RSAConference2015

Singapore | 22-24 July | Marina Bay Sands

CHANGE

Challenge today's security thinking

SESSION ID: **SEC-R01**

Security Risks in SDN and Other New Software Issues

Anthony Lim

Vice-Chair, Application Security, isc2.org

Sr Cybersecurity Advisor, Frost & Sullivan

(ISC)² INSPIRING A SAFE AND SECURE CYBER WORLD.

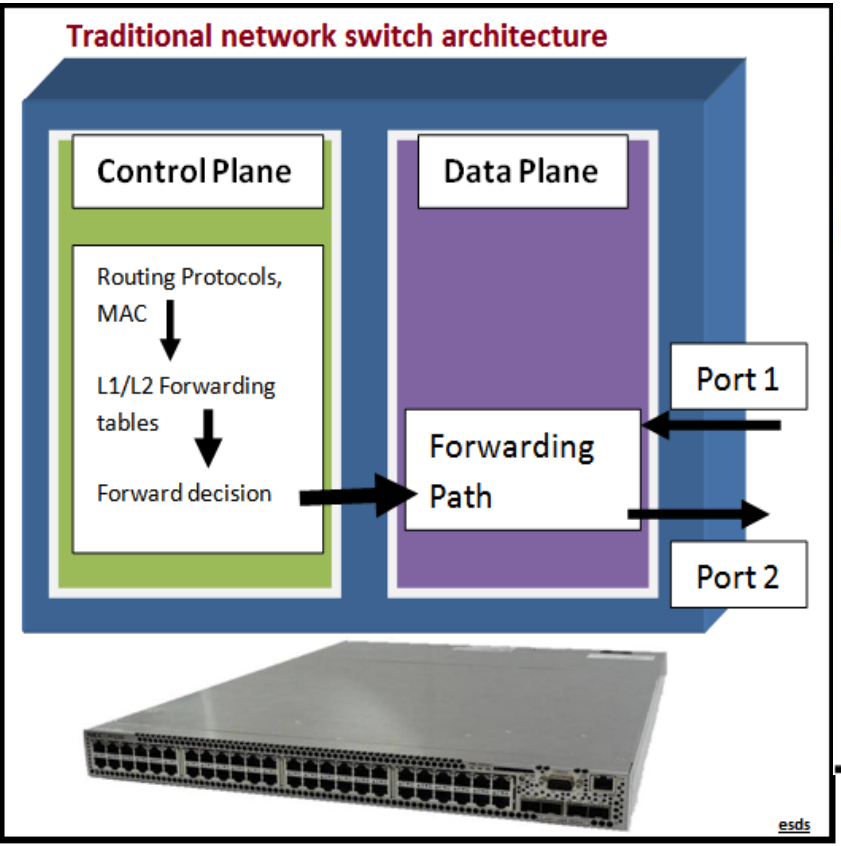
FROST & SULLIVAN

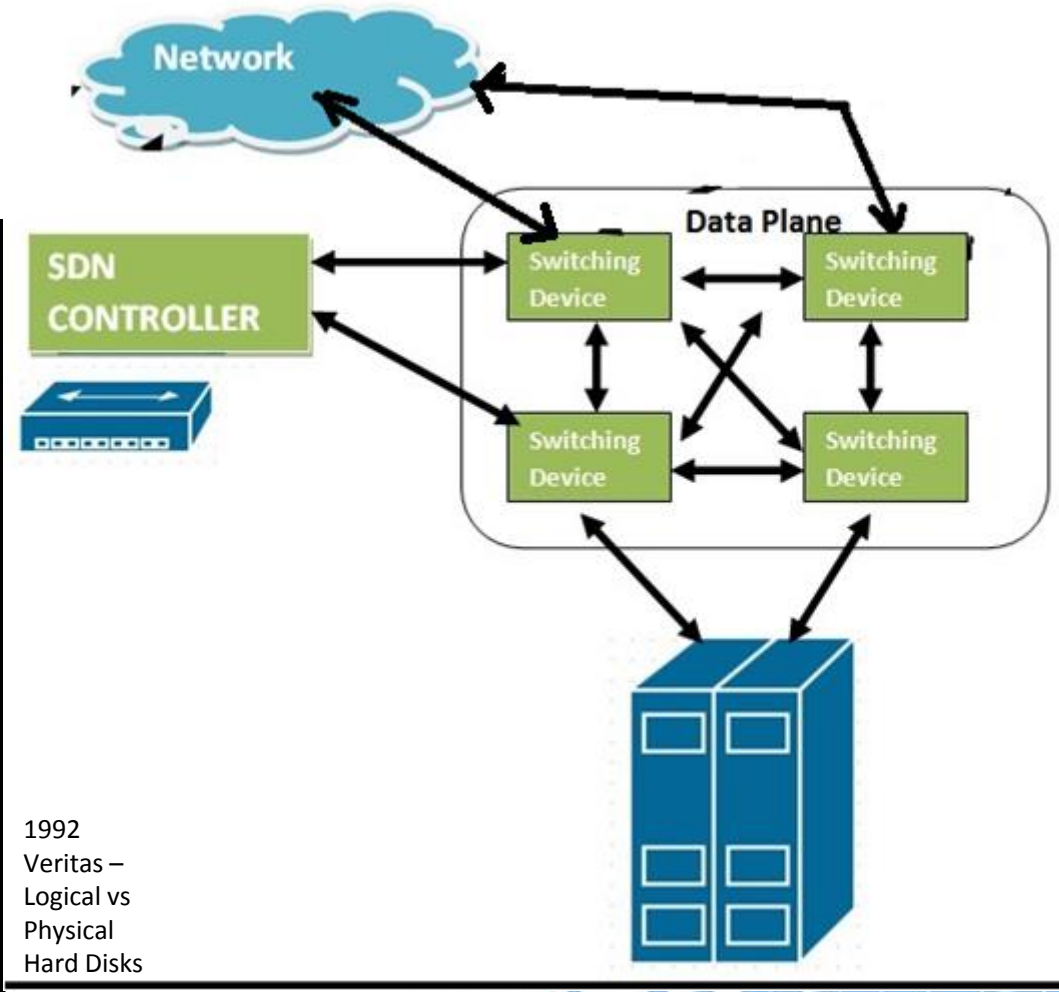
#RSAC

Introduction : Software (Application Security)

- **Greatest cybersecurity threat and attack methodology today**
  - but ironically worst-defended, because it is least-understood by users.
- **Network firewalls do not stop application attacks.**
  - Network & system administrators usually don't know or care much about App Devt.
- **More and more software in our increasingly-connected world today:**
  - Web 2.0, HTML5, Cloud Services, Mobile services (BYOD / Android)
  - Internet-of-Things (IOT), Software-defined Networks (SDN), Software-defined Datacenters (SDDC)
  - "old" targets – Heartbleed, Shellshock, ICS/SCADA, Win XP, Legacy systems …

SOFTWARE-DEFINED NETWORK (SDN)

