Chapitre 1

Web basics and recent trends

Business Training

1

Outline

- Overview HTTP and related web development technologies
- Modern web application architecture : from monolithic apps to microservices
- DevOps practices and Cloud deployment
- Key security concepts
- Open Web Application Security Project
- Recent attack trends

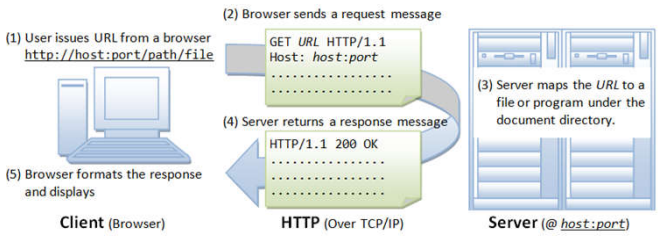
2

Overview HTTP and related web development technologies

3

HTTP ?

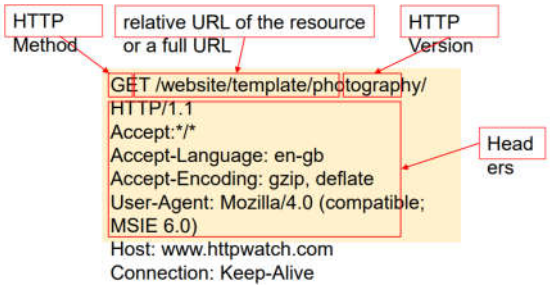
- **The Hyper Text Transfer Protocol (HTTP) is a client-server network protocol**
  - In use by the World-Wide Web since 1990.
  - It is based on Request – Response Paradigm
- **HTTP is a stateless protocol. In other words, the current request does not know what has been done in the previous requests.**
- **HTTP permits negotiating of data type and representation, so as to allow systems to be built independently of the data being transferred.**



4

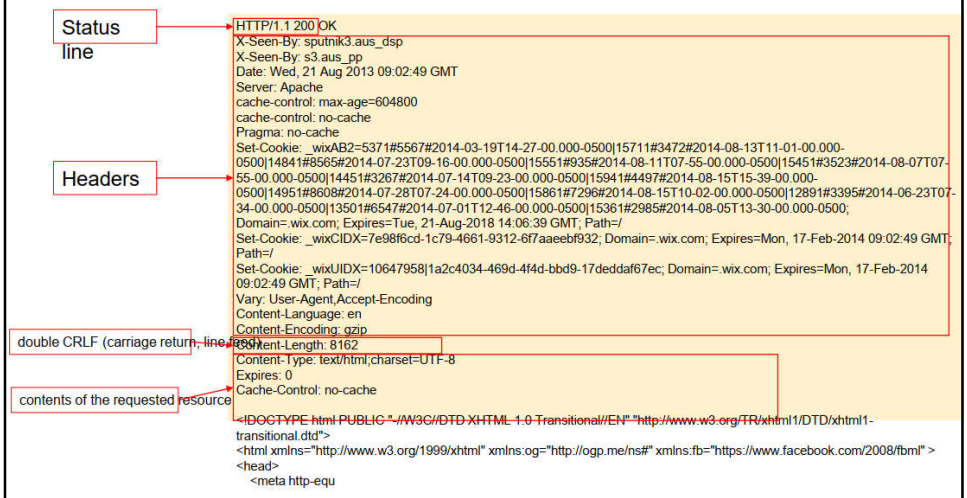
HTTP Request Message

- **The format of an HTTP request message is as follow:**
  - Request Line
  - Request Headers
  - Blank Line
  - Request Message Body (Optional)



5

HTTP Response



6

HTTP Methods

- **HTTP protocol defines a set of request methods. A client can use one of these request methods to send a request message to an HTTP server. The methods are:**
  - **GET**: A client can use the GET request to get a web resource from the server.
  - **HEAD**: A client can use the HEAD request to get the header that a GET request would have obtained. Since the header contains the last-modified date of the data, this can be used to check against the local cache copy.
  - **POST**: Used to post data up to the web server.
  - **PUT**: Ask the server to store the data.
  - **DELETE**: Ask the server to delete the data.
  - **TRACE**: Ask the server to return a diagnostic trace of the actions it takes.
  - **OPTIONS**: Ask the server to return the list of request methods it supports.
  - **CONNECT**: Used to tell a proxy to make a connection to another host and simply reply the content, without attempting to parse or cache it. This is often used to make SSL connection through the proxy.

