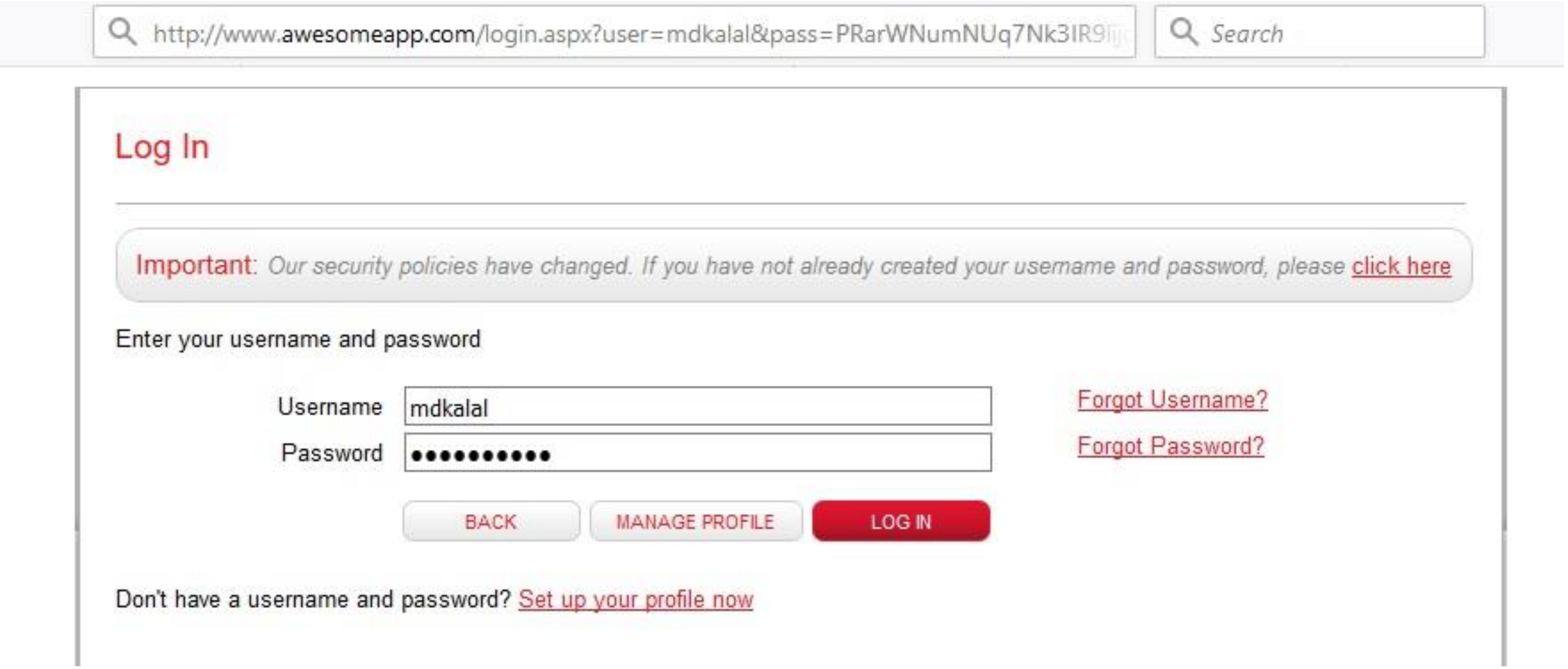
Secure Coding

Examples



http://www.awesomeapp.com/login.aspx?user=mdkalal&pass=PRarWNumNUq7Nk3IR9

Search

Log In

Important: *Our security policies have changed. If you have not already created your username and password, please* click here