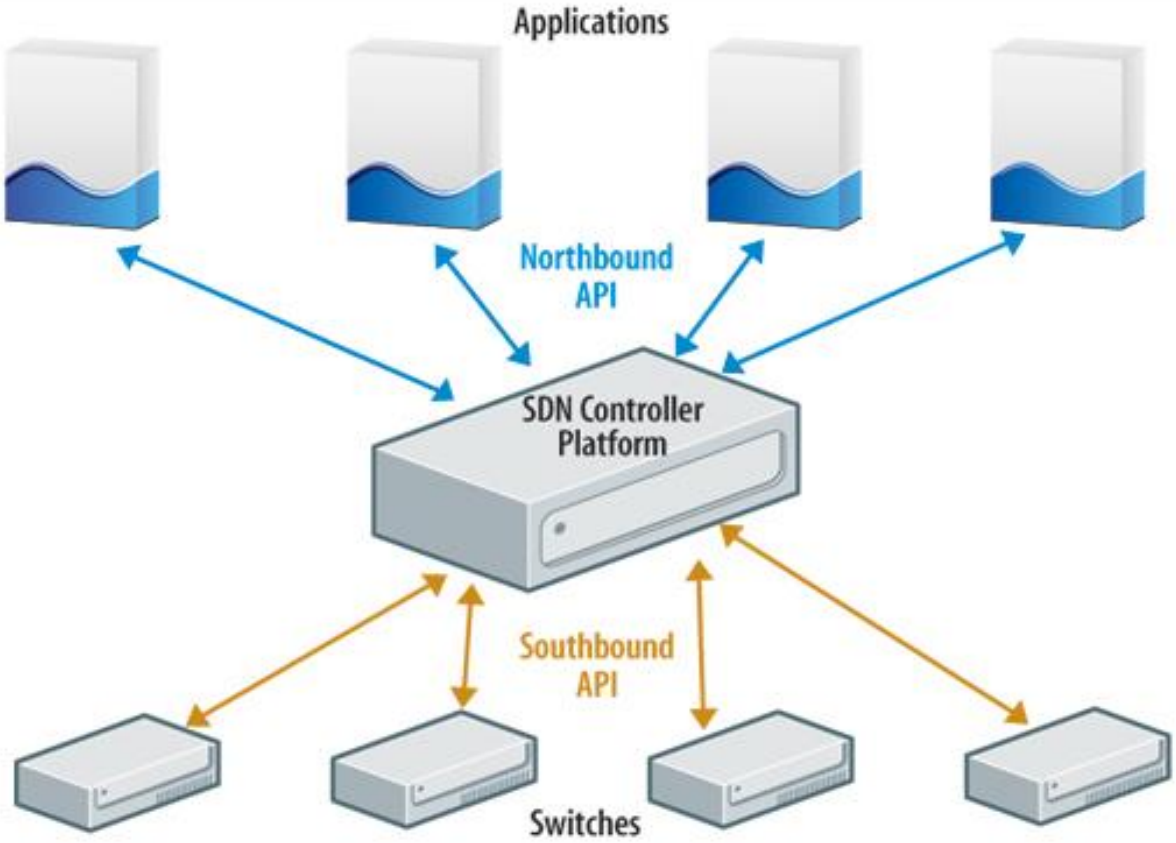




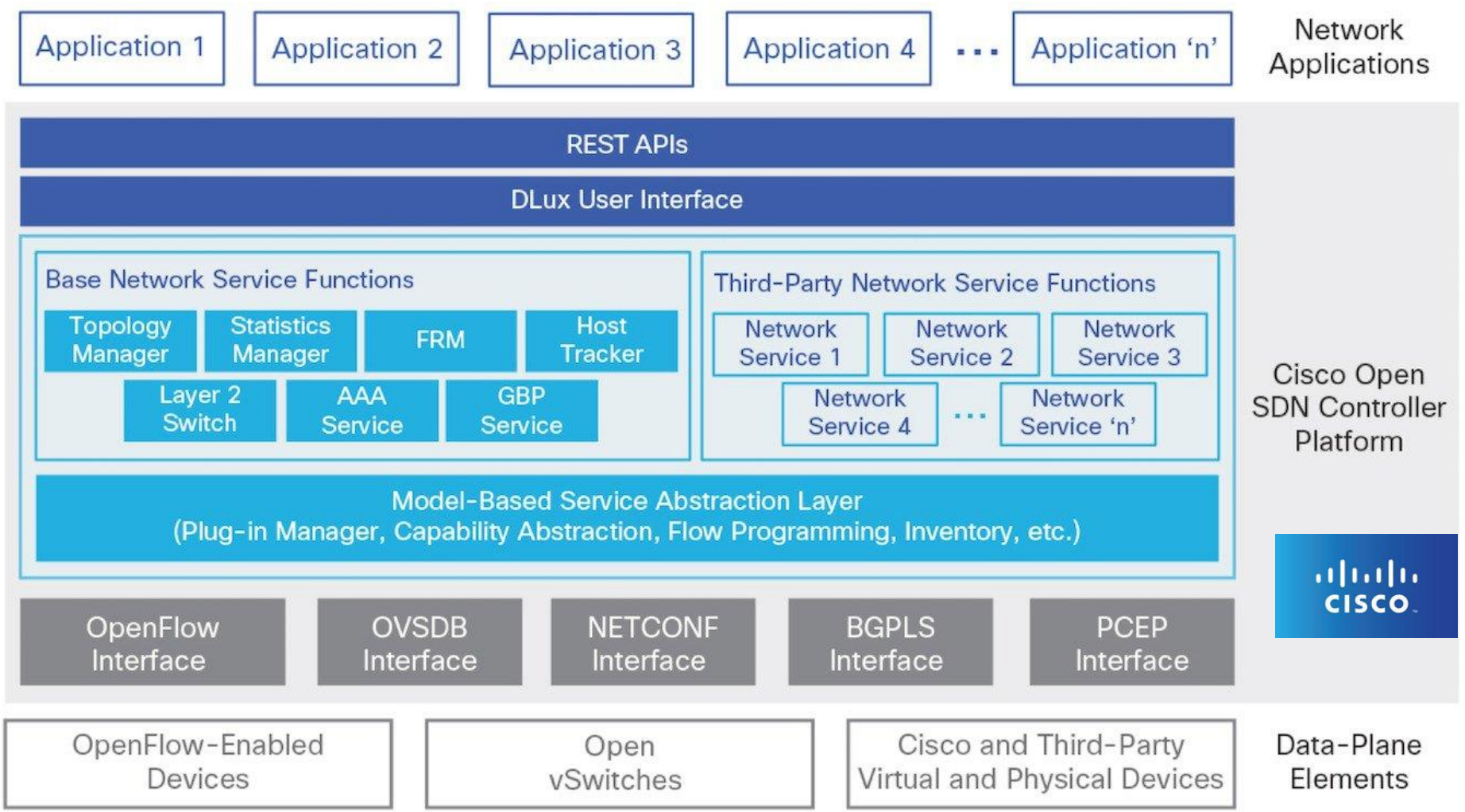
1992
Veritas –
Logical vs
Physical
Hard Disks

SDN Concept:

An architecture meant to be dynamic, manageable, cost-effective, and adaptable, seeking to be suitable for the high-bandwidth, dynamic nature of today's applications. SDN architectures decouple network control and forwarding functions, enabling network control to become directly programmable and the underlying infrastructure to be abstracted from applications and network services,



In one approach to a software-defined network, a centralized controller directs switches on how best to forward traffic. The controller uses a protocol such as OpenFlow to communicate with the switches. At the same time, a northbound API allows third-party applications to interact with the controller to simplify common tasks or enable new network functions.



The Wonders of Software-defined Network (SDN)

- Know the state of the network rather than inferring it
- Run development and production side-by-side
- More practical
  - – easy to control the whole network & traffic flows

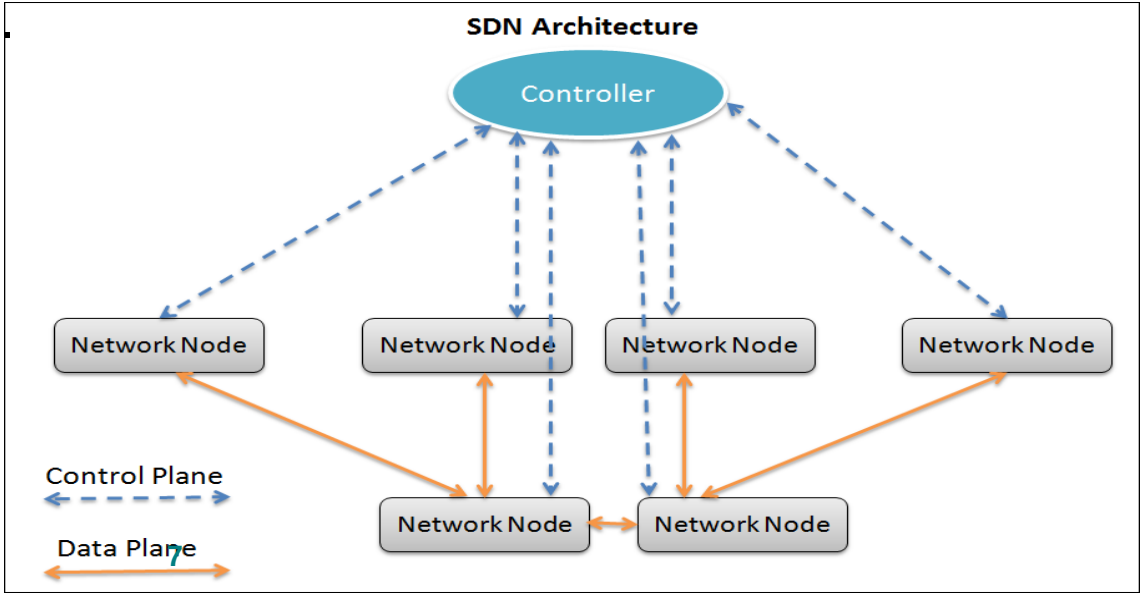
- Less hardware expense
- BGP (Border Gateway Protocol)
  - customer express selection
  - better BGP security
  - faster convergence
  - granular peering at IXPs (internet exchange points)

- Real-world network slicing of flow space
- Network & server load-balancing
- Security
  - dynamic access control
  - adaptive traffic monitoring
  - attack detection and mitigation

Software Defined Networking was developed in an attempt to simplify networking and make it more secure. By separating the control plane (the controller)—which decides where packets are sent—from the data plane (the physical network)—which forwards traffic to its destination—the creators of SDN hoped to achieve scalability and agility in network management. The application layer (virtual services) is also separate. SDN increasingly uses elastic cloud architectures and dynamic resource allocation to achieve its infrastructure goals.

3 Key Elements of Software-Defined Networking

1. Ability to manage the forwarding of frames/packets and apply policy;
2. Ability to perform this at scale in a dynamic fashion;
3. Ability to be programmed.



SDN Security – What is the Issue?

