Real World Threat Modeling Using the PASTA Methodology

Tony Uceda Velez
Managing Partner, VerSprite
OWASP AppSec EU 2012
Why Threat Modeling?

Threat Dissection

Targeted Analysis

- Focused on understanding targeted attacks
- You can’t mitigate all of your threats
- So, what are your most likely threats to your published sites/services?
Taxonomy of Terms
Asset

**Asset.** An asset is a resource of value. It varies by perspective. To your business, an asset might be the availability of information, or the information itself, such as customer data. It might be intangible, such as your company's reputation.
Threat. A threat is an undesired event. A potential occurrence, often best described as an effect that might damage or compromise an asset or objective. Relative to each site, industry, company; more difficult to uniformly define.
**Vulnerability (Weakness)**

**Vulnerability.** A vulnerability is a weakness in some aspect or feature of a system that makes an exploit possible. Vulnerabilities can exist at the network, host, or application levels and include operational practices.
An attack is an action taken that utilizes one or more vulnerabilities to realize a threat.
Countermeasures

Countermeasure. Countermeasures address vulnerabilities to reduce the probability of attacks or the impacts of threats. They do not directly address threats; instead, they address the factors that define the threats.
Use Case.
Functional, as designed function of an application.
Abuse Case

Abuse Case.
Deliberate abuse of functional use cases in order to yield unintended results.
Attack Vector

Point & channel for which attacks travel over (card reader, form fields, network proxy, client browser, etc)
Attack Surface.
Logical area (browser stack, infrastructure components, etc) or physical area (hotel kiosk)
Actor (Threat Agent)

**Actor.** Legit or adverse caller of use or abuse cases.
Impact

Figure 1: Average per-record cost of a data breach, 2005–2008

Attack Tree

Diagram of relationship amongst asset-actor-use case-abuse case-vuln-exploit-countermeasure
What is PASTA?

Why should I eat this?

Current menu of application testing doesn’t provide a full security meal

- Pen Tests: Exploit driven
- Risk Assessments: Subjective; lacks meat
- Static Analysis: Weakness, flaw driven; disregards threats, narrow focus
- Vuln Scans: (C’mon! As if this could provide a decent meal!)
- Security testing deliverables are adversarial
- Integrated disciplines are needed via a unifying methodology

Better form of risk analysis w/ more substance
Encapsulates other security efforts

What is PASTA?

- Process for Attack Simulation & Threat Analysis
  - Integrated application threat analysis
  - Application threat modeling methodology
  - Risk or asset based approach; great for business integration
  - 7 stages building up to impact of threat to application & business.

- Aimed at addressing most viable threats to a given application target
Threat Modeling Comparisons

**MS Approach**

1. Identify Assets
2. Create an Architecture Overview
3. Decompose the Application
4. Identify the Threats
5. Document the Threats
6. Rate the Threats

**Process for Attack Simulation & Threat Analysis (PASTA)**

1. Define Biz Objectives
2. Define Tech Scope
3. App Decomposition
4. Threat Analysis
5. Vuln Detection
6. Attack Enumeration
7. Risk/ Impact Analysis
STAGE I
Define The Business & Security Objectives: “Capture requirements for the analysis and management of web based risks”
Stage 1 Walkthru – Understand Biz Objectives

(1.0) Stage I: Define the Objectives

(1.1) Define Business Objectives
(1.2) Define Security Requirements
(1.3) Define Compliance Requirements
(1.4) Perform preliminary Business Impact Analysis (BIA)

Input:
- Business requirement documents
- Functional requirement documents
- Information Security Policies
- Regulatory Compliance Standards
- Security Standards & Guidelines
- Data Classification Documents

Output:
- Description of the application functionality
- List of business objectives
- Definition of the application security and compliance requirements
- Business Impact Analysis Report
Business Objectives affect Web Apps

• Function req of supercookies (marketing)
  – Persistent storage of PII
• Easily accessible web services for internal APIs
  – ‘Internal’ lets security guard down w/ authentication
• Over-scoping of functional requirements
  – Orphaned features that lose maintenance
• Change Management System Web App Example
  – Biz Objective: Track & Manage Changes Across Groups; Easily accessible; Control Changes; Role based access
  – Discovered Threats/ Vulnerabilities: Internet accessible, elevation of privileges,
**Application Profile: Online Banking Application**

