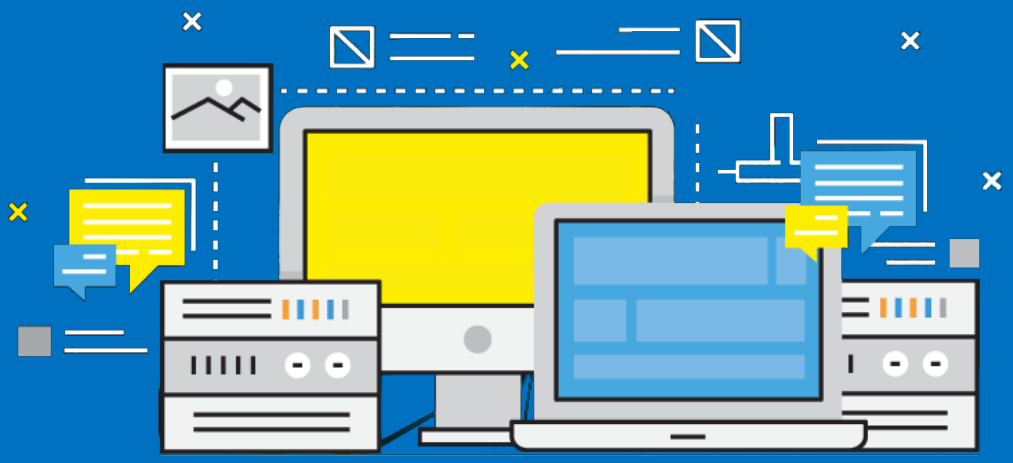
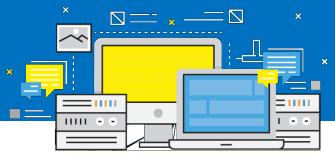


Applying machine learning to cybersecurity

Diego Soto & Luis Marqueta

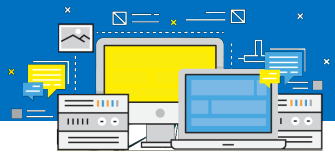
TECDAY #1 '18



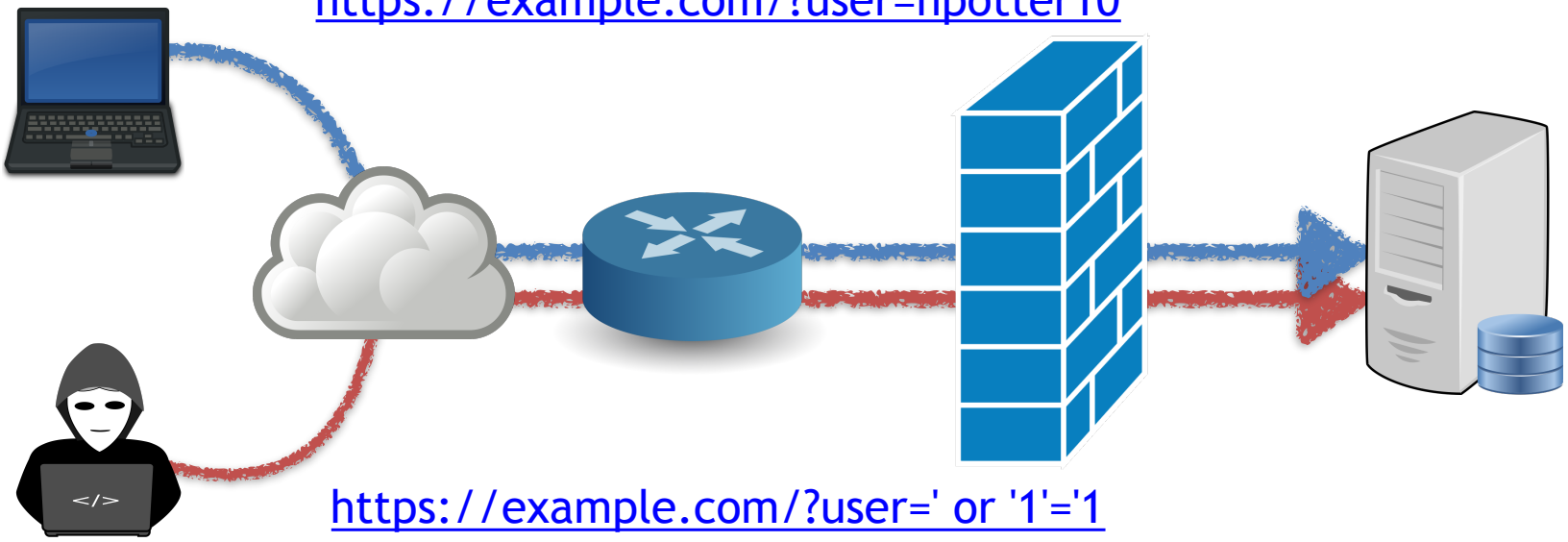
WHAT IS A WAF?