- SDN technology currently in infancy
- hackers will go for anything new
  - Today software attacks are in vogue
- SDN operators are usually not security folks and vice versa

- Security cannot be enforced by physical topology
- Requires complete Trust in SDN Applications and Controller
- We don't understand the consequences

Rob Hinden, Check Point, RSA SFO 2014

Attacker can:

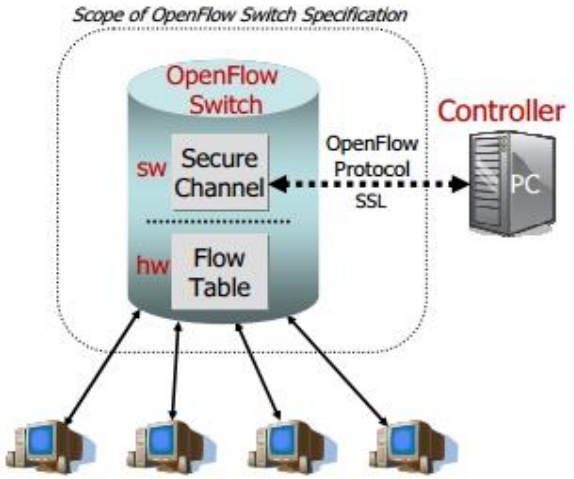
- Identify targets
- enumerate ACLs
- find sensors
- gain access to server
- attack the server
- modify content
- isolate the administrator
- hide from the IDS
- attack the server
- do a Man-in-the-Middle
- subvert DNS responses
- insert malware
- monitor traffic
- PWN the network

Gregory Pickett, Hellfiresecurity.com, DefCon 2014

The Basic SDN Security Issue – The Controller

Per Open Networking Foundation (ONF), steward of the OpenFlow protocol

- **The centralized controller is a "potential single point of attack and failure."**
- The southbound interface -- such as OpenFlow -- between the controller and data-forwarding devices is "vulnerable to threats that could degrade the availability, performance and integrity of the network."



Programmability of SDN controllers presents a double-edged sword.

- Engineers can install security applications on the controller's northbound interface to open up new ways to apply security policies on a network.

- However, this is a potential vulnerability. Those applications can reprogram the network through the controller. Hackers can trick engineers into installing compromised applications

The basic SDN Security Issue – the Controller

Controller weaknesses

- No / weak / incomplete encryption or authentication
- could lead to information disclosure, unauthroized access
- perpetrator can add / remove access, hide or change traffic.

The controller is the most vital component in SDN architecture mainly because it defines the data flows that occur in the Data Plane (traffic flow manager). It is the “brain” of the network and therefore the attacks on and vulnerabilities in controllers are probably the most severe threats to SDN architecture. If the controller is compromised, the attacker can disrupt the data path.

- SDN Applications and Controller have complete control of the network
- Controllers/Applications are built on general-purpose computing platforms, which have vulnerabilities
- If Controller or Application is compromised, the whole Network is so too.

Rob Hinden, Check Point, RSA SFO 2014

If somebody is not paying attention to [the controller], it becomes an extraordinarily high-profit target for an attacker, who could very easily ... modify some of your code base and rescript control of your traffic.

Dave Shackleford, Voodoo security consultant and IANS lead faculty member

Gregory Pickett, Hellfiresecurity.com, DefCon 2014

More basic SDN Security Issues

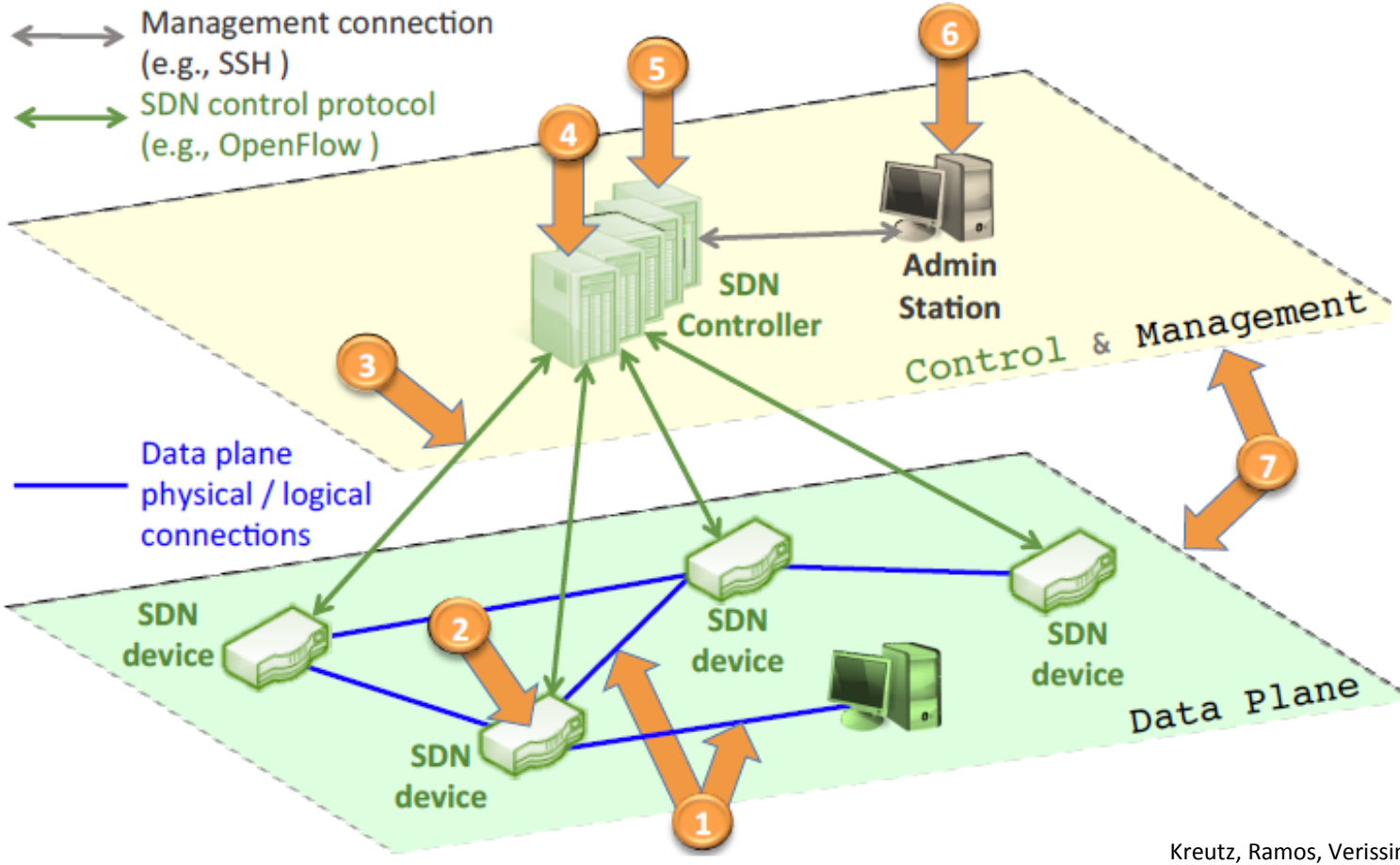
- **Protocol Weaknesses**
  - Encryption and authentication via TLS
    - v.1.0.0 over TLS
    - v.1.4.0 over TCP or TLS

- **Information disclosure through interception**
  - topology, credentials
  - Modification through man-in-the-middle attack
  - Denial-of-service attack potential

- **Debug Ports**
  - no encryption or authentication
  - just full control of the switch
  - All via "dpctl" command-line tool

- **Openflow switch can be impersonated**
  - Openflow controller can be flooded
  - Network can be disrupted and brought down
  - Runs as "root"

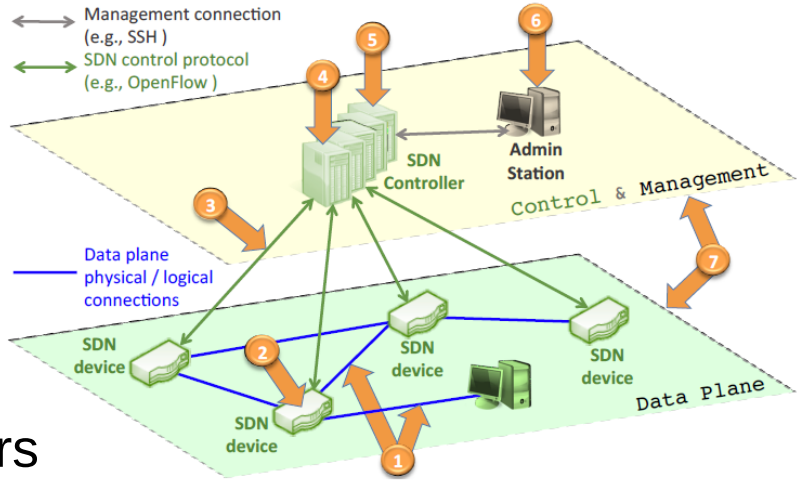
SDN Security Threat Vectors Map



1. forged or faked traffic flows
2. attack on vulnerabilities in switches
3. attacks on control plane communications
4. attacks on and vulnerabilities in controllers
5. lack of mechanisms to ensure trust between the controller and management applications
6. attacks on and vulnerabilities in administrative stations
7. lack of trusted resources for forensics and remediation

Kreutz, Ramos, Verissimo, Univ Lisbon, PT

SDN Security Threat Vectors



1. forged or faked traffic flows
2. attack on vulnerabilities in switches
3. attacks on control plane communications
4. attacks on and vulnerabilities in controllers
5. lack of mechanisms to ensure trust between the controller and management applications
6. attacks on and vulnerabilities in administrative stations
7. lack of trusted resources for forensics and remediation

SDN Security Issues – There's Hope Yet…

Harden the SDN System

1. secure the Data Plane layer

- Use TLS for authentication and encryption to avoid eavesdropping, sniffing and spoofing of traffic.