Enter your username and password

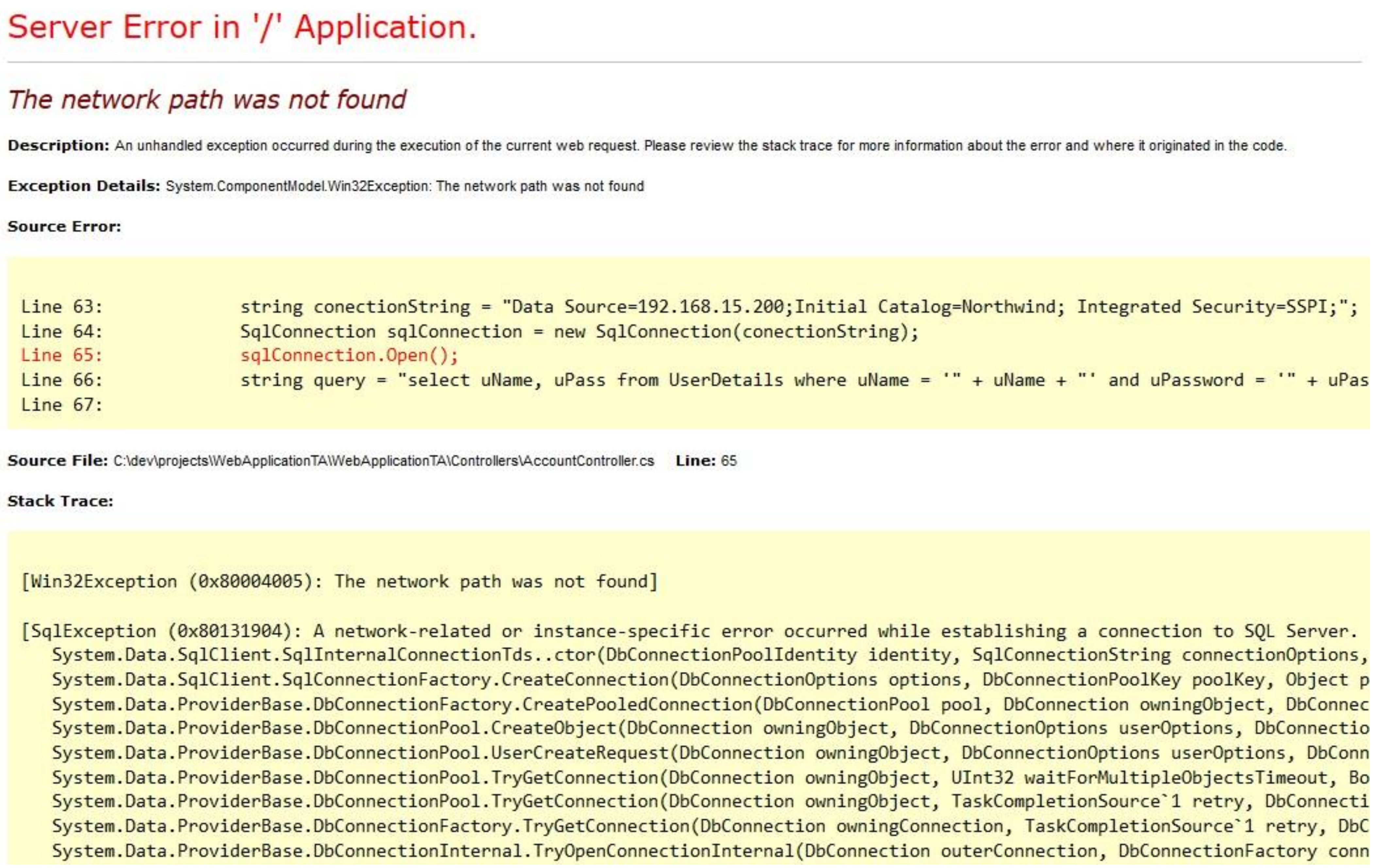
Username: mdkalal

Password: ●●●●●●●●●●

Forgot Username?

Forgot Password?

Don't have a username and password? Set up your profile now



Server Error in '/' Application.

The network path was not found

Description: An unhandled exception occurred during the execution of the current web request. Please review the stack trace for more information about the error and where it originated in the code.

Exception Details: System.ComponentModel.Win32Exception: The network path was not found

Source Error:

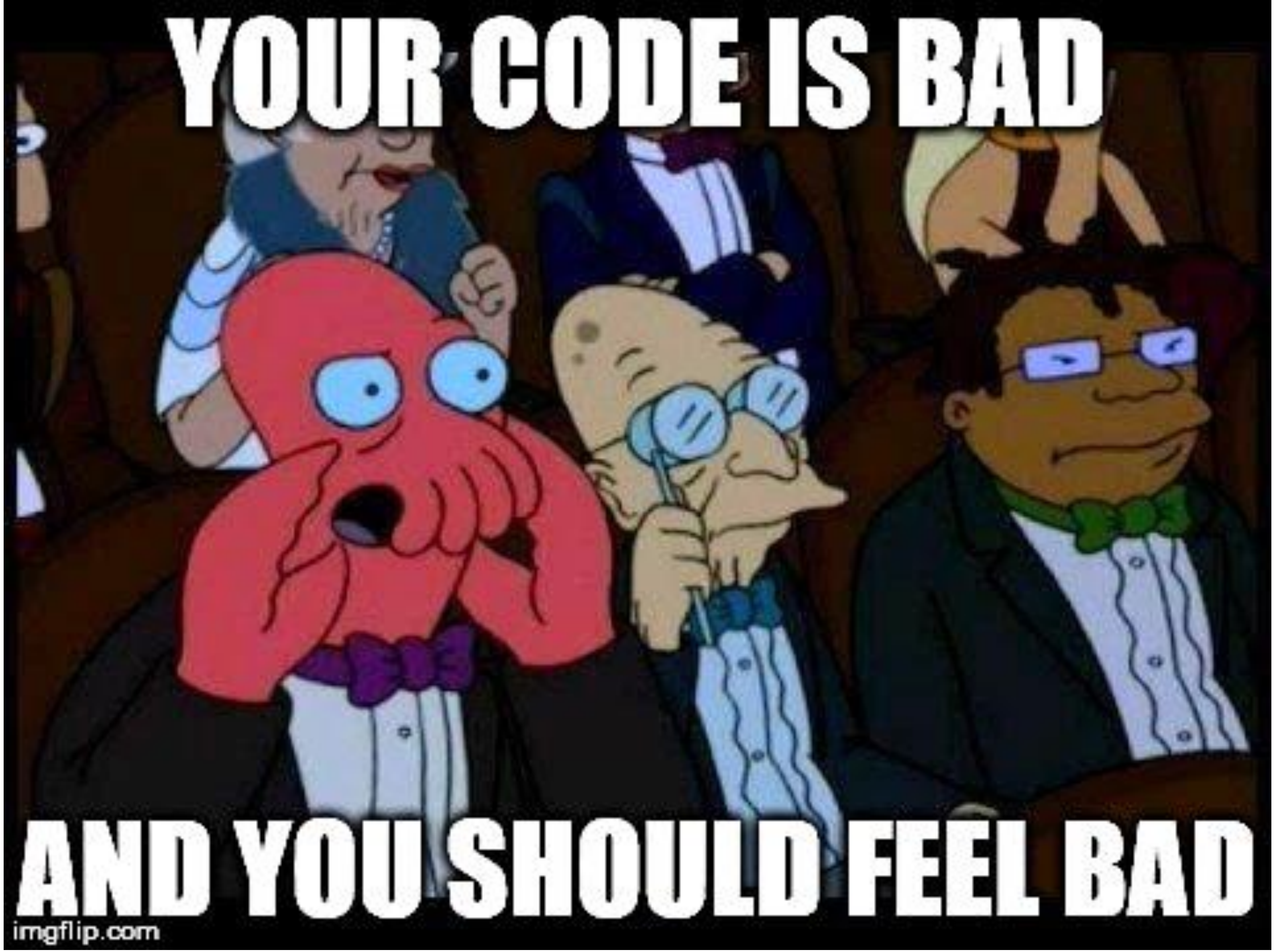
```
Line 63:            string conectionString = "Data Source=192.168.15.200;Initial Catalog=Northwind; Integrated Security=SSPI;";
Line 64:            SqlConnection sqlConnection = new SqlConnection(conectionString);
Line 65:            sqlConnection.Open();
Line 66:            string query = "select uName, uPass from UserDetails where uName = '" + uName + "' and uPassword = '" + uPas
Line 67:
```

