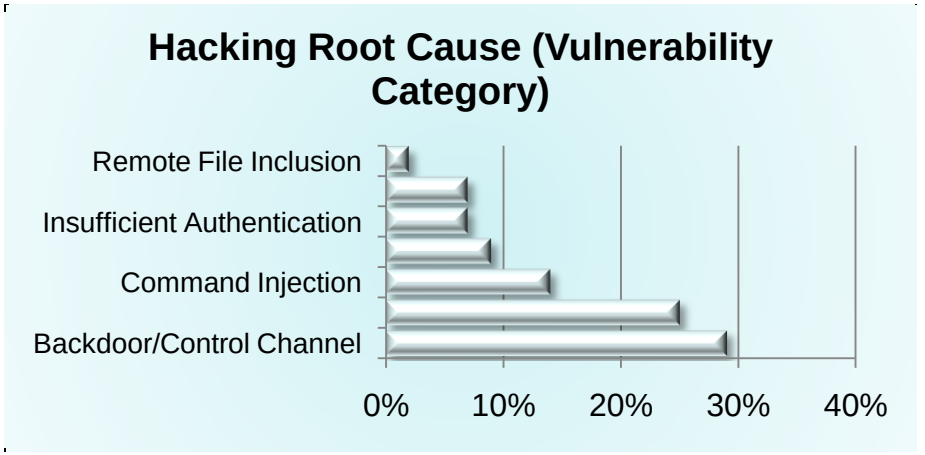
Communication	Consumer	Education	Energy	Financial	Healthcare	Hotel & Leisure	Manufacturing	Media	Pharma	Research	Retail	Services	Technology	Transportation
209	159	203	237	**248**	294	153	136	149	310	266	133	256	192	121

Ponemon per-capita data by US industry sector (US Dollars)

Threat Space Data



40% of data breaches are due to hacking



Top 7 application vulnerability categories

Source: Verizon 2010 Data Breach Investigations Report

62% of organizations experienced breaches in critical applications in 12 month period

Source: Forrester 2009 Application Risk Management and Business Survey

How to Derive Your Expected Loss

$$expected\ loss_{\ vulnerability\ category} = f\left(\begin{array}{c} \%\ of\ orgs\ breached, * \\ breach\ cost, * \\ breach\ likelihood\ from\ vuln.category \end{array}\right)$$

Baseline expected loss for your organization due to SQL Injection*

$$expected\ loss_{SQL\ Injection} = f\left(\begin{array}{c} 62\%, * \\ \$248 * 100{,}000 \\ * 25\% \end{array}\right) = \$3{,}844{,}000$$

If your SQL Injection prevalence is similar to average SQL Injection prevalence, assumes 100,000 records

Monetary Risk Derived From Relative Prevalence

Vulnerability Category	Breach Likelihood	Baseline Expected loss	Average % of Apps Affected[1]	Your % of Apps Affected[2]	Your Monetary Risk
Backdoor/ Control Channel	29%	$4,459,040	8%	15%	higher
SQL Injections	25%	3,844,000	24%	10%	lower
Command Injection	14%	2,152,640	7%	6%	same
XSS	9%	1,383,840	34%	5%	lower
Insufficient Authentication	7%	1,076,320	5%	2%	lower
Insufficient Authorization	7%	1,076,320	7%	7%	same
Remote File Inclusion	2%	307,520	<1%	<1%	same

1. Veracode 2010 State of Software Security Report, Vol. 2
2. De-identified financial service company data from Veracode industry data

Assume 100,000 customer records.
For SQLi the expected loss is:
62% * $248 * 100,000 * 25% = $3,844,000