2. secure the controller layer

- basically harden the host O/S (usually Linux);
- prevent unauthorised access to SDN control administrator; have logging and trails.
-have a High Availability functionality to guard against Denial-Of-Service attack.

3. secure the SDN layer

- use an out-of-band network for control-traffic.
- use TLS or SSH to secure north-bound communications and controller management.
- use application security and secure coding best practices in the software development.

http://www.networkworld.com/article/2840273/sdn/sdn-security-attack-vectors-and-sdn-hardening.html

(Per Scott Hogg, CTO, GTRI)

SDN Security Issues – There's Hope Yet…

1. **Uniform SDN Security Policy - Security Application**
   - Couple Security Policy to SDN policy/rules and validate SDN flows against the security policy
   - Ensure that Security Policy is implemented for all traffic
   - Maintain regulatory and compliance requirements

2. **Security Everywhere**
   - All Routers and Switches have Security capabilities
   - Security Application can take advantage of this and push rules to all network devices

3. **Control Security Treatment Traffic Receives - Range of Security Control**
   - Full Firewall
   - ACL style filtering
   - No filtering

4. **Isolate Compromised Hosts**
   - Once compromised host is detected, it can be isolated.

SDN Security : Protect The Controller

- Secure the Controller: as the centralized decision point, access to the SDN Controller needs to be tightly controlled.
- Protect the Controller: if the SDN Controller goes down (for example, because of a DDoS attack), so goes the network, which means the availability of the SDN Controller needs to be maintained.
- Establish Trust: protecting the communications throughout the network is critical. This means ensuring the SDN Controller, the applications loaded on it, and the devices it manages are all trusted entities that are operating as they should.
- Create a Robust Policy Framework: what's needed is a system of checks and balances to make sure the SDN Controllers are doing what you actually want them to do.
- Conduct Forensics and Remediation: when an incident happens, you must be able to determine what it was, recover, potentially report on it, and then protect against it in the future.

SDN security – Protect The Controller

- The controllers need to be placed at secure location in the network with stringent access policy.
- Out of band management to establish dedicated channel between the controller and SDN devices.
- Secure communication channel between the controller and SDN devices.
- Establish trust relationship between the controller and SDN devices. In addition, the network will still have to cope with existing threats such as attacks on vulnerabilities in network devices and management stations.

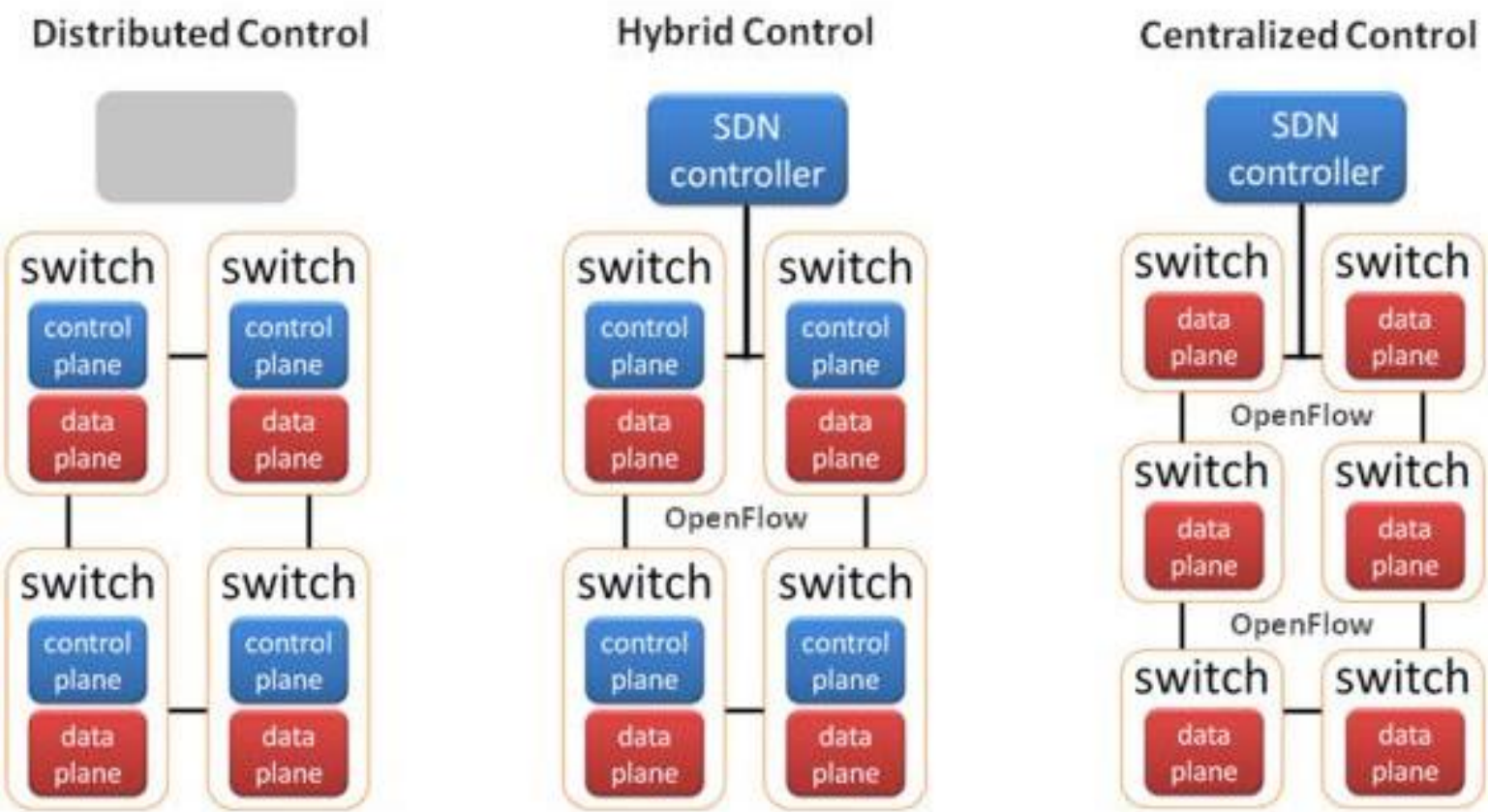
Securing the SDN – per Cisco Systems

Element/Layer	Security Mechanisms
Controller-specific hardening	Secure management protocols; AAA; OS patches; enable only used services, ports, and protocols
Device hardening for agents and controllers	Control plane, management plane and data plane security mechanisms; physical and Layer 2 security
Network services	Disable unused ports, protocols and services; infrastructure access lists; and firewall protection
Applications/APIs	Secure coding practices; digital signing of code; integrity checks
Management/provisioning	Role-based access control; encryption; logging; change management processes
Communications channels	Authentication and authorization; encryption
Agent security services	Firewalls; identity; threat mitigation