HTTP Status Codes and Errors

- **1xx – Informational**
  - Intermediate response and indicates that the server has received the request but has not finished processing it.
- **2xx – Successful**
  - 200 OK
- **3xx – Redirection:**
  - 301-permanent, 302-temporary
- **4xx - Client Error:**
  - 400-bad request, 403-forbidden, 404-not found, 418 I'm a teapot
- **5xx - Server Error:**
  - 500 Internal Server Error,
  - 503-Service Unavailable,504-Gateway Timeout

HTTP Headers

- **HTTP Headers**
  - Accept: text/plain; - specify certain media types which are acceptable for the response.
  - Accept-Encoding: compress, gzip ;
  - Accept-Language: da, en-gb;q=0.8, en;q=0.7
  - Connection: Close|Keep-Alive - HTTP/1.1 uses persistent (keep-alive) connection by default. HTTP/1.0 closes the connection by default.
  - Referer: referer-URL –
  - User-Agent: browser-type
  - Content-Length: number-of-bytes - Used by POST request, to inform the server the length of the request body.
  - Content-Type: mime-type -
  - Cache-Control: no-cache|... - The client can use this header to specify how the pages are to be cached by proxy server. "no-cache" requires proxy to obtain a fresh copy from the original server, even though a local cached copy is available. (HTTP/1.0 server does not recognize "Cache-Control: no-cache". Instead, it uses "Pragma: no-cache". Included both request headers if you are not sure about the server's version.)
  - Authorization: Used by the client to supply its credential (username/password) to access protected resources. (This header will be described in later chapter on authentIcation.)
  - Cookie: cookie-name-1=cookie-value-1, cookie-name-2=cookie-value-2, ... - The client uses this header to return the cookie(s) back to the server, which was set by this server earlier for state management. (This header will be discussed in later chapter on state management.)
  - If-Modified-Since: date - Tell the server to send the page only if it has been modified after the specific date.

9

Cookies

- **Servers supply cookies by populating the set-cookie** response header with the following details:
  Set-Cookie: name=value
  - **Name** Name of the cookie
  - **Value** Textual value to be held by the cookie
  - **Expires** Date/time when the cookie should be discarded by the browser.
    If this field is empty the cookie expires at the end of the current browser session. This field can also be used to delete a cookie by setting a date/time in the past.
  - **Path** Path below which the cookie should be supplied by the browser.
  - **Domain** Web site domain to which this cookie applies.
    This will default to the current domain and attempts to set cookies on other domains are subject to the privacy controls built into the browser

10

Type of cookies - Terminology

- **Session cookie -** Web browsers normally delete session cookies when the user closes the browser
- **Persistent cookie -** A persistent cookie will outlast user sessions – expires on a set timestamp
- **Secure cookie -** A secure cookie has the secure attribute enabled and is only used via HTTPS
- **HttpOnly cookie -** On a supported browser, only when transmitting HTTP (or HTTPS) requests. Not accessible to Javascript.

11

HTTP Security Headers

- **HTTP security headers that tell your browser how to behave when handling your website's content.**
- **Here some Security Headers**
  - Content Security Policy
  - X-XSS-Protection
  - X-Frame-Options
  - X-Content-Type-Options
  - HTTP Strict Transport Security (HSTS)
  - HTTP Public Key Pinning (HPKP)
  - https://www.netsparker.com/whitepaper-http-security-headers/