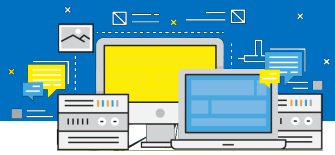
OWASP:

A web application firewall (WAF) is an application firewall for HTTP applications. It applies a set of rules to an HTTP conversation. Generally, these rules cover common attacks such as cross-site scripting (XSS) and SQL injection.

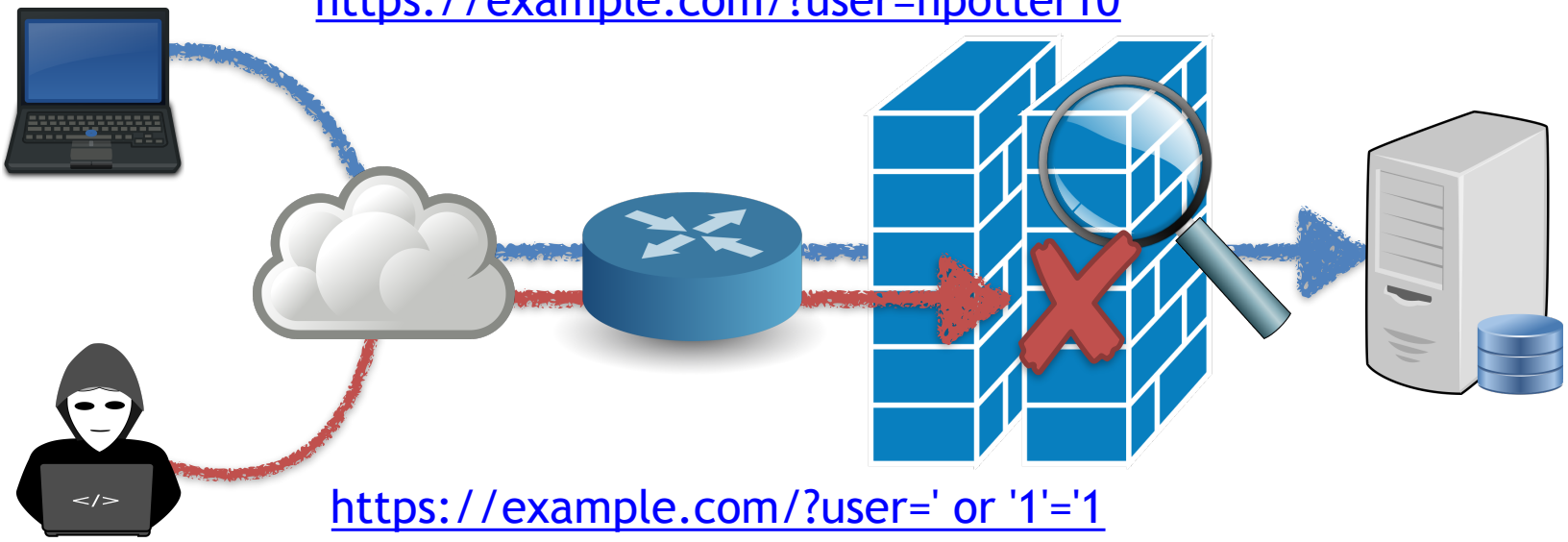


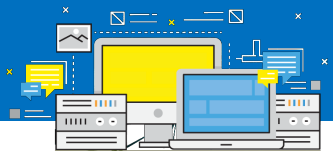
WTF IS A WAF?


https://example.com/?user=hpotter10
</>
https://example.com/?user=' or '1'='1



WTF IS A WAF?


https://example.com/?user=hpotter10
https://example.com/?user=' or '1'='1



WAF PROTECTION

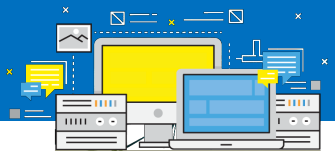
- HTTP flood
- Slow loris
- OWASP top 10
- Compliance
  - *Installing an automated technical solution that detects and prevents web-based attacks (for example, a web-application firewall) in front of public-facing web applications, to continually check all traffic*