<table>
<thead>
<tr>
<th>General Description</th>
<th>The online banking application allows customers to perform banking activities such as financial transactions over the internet. The type of transactions supported by the application includes bill payments, wires, funds transfers between customer’s own accounts and other bank institutions, account balance inquires, transaction inquires, bank statements, new bank accounts loan and credit card applications. New online customers can register an online account using existing debit card, PIN and account information. Customers authenticate to the application using username and password and different types of Multi Factor Authentication (MFA) and Risk Based Authentication (RBA)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Application Type</td>
<td>Internet Facing</td>
</tr>
<tr>
<td>Data Classification</td>
<td>Public, Non Confidential, Sensitive and Confidential PII</td>
</tr>
<tr>
<td>Inherent Risk</td>
<td>HIGH (Infrastructure , Limited Trust Boundary, Platform Risks, Accessability)</td>
</tr>
<tr>
<td>High Risk Transactions</td>
<td>YES</td>
</tr>
<tr>
<td>User roles</td>
<td>Visitor, customer, administrator, customer support representative</td>
</tr>
<tr>
<td>Number of users</td>
<td>3 million registered customers</td>
</tr>
<tr>
<td>Project Business Objective</td>
<td>Security and Compliance Requirement</td>
</tr>
<tr>
<td>----------------------------------------------------------------------------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Perform an application risk assessment to analyze malware banking attacks</td>
<td>Risk assessment need to assess risk from attacker perspective and identify on-line banking transactions targeted by the attacks</td>
</tr>
<tr>
<td>Identify application controls and processes in place to mitigate the threat</td>
<td>Conduct architecture risk analysis to identify the application security controls in place and the effectiveness of these controls. Review current scope for vulnerability and risk assessments.</td>
</tr>
<tr>
<td>Comply with FACT Act of 2003 and FFIEC guidelines for authentication in the banking environment</td>
<td>Develop a written program that identifies and detects the relevant warning signs – or “red flags” – of identity theft. Perform a risk assessment of online banking high risk transactions such as transfer of money and access of Sensitive Customer Information</td>
</tr>
<tr>
<td>Analyze attacks and the targets that include data and high risk transactions</td>
<td>Analyze attack vectors used for acquisition of customers’PII, logging credentials and other sensitive information. Analyze attacks against user account modifications, financial transactions (e.g. wires, bill-pay), new account linkages</td>
</tr>
<tr>
<td>Identify a Risk Mitigation Strategy That Includes Detective and Preventive Controls/Processes</td>
<td>Include stakeholders from Intelligence, IS, Fraud/Risk, Legal, Business, Engineering/Architecture. Identify application countermeasures that include preventive, detective (e.g. monitoring) and compensating controls against malware-based banking Trojan attacks</td>
</tr>
</tbody>
</table>
Baking in GRC

• Serve as inherent countermeasures in the form of people, process, technology
  – Policies (for people)
  – Standards (for technology)

• Prior risk assessments help build app risk profile
  – Historical RAs provide prior risk profile of app

• Regulatory landscape taken into consideration, but not the driver
  – Key here is to not retrofit compliance; more costly

• Web Related Example:
  – Tech: Using Nessus OWASP template to audit for PHP & ColdFusion hardening guidelines
  – OWASP Input Validation Cheat Sheets
  – CIS Web Technology Benchmarks
STAGE II
Define The Technical Scope: “Defining the scope of technical assets/ components for which threat enumeration will ensue”
Stage 2 Walkthru – Define Tech Scope

(2.0) Stage II: Define the Technical Scope

- (2.1) Identify Application Boundaries
- (2.2) Identify Application Dependencies from Network Environment
- (2.3) Identify Application Dependencies from Servers/Infrastructure
- (2.4) Identify Application Dependencies from Software

Input: High level design documents, sketches from white-board exercises, network diagrams, logical and physical architecture diagrams, software and technical specifications

Output: High level, end-to-end view list of all protocols and data, list of all the application servers, list of all hosts and servers and type of software/technology dependencies, list of all network devices/appliances

Versprite
Navigate Beyond Risk
The Application Architecture Scope
Technical Scope Definition

• Define the scope from design artifacts:
  – Application components with respect to the application tiers (presentation, application, data)
  – Network topology
  – Protocol/services being used/exposed from/to the user to/from the back end (e.g. data flow diagrams)
  – Use case scenarios (e.g. sequence diagrams)

• Model the application in support of security architecture risk analysis
  – The application assets (e.g. data/services at each tier)
  – The security controls of the application (e.g. authentication, authorization, encryption, session management, input validation, auditing and logging)
  – Data interactions between the user of the application and between servers for the main use case scenarios (e.g. login, registration, query etc)

• End of this stage results in inherent countermeasures (people, process, technology)
STAGE III

