



# Human vs Artificial intelligence – Battle of Trust

Hemil Shah

Co-CEO & Director

Blueinfy Solutions Pvt Ltd



**Blueinfy**

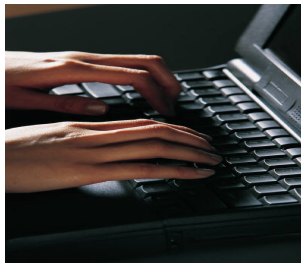
# About

- **Hemil Shah** – [hemil@blueinfy.net](mailto:hemil@blueinfy.net)
- **Position** - , Co-CEO & Director at BlueInfy Solutions, - Founder & Director, eSphere Security & ExtendedITArms, Member of advisory board on several security consulting companies, Speaker & trainer at different conferences
- **Blueinfy** – Professional Application Security Services Company
- **Blog** – [blog.Blueinfy.com](http://blog.Blueinfy.com)
- **Interest**
  - ♦ Application security research (Web & Mobile)
- **Published research**
  - ♦ Articles / Papers – Packstroem, etc.
  - ♦ Tools – DumpDroid, CheckDebugable, FSDroid, iAppliScan, wsScanner, scanweb2.0, AppMap, AppCodeScan, AppPrint etc.

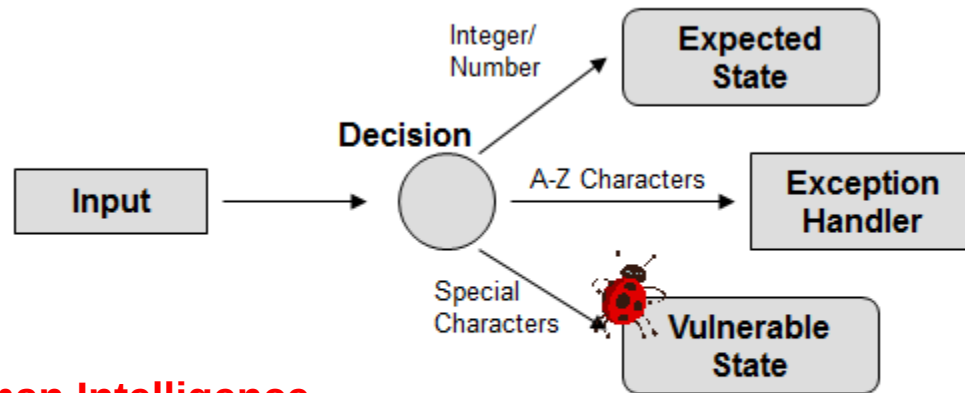
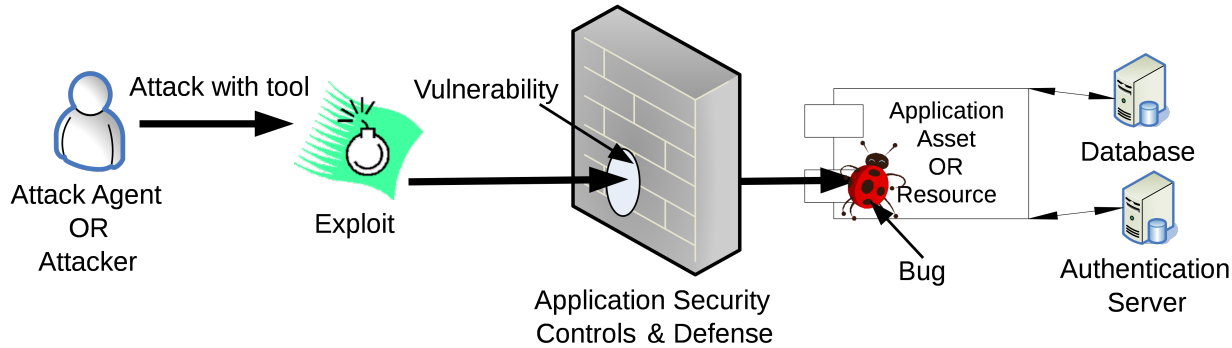


# Key Points

- Importance of manual application security reviews
- Unique cases used by humans to identify high value security vulnerabilities in applications
- Methodology of analyzing modern era applications to find complex security vulnerabilities
- Suggestions to protect against the vulnerabilities



# Vulnerabilities




## Human Intelligence

2 Medium = 1 Critical = Compromise

## Automated

2 Medium = 2 Medium

Potential Exploitation 



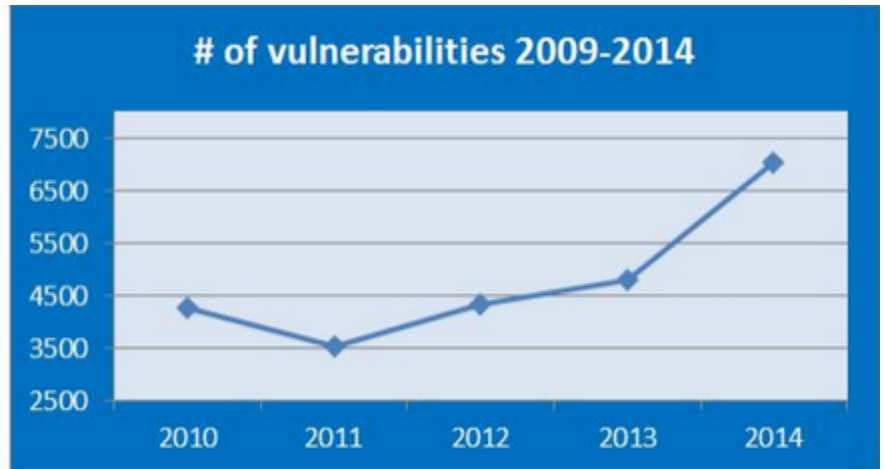
# Complex Structures

- Stack
  - ◆ NodeJS, Mashup driven, Cross Domain integrations, Web services, Webhooks, Browser Extensions, Entity frameworks etc
- Protocols
  - ◆ JSON, GWT, WebSockets, Custom, AMF 3.0, etc
- Database and File System
  - ◆ MongoDB, Hadoop, IndexDB, FileAPI, etc
- Client side
  - ◆ HTML5, AngularJS, ExpressJS, Knockout.js, etc

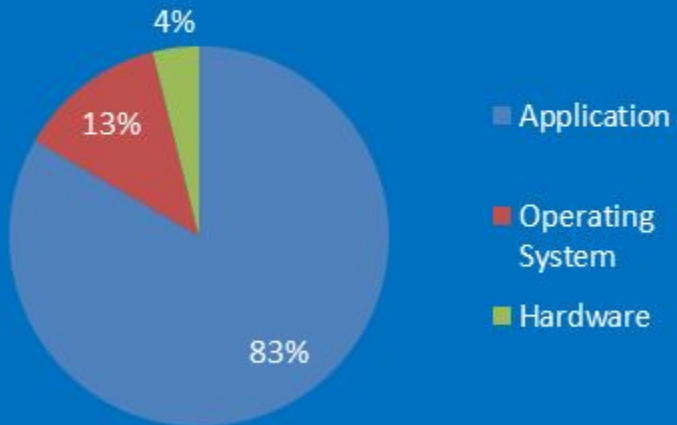


# Where are we standing?

Year	# of vulnerabilities
2010	4,258
2011	3,532
2012	4,347
2013	4,794
2014	7,038



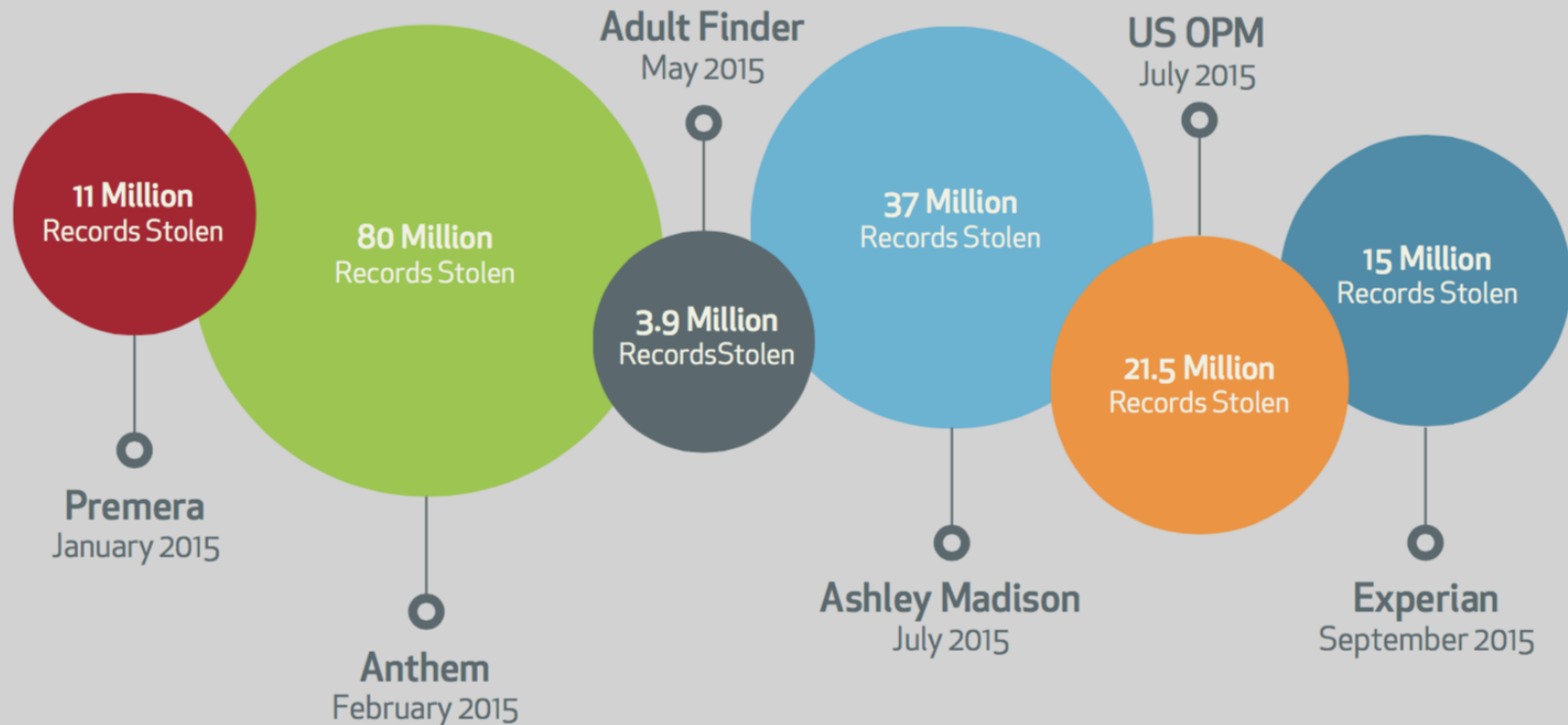
## Vulnerability distribution by product type - 2014



Source: [Most vulnerable operating systems and applications in 2014](#) by GFI Languard

# 2015 – A Year that has been

2015 has been a year of large security breaches:



The above are examples of significant system breaches resulting in millions of records and personal data being stolen.