Source File: C:\dev\projects\WebApplicationTA\WebApplicationTA\Controllers\AccountController.cs **Line:** 65

Stack Trace:

```
[Win32Exception (0x80004005): The network path was not found]

[SqlException (0x80131904): A network-related or instance-specific error occurred while establishing a connection to SQL Server.
   System.Data.SqlClient.SqlInternalConnectionTds..ctor(DbConnectionPoolIdentity identity, SqlConnectionString connectionOptions,
   System.Data.SqlClient.SqlConnectionFactory.CreateConnection(DbConnectionOptions options, DbConnectionPoolKey poolKey, Object p
   System.Data.ProviderBase.DbConnectionFactory.CreatePooledConnection(DbConnectionPool pool, DbConnection owningObject, DbConnec
   System.Data.ProviderBase.DbConnectionPool.CreateObject(DbConnection owningObject, DbConnectionOptions userOptions, DbConnectio
   System.Data.ProviderBase.DbConnectionPool.UserCreateRequest(DbConnection owningObject, DbConnectionOptions userOptions, DbConn
   System.Data.ProviderBase.DbConnectionPool.TryGetConnection(DbConnection owningObject, UInt32 waitForMultipleObjectsTimeout, Bo
   System.Data.ProviderBase.DbConnectionPool.TryGetConnection(DbConnection owningObject, TaskCompletionSource`1 retry, DbConnecti
   System.Data.ProviderBase.DbConnectionFactory.TryGetConnection(DbConnection owningConnection, TaskCompletionSource`1 retry, DbC
   System.Data.ProviderBase.DbConnectionInternal.TryOpenConnectionInternal(DbConnection outerConnection, DbConnectionFactory conn
```


YOUR CODE IS BAD
AND YOU SHOULD FEEL BAD
imgflip.com

Outline

- What is secure coding?
- Philosophy and practice
- Common vulnerabilities / OWASP top ten
- Remediation and prevention
- Data protection – secure password hashing and storage
- Threat landscape

About me

- Mark Kalal
- Software development/technology solutions at AMT Consumer Services
- mark.kalal@amtrustgroup.com or mdkalal@gmail.com

Secure Coding

- Awareness and techniques to prevent the introduction of vulnerabilities in application code
- Making individual apps secure, rather than relying on a firewall or network/server/container “perimeter”
- A way to help developers avoid doing the sort of things that the bad guys exploit

So what?

- Insecure code is potentially responsible for any variety of system attacks and data breaches
- Secure coding and testing is challenging
- Important for all developers, but critically so for those who develop applications that store/process sensitive information.

CYBEREDGE GROUP

2017 CYBERTHREAT DEFENSE REPORT

SURVEY DEMOGRAPHICS

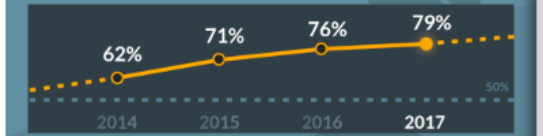
15 Countries represented around the world

19 Industries represented

1,100 Qualified IT security decision makers & practitioners

RISING CYBERATTACKS

The percentage of respondents affected by successful attacks has risen the last three years with no end in sight.



SUSCEPTIBLE NATIONS

The percentage of respondents affected by successful attacks in 2016 varied by nation.