Decompose the Application: “Identify the application controls that protect high risk web transactions sought by adversaries”
Stage 3 Walkthru – App Decomposition

(3.0) Stage III: Decompose the Application

- (3.1) Data Flow Diagramming & Trust Boundaries
- (3.2) Identify Users-Actors and their Roles-Permissions
- (3.3) Identify Assets, Data, Services, Hardware and Software
- (3.4) Identify Data Entry Points and Trust Levels

Input → Output

Architecture diagrams-design documents, Sequence diagrams, Use cases, Users, roles and permissions, Logical diagrams, Physical-network diagrams

Data Flow Diagrams
Access control matrix
List of assets including data and data sources
List of interfaces and trust levels
Mapping of use cases with actors and assets

VerSprite
Navigate Beyond Risk
What’s your web app cooking with?

- What/who are the actors?
- What calls do the actors make?
  - Key aspect of this phase
- Enumerate all use cases
- Define trust boundaries (implicit vs explicit trust)
  - Domains, networks, hosts, services, etc
- Further identify data sources and their relevant data flows
On-line Banking Application Data Flow Diagram (DFD) Example

- User/Browser
  - HTTPs Request
  - HTTPs Responses

- Web Server
  - Application Calls (.do)
  - Application Responses

- DMZ (User/Web Server Boundary)
  - Message XML/JMS
  - XML/HTTPS

- Application Server
  - XML/HTTPS
  - Service Message Response
  - SQL Query Call/JDBC
  - Auth Data

- Internal (Web Server/App & DB Server Boundary)
  - Messaging Bus

- Restricted Network (App & DB Server/Financial Server Boundary)
  - Financial Transaction Processing MainFrame
  - Financial Transactions (ACH, wires/external transfer)

- MFA RBA/Fraud Detection
  - XML/HTTPS

- Authentication Credential Store
# Transactional Security Control Analysis

## Online Banking Application Transaction Analysis

<table>
<thead>
<tr>
<th>Transaction</th>
<th>Risk</th>
<th>Data Classification</th>
<th>Security Functions Invoked</th>
</tr>
</thead>
<tbody>
<tr>
<td>Password Reset</td>
<td>HIGH</td>
<td>Sensitive</td>
<td>Debit Card, PIN, Account#</td>
</tr>
<tr>
<td>Username Recovery</td>
<td>HIGH</td>
<td>Sensitive</td>
<td>Debit Card, PIN, Account#</td>
</tr>
<tr>
<td>Registration</td>
<td>MEDIUM</td>
<td>Confidential PII &amp; Sensitive</td>
<td>Debit Card, PIN, Account#, PII (e.g. SSN), Demographics</td>
</tr>
<tr>
<td>Logon</td>
<td>HIGH</td>
<td>Confidential PII &amp; Sensitive</td>
<td>Username /Password</td>
</tr>
<tr>
<td>Wires</td>
<td>HIGH</td>
<td>Confidential PII &amp; Sensitive</td>
<td>Amount, Account#, IBAN/BIC</td>
</tr>
<tr>
<td>Bill Pay</td>
<td>HIGH</td>
<td>Confidential PII &amp; Sensitive</td>
<td>Amount, Payee Account#</td>
</tr>
</tbody>
</table>
STAGE IV

Threat Analysis:

“Identifying and extracting threat information from sources of intelligence to learn about threat-attack scenarios used by web focused attack agents”
Stage 4 Walkthru – Threat Intelligence/Analysis

(4.0) Stage IV: Analyze the Threats

(4.1) Analyze Probabilistic Attack Scenarios
(4.2) Analyze Incidents and Fraud-Case Management Reports
(4.3) Analyze Application Logs And Security Events
(4.4) Correlate Incidents and Fraud with Threat Intelligence

Input

Output

Threat agents and motives, Security incidents (SIRT) report, Fraud detection report, Secure incident event monitoring (SIEM) reports, Application and server logs, Threat intelligence reports

Attack scenario-landscape report
List of threat agents and attacks
Report on incidents-events relevant to the likelihood of threats and attack scenarios
Correlation to threat intelligence reports for attack scenarios

VerSprite
Navigate Beyond Risk
Threat Intelligence is Golden

• **Threat Enumeration Based upon Good Intel**
  - Threats based upon known intel
  - Prior assessment info (where applicable & useful)
  - Other application assessments from 3rd parties
  - SIEM feeds/ Syslog data/ Application Logs/ WAF logs
    - Denote attacks but will reveal overarching threats
  - Threat Intel/ Feeds
  - Security Operations/ Incident Reports
    - Personnel/ Infrastructure