12

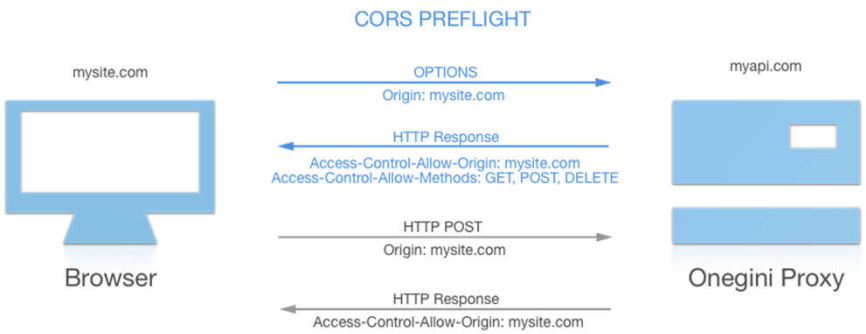
HTTP CORS

- **The Same-origin policity (SOP)**
  - **The same-origin policy is very restrictive**. Under this policy, a document (i.e., like a web page) hosted on server A can only interact with other documents that are also on server A. In short, the same-origin policy enforces that documents that interact with each other have the same origin.
- **WHAT IS CORS?**
  - A request for a resource (like an image or a font) outside of the origin is known as a ***cross-origin*** request. CORS (cross-origin resource sharing) manages cross-origin requests.
    - With CORS, a server can specify who can access its assets and which HTTP request methods are allowed from external resources.
- **HOW DOES CORS MANAGE REQUESTS FROM EXTERNAL RESOURCES?**
  - Headers are passed back and forth between your web browser (also referred to as a client) and a server when the web page you are on wants to use resources hosted on a different server.
    - Access-Control-Allow-Origin / Access-Control-Allow-Credentials/
    - Access-Control-Allow-Headers/ Access-Control-Allow-Methods
    - Access-Control-Expose-Headers / Access-Control-Max-Age
    - Access-Control-Request-Headers / Access-Control-Request-Method/ Origin

CORS, How it works ?

- **This mechanism prevents attackers that plant scripts on various websites (eg. in ads displayed via Google Ads) to make an AJAX call to www.yourbank.com and in case you were logged in making a transaction using *your* credentials.**
  - If the server does not respond with specific headers to a “simple” GET or POST request — it will still be send, the data still received but the browser **will not allow JavaScript to access the response**.
  - If your browser tries to make a “non simple” request (eg. an request that includes cookies, or which Content-type is other than application/x-ww-form-urlencoded, multipart/form-data or text-plain) an mechanism called preflight will be used and an **OPTIONS** request will be sent to the server.
    - An common example of “non simple” request is to add cookies or custom headers — it your browser sends such a request and the server does not respond properly, only the **preflight call** will be made (without the extra headers) but the actual HTTP request the brower meant to make will not be sent.
- **Access-Control-Allow-What?**
  - **Access-Control-Allow-Origin**
    - This header is meant to be returned by the server, and indicate what client-domains are allowed to access its resources.
  - **Origin**
    - This header is part of the request that the client is making, and will contain the domain from which the application is started. For security reasons browsers will not allow you to overwrite this value.

CORS Preflight



15

HTTP Caching

- **Preventing Caching** Cache-Control: no-cache (HTTP 1.1); Pragma: no-cache (HTTP 1.0)
- Last-Modified: Wed, 15 Sep 2004 12:00:00 GMT – Browser can check the server for changes
- Expires: Sun, 17 Jan 2038 19:14:07 GMT - browser can reuse the content without having to check the server