PCI DSS COMPLIANT

T10 OWASP Top 10 Application Security Risks – 2017

6

A1:2017-Injection	Injection flaws, such as SQL, NoSQL, OS, and LDAP injection, occur when untrusted data is sent to an interpreter as part of a command or query. The attacker's hostile data can trick the interpreter into executing unintended commands or accessing data without proper authorization.
A2:2017-Broken Authentication	Application functions related to authentication and session management are often implemented incorrectly, allowing attackers to compromise passwords, keys, or session tokens, or to exploit other implementation flaws to assume other users' identities temporarily or permanently.
A3:2017-Sensitive Data Exposure	Many web applications and APIs do not properly protect sensitive data, such as financial, healthcare, and PII. Attackers may steal or modify such weakly protected data to conduct credit card fraud, identity theft, or other crimes. Sensitive data may be compromised without extra protection, such as encryption at rest or in transit, and requires special precautions when exchanged with the browser.
A4:2017-XML External Entities (XXE)	Many older or poorly configured XML processors evaluate external entity references within XML documents. External entities can be used to disclose internal files using the file URI handler, internal file shares, internal port scanning, remote code execution, and denial of service attacks.
A5:2017-Broken Access Control	Restrictions on what authenticated users are allowed to do are often not properly enforced. Attackers can exploit these flaws to access unauthorized functionality and/or data, such as access other users' accounts, view sensitive files, modify other users' data, change access rights, etc.
A6:2017-Security Misconfiguration	Security misconfiguration is the most commonly seen issue. This is commonly a result of insecure default configurations, incomplete or ad hoc configurations, open cloud storage, misconfigured HTTP headers, and verbose error messages containing sensitive information. Not only must all operating systems, frameworks, libraries, and applications be securely configured, but they must be patched and upgraded in a timely fashion.
A7:2017-Cross-Site Scripting (XSS)	XSS flaws occur whenever an application includes untrusted data in a new web page without proper validation or escaping, or updates an existing web page with user-supplied data using a browser API that can create HTML or JavaScript. XSS allows attackers to execute scripts in the victim's browser which can hijack user sessions, deface web sites, or redirect the user to malicious sites.
A8:2017-Insecure Deserialization	Insecure deserialization often leads to remote code execution. Even if deserialization flaws do not result in remote code execution, they can be used to perform attacks, including replay attacks, injection attacks, and privilege escalation attacks.
A9:2017-Using Components with Known Vulnerabilities	Components, such as libraries, frameworks, and other software modules, run with the same privileges as the application. If a vulnerable component is exploited, such an attack can facilitate serious data loss or server takeover. Applications and APIs using components with known vulnerabilities may undermine application defenses and enable various attacks and impacts.
A10:2017-Insufficient Logging & Monitoring	Insufficient logging and monitoring, coupled with missing or ineffective integration with incident response, allows attackers to further attack systems, maintain persistence, pivot to more systems, and tamper, extract, or destroy data. Most breach studies show time to detect a breach is over 200 days, typically detected by external parties rather than internal processes or monitoring.



LOTS OF OPTIONS!

FORTINET®

CLOUDFLARE

Azure Web Application Firewall (WAF)

f5®

AWS WAF

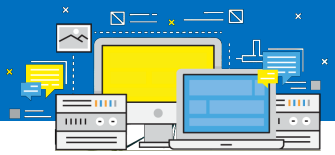
SUCURI

IMPERVA INCAPSULA

IMPERVA SECURESPHERE

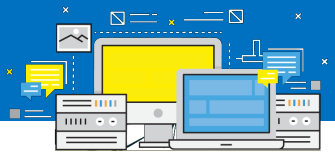
modsecurity

Open Source Web Application Firewall



COMMERCIAL VS OPEN SOURCE

	CloudFlare	ModSecurity	Incapsula
Total SQL Injection Tests	54	54	54
SQL Injection Bypassed	54	0	1
SQL Injection Blocked	0	54	53
Total XSS Tests	46	46	46
XSS Bypassed	46	0	3
XSS Blocked	0	46	43
Total LFI/RFI Tests	23	23	23
LFI/RFI Bypassed	23	2	4
LFI/RFI Blocked	0	21	19