In summary … basically, SDN =

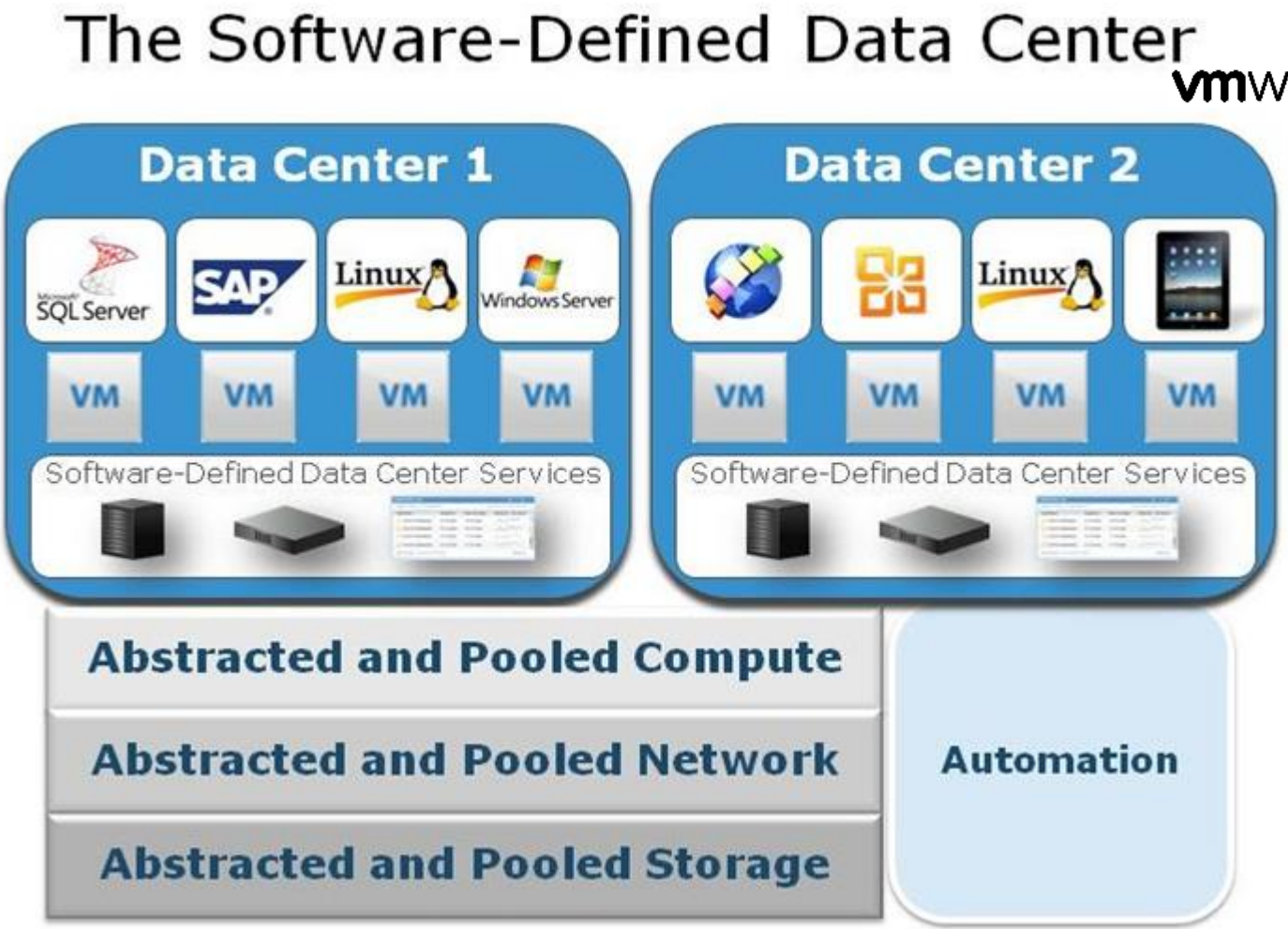
MORE SOFTWARE

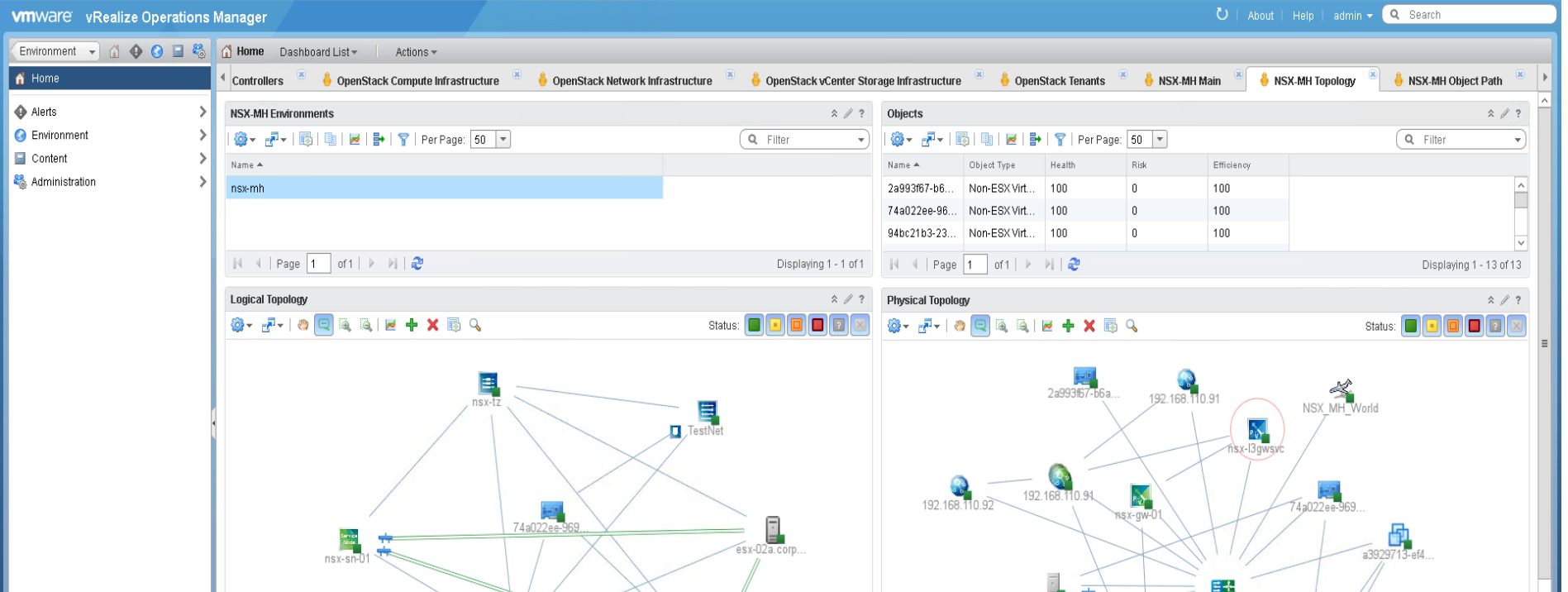
SOFTWARE-DEFINED NETWORK



Now not only SDN
But … ie. Even

MORE SOFT-WARE





Admin Web GUI Dashboard 21

Hackers Attack Web Applications

- **Applications can be CRASHED**
  - to reveal source, logic, script or infrastructure information that can give a hacker intelligence.
- **Applications can be COMPROMISED**
  - to make it provide unauthorized entry access or unauthorized access to read, copy or manipulate data stores, or reveal information that it otherwise would not.
  - Eg. Parameter tampering, cookie poisoning.
- **Applications can be HIJACKED ("PWNED")**
  - to make it perform its tasks but for an authorized user, or send data to an unauthorized recipient, etc.

```
Copyright (c) 2009 Microsoft Corporation. Reservados todos los derechos.

C:\Users\USUARIO>cd /

C:\>cd CMSmap-master

C:\CMSmap-master>cmsmap.py
CMSmap tool v0.6 - Simple CMS Scanner
Author: Mike Manzotti mike.manzotti@dionach.com
Usage: cmsmap.py -t <URL>
          -t, --target    target URL (e.g. 'https://abc.test.com:8080/')
          -v, --verbose   verbose mode (Default: false)
          -T, --threads   number of threads (Default: 5)
          -u, --usr       username or file
          -p, --psw       password or file
          -i, --input     scan multiple targets listed in a given text file
          -o, --output    save output in a file
          -k, --crack     password hashes file (WordPress and Joomla only)
          -w, --wordlist  wordlist file (Default: rockyou.txt)
          -a, --agent     set custom user-agent
          -U, --update    (C)MSmap, (W)ordpress plugins and themes, (J)oomla com
ponents, (D)rupal modules
          -f, --force     force scan (W)ordpress, (J)oomla or (D)rupal
          -F, --fullscan  full scan using large plugin lists. Slow! (Default: fa
lse)
          -h, --help      show this help

Example: cmsmap.py -t https://example.com
         cmsmap.py -t https://example.com -f W -F
         cmsmap.py -t https://example.com -i targets.txt -o output.txt
         cmsmap.py -t https://example.com -u admin -p passwords.txt
         cmsmap.py -k hashes.txt

C:\CMSmap-master>_
```

WhiteHat Top 10 Web App Attacks 2013

1. Mutation XSS
2. BREACH

(Browser Recon n Exfiltration via Adaptive Compression of Hypertext) attack, similar to CRIME (compression Ratio Info-leak Made Easy)

3. HTML5 Pixel-Perfect Timing attack
4. "Lucky 13" attack

(against TLS Transport Layer Security & Datagram TLS; header has 13 bytes).

5. TLS RC4 Encryption weaknesses
6. XML out-of-band data retrieval
7. Million-browser Botnet
8. DOM-based XSS (Type "0" XSS) (Document Object Model)
9. TOR Hidden-service Passive De-cloaking
10. HTML5 Hard Disk Filler API
11. Serialized YAML Remote Code execution

A Bit On Mobile Security

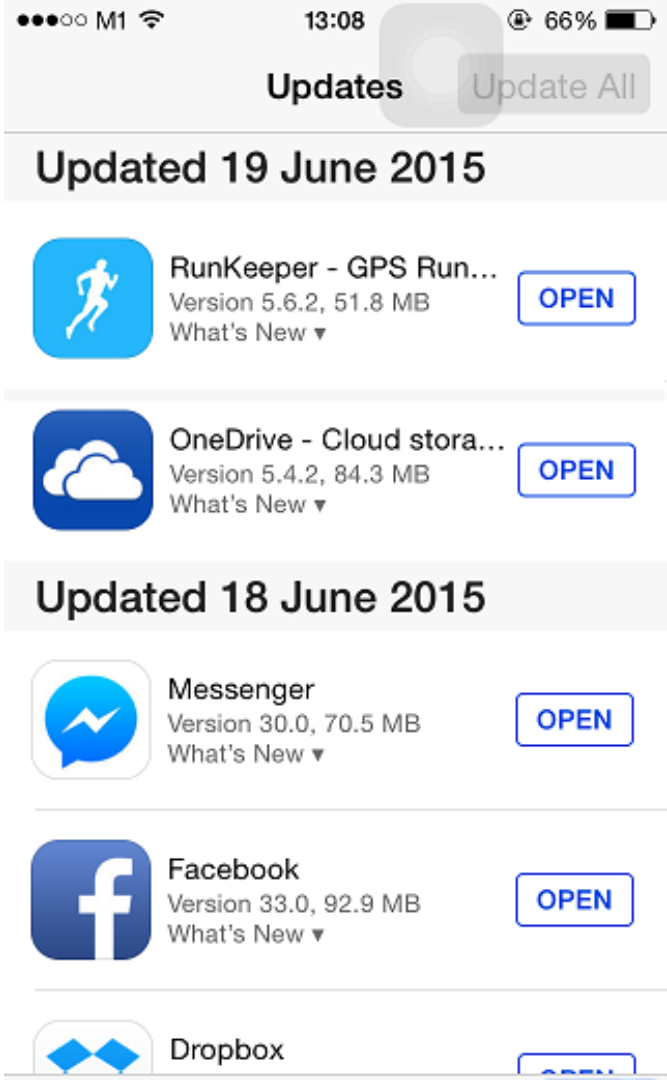
It's not always YOUR phone data the hacker is after …