Source – 2015 – Edgescan Status Report



# 2015 – A Year that has been

## SECURITY TESTING

**98% of applications tested were vulnerable**

---

**20: Median number of vulnerabilities per application (up six from 2013)**

---

**95% of mobile applications were vulnerable**

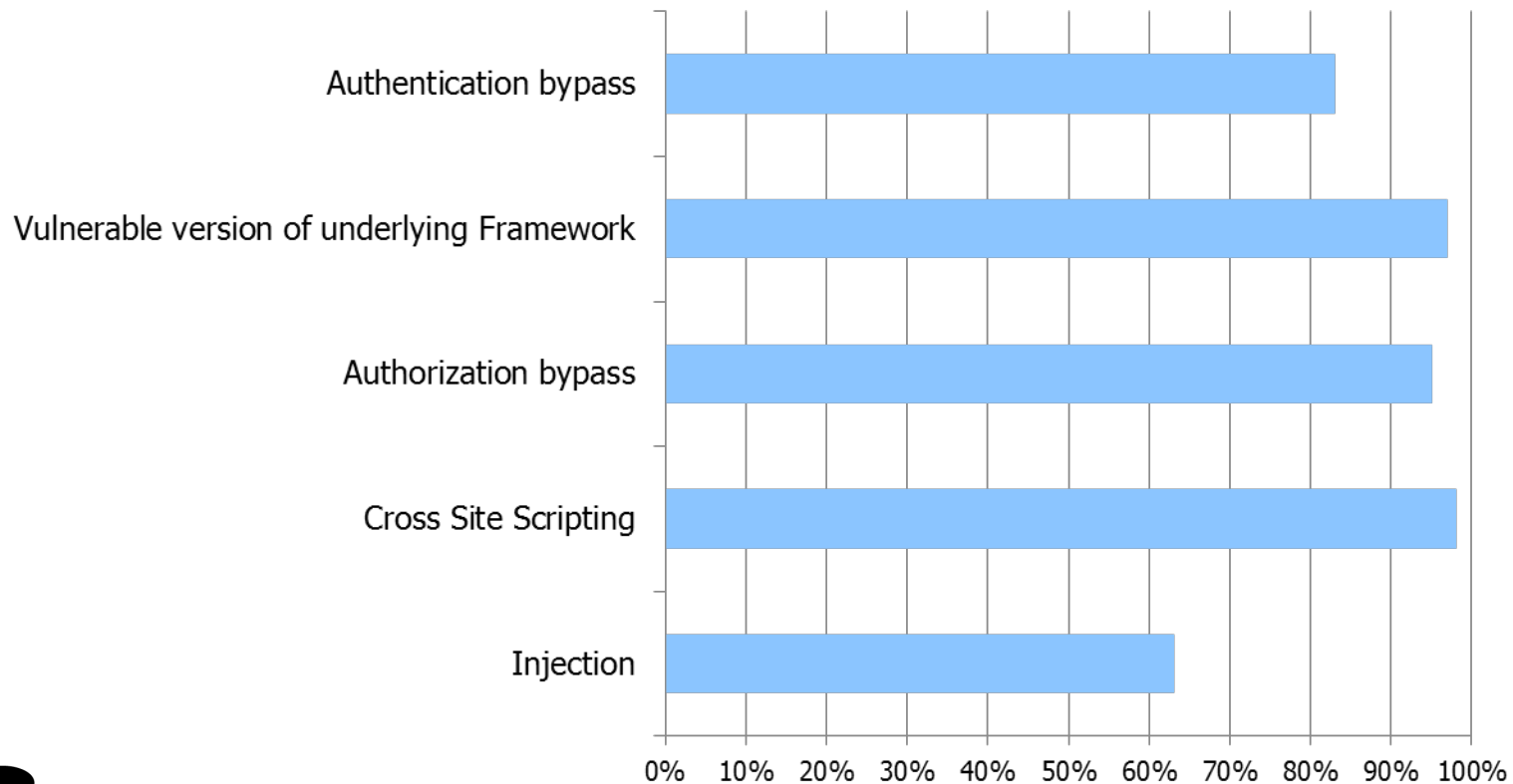
- 6.5: Median number of vulnerabilities per mobile application
- 35% had critical issues
- 45% had high-risk issues





# Top 5 Vulnerabilities

- Zero application with NO vulnerability in first review
- From the stats of Blueinfy's data –

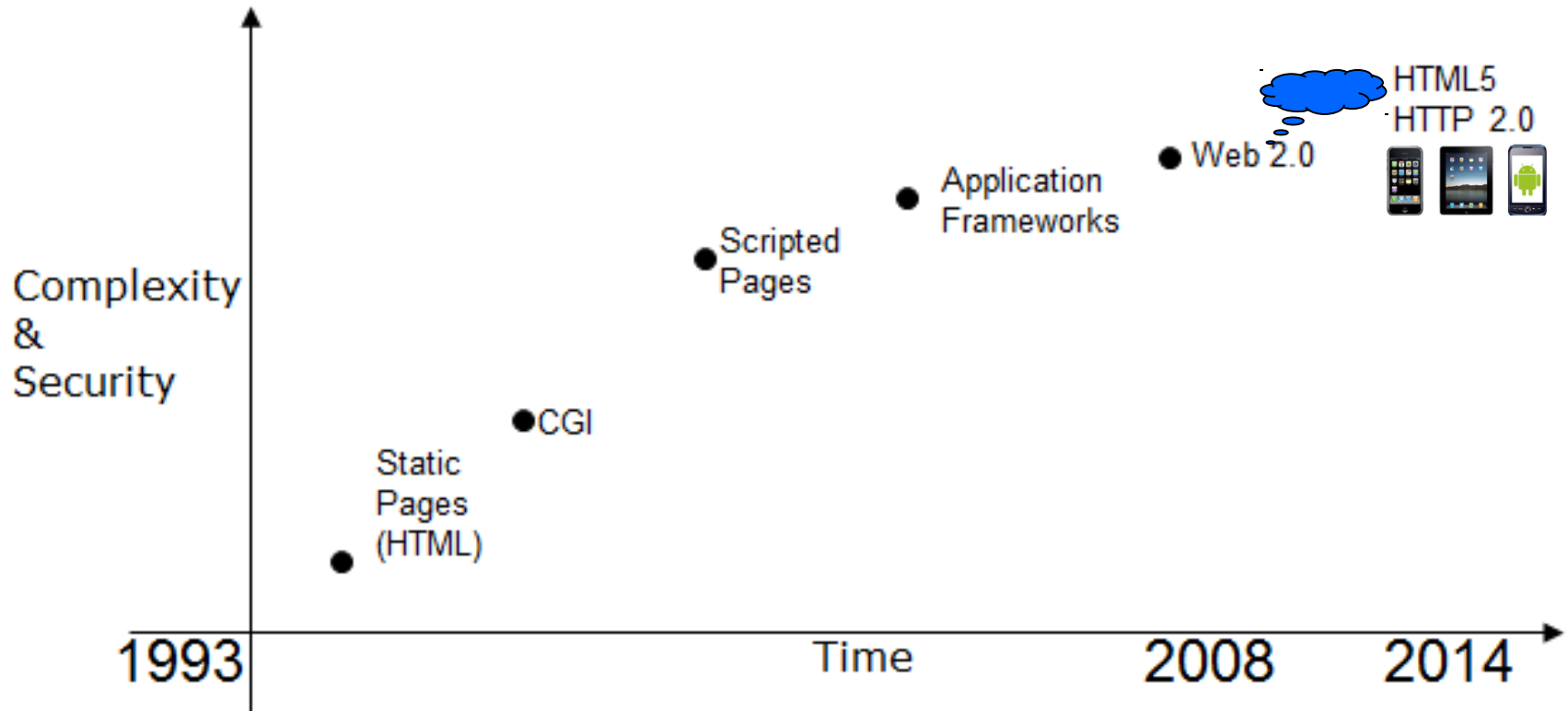


# Attacks and Hacks

- 98% applications are having security issues
- Web Application Layer vulnerabilities are growing at higher rate in security space
- Client side hacking and vulnerabilities are on the rise
- Web browser vulnerabilities are growing at higher rate
- End point exploitation shifting from OS to browser and its plugins



# Technology Trend



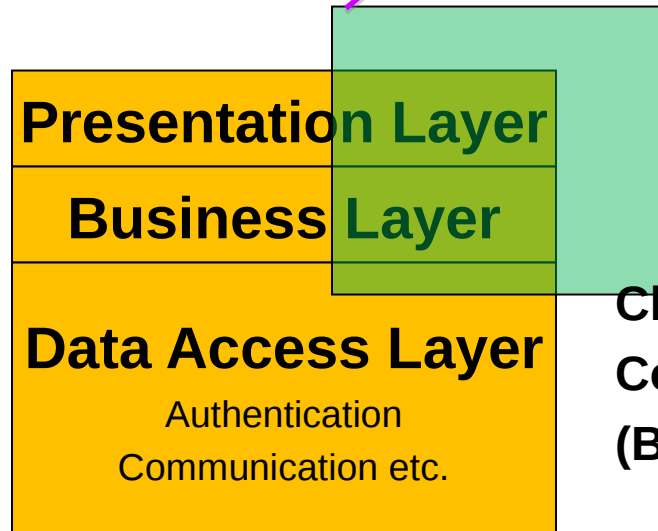
# Today's Development Life Cycle

- Rapid/Agile development
- Catching up with new technologies – Web 2.0, HTML5, Framework driven, Mobile, Cloud and on on on on on...
- Tight deadlines
- Frequent release cycles
- Business requirement
- Security requirement