OOPS

CLOUDFLARE

Web Application Firewall

Protect your website against SQL injections, cross-site scripting attacks and more

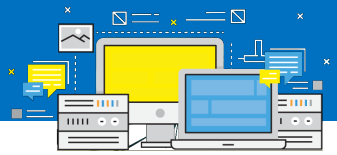
Cloudflare's Web Application Firewall (WAF) protects your website from SQL injection, cross-site scripting (XSS) and zero-day attacks, including OWASP-identified vulnerabilities and threats targeting the application layer. Customers include the Alexa-ranked Top 50, financial institutions, ecommerce companies and major enterprises. Fully-integrated with our DDoS protection, our WAF blocks millions of attacks daily, automatically learning from each new threat.

A robust rules engine to customize to your needs

Our WAF runs ModSecurity rule sets out of the box, protecting you against the most critical web application security flaws as identified by OWASP. It can also handle your existing rule sets and custom rules. Rules become effective in under 30 seconds.

Highlights:

- **Automatic protection** from diverse threats, with strong default rule sets and extensive customization providing Layer 7 protection that is fully integrated with DDoS mitigation
- **Lightning-fast 0.3 ms processing times**, with instant global updates
- **Compliance for PCI DSS requirement 6.6** — Cloudflare's WAF enables you to cost-effectively fulfill PCI compliance
- **Real-time reporting** — robust logging lets you see what's happening instantaneously

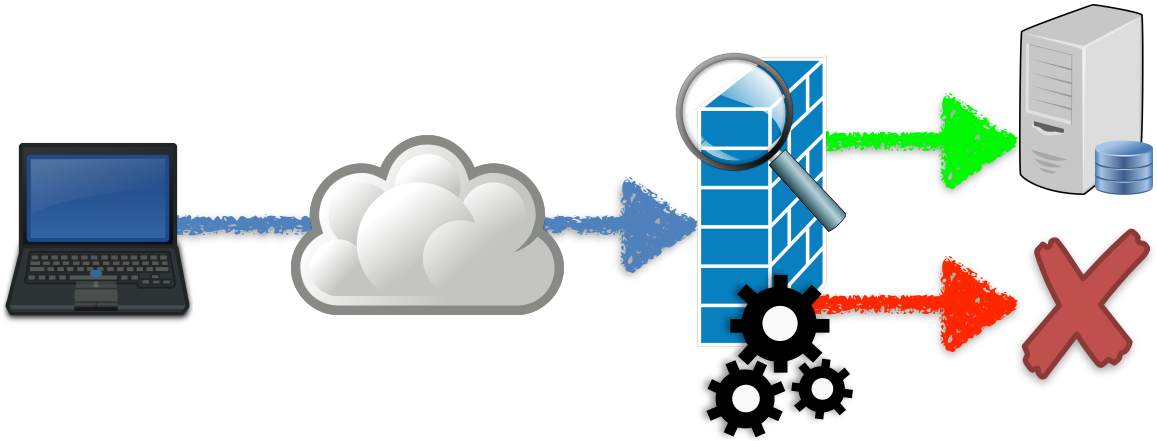


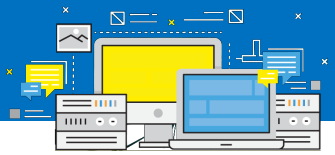
MODSECURITY

Ex:
index.php?id_product=190
index.php?id_product=193
index.php?id_product=210
index.php?id_product=450

Ex:
index.php?id_product=‘ or ‘1’=‘1

SecRule ARGS_GET:id_product “!@rx /d+” “action, id:1”





USER FRIENDLY CONFIGURATION

New Custom Rule

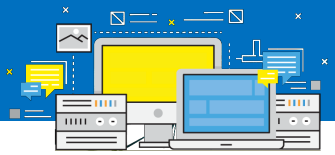
- Cross Site Scripting
- Cross Site Scripting (Extended)
- SQL Injection
- SQL Injection (Extended)
- Generic Attacks
- Generic Attacks(Extended)
- Known Exploits
- Trojans
- Information Disclosure
- Bad Robot

Custom Signature

Custom Signature Type — Custom Signature Group / Custom Signature Rule

The chosen signature category must also be enabled in the Signature policy itself.