- Sync
- Charge

cable

MDM

1 Passcode
2 Multiple O/S
3 Email
4 Apps
5 Remote Wipe
6 3G/4G & WLAN
7 Encryption
8 Security Policies
9 Tracking



OWASP Mobile Top 10 - Software Security Issues

1. Weak Server Side Controls
2. Insecure Data Storage
3. Insufficient Transport Layer Protection
4. Unintended Data Leakage
5. Poor Authorization and Authentication
6. Broken Cryptography
7. Client Side Injection
8. Security Decisions Via Untrusted Inpu
9. Improper Session Handling
10. Lack of Binary Protections

MORE SOFTWARE
"IOT" – INTERNET OF THINGS

Vehicle,asset,person & pet monitoring & controlling

Agriculture automation

Energy consumption

Security & surveillance

Building managment

Embedded

Mobile

Internet of things

Everyday things get connected for smarter tomorrow

M2M & wireless sensor network

Everyday things

Smart homes & cities

Telemedicine & helthcare

SMART CITIES

More Software

Information Collection

Traffic Data Center & Data Exchange

Applications, services

Home Broadband Router Firmware Risks

NSHC.net

HUAWEI

PWN3D!

TOTO LINK
The Smartest Network Device

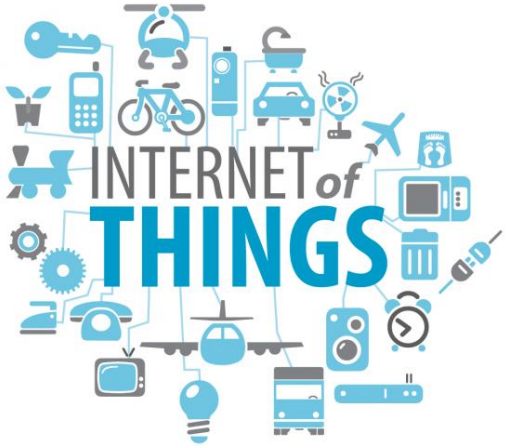
(ISC)

OWASP IoT Top 10 (Software Security Issues)

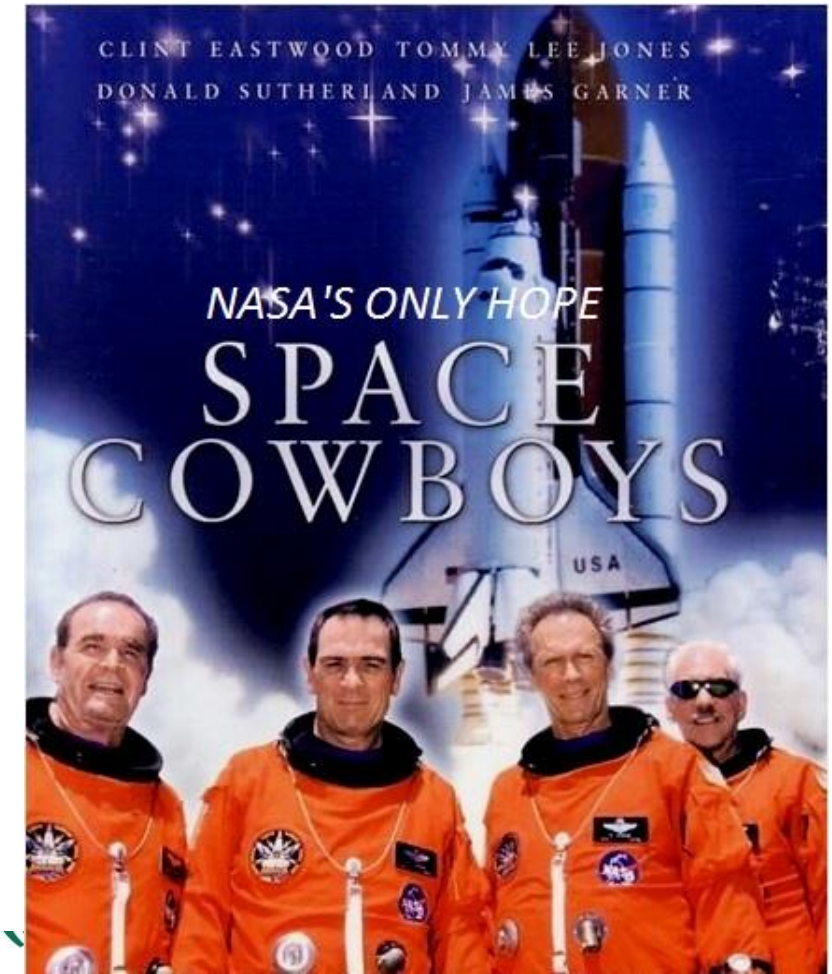
The Open Web Application Security Project (OWASP) has the primary intent to divulge best practices to improve the security of software. It is natural that the project also analyzed the top 10 security issues related to the popular paradigm.

OWASP
The Open Web Application Security Project

1. Insecure Web Interface
2. Insufficient Authentication/Authorization
3. Insecure Network Services
4. Lack of Transport Encryption
5. Privacy Concerns
6. Insecure Cloud Interface
7. Insecure Mobile Interface
8. Insufficient Security Configurability
9. Insecure Software/Firmware
10. Poor Physical Security



OLD" & "Other" Software ISSUES = NEW RISKS



- HEARTBLEED
- SHELLSHOCK /BASH
- "BEAST" attack

(Browser Exploit Against SSL & TLS)
- Lets hacker get plain text from encrypted traffic

- "POODLE" (Oracle)

- Tricks your app /service into lowering encryption

- Old O/S, eg. MS Win XP, Android 3
- Apache / Samba Server, O/S Kernel
- Adobe Flash, Apple Quicktime,
- Wordpress
- Other Middleware / Legacy systems
- ICS / SCADA

Whitehat Top 10 Web App Attacks 2014

1. Heartbleed
2. ShellShock
3. Poodle
4. Rosetta Flash
5. Residential Gateway “Misfortune Cookie”
6. Hacking PayPal Accounts with 1 Click
7. Google Two-Factor Authentication Bypass
8. Apache Struts ClassLoader Manipulation Remote Code Execution and Blog Post
9. Facebook hosted DDOS with notes app
10. Covert Timing Channels based on HTTP Cache Headers

Some SCADA /ICS Software Security Issues

"Black Box" issues

1. Security patches can't be applied to applications and operating systems.
2. You can't install anti-malware on the endpoint, HIPS or firewalls.
3. You can't perform system hardening.
4. Often come built-in with Sample programs & Debugging code.
5. Use of old, outdated, legacy operating systems eg. Win XP or earlier.
6. More, newer and faster equipment introduced with software written with security as only an afterthought, if at all.
7. Use of proprietary protocols.
8. No proper and documented configuration management processes.

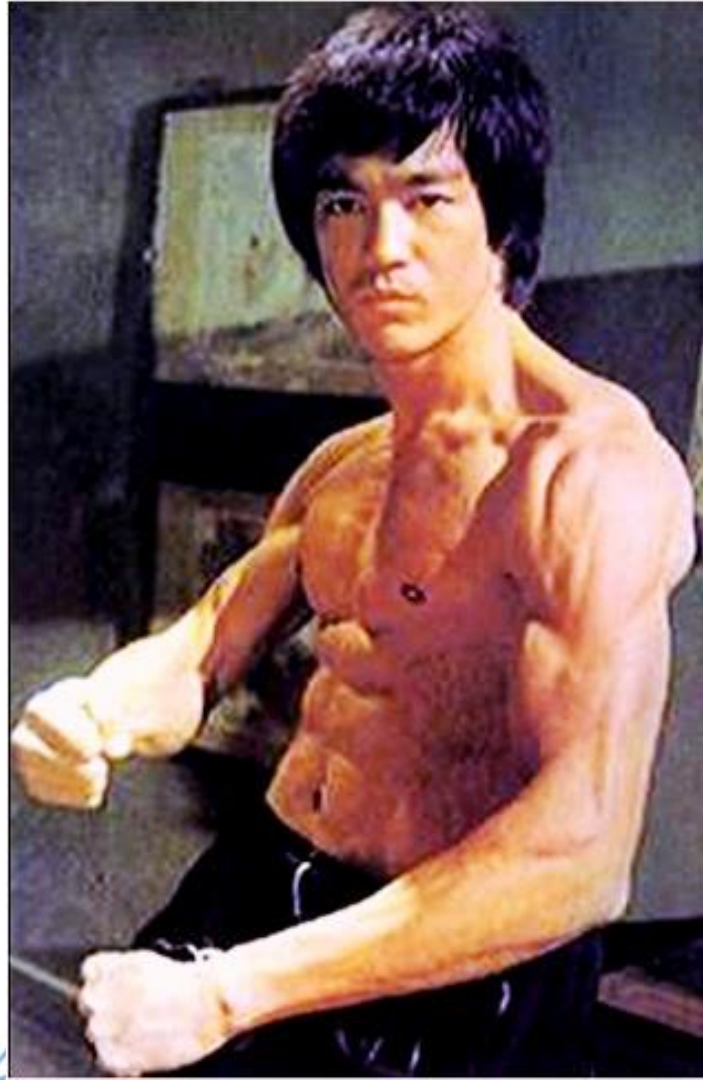
A Trip Down Memory Lane …

Trends?! We Never Learn?!