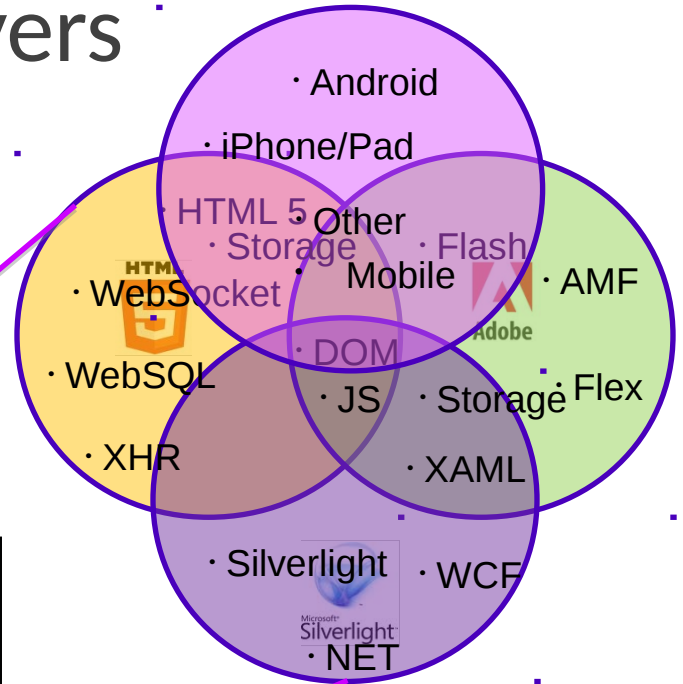


# Stack/Logic - Layers

Server side  
Components



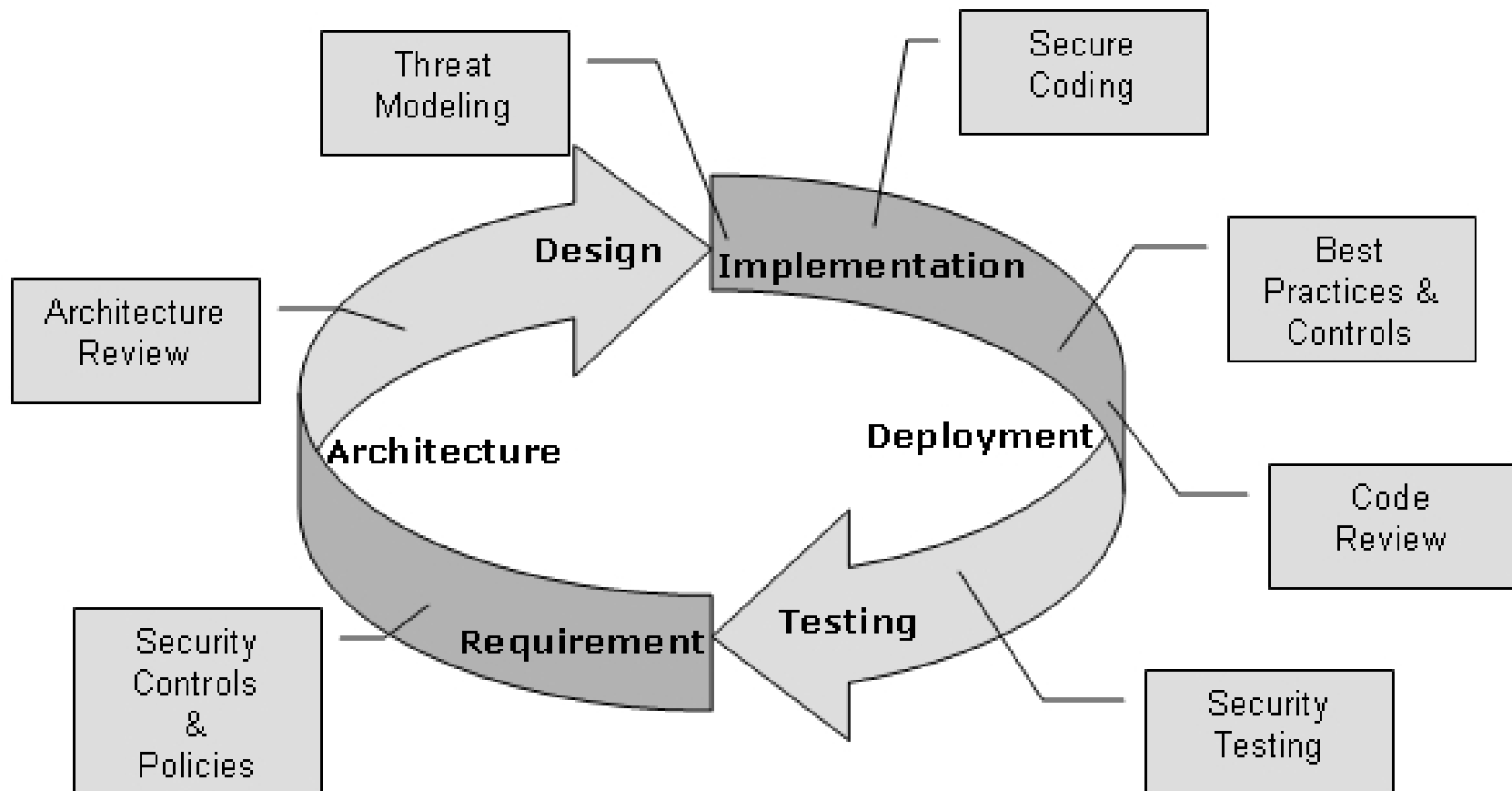
Client side  
Components  
(Browser)



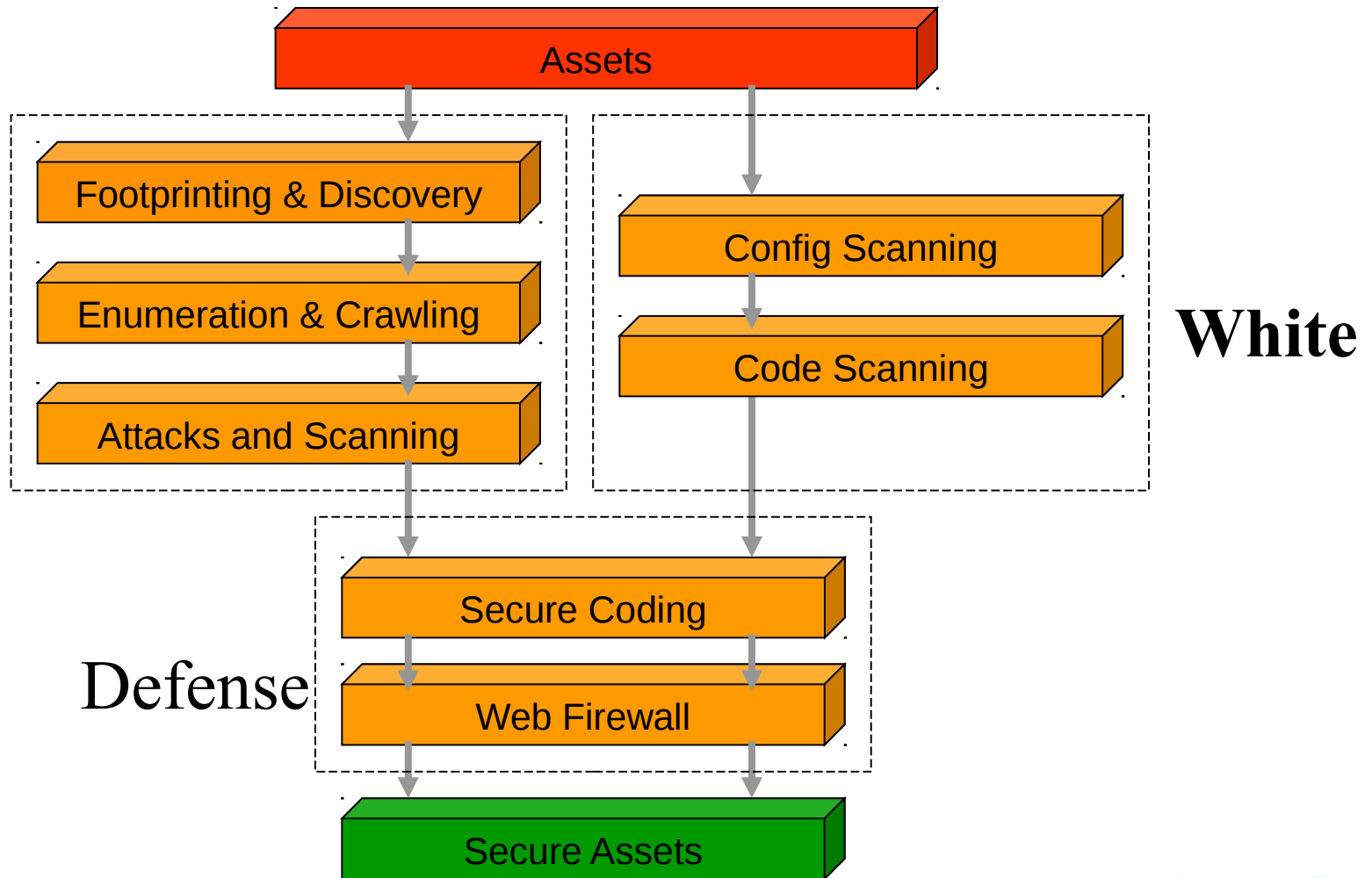
Runtime, Platform, Operating System Components