16

HTTPS

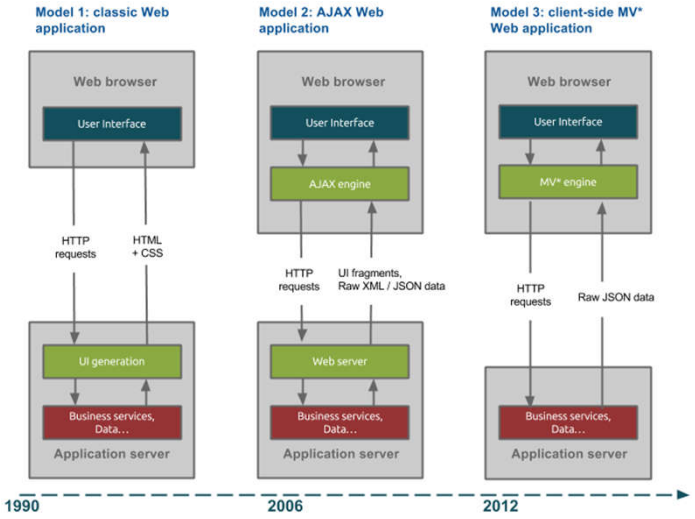
- The SSL/TLS was designed to encrypt any TCP/IP based network traffic and provide the following capabilities
  - Prevents sniffing
  - Prevents tampering or replaying of messages
  - Uses certificates to authenticate servers and optionally clients
  - The HTTPS protocol is the same text based protocol as HTTP but is run over an encrypted SSL session.

17

Modern web application architecture: from monolithic apps to microservices

18

Web Applications Evolution

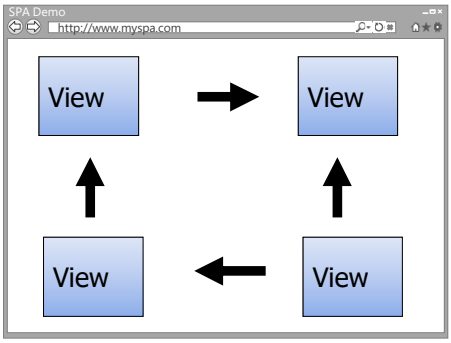


[http://blog.octo.com/les-nouvelles-architectures-front-web-et-leur-impact-sur-les-dsi-partie-1/]

19

Single Page Application (SPA)

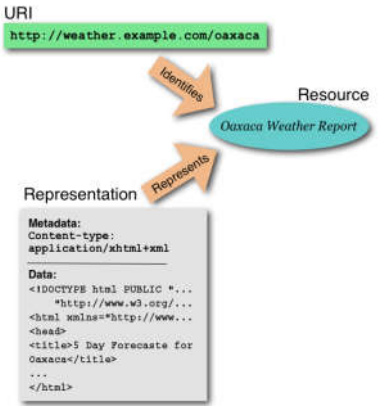
- **Advantages of Single Page Apps**
  - Limit page requests for UX
  - Load content up front (bundled)
  - Load additional data through async requests
  - Route-first
  - Data-binding
  - Module management



20

What is REST ?

- **State Transfer (*REST*) is a style of *architecture* based on a set of principles that describe how networked resources are defined and addressed.**
  - These principles were first described in 2000 by Roy Fielding as part of his doctoral dissertation.
- **A REST API should spend almost all of its descriptive effort in**
  - **defining the media type(s) used for representing resources and driving application state, or in**
  - **defining extended relation names and/or hypertext-enabled mark-up for existing standard media types.**