• **Threat examples:**
  - IP Theft
  - Data Theft
  - Sabotage
  - Infrastructure compromise
  - Ransom
Threat Analysis Prefaces Attack Enumeration

• Threat analysis will lead to attack enumeration
  – PII theft
    – XSS
    – SQL Injection
    – MITM
  – Sabotage driven threats
    – CMS exploits to web application (Zope, Joomla, Mambo, etc)
    – FTP Brute Force attacks
    – iFrame Injection attacks
  – Malware upload
• Identify most likely attack vectors
  – Address entire application footprint (email, client app, etc)
  – Web Forms/ Fields
  – WSDLs/ SWF Objects
  – Compiled Libraries/ Named Pipes
STAGE V
Weakness and Vulnerabilities Analysis:
Analyzing the weaknesses and vulnerabilities of web application security controls
Stage 5 Walkthru – Vuln Analysis

Library of threat trees
Attack scenarios (from Stage IV)
Vulnerability Assessment reports Standards for vulnerability enumeration (MITRE CWE, CVE)
Standards for vulnerability scoring (CVSS,CWSS)

(5.0) Stage V: Vulnerabilities & Weaknesses Analysis

(5.1) Correlate Vulnerabilities to Application Assets
(5.2) Map Threat to Vulnerabilities Using Threat Trees
(5.3) Map Threat To Security Flaws Using Use and Abuse Cases
(5.4) Enumerate and Score Vulnerabilities

Map of existing vulnerabilities to the nodes of a threat tree
Enumeration of these vulnerabilities using CVE-CWE
Scoring of these using CVSS-CWSS
List of threats-attacks-vulnerabilities-assets

Versprite
Navigate Beyond Risk
Leveraging Web App Testing
MITRE CWE Cross-Section: 20 of the Usual Suspects

- Absolute Path Traversal (CWE-36)
- Cross-site scripting (XSS) (CWE-79)
- Cross-Site Request Forgery (CSRF) (CWE-352)
- CRLF Injection (CWE-93)
- Error Message Information Leaks (CWE-209)
- Format string vulnerability (CWE-134)
- Hard-Coded Password (CWE-259)
- Insecure Default Permissions (CWE-276)
- Integer overflow (wrap or wraparound) (CWE-190)
- OS Command Injection (shell metacharacters) (CWE-78)
- PHP File Inclusion (CWE-98)
- Plaintext password Storage (CWE-256)
- Race condition (CWE-362)
- Relative Path Traversal (CWE-23)
- SQL injection (CWE-89)
- Unbounded Transfer ('classic buffer overflow') (CWE-120)
- UNIX symbolic link (symlink) following (CWE-61)
- Untrusted Search Path (CWE-426)
- Weak Encryption (CWE-326)
- Web Parameter Tampering (CWE-472)
## Vulnerability/Weakness Classification

<table>
<thead>
<tr>
<th>WASC Threat Classification v2</th>
<th>OWASP Top Ten 2010 RC1</th>
</tr>
</thead>
<tbody>
<tr>
<td>WASC-19 SQL Injection</td>
<td>A1 - Injection</td>
</tr>
<tr>
<td>WASC-23 XML Injection</td>
<td></td>
</tr>
<tr>
<td>WASC-28 Null Byte Injection</td>
<td></td>
</tr>
<tr>
<td>WASC-29 LDAP Injection</td>
<td></td>
</tr>
<tr>
<td>WASC-30 Mail Command Injection</td>
<td></td>
</tr>
<tr>
<td>WASC-31 OS Commanding</td>
<td></td>
</tr>
<tr>
<td>WASC-39 XPath Injection</td>
<td></td>
</tr>
<tr>
<td>WASC-46 XQuery Injection</td>
<td></td>
</tr>
<tr>
<td>WASC-08 Cross-Site Scripting</td>
<td>A2 - Cross Site Scripting (XSS)</td>
</tr>
<tr>
<td>WASC-01 Insufficient Authentication</td>
<td>A3 - Broken Authentication and Session</td>
</tr>
<tr>
<td>WASC-18 Credential/Session Prediction</td>
<td></td>
</tr>
<tr>
<td>WASC-37 Session Fixation</td>
<td></td>
</tr>
<tr>
<td>WASC-47 Insufficient Session Expiration</td>
<td></td>
</tr>
<tr>
<td>WASC-01 Insufficient Authentication</td>
<td>A4 - Insecure Direct Object References</td>
</tr>
<tr>
<td>WASC-02 Insufficient Authorization</td>
<td></td>
</tr>
<tr>
<td>WASC-33 Path Traversal</td>
<td></td>
</tr>
<tr>
<td>WASC-09 Cross-Site Request Forgery</td>
<td>A5 - Cross-Site Request Forgery</td>
</tr>
<tr>
<td>WASC-14 Server Misconfiguration</td>
<td>A6 - Security Misconfiguration</td>
</tr>
<tr>
<td>WASC-15 Application Misconfiguration</td>
<td></td>
</tr>
<tr>
<td>WASC-02 Insufficient Authorization</td>
<td>A7 - Failure to Restrict URL Access</td>
</tr>
<tr>
<td>WASC-16 Denial of Service</td>
<td></td>
</tr>
<tr>
<td>WASC-11 Brute Force</td>
<td></td>
</tr>
<tr>
<td>WASC-21 Insufficient Anti-automation</td>
<td></td>
</tr>
<tr>
<td>WASC-04 Predictable Resource Location</td>
<td></td>
</tr>
<tr>
<td>WASC-38 URL Redirector Abuse</td>
<td>A8 - Unvalidated Redirects and Forwards</td>
</tr>
<tr>
<td>WASC-50 Insufficient Data Protection</td>
<td>A9 - Insecure Cryptographic Storage</td>
</tr>
<tr>
<td>WASC-04 Insufficient Transport Layer Protection</td>
<td>A10 - Insufficient Transport Layer Protection</td>
</tr>
</tbody>
</table>