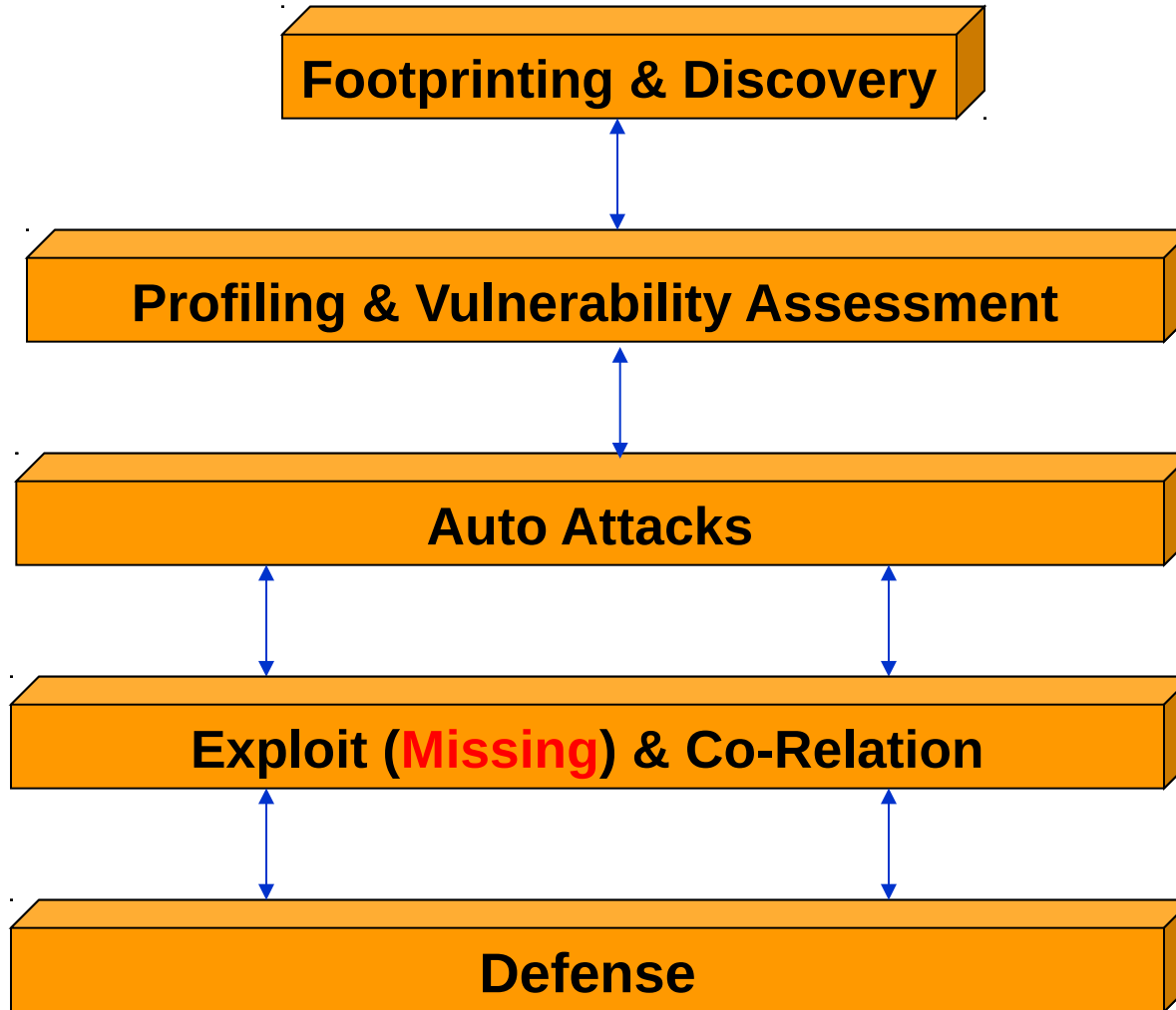
# Secure ADLC – Is it enough???



# Methodology, Scan and Attacks

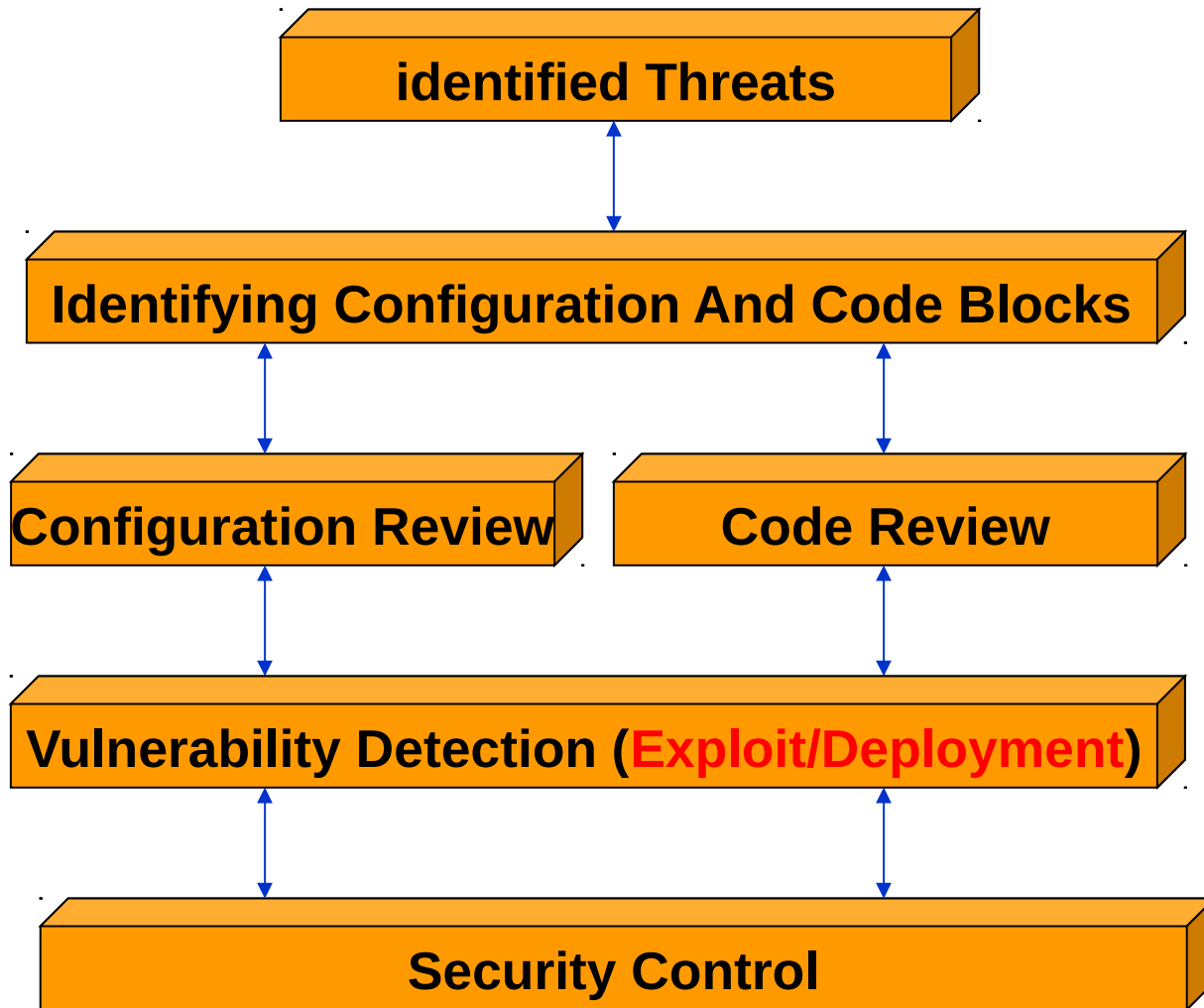


# Automated Blackbox





# Automated Whitebox



# Automated - Whitebox & Blackbox

- Blackbox method uses crawling and spidering to determine all possible resources
- Application assets are residing in JavaScript and various other tags in HTML, it makes asset detection very difficult and blackbox approach fails in many cases.
- Automated tools are headless and generate so many False Positives that usually real vulnerabilities are missed while eliminating false positives
- Automated tools may not perform behavioral analysis effectively



# Modern Application Challenges

- How to identify possible hosts running the application? – Cross Domain
- Identifying Ajax and HTML5 calls
- Dynamic DOM manipulations
- Identifying XSS and XSRF vulnerabilities for Web 2.0
- Discovering back end Web Services - SOAP, XML-RPC or REST.
- How to fuzz XML and JSON structures?
- Client side code review
- Mashup and networked application points



# Game is complex



# Value of Human in Application Security Reviews



DEEPSEC

---

**Blue**infy

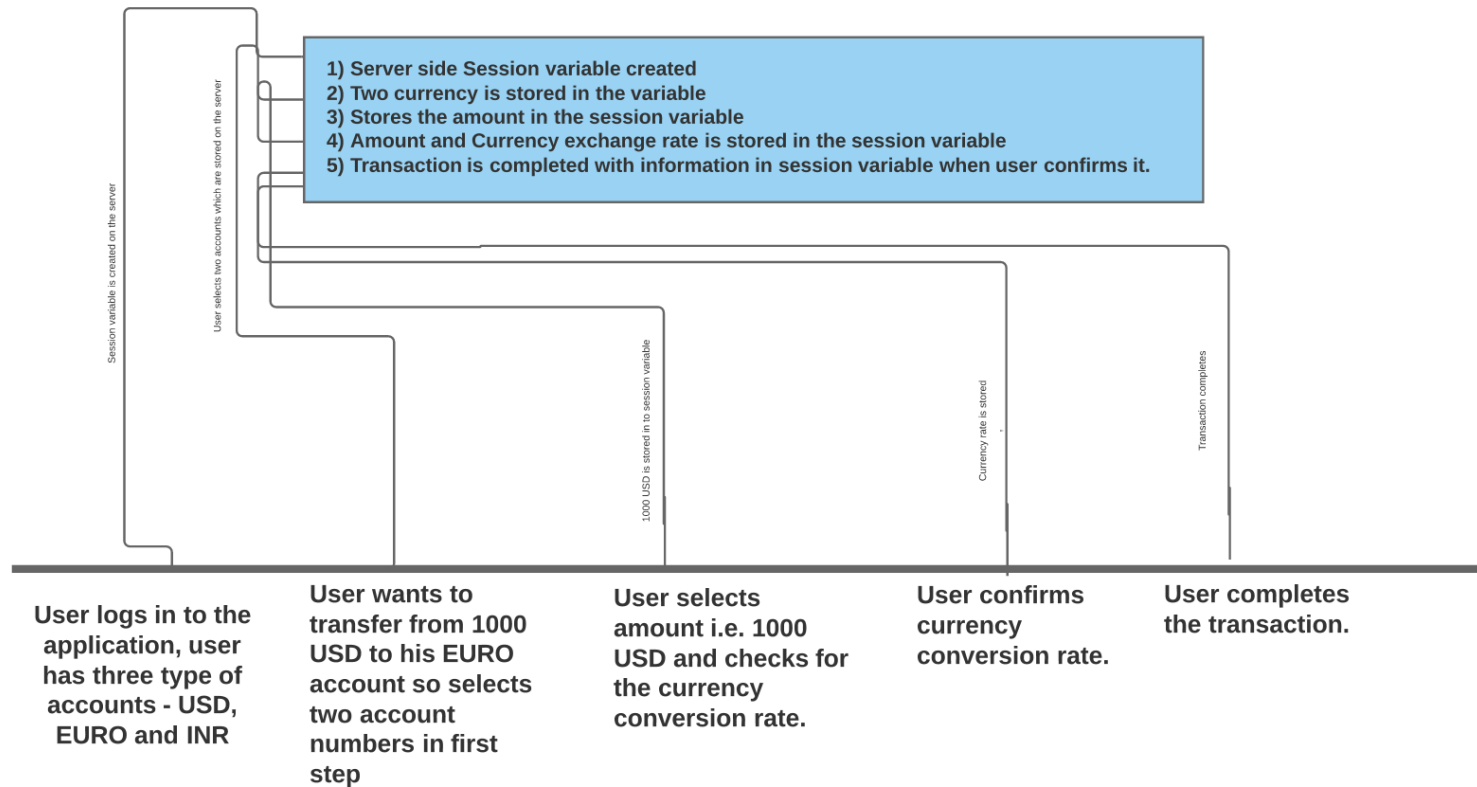
# Server Side Session over riding

- Typically found in multi-step or wizard based functionality implementation
- The application is writing in to server side session to remember user selection/input from the previous page/step
- The application performs validation on each step but fails to perform validation at the end before performing actual transaction
- E-commerce application stores information when the user selects item
- Banking/Currency conversion will require to ask for user's authorization or transaction passwords