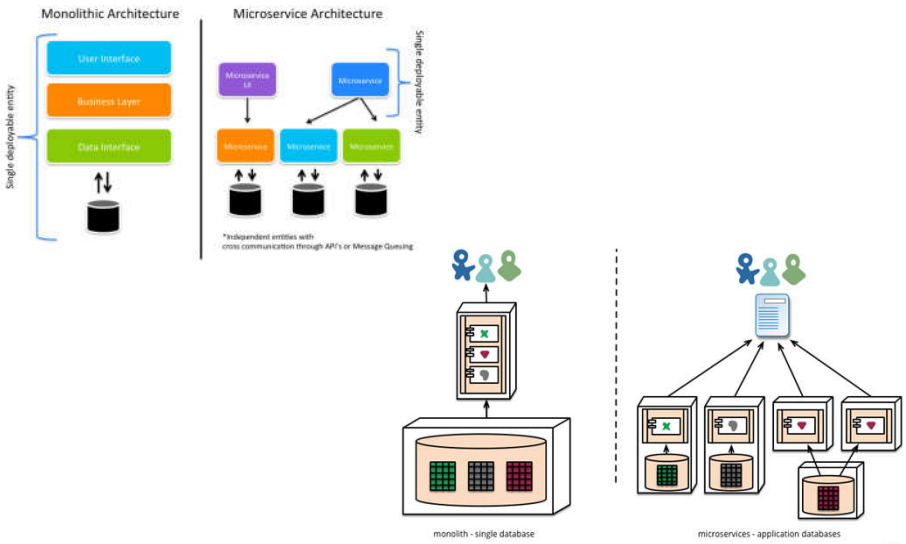
21

HTTP Verbs (CRUD)

Resource	GET (Read)	POST (Create)	PUT (Update)	DELETE (Delete)
/users	Returns a list of users	Creates a new user	Bulk update of users	Delete all users
/users/123	Returns a specific User	Method not allowed (405)	Updates a specific user	Deletes a specific user

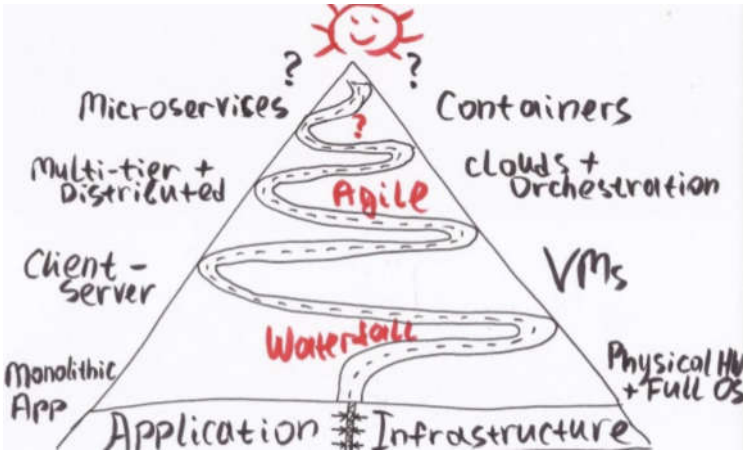
22

Architecture: From Monoliths to Microservices



23

Software deployment



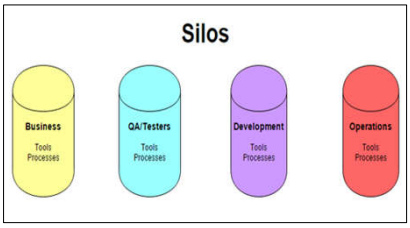
24

DevOps practices and Cloud deployment

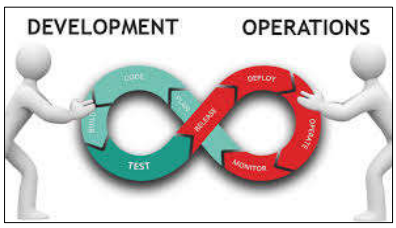
25

Breaking Down the silos!

- **Silos inhibit collaboration and reinforce the waterfall methodology**
  - A cultural movement in as much as it is a development cycle [1]
    - "…that's not my Job!" "That's is my Job, stay out!..." (Ownership ideology)
    -
  - Everyone becomes a first class citizen and works collaboratively across the disparate areas: Requirements->Design->Implementation->verification>maintenance
    - e.g. Operations (security staff) should be involved in requirements and design phases else domain expertise is lost!



[2] - http://www.agiletestingframework.com/devops-getting-old-deja-vu-feeling/



[3] - http://www.agilebuddha.com/agile/demystifying-devops

26

DevOps Core Principles

- **Collaboration**
  - Efficiency (communication and processes) between disparate roles, both within teams between teams in an organizations
- **Infrastructure as Code (IaC)**
  - Script and version control as much of infrastructure as possible, principally for capturing the disparate environment details (.e.g. development, testing, production, etc.)
    - Goal: (repeatable, recoverable, reusable, sharable)
- **Automation**
  - Automate human (error processes) wherever possible
    - Goal: Increase frequency of testing, continuous (consistent/uniform) integration, delivery, and deployment processes
- **Monitoring**
  - Data to needed to inform development/operational decisions
    - policies and priorities are domain dependent
      - Is the System working? Performance? Are users consuming the system as expected