### OWASP Top Ten 2010 RC1

<table>
<thead>
<tr>
<th>2010 Top 25</th>
</tr>
</thead>
<tbody>
<tr>
<td>A1 - Injection</td>
</tr>
<tr>
<td>A2 - Cross Site Scripting (XSS)</td>
</tr>
<tr>
<td>A3 - Broken Authentication and Session Management</td>
</tr>
<tr>
<td>A4 - Insecure Direct Object References</td>
</tr>
<tr>
<td>A5 - Cross Site Request Forgery (CSRF)</td>
</tr>
<tr>
<td>A6 - Security Misconfiguration</td>
</tr>
<tr>
<td>A7 - Failure to Restrict URL Access</td>
</tr>
<tr>
<td>A8 - Unvalidated Redirects and Forwards</td>
</tr>
<tr>
<td>A9 - Insecure Cryptographic Storage</td>
</tr>
<tr>
<td>A10 - Insufficient Transport Layer Protection</td>
</tr>
</tbody>
</table>

### CWE Numbers

- CWE-89 (SQL injection), CWE-78 (OS Command injection)
- CWE-306, CWE-307, CWE-798
- CWE-285
- CWE-352
- No direct mappings; CWE-209 is frequently the result of misconfiguration.
- CWE-285
- CWE-601
- CWE-327, CWE-311
- CWE-311
MITRE CWE Cross-Section: 22 More Suspects

• Design-Related
  • High Algorithmic Complexity (CWE-407)
  • Origin Validation Error (CWE-346)
  • Small Space of Random Values (CWE-334)
  • Timing Discrepancy Information Leak (CWE-208)
  • Unprotected Windows Messaging Channel ('Shatter') (CWE-422)
  • Inherently Dangerous Functions, e.g. gets (CWE-242)
  • Logic/Time Bomb (CWE-511)

• Low-level coding
  • Assigning instead of comparing (CWE-481)
  • Double Free (CWE-415)
  • Null Dereference (CWE-476)
  • Unchecked array indexing (CWE-129)
  • Unchecked Return Value (CWE-252)
  • Path Equivalence - trailing dot - 'file.txt.' (CWE-42)

• Newer languages/frameworks
  • Deserialization of untrusted data (CWE-502)
  • Information leak through class cloning (CWE-498)
  • .NET Misconfiguration: Impersonation (CWE-520)
  • Passing mutable objects to an untrusted method (CWE-375)

• Security feature failures
  • Failure to check for certificate revocation (CWE-299)
  • Improperly Implemented Security Check for Standard (CWE-358)
  • Failure to check whether privileges were dropped successfully (CWE-273)
  • Incomplete Blacklist (CWE-184)
  • Use of hard-coded cryptographic key (CWE-321)

... and about 550 more
STAGE VI
Attacks/Exploits Enumeration & Modeling
Stage 6 Walkthru – Attack Enumeration

Application Technical Scope (Stage II) and Application Decomposition (Stage III) Attack libraries-patterns List of threats, attacks and vulnerabilities to the application assets (Stage V)