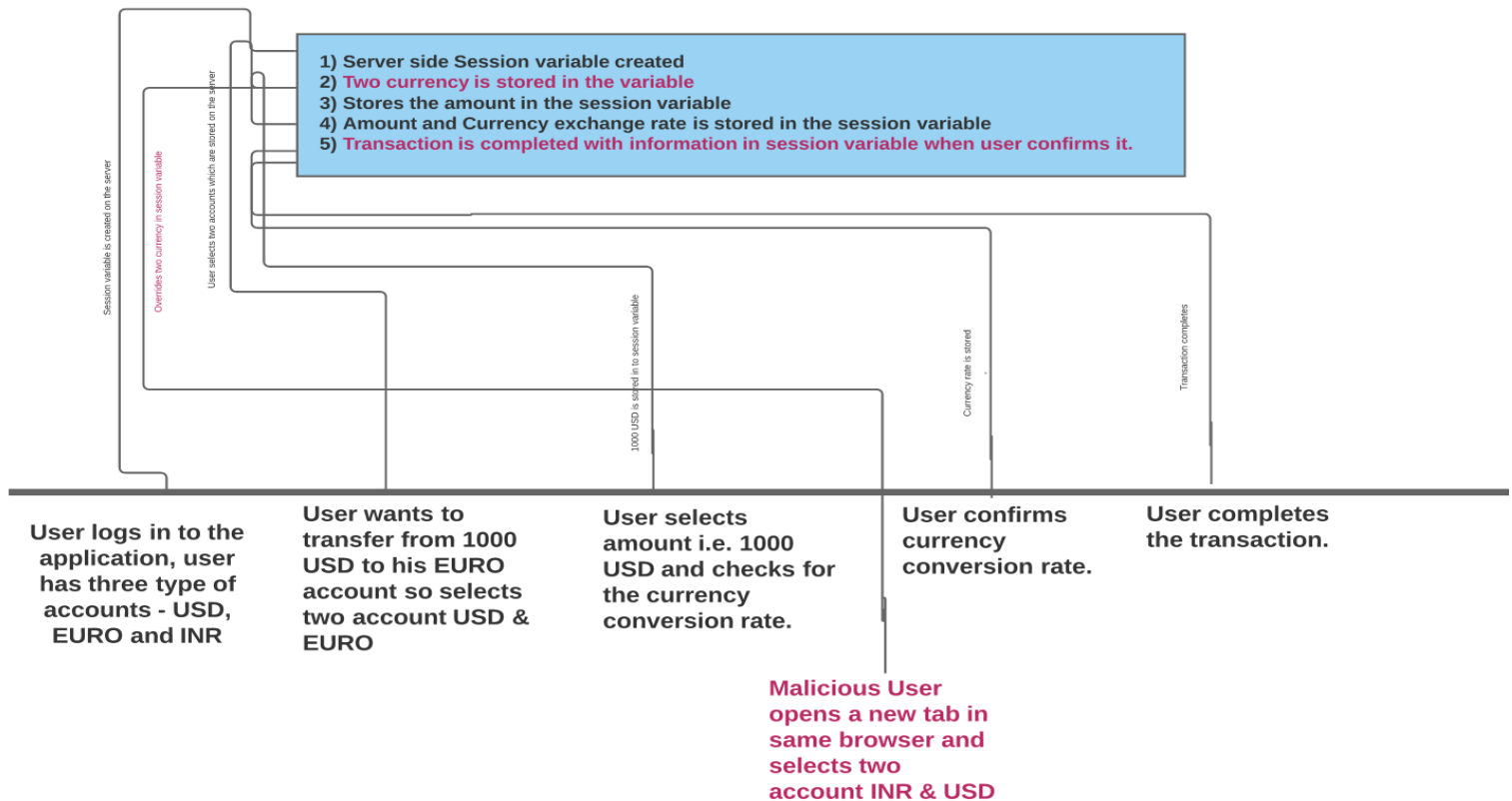
# Server Side Session overriding

## SESSIONOVERRIDING



# Server Side Session overriding - Attack

## SESSIONOVERRIDING





# Server Side Session over riding - Demo



# Bypassing Authorization with Response Fuzzing

- Traditionally penetration testing has been focusing only on modification of requests.
- As the browsers have become smart and feature rich, it provides a lot of functionalities along with storage on the client side
- Developers are leveraging it
- As a result, some of the business logic has been shifted from server side to client side
- Thus, it has become important to fuzz response along with request
- At times, flag change in response opens up new functionality or new menu items or access to an admin application



# Bypassing Authorization with Response Fuzzing



# HTML5 Client Side Storage Bypass

- HTML5 brings two different types of storage in to browser
- XSS was pain as it was stealing my cookie, now I can save my cookies in to client side storage in browser and XSS cannot steal it – REALLY???
- Brute force DOES not work any more as the accounts are locked – REALLY???
- Great, now I can save all my counters and information on the client side storage
- All these are wrong assumption as automated scanners do not find these vulnerabilities
- The applications are counting bad attempts and setting a value on the client side either in cookies or LocalStorage options provided by browsers



# HTML5 Client Side Storage Bypass - Demo

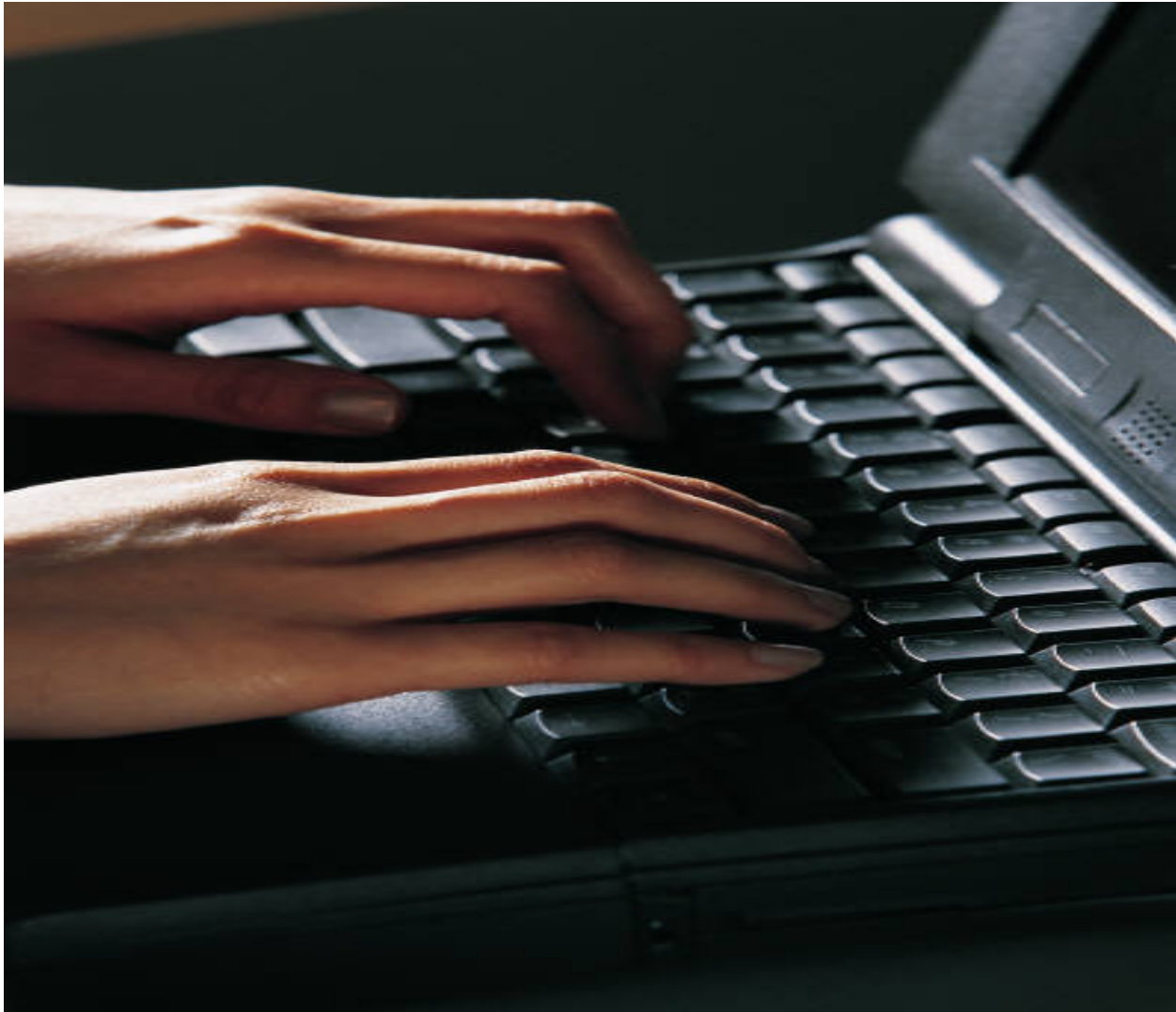


# Command Execution with File Upload

- The application has a upload functionality
- My favorite place to attack
- Result can be command execution, path traversal to get access to important files, impact system with virus
- Key is to find out where files are stored
- If uploads are not validated, can be very RISKY



# Command Execution with File Upload - Demo



# SQL Injection on Parameter Name

- SQL Injection in parameter value is very common – isn't it
- What if you are validating all user input value
- Should input validation be only on the VALUE and not on the NAME if you are using it to construct query???





