Stop complaining and...

Solve a Security Problem Instead

By Ivan Ristic

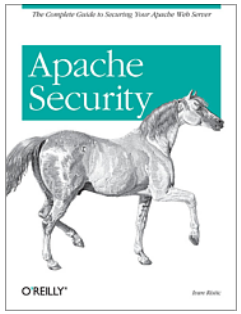
QUALYS®

I am a compulsive builder

1) ModSecurity (open source web application firewall), **2)** *Apache Security* (O'Reilly, 2005), **3)** SSL Labs (research and assessment platform), **4)** *ModSecurity Handbook* (Feisty Duck, 2010)

modsecurity

QUALYS® SSL LABS



Message for today Software is universally insecure, and we are not doing enough to make things right.

Morris Worm

In November 1998, the first computer worm infected about 10% of the Internet (about 6,000 servers). The worm was written by Robert T. Morris.

(The worm source code is available from www.foo.be/docs-free/morris-worm/.)

The Morris Worm spread using **password cracking**, **server misconfiguration**, **buffer overflows**, and **remote code execution**.

Same as today, eh? We haven't seen an improvement in computer security in the **22 years** since the first worm.

In fact, the situation has become **much worse** because of the wide adoption of computers and the Internet.

Why? Four reasons:

1) *ignorance*, **2)** *convenience*,

3) *economics*, and **4)** *no single point of control*, but ultimately because

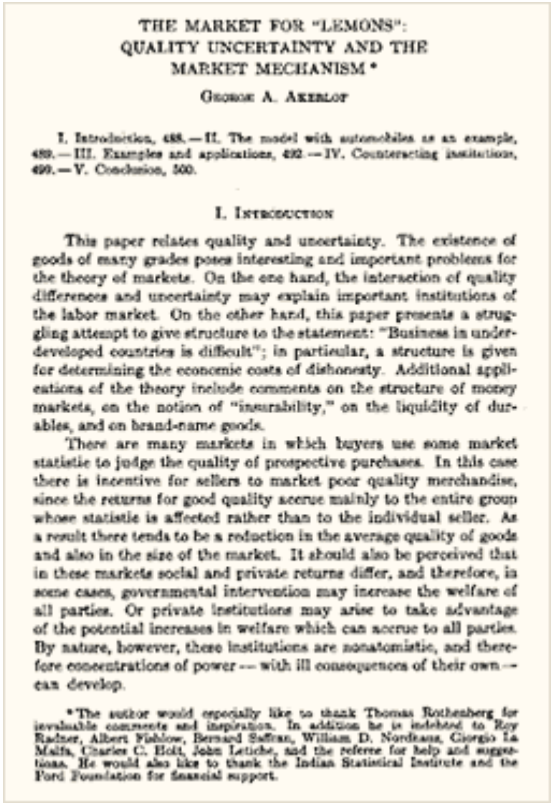
security is not important to us.

Software is a

market for lemons.

George A. Akerlof

The Market for "Lemons": Quality Uncertainty and the Market Mechanism



THE MARKET FOR "LEMONS":
QUALITY UNCERTAINTY AND THE
MARKET MECHANISM*

GEORGE A. AKERLOF

I. Introduction, 488.—II. The model with automobiles as an example, 489.—III. Examples and applications, 492.—IV. Counteracting institutions, 499.—V. Conclusion, 500.

I. INTRODUCTION

This paper relates quality and uncertainty. The existence of goods of many grades poses interesting and important problems for the theory of markets. On the one hand, the interaction of quality differences and uncertainty may explain important institutions of the labor market. On the other hand, this paper presents a struggling attempt to give structure to the statement: "Business in underdeveloped countries is difficult"; in particular, a structure is given for determining the economic costs of dishonesty. Additional applications of the theory include comments on the structure of money markets, on the notion of "insurability," on the liquidity of durables, and on brand-name goods.

There are many markets in which buyers use some market statistic to judge the quality of prospective purchases. In this case there is incentive for sellers to market poor quality merchandise, since the returns for good quality accrue mainly to the entire group whose statistic is affected rather than to the individual seller. As a result there tends to be a reduction in the average quality of goods and also in the size of the market. It should also be perceived that in these markets social and private returns differ, and therefore, in some cases, governmental intervention may increase the welfare of all parties. Or private institutions may arise to take advantage of the potential increases in welfare which can accrue to all parties. By nature, however, these institutions are nonatomistic, and therefore concentrations of power — with ill consequences of their own — can develop.

*The author would especially like to thank Thomas Rothenberg for invaluable comments and inspiration. In addition he is indebted to Roy Radner, Albert Fishlow, Bernard Saffran, William D. Nordhaus, Giorgio La Malfa, Charles C. Holt, John Letiche, and the referee for help and suggestions. He would also like to thank the Indian Statistical Institute and the Ford Foundation for financial support.