China	96%
Mexico	94%
Colombia	87%
South Africa	83%
UK	82%
Turkey	82%
USA	81%
France	79%
Australia	76%
Germany	74%
Canada	74%
Singapore	73%
Saudi Arabia	69%
Japan	66%
Brazil	65%

INCREASING SECURITY BUDGETS

Although three in four IT security budgets are increasing in 2017, the percentage of growing budgets varies by industry.

Retail	83%
Finance	82%
Manufacturing	78%
Health Care	77%
Telecom & Technology	75%
Education	74%
Government	47%

SECURITY'S WEAKEST LINKS

These areas are rated as most difficult to secure...

1. Mobile devices
2. Laptops
3. Desktops

CYBERTHREAT HEADACHES

Cyberthreats causing the greatest concern include...

1. Malware (viruses, worms, Trojans, ransomware)
2. Phishing/ spear-phishing attacks
3. Insider threats/ data exfiltration by employees

HELD HOSTAGE BY RANSOMWARE

Three in five respondents indicated their organization was victimized by ransomware last year. Only one in three victims actually paid the ransom.

SECURITY'S BIGGEST OBSTACLES

These obstacles inhibit IT from defending cyberthreats...

1. Low security awareness among employees
2. Lack of skilled personnel
3. Too much data to analyze

NETWORK SECURITY ACQUISITIONS

The top four network security technologies targeted for acquisition in 2017 are...

1. Honeypots / network deception
2. Next-generation firewall (NGFW)
3. User & entity behavior analytics (UEBA)
4. Threat intelligence service

ENDPOINT SECURITY ACQUISITIONS

The top four endpoint security technologies targeted for acquisition in 2017 include...

1. Containerization/ micro-virtualization
2. Self-remediation for infected endpoints
3. Endpoint deception
4. Digital forensics/ incident resolution

CHALLENGING IT SECURITY FUNCTIONS

The most-challenging internal IT security functions are...

1. Application development & testing (SDLC)
2. Attack surface reduction
3. User security awareness/education

RESEARCH SPONSORS

PLATINUM: CODE42, IMPERVA, SecureWorks, Symantec

GOLD: bitglass, exabeam, Hewlett Packard Enterprise, WEBROOT Smarter Cybersecurity

SILVER: ENDGAME., foxtechnologies, illusive, Soliton, sumologic

Philosophy

Philosophy

- “Security is an illusion”
- Work is never done, landscape constantly changes
- Consider potential vulnerabilities a risk, rather than a weakness
- Tenets include:
    - Untrusted data – never assume data/user input is “good”
    - Strictest level of permission, fewest users
    - Untrusted sources - anonymous external users as well as users with their own accounts may attempt to compromise data/system. Also consider insiders wanting to disguise their actions

“Security in IT is like locking your house or car – it doesn't stop the bad guys, but if it's good enough they may move on to an easier target.”

— Paul Herbka

Common Vulnerabilities

- Unvalidated input
    - Validate at the server – client code is more susceptible to manipulation
- Denial of service (resource exhaustion)
    - Typically resolved via firewall, proxy servers
- Insecure communications
    - HTTPS vs HTTP

Select OWASP Top Ten vulnerabilities

- Injection flaws
- Broken session / authentication
- Sensitive data exposure
- Security misconfiguration
- Cross-site scripting
- Using components with known vulnerability

Injection Flaws

- Attacker can trick application into executing commands
- Can occur when untrusted data is sent to an interpreter
    - From the user (query string, form post)
    - From a browser (cookies, request headers)
    - External services
    - Even your own database
- Vulnerable applications
    - User supplied data is not validated
    - Dynamic queries or non-parameterized calls used without escaping