# SQL Injection on Parameter Name - Demo



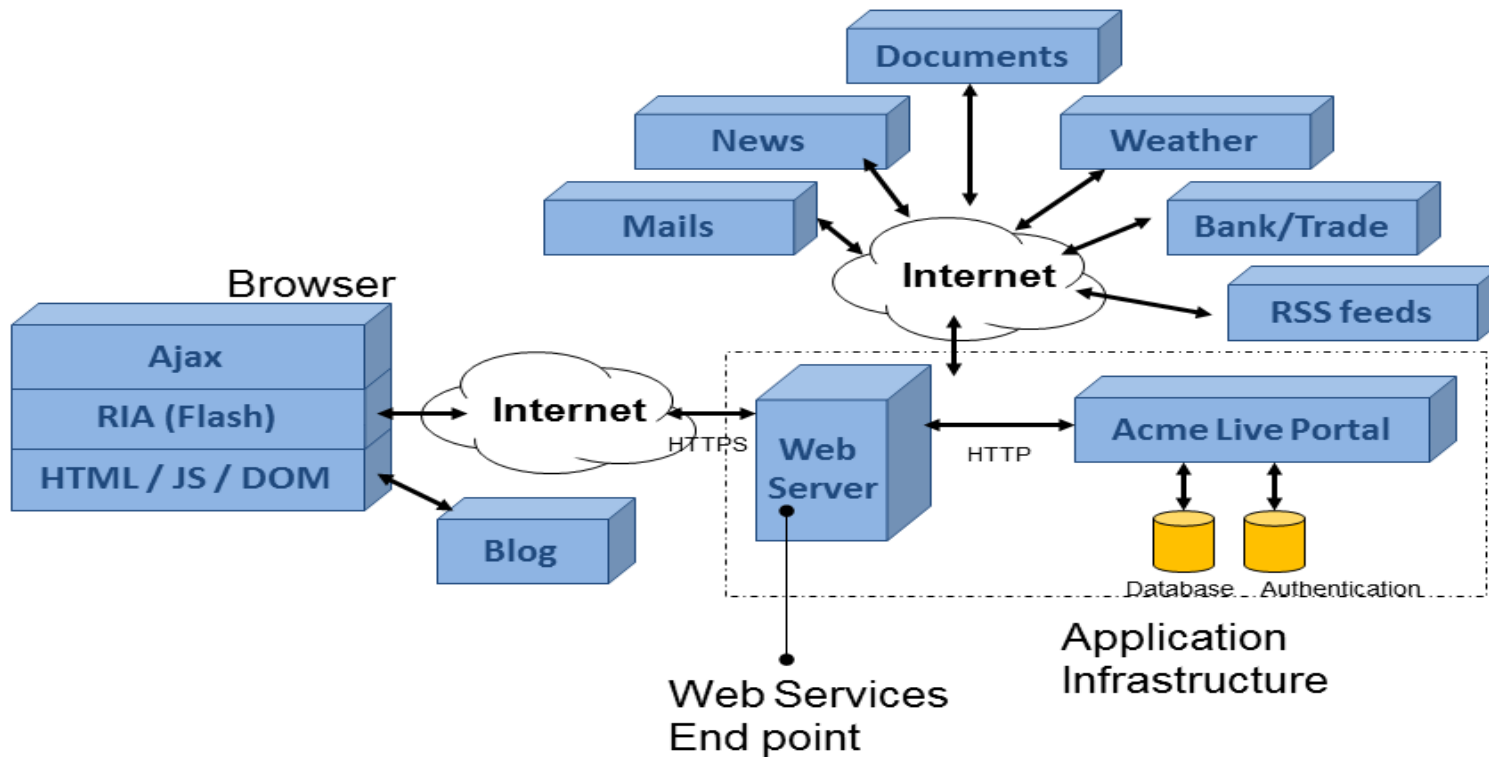
# Third party posting, injection and streaming

- There are number of applications out there which works as a proxy
- An application will allow user to create custom widgets and configure what they would like to see on the widget
- These widgets can make calls to third party server to fetch email, RSS feeds, blog, News, twitter, facebook and so on...
- An attacker does not need to attack the application but needs to attack source of the application to compromise user
- XSS attack from RSS feed can lead in to user's loosing his session for the application and end up giving access to all other widgets of the user.



# Third party posting, injection and streaming

## Acme Live Portal Architecture



# Third party posting, injection and streaming - Demo



DEEP SEC  
INTEGRITY SECURITY

Blueinfy

# Asynchronous injections across critical functions

- Applications are leveraging Asynchronous channels i.e. email, OTP on the Mobile to implement critical functionalities
- SOA has made it easy as third party API integration has become really easy
- In the application, not all tasks need to be completed right away.
- Possibly another program can take care of the task later
- If you request an old statement in bank, bank will have transaction table where it enters the account number and run another program which runs at particular time
- The program will run the job and send results to user over outside communication channel i.e. email



# Asynchronous injections across critical functions

- What if we find SQL Injection and enter payload with account number???
- The application will end up sending statement over email of all users



# Custom protocol handling

- AJAX calls making calls using non HTTP protocol
- Tough to analyze and craft request
- Only HUMAN can do it unless custom program is written to attack using custom protocol



# Information Leakage via Analytics calls

- Most applications leverages analytics to understand user's behavior
- Sends information to third party analytical services
- Easy to implement across the application BUT should not be implemented at key functionalities -
  - ◆ Forgot password
  - ◆ Change password page





# Protect

- Input validation on the server side is key – Perform validation for type, size, length and expected characters
- Client side validation is NO VALIDATION
- Implement defense in depth – Defense at all possible layer
- Sanitize all output or perform output encoding
- Do not store files in webroot
- Have proper permission on directories which stored uploaded files



# Summary

- HUMANS can find what machine can not
- No matter what we do, HUMAN is going to be superior than machines to find HIGH risk vulnerabilities
- Leverage combination of human intelligence and machine (Automated) to achieve maximum security (Minimize Risk)



# DEEP SEC

THANK YOU!

[Hemil Shah, [hemil@blueinfy.net](mailto:hemil@blueinfy.net)]



DEEP SEC

Blueinfy

# DEEPSEC

## Human vs Artificial intelligence – Battle of Trust

Hemil Shah

Co-CEO & Director

Blueinfy Solutions Pvt Ltd



Notes

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

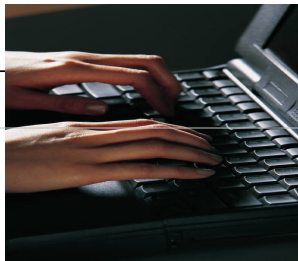
---

---



# Key Points

- Importance of manual application security reviews
- Unique cases used by humans to identify high value security vulnerabilities in applications
- Methodology of analyzing modern era applications to find complex security vulnerabilities
- Suggestions to protect against the vulnerabilities



## Notes

---

---

---

---

---

---

---

---

---

---

---



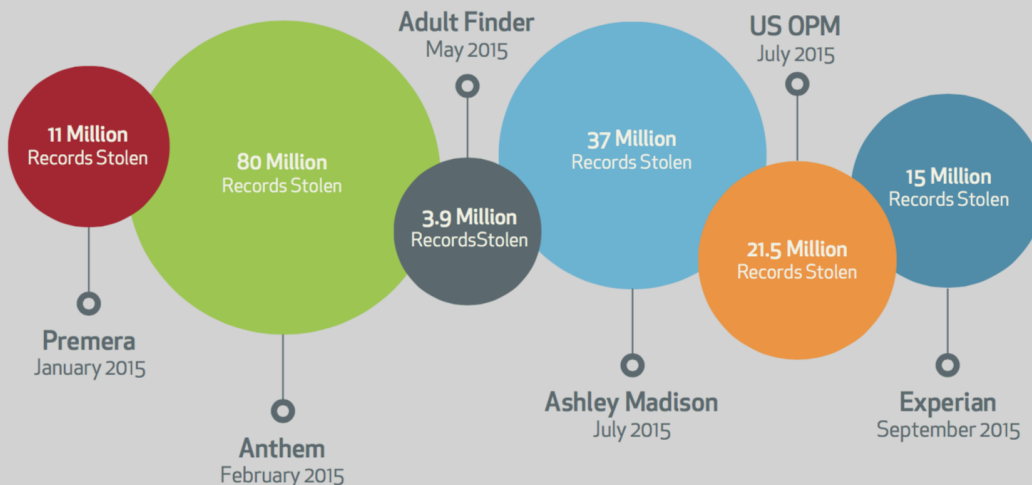






# 2015 – A Year that has been

2015 has been a year of large security breaches:



The above are examples of significant system breaches resulting in millions of records and personal data being stolen.



Source – 2015 – Edgescan Status Report

Slide 7

## Notes

---

---

---

---

---

---

---

---

---

---

---

---

---

# 2015 – A Year that has been

## SECURITY TESTING

**98% of applications tested were vulnerable**

**20: Median number of vulnerabilities per application (up six from 2013)**

**95% of mobile applications were vulnerable**

- 6.5: Median number of vulnerabilities per mobile application
- 35% had critical issues
- 45% had high-risk issues



Source – 2015 – Trustwave Global Security Report

Slide 8

## Notes

---

---

---

---

---

---

---

---

---

---

---

---

---



# Attacks and Hacks

- 98% applications are having security issues
- Web Application Layer vulnerabilities are growing at higher rate in security space
- Client side hacking and vulnerabilities are on the rise
- Web browser vulnerabilities are growing at higher rate
- End point exploitation shifting from OS to browser and its plugins