"[…] the presence of people who wish to pawn bad wares as good wares tends to ***drive out the legitimate business****".*

Security comes from making sensible decisions, thinking things trough, taking your time… It is **boring** and it **doesn't make anyone rich**.

Open source projects just want to succeed, companies want to make profit, people want to get things done.

Security is standing in everyone's way.

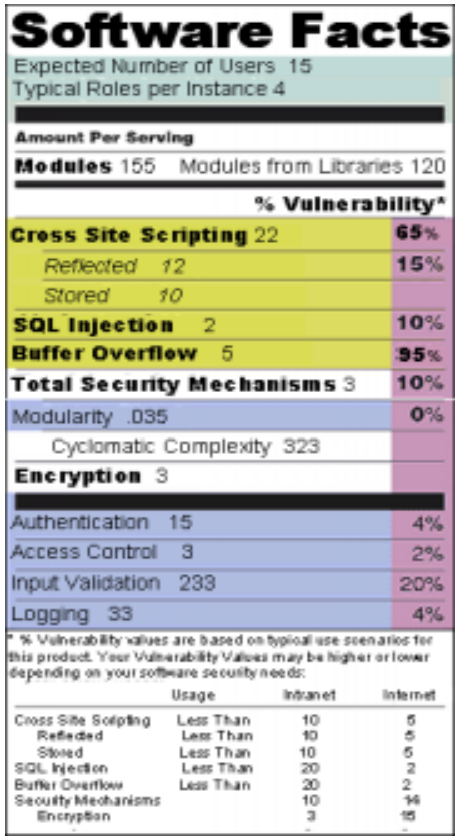
Only one solution long-term: **make the parties involved accountable** for the quality.

But we are probably not ready yet.

Self-certification

Could help us focus on those who really should be liable.

(*The Software Facts label taken from Jeff Williams's talk at AppSec Europe 2005.*)



Software Facts

Expected Number of Users 15
Typical Roles per Instance 4

Amount Per Serving

Modules 155 Modules from Libraries 120

% Vulnerability*

Cross Site Scripting 22	**65%**
Reflected 12	**15%**
Stored 10	
SQL Injection 2	**10%**
Buffer Overflow 5	**95%**
Total Security Mechanisms 3	**10%**
Modularity .035	**0%**
Cyclomatic Complexity 323	
Encryption 3	
Authentication 15	4%
Access Control 3	2%
Input Validation 233	20%
Logging 33	4%

* % Vulnerability values are based on typical use scenarios for this product. Your Vulnerability Values may be higher or lower depending on your software security needs:

	Usage	Intranet	Internet
Cross Site Scripting	Less Than	10	5
Reflected	Less Than	10	5
Stored	Less Than	10	5
SQL Injection	Less Than	20	2
Buffer Overflow	Less Than	20	2
Security Mechanisms		10	14
Encryption		3	15

How to… really fix security issues

Design platforms, libraries, and components in such a way that vulnerabilities cannot exist.

Then use them.

Start small Do one thing, no matter how small. Repeat.

Kaizen Philosophy of continuous improvement.

Kaizen Continuous small improvements will yield large compound improvement over time.

Start small In your current project, make all new work secure.

Start small In your next project, replace as many insecure components and practices as possible.

Start small Think about how to solve a known security problem. Think some more next week. Help solve it.

Start small Reach out and inspire someone else to do start small.

Start small Find an influential person. Inspire her.

Start small Become an influential person. Join a popular open source project, or an important company.

Change the world.

Summary What we can do:
1) change ourselves, 2) contribute to the body of knowledge, 3) inspire others, and 4) make a difference.

Example We need to transition to a world without plain-text protocols. How? Start by fixing SSL.

Example: Fixing SSL (1)

Performance 1) Improve protocols to address latency issues, 2) major sites support improvements, 3) one browser gets a performance edge, 4) other browsers follow.

Google is already doing this, and we should help them.

Example: Fixing SSL (2)

No support for modern TLS features 1) Realise that the underlying libraries are lacking, 2) understand why, 3) fund development, and 4) continue funding development.

Example: Fixing SSL (3)

Bad configuration 1) Raise awareness (but that won't work), 2) target library developers to drop obsolete features, 3) target vendors to ship with secure defaults

Example: Fixing SSL (4)

Virtual SSL hosting 1) Realise that we won't get virtual SSL hosting until Windows XP is retired, 2) put pressure on Microsoft to change their mind, 3) find one person at Microsoft who can change things.

Example: Fixing SSL (5)

Certificate authority trust issues 1) Wait for a wide adoption of DNSSEC, 2) put certificates into DNS, and 3) improve browser user interfaces.

Example: Fixing SSL (6)

Plain-text support issues 1) Use SRV records to enable sites to opt-out from supporting HTTP, then 2) support SRV records in web browsers, and 3) use Strict Transport Security in the meantime.

Message for today Do one thing, no matter how small.

Repeat.

Thank you!

The slides will be available for download
from **http://blog.ivanristic.com**

QUALYS®