Injection cont

- Example
  - **http://sqlidemo.altervista.org/login1.php**

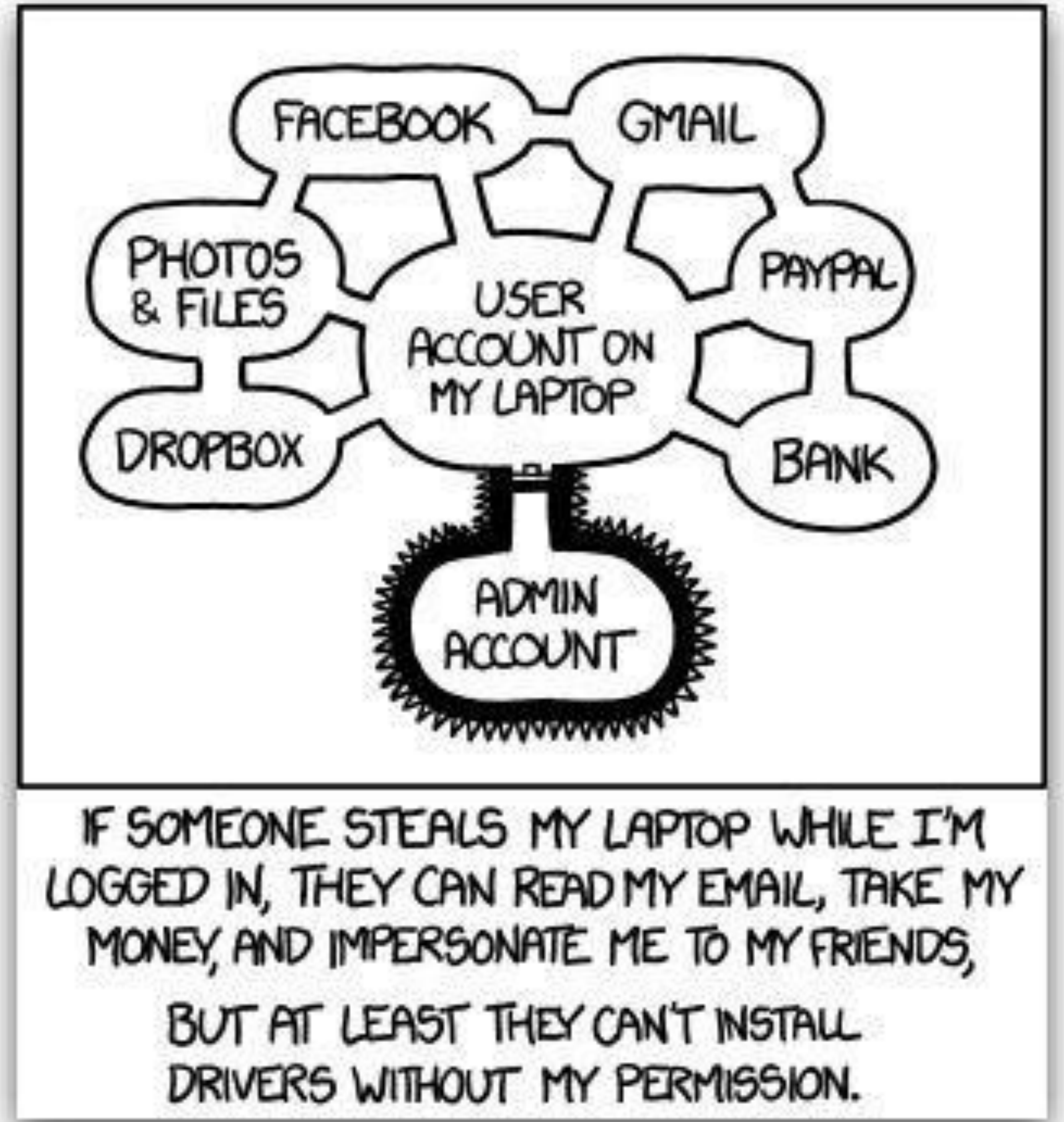
- Prevention – keep untrusted data separate from commands/queries
  - Parameterized API
  - Escape characters
  - Whitelist/blacklist validation

Broken Session/Authentication

- Authentication and session management may be done insecurely
- Can be flaws in things like login/logout, password management, timeouts, etc
- Vulnerable applications
    - Credentials aren’t protected when stored, or can be guessed or overwritten
    - Session ID’s or sensitive data is exposed in the URL
    - Sessions don’t timeout

Broken Session cont

- Example
  - Shopping site-
    http://awesomeshop.com/saleitems;jsessionid=2P0OCETSNDLPKHCJUN2V?item=stereo
    Authenticated user emails url to his friends, without knowing his friends may use his session/credit card.
  - Application does not timeout, user on public computer doesn’t close browser
  - Attacker gains access to password store, unprotected passwords are compromised
- Prevention
  - Strong session/authentication controls (timeout, multi factor authentication)


FACEBOOK
GMAIL
PHOTOS & FILES
PAYPAL
USER ACCOUNT ON MY LAPTOP
DROPBOX
BANK
ADMIN ACCOUNT
IF SOMEONE STEALS MY LAPTOP WHILE I'M LOGGED IN, THEY CAN READ MY EMAIL, TAKE MY MONEY, AND IMPERSONATE ME TO MY FRIENDS,
BUT AT LEAST THEY CAN'T INSTALL DRIVERS WITHOUT MY PERMISSION.

Sensitive Data Exposure

- PII, PHI, payment cards, and similar are normally considered sensitive. Attackers can gain access if data is not protected.

- Vulnerable applications
  - Sensitive data stored unencrypted/clear text (including backup)
  - Data unencrypted in transit
  - Weak encryption used

Sensitive Data Exposure cont

- Example
  - Website doesn’t use SSL/TLS for all pages, attacker monitors traffic, and data is intercepted
  - Passwords in a table aren’t hashed properly, attacker gains access to file and passwords are compromised
  - Credit card numbers are encrypted at rest, but in clear text in use. SQL injection flaw used to retrieve card numbers
- Prevention
  - Encrypt all sensitive data at rest and in transit
  - Don’t store what you don’t need, discard as soon as possible
  - Key management, correct password storage
  - Mask data entry, disable auto-complete

Security Misconfiguration

- Entire system should be configured specific to use. The platform, server, database, framework, and code can have default settings, enabled guest accounts, unneeded accounts, unused pages, unprotected files, etc.
- Error messages and other user output can reveal system details
- Vulnerable applications
  - Out of date software/patches
  - Unnecessary features/accounts installed or enabled (incl ports, services, privileges, etc)
  - Error handling reveal system details, stack traces, overly informative messages

Security Misconfiguration cont

- Example
    - An app server admin console is installed by default, and not removed. Default accounts remain. Attacker discovers the admin console, logs in with default admin, and takes over
    - Directory listing is not disabled on server. Attacker locates and downloads program code
    - Application error message returns stack trace to user, and exposes libraries used which contain known flaws
- Prevention
    - All environments (Prod, Dev, QA) configured the same (different passwords)
    - Keep abreast of patches and updates
    - Sensible software design/architecture (no needless dependencies)

Cross Site Scripting (XSS)

- Attacker injects code/scripts into web pages, which can be executed on the victims browser.
- Multiple types – Reflected, Stored, etc
- Vulnerable applications
  - User supplied data is not validated
  - Not using parameterized API
- Can be used to hijack user sessions, deface websites, insert hostile content, redirect users, etc.