## Notes

---

---

---

---

---

---

---

---

---

---

---

---

---

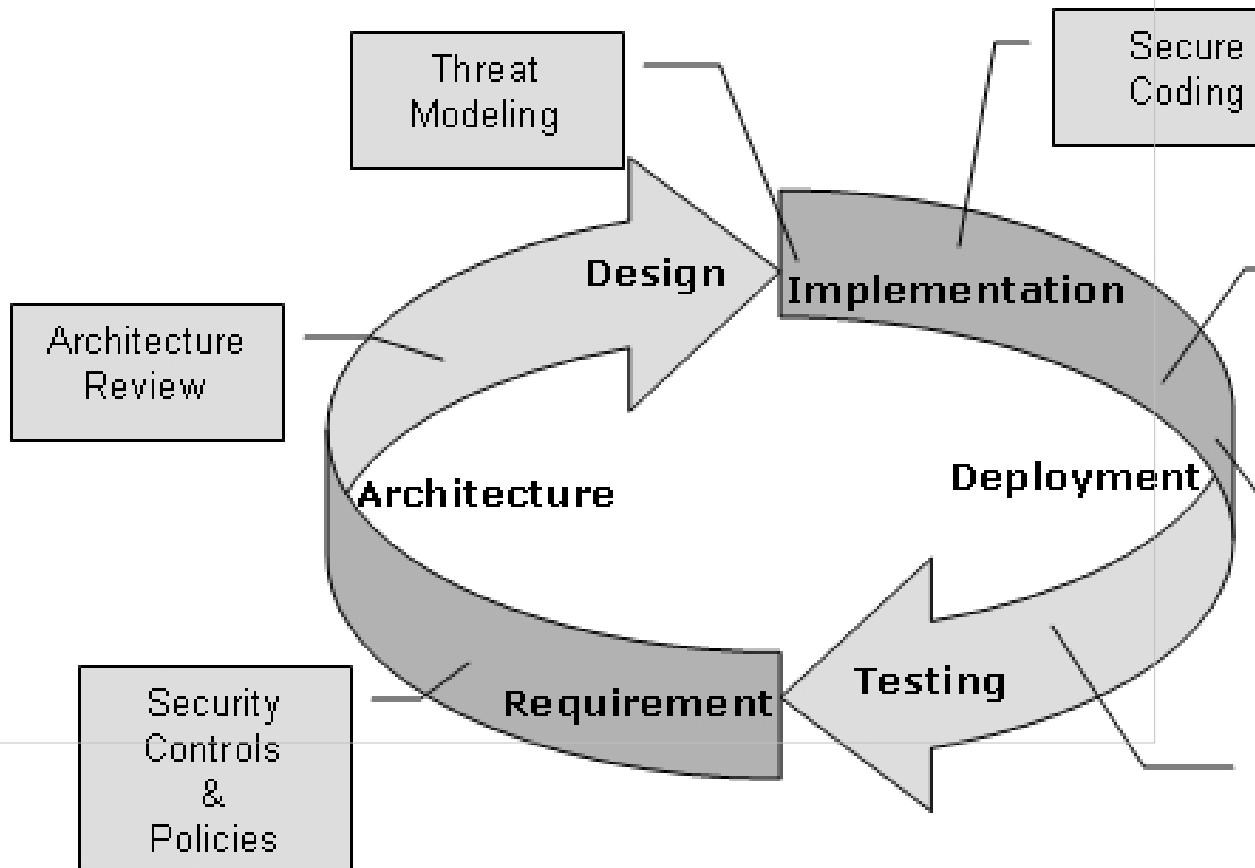








# Secure ADLC – Is it enough???



Not

---

---

---

---

---

---

---

---

---

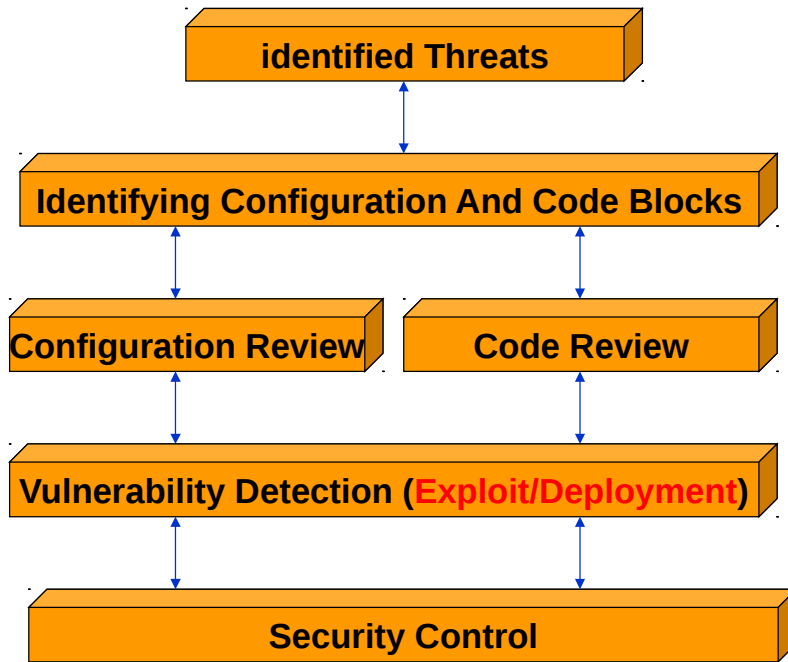
---

---





# Automated Whitebox



## Notes

---

---

---

---

---

---

---

---

---

---

---

---

---

---





# Game is complex



---

---

---

---

---

---

---

---





# Server Side Session over riding

- Typically found in multi-step or wizard based functionality implementation
- The application is writing in to server side session to remember user selection/input from the previous page/step
- The application performs validation on each step but fails to perform validation at the end before performing actual transaction
- E-commerce application stores information when the user selects item
- Banking/Currency conversion will require to ask for user's authorization or transaction passwords



## Notes

---

---

---

---

---

---

---

---

---

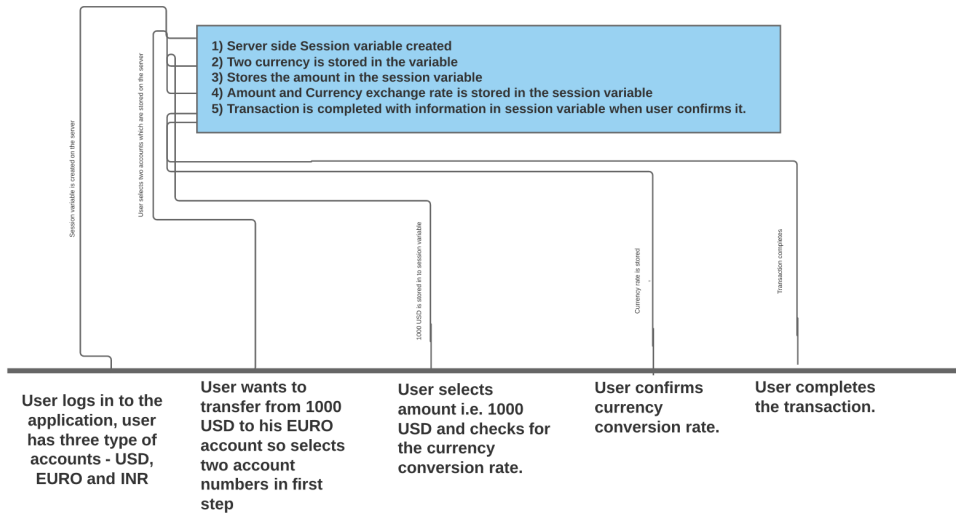
---

---

---

# Server Side Session over riding

## SESSIONOVERRIDING



User logs in to the application, user has three type of accounts - USD, EURO and INR

User wants to transfer from 1000 USD to his EURO account so selects two account numbers in first step

User selects amount i.e. 1000 USD and checks for the currency conversion rate.

User confirms currency conversion rate.

User completes the transaction.