Source: https://www.sei.cmu.edu/education-outreach/courses/course.cfm?coursecode=V38

27

DevOps Toolbox

https://hostadvice.com/blog/devops-toolbox-jenkins-ansible-chef-puppet-vagrant-saltstack/

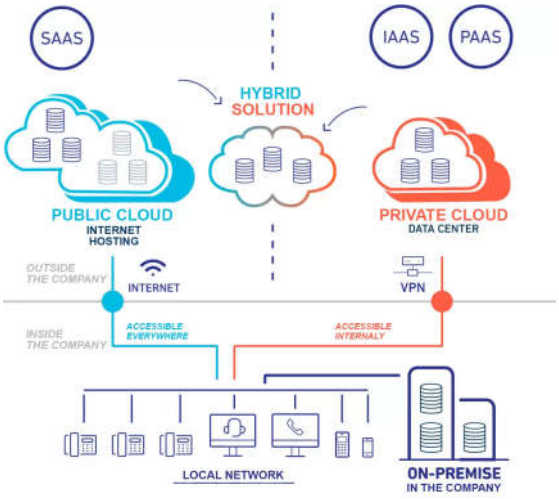
28

Cloud Deployment

- **The 5 characteristics of cloud computing are:**
  - On-demand self-service;
  - Broad network access;
  - Resource pooling;
  - Rapid elasticity;
  - Measured service to know its real consumption.
- **There are 3 models of cloud computing services that are:**
  - **IaaS (Infrastructure as a Service)**: rental of the hardware IT infrastructure;
  - **PaaS (Platform as a Service)**: rental of hardware infrastructure but also middleware applications;
  - **SaaS (Software as a Service)**: Cloud service all-inclusive, accessible via a web browser.
- **There are 3 main deployment models:**
  - Private
  - Public
  - Hybrid

29

Hybrid Cloud



https://www.padok.fr/en/blog/devops-cloud-migration

30

Are Cloud and DevOps complementary ?

- **Cloud and DevOps are independent and yet they hardly work separately.... but the Cloud is boosting DevOps.**
- **This winning combo Cloud and DevOps have several advantages:**
  - **Continuous improvement...**
    - Production cycles must be reduced. To do this, the DevOps culture needs tools that allow this flexibility and speed of execution. Cloud providers such as Amazon Web Services (AWS), Google Cloud Platform (GCP), OVH and Azure offer DevOps tools directly integrated into their services (Source Code Managers, etc.).
  - **The Cloud, a flexible solution**
    - With the SaaS model, the software is accessible everywhere, at any time, all you need is an internet connection. This allows your development and operational teams to work from anywhere. With the democratization of remote, it is a factor to be taken into account.
- **The combination of DevOps and Cloud has benefited European companies. Indeed, according to a CA Technologies study, they recorded a 129% increase in their deployment performance. They also noted a 108% improvement in their customer experience compared to traditional deployment models. They also announce having better control over the costs and resources used by DevOps teams.**
  [https://www.ca.com/content/dam/ca/us/files/msf-hub-assets/research-assets/devops-cloud-computing-exploiting-synergy-business-advantage.pdf]

31

Key Security Concepts

32

What Is Security?

- **"The quality or state of being secure – to be free from danger"**
  - To be protected from adversaries
- **Security refers to techniques for ensuring that information /data stored in a computer cannot be read or compromised by any individuals without authorization or**
  - The protection of information assets through the use of technology, processes, and training

33

Definition Cont…

- **"A computer is secure if you can depend on it and its software to behave as intended."**
- **A secure system is a system which does exactly what we want it to do even when someone else tries to make it behave differently.**
- **To achieve this we have to understand all the components of some service infrastructure and their interrelationships.**

34

Fundamental goals

Systems security has fundamental goals;

1. **Privacy**: keep private documents private using encryption, password and access controls.
2. **Integrity** :data and applications should be safe from modification without owner's consent
3. **Authentication** : ensure that people using the computer are authorized users of the system
4. **Availability** : the data and information should be available when needed by authorized users.