XSS cont

- Example
  - https://xss-game.appspot.com/

- Prevention
  - Escape untrusted data / HTML encoding
  - Microsoft AntiXSS libraries
  - Blacklist/whitelist HTML “sanitation”
  - Manual code review, automated testing
  - Content Security Policy (CSP)

Using components with known vulnerabilities

- Vulnerabilities and exploits can be found in components / framework libraries.

- Vulnerable applications
  - Potentially any application, since most use components/libraries
  - Unknown versions/out of date components

Components cont

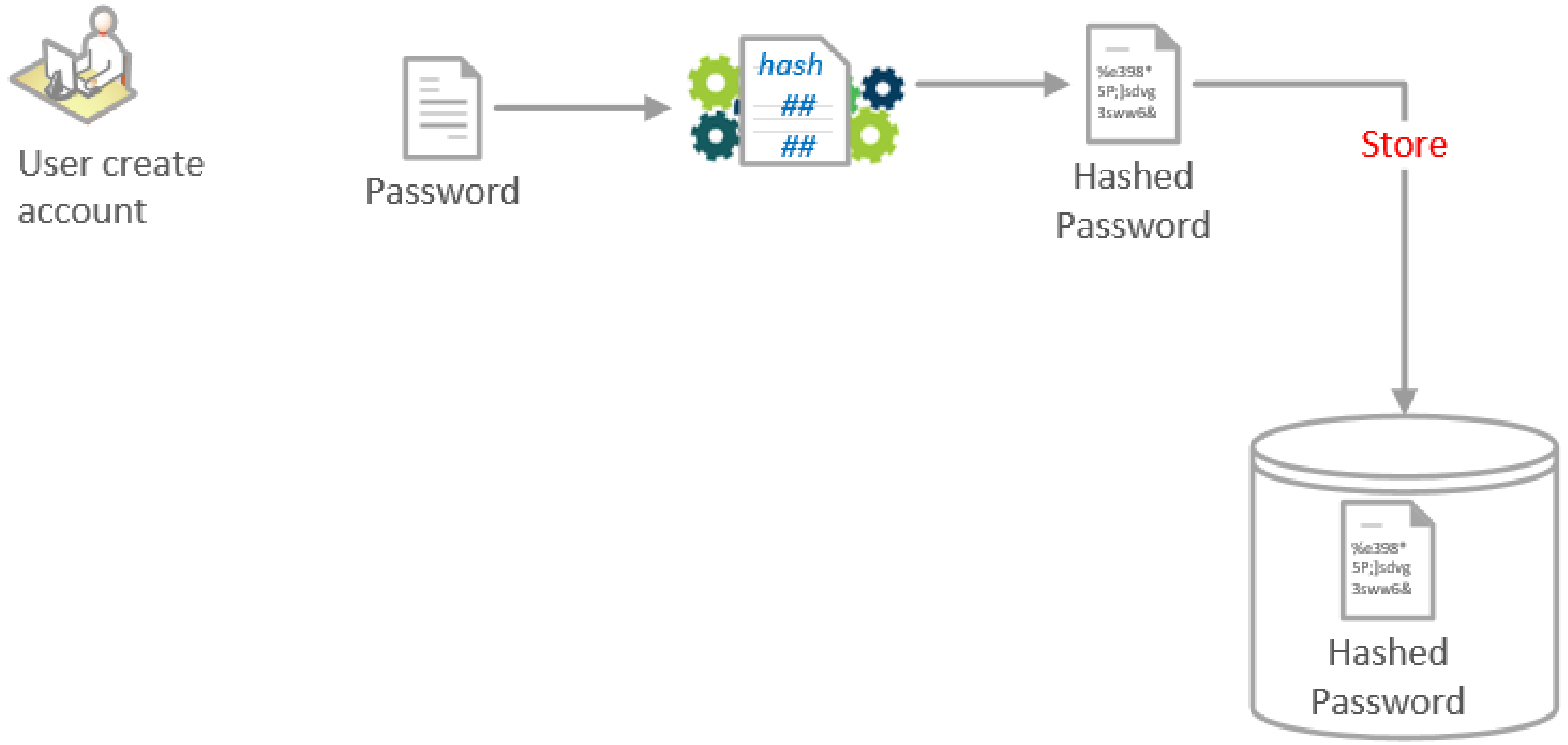
- Prevention
  - Keep versions current
  - Monitor updates on public sources
    - https://cve.mitre.org/, https://nvd.nist.gov/
  - Package / dependency manager
  - Establish security focused policies – evaluation prior to implementation, remove unused dependencies, disable unused features, etc.