OWASP Top 10 2004	2007
1. Unvalidated Input	1. XSS
2. Broken Access Control	2. Injection Flaws
3. Broken Authentication & Session Management	3. Malicious File Executing
4. Cross Site Scripting (“XSS”)	4. Insecure Direct Object Reference
5. Buffer Overflow	5. Cross Site Request Forgery (CSRF)
6. Injection Flaws	6. Data Leakage & Improper Error Handling
7. Improper Error Handing	7. Broken Authentication & Session Management
8. Insecure Storage	8. Insecure Cryptographic Storage
9. Application Denial of Service	9. Insecure Communications
10. Insecure Configuration Management	10. Failure to restrict URL Access

Conclusion: Security by Application Development Quality Assurance (QA)

- **The Application Must Defend Itself**
  - **(ie. Write the programs properly)**
  - **Network security solutions do not stop application attacks**
  - **Existing network security solutions do not automatically work well in cloud environments**
  - THIS IS THE BEST AND ONLY WAY TO MINIMISE SOFTWARE ATTACKS
  - Both security and development teams need to be in harmony
  - DEVELOPERS NEED TO BE TRAINED APPROPRIATELY IN SECURE CODING
  - MANAGEMENT MUST ACTIVELY SUPPORT AND FINANCE A SOFTWARE SECURITY POLICY, RESOURCE AND ONGOING PRACTICE



RSAConference2015

Singapore | 22-24 July | Marina Bay Sands

(ISC)² INSPIRING A SAFE AND SECURE CYBER WORLD.

FROST & SULLIVAN

Anthony Lim

Isc2.org

Frost.com

THANK YOU

Security Risks in SDN and other new software issues

#RSAC

RSAConference2015

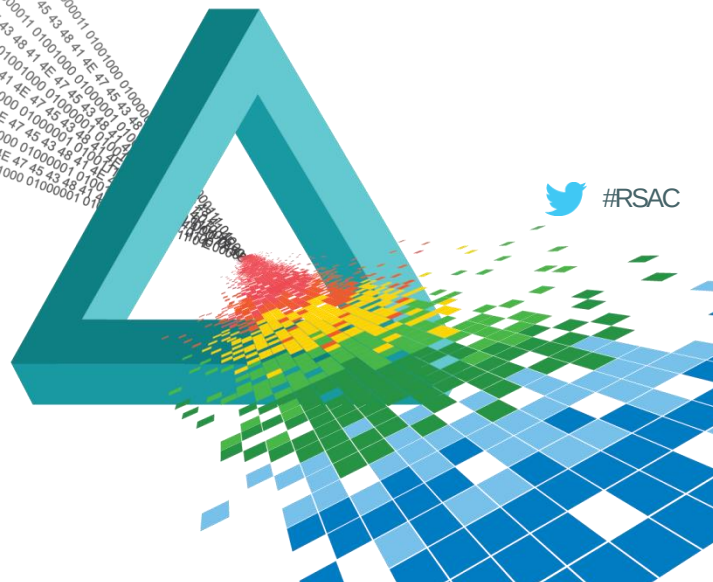
Singapore | 22-24 July | Marina Bay Sands

SEC-RO1

Security Risks in SDN and Other New Software Issues

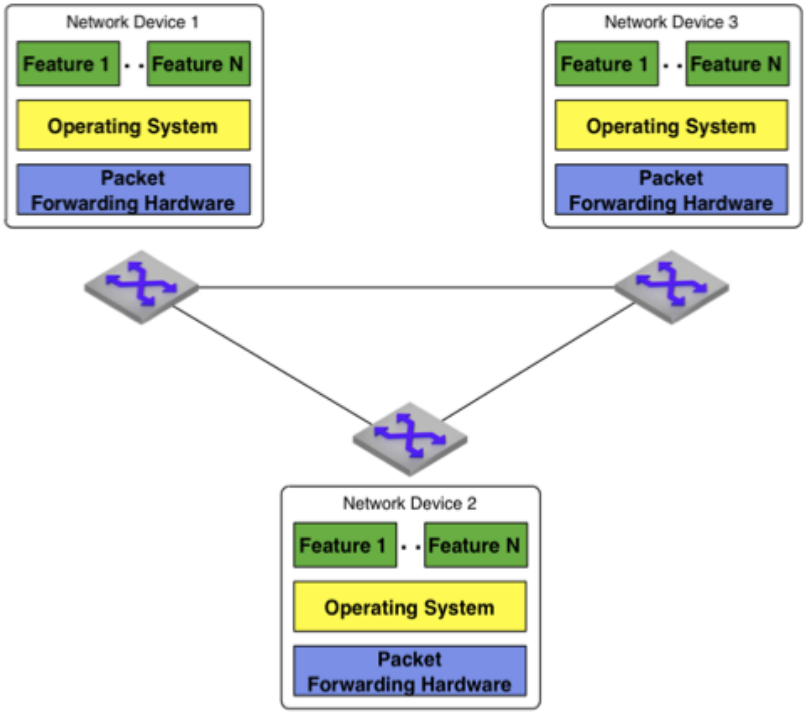
Anthony Lim
Isc2.org
Frost.org

APPENDIX

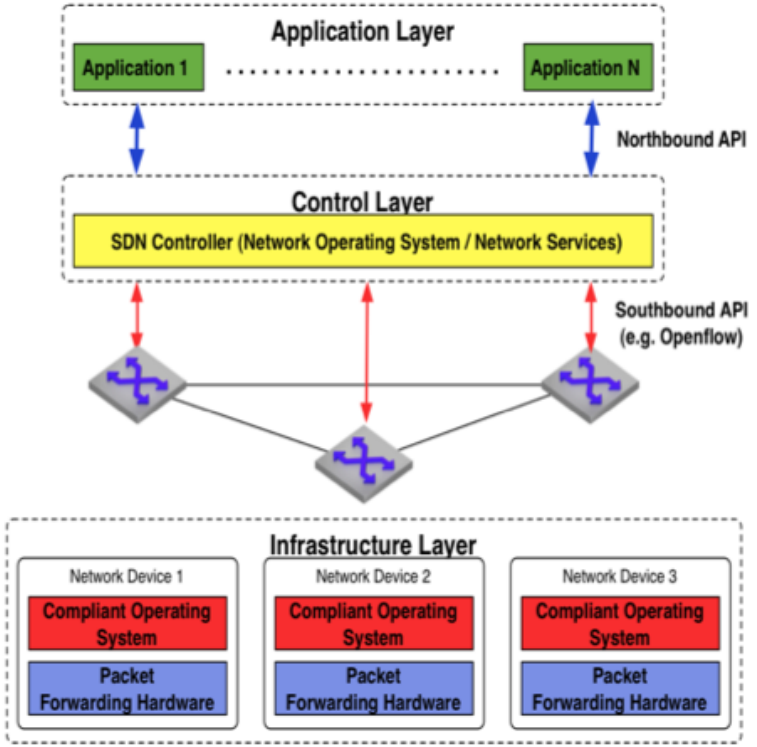


SOFTWARE-DEFINED NETWORK (SDN)