Slide 23

### Notes

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

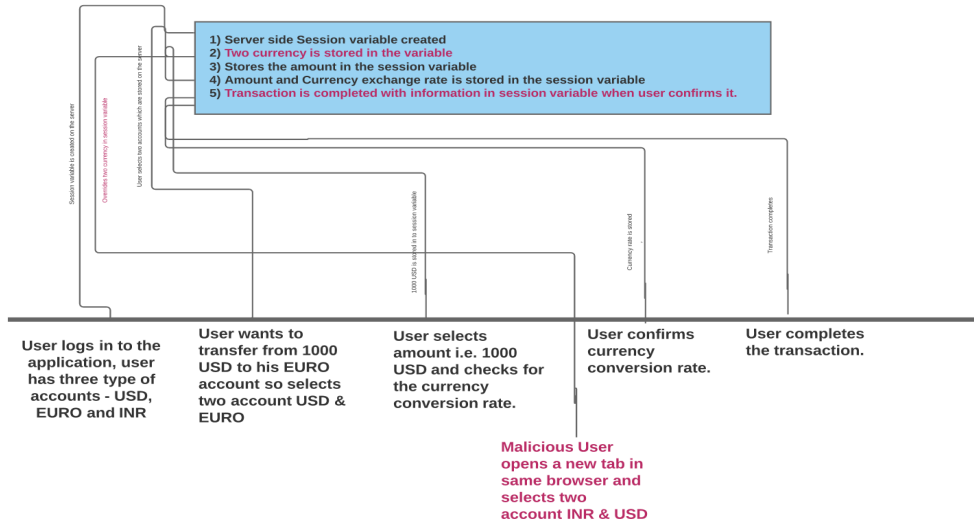
---

---

---

# Server Side Session over riding - Attack

## SESSIONOVERRIDING



DEEPSEC

Slide 24

## Notes

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

# Server Side Session over riding - Demo



## Notes

---

---

---

---

---

---

---

---

---

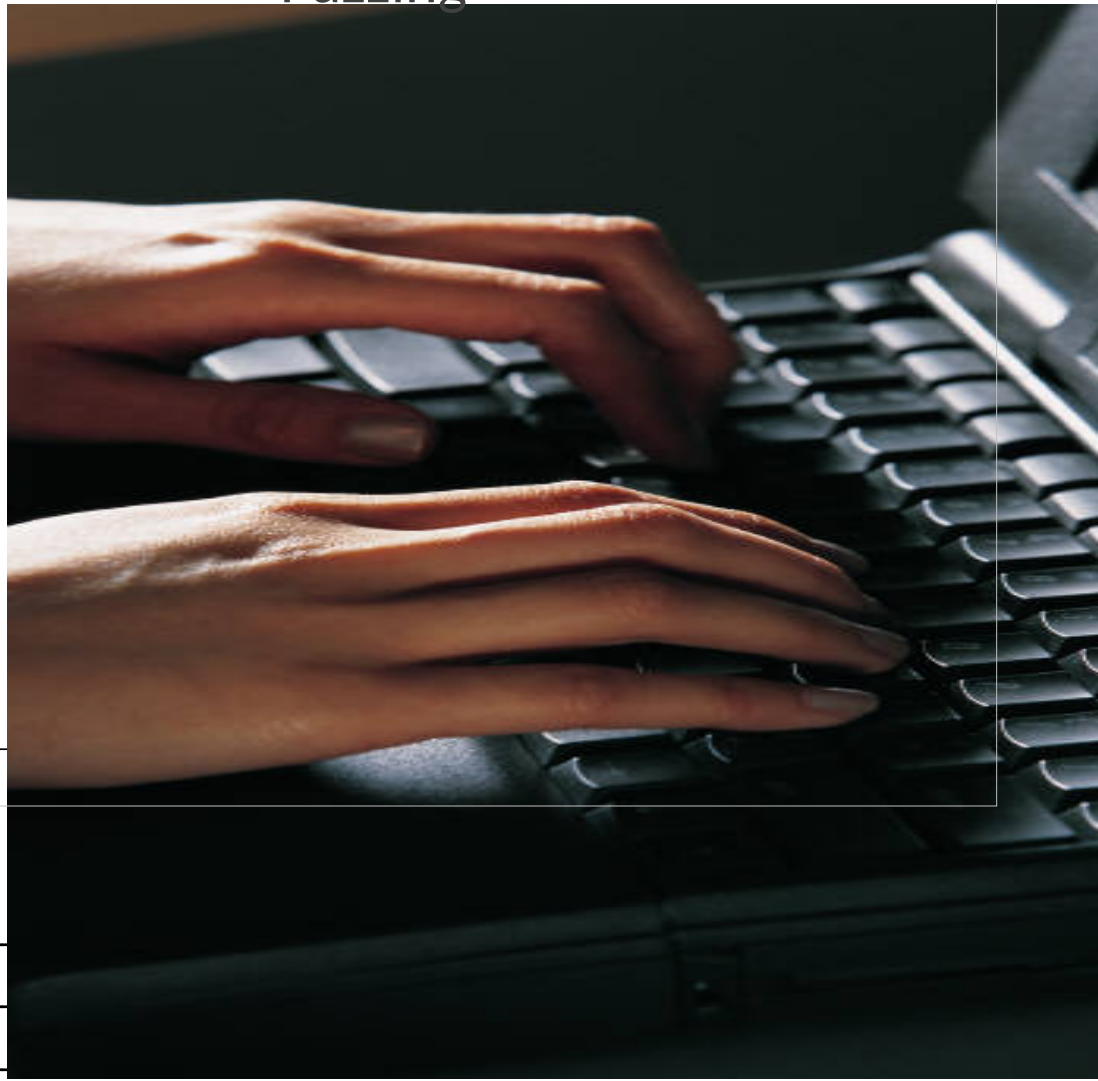
---

---

---



# Bypassing Authorization with Response Fuzzing



## Notes

---

---

---

---

---

---

---

---

---

---

---

---

---



# HTML5 Client Side Storage Bypass - Demo



Notes

---

---

---

---

---

---

---

---

---

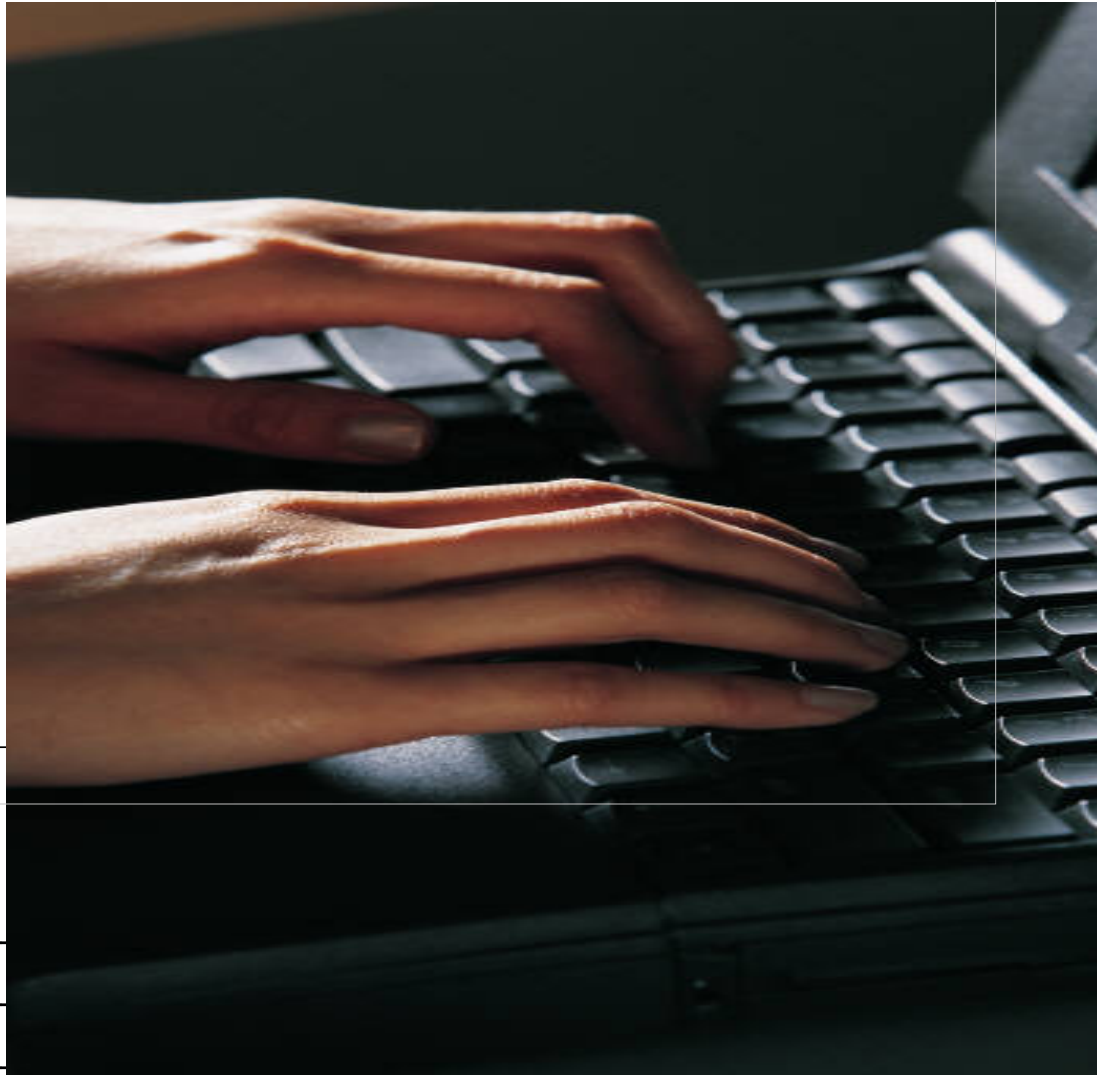
---

---





# Command Execution with File Upload - Demo



Notes

---

---

---

---

---

---

---

---

---

---

---

## SQL Injection on Parameter Name

- SQL Injection in parameter value is very common – isn't it
- What if you are validating all user input value
- Should input validation be only on the VALUE and not on the NAME if you are using it to construct query???



### Notes

---

---

---

---

---

---

---

---

---

---

---

# SQL Injection on Parameter Name - Demo



## Notes

---

---

---

---

---

---

---

---

---

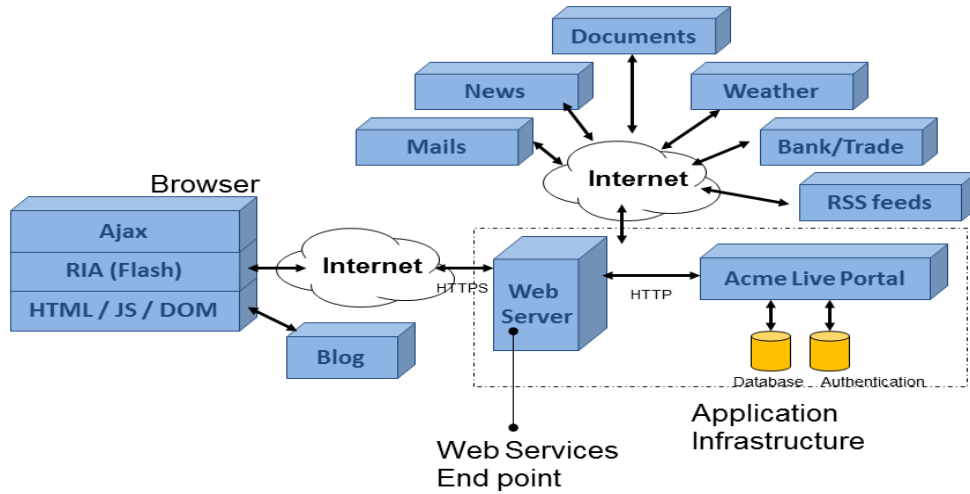
---

---



# Third party posting, injection and streaming

## Acme Live Portal Architecture



Notes

---

---

---

---

---

---

---

---

---

---

---

---

---

# Third party posting, injection and streaming - Demo



Notes

---

---

---

---

---

---

---

---

---

---

---

---

# Asynchronous injections across critical functions

- Applications are leveraging Asynchronous channels i.e. email, OTP on the Mobile to implement critical functionalities
- SOA has made it easy as third party API integration has become really easy
- In the application, not all tasks need to be completed right away.
- Possibly another program can take care of the task later
- If you request an old statement in bank, bank will have transaction table where it enters the account number and run another program which runs at particular time
- The program will run the job and send results to user over outside communication channel i.e. email



DEEPSEC

Slide 37

## Notes

---

---

---

---

---

---

---

---

---

---

---

---



# Asynchronous injections across critical functions

- What if we find SQL Injection and enter payload with account number???
- The application will end up sending statement over email of all users



## Notes

---

---

---

---

---

---

---

---

---

---

---

---

---





# Protect

- Input validation on the server side is key – Perform validation for type, size, length and expected characters
- Client side validation is NO VALIDATION
- Implement defense in depth – Defense at all possible layer
- Sanitize all output or perform output encoding
- Do not store files in webroot
- Have proper permission on directories which stored uploaded files



DEESEC

Slide 41

## Notes

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

# Summary

- HUMANS can find what machine can not
- No matter what we do, HUMAN is going to be superior than machines to find HIGH risk vulnerabilities
- Leverage combination of human intelligence and machine (Automated) to achieve maximum security (Minimize Risk)



## Notes

---

---

---

---

---

---

---

---

---

---

---

# DEEP SEC

THANK YOU!

[Hemil Shah, hemil@blueinfy.net]



## Notes

---

---

---

---

---

---

---

---

---

---

---