(6.0) Stage VI: Model The Attacks

(6.1) Identify Application Attack Surface
(6.2) Derive Attack Trees For Threats and Assets
(6.3) Map Attack Vectors To Nodes of Attack Trees
(6.4) Identify Exploits and Attack Paths using Attack Trees

Application attack surface Attack trees with attack scenarios for targeted assets Attack tree mapping to vulnerabilities for impacted assets List of possible attack paths to exploits including the attack vectors

VerSprite
Navigate Beyond Risk
Analysis Of Attacks Using Attack Trees

Fraudster

- Upload Malware on Vulnerable Site
  - Drive-by Download / Malicious Ads

- Attack Victim's Vulnerable Browser
  - Phishing Email, FaceBook Social Engineering

  - Man In The Browser
    - Phish User To Click Link With Malware

  - Harvest Confidential Data / Credentials From Victim
    - Sends Steal Data To Fraudster's Collection Server

  - Modifies UI Rendered By The Browser
    - Remote Access To Compromised PC Through Proxy

  - Steals Keystrokes with Key-logger

- Upload Banking Malware on Customer's Pc

  - Sends Stolen Data to Fraudster's Collection Server

  - Logs into Victim's Online Bank Account

  - Perform Unauthorized Money Transfer to Mule

  - Use Stolen Banking Credentials, Challenge C/Q

- Delete Cookies Forcing to Login To Steal Logins

- Modify User's Account To rendition by the Browser

- Redirect User To Malicious Sites
## Attack Vectors Used By Different Types of Malware

<table>
<thead>
<tr>
<th>Trojan</th>
<th>Infection Method</th>
<th>Attack Capabilities</th>
<th>Timing</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>MB- MitB</td>
<td></td>
<td>MB</td>
<td>MM</td>
<td>B</td>
</tr>
<tr>
<td>MM-MitM</td>
<td></td>
<td>MB</td>
<td>MM</td>
<td>B</td>
</tr>
<tr>
<td>B-Both</td>
<td></td>
<td>MB</td>
<td>MM</td>
<td>B</td>
</tr>
<tr>
<td>O-Other</td>
<td></td>
<td>MB</td>
<td>MM</td>
<td>B</td>
</tr>
<tr>
<td>Zeus</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
</tr>
<tr>
<td>SpyEye</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
</tr>
<tr>
<td>InfoStealer</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
</tr>
<tr>
<td>SilentBanker</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
</tr>
<tr>
<td>URLZone</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
</tr>
<tr>
<td>Clampi/Bugat/Gozi</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
</tr>
<tr>
<td>Haxdoor</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
</tr>
<tr>
<td>Limbo</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
</tr>
</tbody>
</table>
Analysis of Web App Use and Abuse Cases

User
Login With UserID password over SSL
Includes
Trust connection by IP and machine tagging/browser attributes

Fraudster
Drops Banking Malware on victims/PC
Includes
Communicate with fraudster C&C
Includes

Enter One Time Password (OTP) to authenticate transaction
Includes
Enter Challenge Question (C/Q) to authenticate transaction

Key logger/From grabber captures keystrokes incl. credentials
Includes
Set IP with Proxy/MiTM to same IP gelocation of the victim
Includes
Hijacks SessionIDs, Cookies, Machine Tagging
Includes
Capture OTP on web channel and authenticate on behalf of the user
Includes
Capture C/Qs in transit and authenticate on behalf of user
Includes
Man In The Browser Injected HTML to capture C/Q
Includes

Threatens
Trust connection by IP and machine tagging/browser attributes

Threatens
Enter Challenge Question (C/Q) to authenticate transaction

Threatens
Man In The Browser Injected HTML to capture C/Q

Threatens
Set IP with Proxy/MiTM to same IP gelocation of the victim

Threatens
Key logger/From grabber captures keystrokes incl. credentials

Threatens
Include

Man In The Browser Injected HTML to capture C/Q
Includes

Communicate with fraudster C&C
Includes

Drops Banking Malware on victims/PC
Includes
Mapping Use Cases to Misuse Cases

User

Enter Username and password

Includes

User Authentication

Includes

Brute Force Authentication

Includes

Threatens

Mitigates

Application/Server

Validate Password

Minimum Length and Complexity

Includes

Show Generic Error Message

Valid User Accounts

Harvest (e.g. guess)

Dictionary Attack

Mitigates

Mitigates

Lock Account After N. Failed Login Attempts

Includes

Hacker/Malicious User
STAGE VII
Risk And Impact Analysis: Impact Analysis, Residual Risk, and Countermeasure Development
Stage 7 Walkthru – Residual Risk Analysis