35

maybe THE main principle is…

- Always go for Defense in Depth.
- Never trust any single device.
- There is no such thing as the one tool to secure everything.
- You always have to remain vigilant and adapt things.

36

The reality is;...

- There is absolutely no such thing as 'absolute security'.
- Security is a never-ending process, a goal to strive for. Not a stable state!
- Security is a challenge to every organization
- Always a balance between usability and security

37

Three key objectives (the CIA triad)

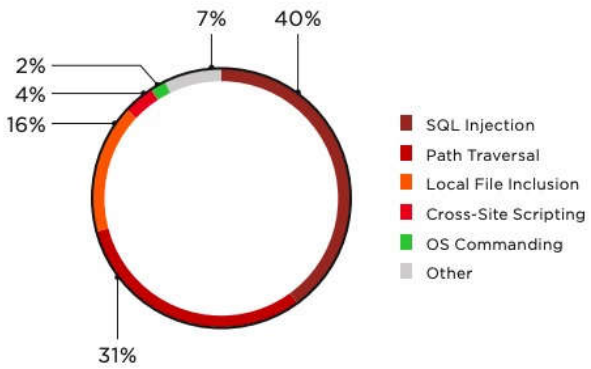
- **Confidentiality**
  - **Data confidentiality**: Assures that confidential information is not disclosed to unauthorized individuals
  - **Privacy**: Assures that individual control or influence what information may be collected and stored
- **Integrity**
  - **Data integrity**: assures that information and programs are changed only in a specified and authorized manner
  - **System integrity**: Assures that a system performs its operations in unimpaired manner
- **Availability: assure that systems works promptly and service is not denied to authorized users**

38

Recent attack trends

39

Top five attacks on web applications of IT companies



https://www.ptsecurity.com/ww-en/analytics/web-application-attacks-2019/

40

Security Trends

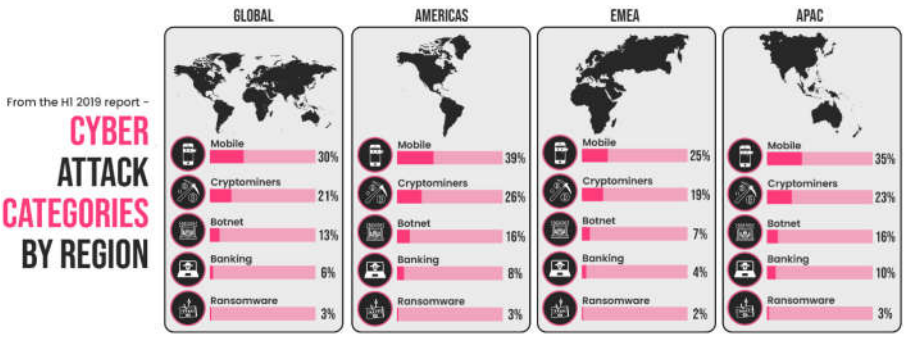
- Trend #1: The phishing landscape is changing, though email still ranks as the biggest of those threats
- Trend #2: Increasing use of mobile as an attack vector
- Trend #3: Targeting of local governments and enterprises via ransomware attacks
- Trend #4: Increasing emphasis on data privacy, sovereignty, and compliance
- Trend #5: Increasing investments in cyber security automation

https://www.thesslstore.com/blog/the-top-cyber-security-trends-in-2019-and-what-to-expect-in-2020/