Remediation / Prevention

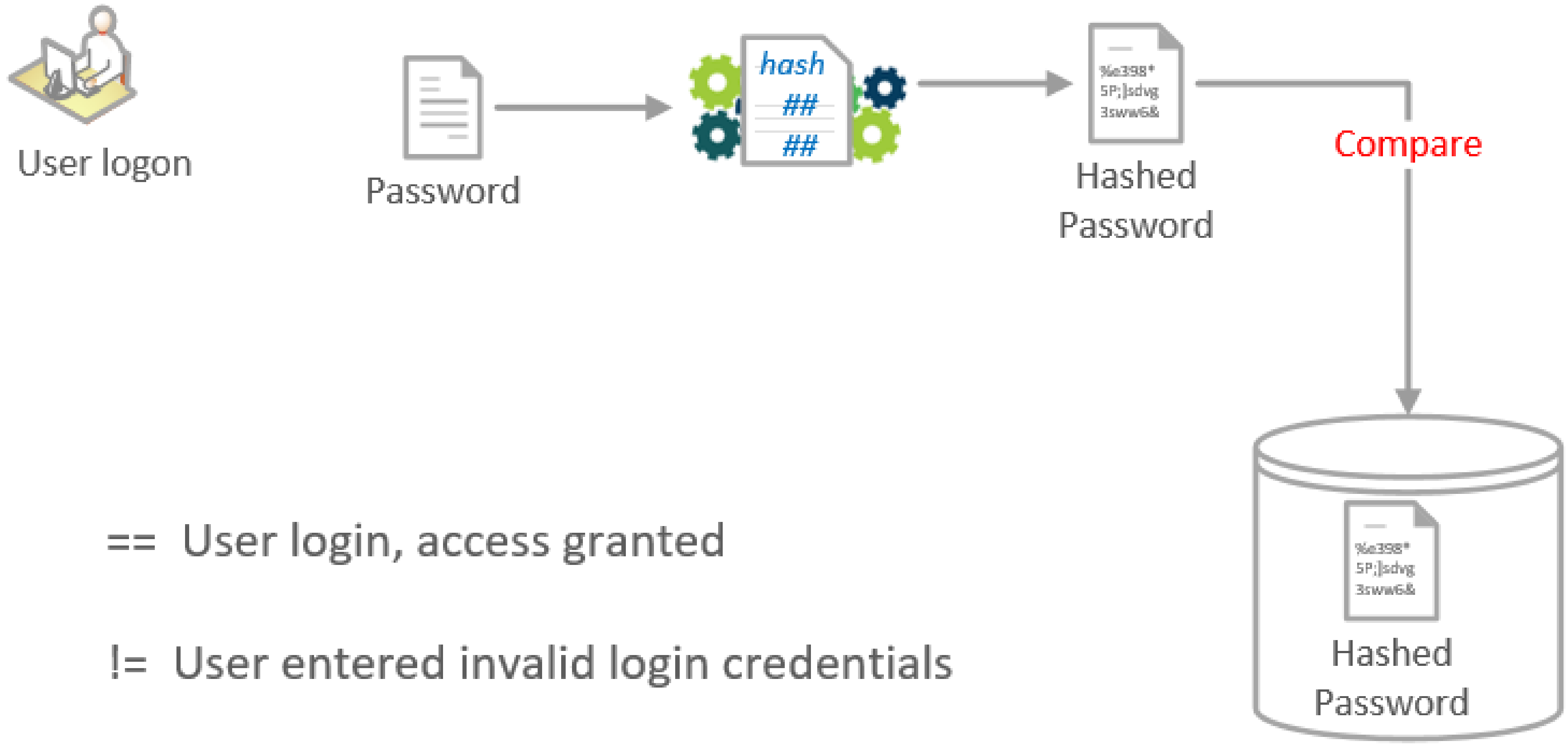
- Education and awareness
- Vulnerability scanning / penetration testing
- Manual review – human experts with good tools
- Two factor authentication
- Security as part of each SDLC phase (security requirements, test for vulnerabilities, security focused code reviews)
- Build and maintain your own list of security requirements (e.g. Common Weakness Enumeration - https://cwe.mitre.org). OWASP top ten probably does not cover all threats that are relevant to your applications.

Secure password hashing and storage

General workflow – create account



General workflow – login



== User login, access granted

!= User entered invalid login credentials

About hashing

- What is hashing?

```
hash("VandaliaExit") = iZ4y693raOVtW8PlHjZv0WNPc5b9e29
hash("mark")         = 7SVBDXMeeazvFj2RpXA5dIRd2fad6af
hash("m6rk")         = tKZ+mHbdYBtRryg1SEDHpuLI32bc6bd
```

- Best practice is to use only cryptographic hash functions (System.Security.Cryptography)
  - PBKDF2, bcrypt, scrypt

- How to “crack” hash
  - Brute Force
  - Dictionary attack / lookup tables / rainbow tables

About hashing cont

- The answer is Salt

```
hash("M6rk")                      = 2cf24dba5fb0a30e26e83b2ac5b9e29e
hash("M6rk" + "QxLUF1bgIAdeQX") = 9e209040c863f84a31e719795b257752
hash("M6rk" + "bv5PehSMfV11Cd") = d1d3ec2e6f20fd420d50e2642992841d
```

- Salt should be random, long, and unique for each password. Don’t reuse salt!
- Best practice is “CSPRNG” (e.g RNGCryptoServiceProvider)
- Salt can be stored with password, and needs to be re-generated when password changes