Preliminary BIA (Stage I)
Technical Scope (Stage II)
Application Decomposition (Stage III)
Threat Analysis (Stage IV)
Vulnerability Analysis (Stage V)
Attack Analysis (Stage VI)
Mapping of attacks to controls
Technical standards for controls

(7.0) Stage VII: Risk and Impact Analysis

(7.1) Qualify and Quantify Business Impacts
(7.2) Identify Gaps in Security Controls
(7.3) Calculate Residual Risks
(7.4) Identify Risk Mitigation Strategies

Input

Output

Application risk profile
Quantitative and qualitative risks report
Threat matrix with threats, attacks, vulnerabilities, business impact
Residual risk to business
Risk mitigation strategy-options

VerSprite
Navigate Beyond Risk
Exploits beget countermeasures

- Unacceptable risks give way to countermeasure development
- Develop countermeasures based upon the net risk of an application environment at multiple levels
  - Baseline configuration
  - Design and programmatic controls
  - 3rd party software/ COTS
<table>
<thead>
<tr>
<th>Range &amp; Type</th>
<th>Problem Type</th>
<th>Description</th>
<th>Consequences</th>
<th>Exploitation</th>
<th>Prevention</th>
<th>Mitigation</th>
<th>Platform</th>
<th>Required Personnel</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Buffer Overflow</td>
<td>Buffer overflow condition exists when a program attempts to store data in a buffer that is too small. In this case, the buffer is a sequential section of memory allocated to contain anything from a character to a string of integers.</td>
<td>Availability: Buffer overflows can lead to crashes. Other attacks leading to lack of availability are possible, including putting the program in an infinite loop.</td>
<td>Design</td>
<td>Design: Mitigating technologies such as safer string libraries and container abstractions could be introduced.</td>
<td>Design: Use abstraction layers to abstract away low-level APIs. Not a complete solution.</td>
<td>Any</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Write-what-where condition</td>
<td>Write-what-where condition occurs when an attacker can write to a memory location, often as the result of a buffer overflow.</td>
<td>Access control (memory protection): Clearly, write-what-where conditions can be used to write to areas of memory outside the scope of a policy. Also, they can be used to execute arbitrary code, which is outside the scope of a program's implicit security policy.</td>
<td>Implementation</td>
<td>Implementation: Many logic errors can lead to this condition. It can be exacerbated by lack of control on inter-messaging technologies.</td>
<td>Operational: Use OS-level pre-emptive functionality, not a complete solution.</td>
<td>Any</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Write-what-where condition</td>
<td>Write-what-where condition occurs when an attacker can write to a memory location, often as the result of a buffer overflow.</td>
<td>Access control (memory protection): Clearly, write-what-where conditions can be used to write to areas of memory outside the scope of a policy. Also, they can be used to execute arbitrary code, which is outside the scope of a program's implicit security policy.</td>
<td>Design</td>
<td>Design: Many write-what-where problems are buffer overflows, and mitigating technologies for this subset of problems can be chosen at this time.</td>
<td>Design: Integrate technologies that try to prevent the consequences of this problem.</td>
<td>Any</td>
<td></td>
</tr>
</tbody>
</table>
The PASTA™ Recipe For Risk Eval of Web Apps

- Focus on **the application** as business-asset target
- Risk != $t \times v \times i$
- Risk! = $t \times v \times i \times p$

- Attack simulation enhances ($p$) probability coefficients
- Considers both inherent countermeasures & those to be developed
- Focused on minimizing risks to applications and associated impacts to business