Traditional Scheme

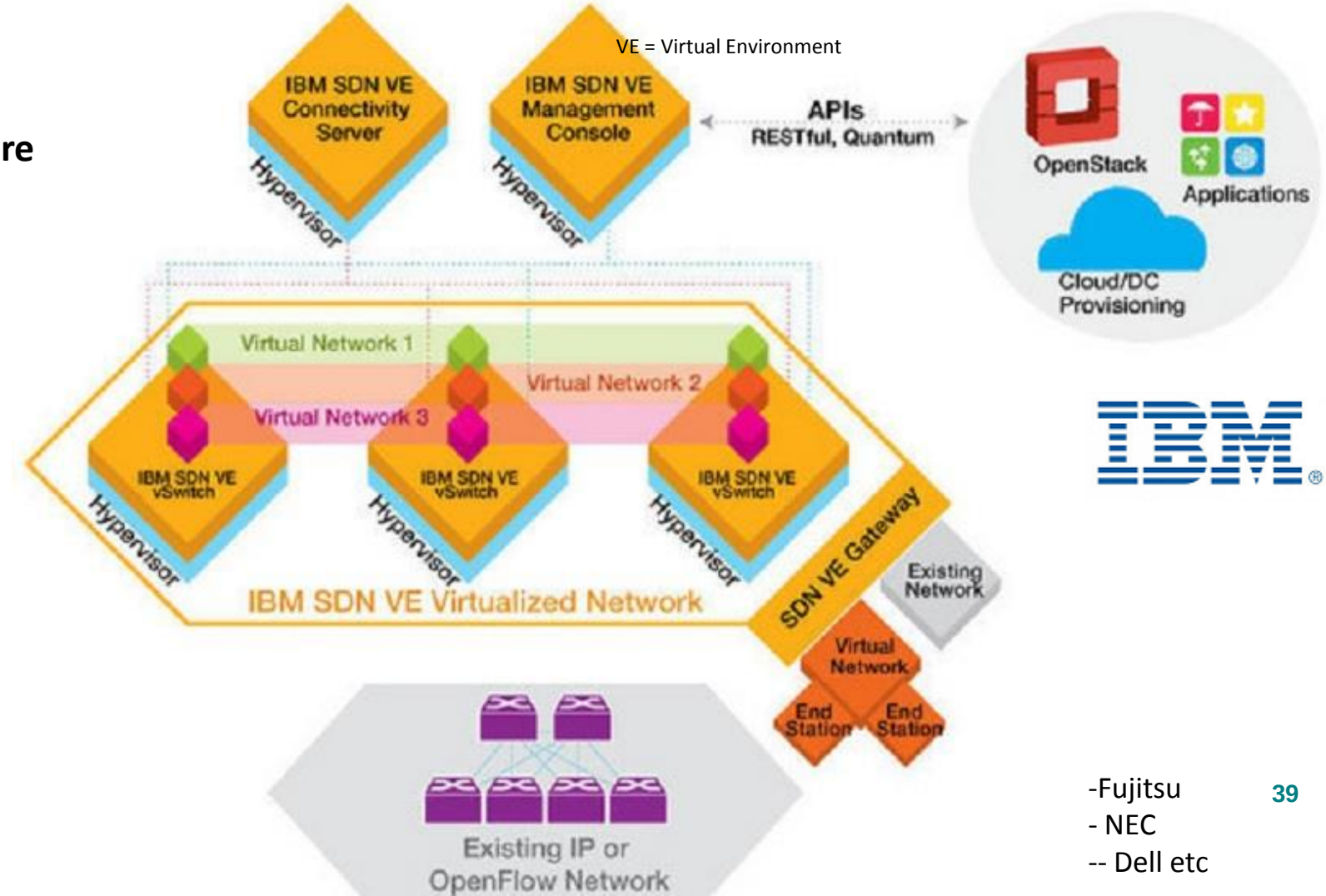


SDN Architecture



VE = Virtual Environment

IBM SDN Architecture

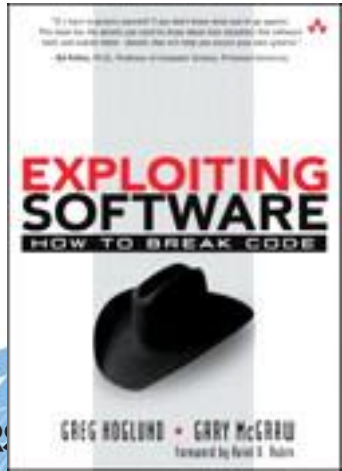


-Fujitsu
- NEC
-- Dell etc

Hackers Today Target Software (eg. Websites)

- **Because they know you have firewalls.**
  - So they need to find a new weak spot to hack
- **Because firewalls do not protect against app attacks!**
  - few people are actively aware of application security issues
  - Most IT security professionals, from network & sys-admin side, have little experience or interest in software development. Programmers have little experience or interest in security or infrastructure.
  - IT security staff are also often overworked and are focusing on other issues
- **Because they can!**
  - Many organizations today still lack a software development security policy!
  - Many applications, including legacy ones still in use, were not built defensively
  - Applications today are hundreds of thousands of lines long. QA is tedious and detailed, ; many will skip or procrastinate; focus of app-dev QA is on feature, function and performance. Security takes 2nd fiddle.
- **Additional loss of control when outsourcing software development work.**

CHEAP
FAST
GOOD



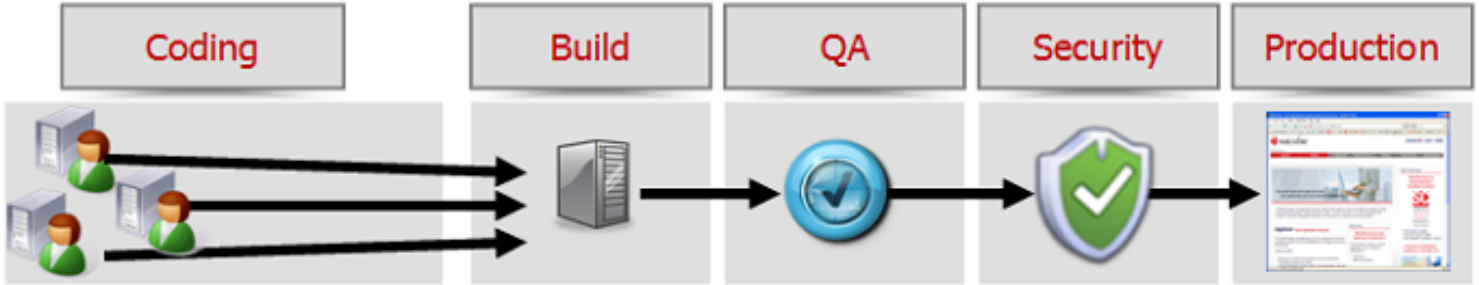
Application Vulnerabilities can start in Development

Developers Lack Security Insights
(or Incentives to Address Security)

- Mandate to deliver functionality on-time and on-budget – but not to develop secure applications
- Developers rarely educated in secure code practices
- Product innovation drives development of increasingly complicated applications

Security Team = SDLC Bottleneck

- Security tests executed just before launch
  - Adds time and cost to fix vulnerabilities late in the process
- Growing number of web applications but small security staff
  - Most enterprises scan ~10% of all web apps
- Continuous monitoring of production apps limited or non-existent
  - Unidentified vulnerabilities & risk



Challenge to share test results and enable self-testing in the SDLC

SOME MOBILE APP SECURITY ISSUES

1. Insecure Data Storage

Eg. A Starbucks app was storing usernames, email addresses, and passwords in clear text.

Design apps in such a way that critical information such as passwords and credit card numbers do not reside directly on a device. If they do, they must be stored securely

2. Weak server-side controls

When creating their first mobile applications, businesses often expose systems that had not previously been accessible from outside of their networks. Often, these formerly sheltered systems are not fully vetted against security flaws. A number of back-end APIs assume (quite wrongly) that an app will be only thing that will access it

3. Unintended Data Leakage

(Eg Angry Birds collect alot of user personal data)

Use caution when choosing analytics providers and implementing advertising. Watching what, how, when, and where data moves can give an attacker a gold mine of information.

4. Broken cryptography

(Weak or wrong algorithms, poor key management)

Many organizations make the mistake of using strong encryption algorithms, but implement their own keys and certificates in areas that are vulnerable to attackers

5. Security Decisions via untrusted Inputs

(Eg. Weak authentication can be bypassed).

Eg. a flaw in Skype security allowed hackers to open the Skype app and dial arbitrary phone numbers using a simple link in the contents of an email.

C Fairchild, InfoWeek

MOBILE APP SECURITY ISSUES

Not different from non-mobile app-dev issues

- **Development is focused on features not security**
- **Developers are unaware of the underlying platform**
- **Users don't even have security on their radar**
- **Users are easily social engineered**

Reversing Android Apps

**** Android apps are written in Java***

- You can use your favorite IDE with a freely downloadable Android SDK plugin (eg. Eclipse)
- Like (unobfuscated) Java apps, they can be easily reversed with the right tools
- bytecode can even be altered and apps repackaged

ATTACKING IOS DEVICE

- iOS strictly enforces application boundaries and sandboxing

The sandbox now can be compromised

- Apps cannot communicate directly from other apps, or access the application directories of other apps

- Written in native ObjectiveC or even C (with the right tools)

- Based on an ARM version of the same XNU kernel from OSX

- Once you breach the walls of the fort, you own the place….

First you must Jailbreak the device -

- Involves finding an exploit in the kernel as well as userland to allow it to run unsigned code

- Use tools like Absinthe, redsn0w limera1n to do the jailbreaking

OWASP Top 10 : 2010 and 2013

OWASP The Open Web Application Security Project

#RSAC

2010

1. Injection
2. Cross-site Scripting (XSS)
3. Broken Authentication & session Management
4. Insecure Direct Object References
5. Cross Site Request Forgery (CSRF)
6. Security Misconfiguration
7. Insecure Cryptographic Storage
8. Failure to restrict URL Access
9. Insufficient Transport Layer Protection
10. Unvalidated Redirects & Forwards

2013

1. Injection
2. Broken Authentication & Session Management
3. XSS
4. Insecure Direct Object References
5. Security Misconfiguration
6. Sensitive Data Exposure
7. Missing Function Level Access Control
8. CSRF
9. Using Components with Known Vulnerabilities
10. Unvalidated Redirects & Forwards