About hashing cont

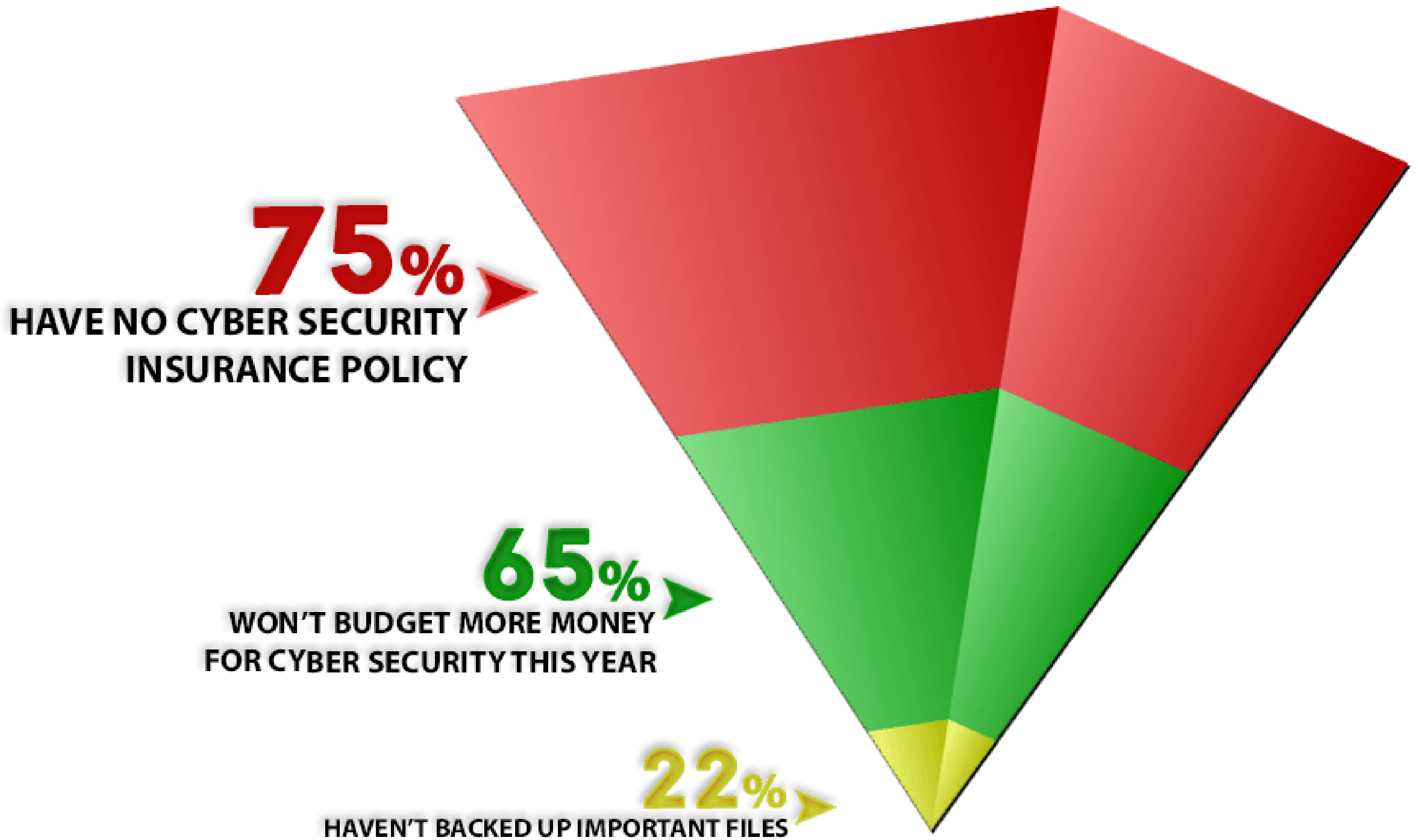
- For web apps – always hash on server
- Use “slow” hashing functions (key stretching)
- Other security measures
  - “keyed” hashes / two factor hashing
  - Penetration testing
  - Monitoring / intrusion detection

Threat Landscape

- Social engineering, phishing are major concerns
- People are the “low hanging fruit”
- Ransomware seems to be the next thing
- Small business seem to be a larger target

Small Business Budgets Little for Cyber Security

As Attacks Rise, Preparedness Not a Priority Just Yet, Either



Small Business TRENDS

Source: IDT911
More Charts: http://sbt.me/charts

Resources

- OWASP - https://www.owasp.org/index.php/Main_Page
- Top Ten - https://www.owasp.org/images/7/72/OWASP_Top_10-2017_%28en%29.pdf.pdf
- Common vulnerabilities and exposures - https://cve.mitre.org/
- Common weakness enumeration - https://cwe.mitre.org/
- National vulnerability database - https://nvd.nist.gov/
- Injection demo - http://sqlidemo.altervista.org
- XSS game - https://xss-game.appspot.com/
- .Net framework security https://docs.microsoft.com/en-us/dotnet/api/system.security.cryptography?view=netframework-4.7.1
- Password hashing - https://nakedsecurity.sophos.com/2013/11/20/serious-security-how-to-store-your-users-passwords-safely/

Questions