$$R_{risk} = \left[\frac{(t \times v_p)}{c}\right] \times i$$
Bonus Round
<table>
<thead>
<tr>
<th>ID</th>
<th>Element Name</th>
<th>Element Type</th>
<th>Element Diagram References</th>
<th>Threat Type</th>
<th>Bug ID</th>
<th>Completion</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>Commands (User to My Process)</td>
<td>DataFlow</td>
<td>Context</td>
<td>Tampering</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>Commands (User to My Process)</td>
<td>DataFlow</td>
<td>Context</td>
<td>Information Disclosure</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>Commands (User to My Process)</td>
<td>DataFlow</td>
<td>Context</td>
<td>DenialOfService</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>Configuration (My Process to Data)</td>
<td>DataFlow</td>
<td>Context</td>
<td>Tampering</td>
<td></td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>Configuration (My Process to Data)</td>
<td>DataFlow</td>
<td>Context</td>
<td>Information Disclosure</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>Responses (My Process to User)</td>
<td>DataFlow</td>
<td>Context</td>
<td>Tampering</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>Responses (My Process to User)</td>
<td>DataFlow</td>
<td>Context</td>
<td>Information Disclosure</td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>Responses (My Process to User)</td>
<td>DataFlow</td>
<td>Context</td>
<td>DenialOfService</td>
<td></td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>Results (Data to My Process)</td>
<td>DataFlow</td>
<td>Context</td>
<td>Tampering</td>
<td></td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>Results (Data to My Process)</td>
<td>DataFlow</td>
<td>Context</td>
<td>Information Disclosure</td>
<td></td>
<td></td>
</tr>
<tr>
<td>14</td>
<td>Results (Data to My Process)</td>
<td>DataFlow</td>
<td>Context</td>
<td>DenialOfService</td>
<td></td>
<td></td>
</tr>
<tr>
<td>21</td>
<td>Data</td>
<td>DataStore</td>
<td>Context</td>
<td>Tampering</td>
<td></td>
<td></td>
</tr>
<tr>
<td>22</td>
<td>Data</td>
<td>DataStore</td>
<td>Context</td>
<td>Repudiation</td>
<td></td>
<td></td>
</tr>
<tr>
<td>23</td>
<td>Data</td>
<td>DataStore</td>
<td>Context</td>
<td>Information Disclosure</td>
<td></td>
<td></td>
</tr>
<tr>
<td>24</td>
<td>Data</td>
<td>DataStore</td>
<td>Context</td>
<td>DenialOfService</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>User</td>
<td>Interactor</td>
<td>Context</td>
<td>Spoothing</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>User</td>
<td>Interactor</td>
<td>Context</td>
<td>Repudiation</td>
<td></td>
<td></td>
</tr>
<tr>
<td>15</td>
<td>My Process</td>
<td>Process</td>
<td>Context</td>
<td>Spoothing</td>
<td></td>
<td></td>
</tr>
<tr>
<td>16</td>
<td>My Process</td>
<td>Process</td>
<td>Context</td>
<td>Tampering</td>
<td></td>
<td></td>
</tr>
<tr>
<td>17</td>
<td>My Process</td>
<td>Process</td>
<td>Context</td>
<td>Repudiation</td>
<td></td>
<td></td>
</tr>
<tr>
<td>18</td>
<td>My Process</td>
<td>Process</td>
<td>Context</td>
<td>Information Disclosure</td>
<td></td>
<td></td>
</tr>
<tr>
<td>19</td>
<td>My Process</td>
<td>Process</td>
<td>Context</td>
<td>DenialOfService</td>
<td></td>
<td></td>
</tr>
<tr>
<td>20</td>
<td>My Process</td>
<td>Process</td>
<td>Context</td>
<td>ElevationOfPrivilege</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Threat Analysis Using Threat Modeling Tool
Threat Modeling Web Apps via SDLC

- Asset based threat model is able to address inherent and new risks that should be mitigated based upon baseline of info.
  - Pen Tests, Risk Assessments, Compliance Audits, etc
  - Business Risk Mitigation Key
- Software based threat models will build upon understood threats to software environ
  - Comparable web apps, prior static/dynamic analysis, and other web app assessments
  - Safeguarding software integrity is key and fosters building security in
- Security centric threat model focused on security of web application environment
  - More focused on attack identification and applying countermeasures
- PMs, business analysts, business owners devise functional requirements (Definition Phase)
- Architects and IT Leaders speak to architectural design and platform solutions (Design Phase)
- Governance leaders inject compliance & standards requirements for during he design phase; BIA
- Threat Model* (SOC/ NOC fed), DFDs Introduced, Trust Boundaries defined, Countermeasures proposed

**Define**
- Biz Objectives
- The C Word

**Design**
- Security Arch
- Security Frameworks
- AntiSamy (Java, .NET)
- OWASP ModSecurity

**Develop**
- OWASP Top 10
- OWASP Development Guide
- ESAPI
- OWASP Dev Guide/OWASP .NET Project

**Test (QA)**
- ASVS (3rd Party Dev)
- OWASP Testing Guide (Internal)
The Beneficiaries of PASTA™

- Business managers can incorporate which security requirements that impact business
- Architects understand security/design flaws and how countermeasure protect data assets
- Developers understand how software is vulnerable and exposed
- Testers can use abuse cases to security tests of the application
- Project managers can manage security defects more efficiently
- CISOs can make informed risk management decisions
https://www.surveymonkey.com/s/Research12_TonyUVRS