41

Checkpoint Mid 2019 Security Report

- Cyber Attack categories by region

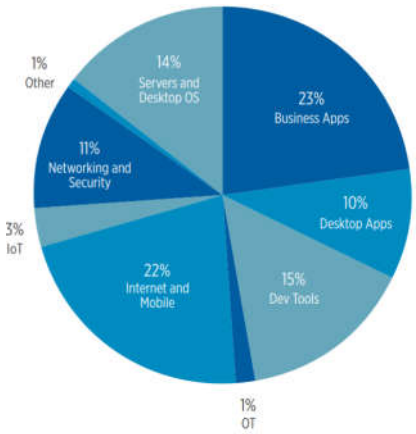


https://research.checkpoint.com/cyber-attack-trends-2019-mid-year-report/

42

2018 Vulnerabilities by Category

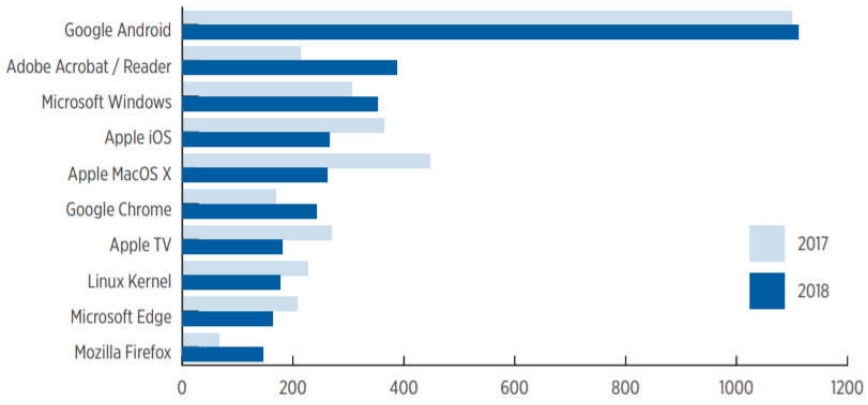
- **2018 Vulnerabilities by category**



https://lp.skyboxsecurity.com/rs/440-MPQ-510/images/Skybox_Report_Vulnerability_and_Threat_Trends_2019.pdf

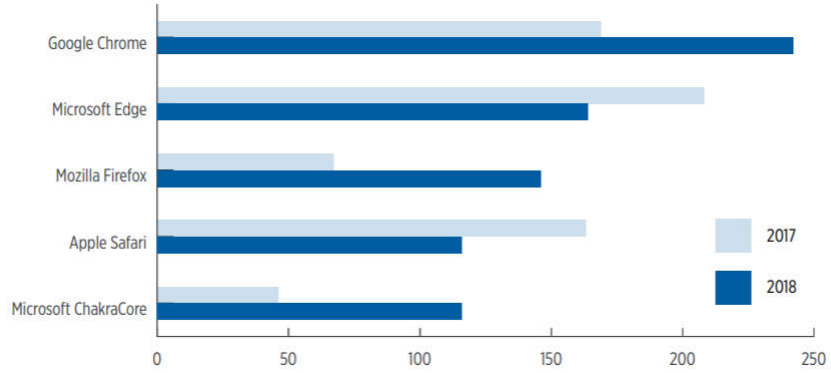
Top 10 Most Vulnerable Products

- **Vendors with the most newly published vulnerabilities**



Web browsers still favored by attackers

- **Browsers with the most newly published vulnerabilities**



45

Ransomware prevention

- **These are tips on how to prevent ransomware attacks**
  - Never click on unverified links
    - Avoid clicking links in spam emails or on unfamiliar websites. Downloads that start when you click on malicious links is one way that your computer could get infected.
  - Do not open untrusted email attachments
    - Do not open email attachments from senders you do not trust. Look at who the email is from and confirm that the email address is correct. Be sure to assess whether an attachment looks genuine before opening it. If you're not sure, contact the person you think has sent it and double check.
  - Only download from sites you trust
    - To reduce the risk of downloading ransomware, do not download software or media files from unknown. Most reputable websites will have markers of trust that you can recognize. Just look in the search bar to see if the site uses 'https' instead of 'http.' A shield or lock symbol may also show in the address bar to verify that the site is secure.
  - Avoid giving out personal data
  - Use mail server content scanning and filtering
  - Never use unfamiliar USBs
  - Keep your software and operating system updated
  - Backup your data

46