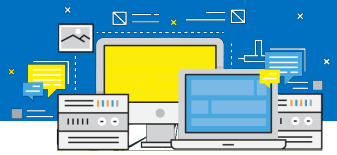
OK Cancel



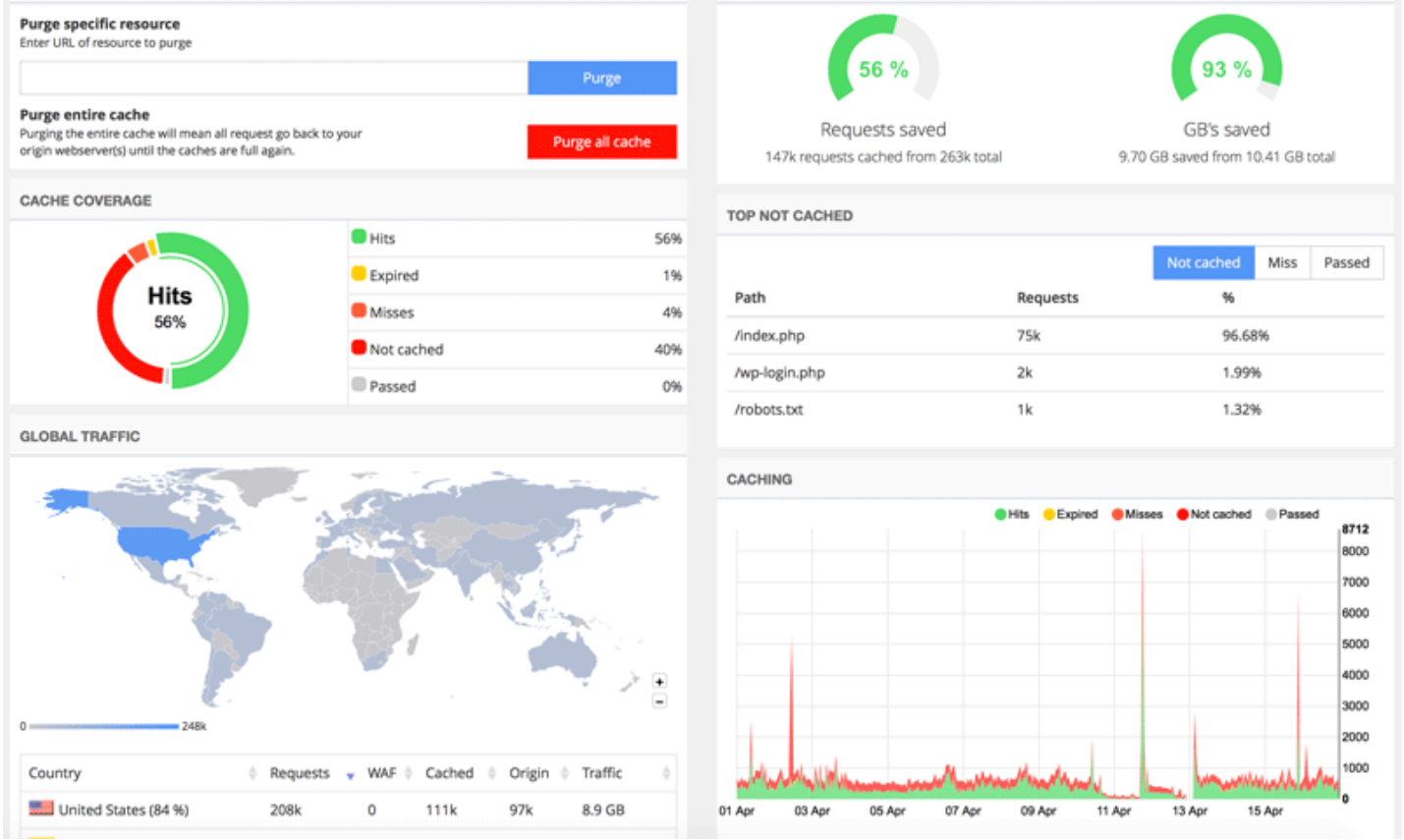
MAGIC RULESETS

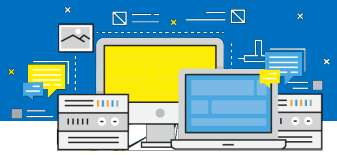
Rule Packages

Package Name	Description	Sensitivity	Action
OWASP ModSecurity Core Rule Set	Covers OWASP Top 10 vulnerabilities, and more.	Low ▼	Challenge ▼
OWASP Protocol Violations	Detection of violations of the HTTP protocol that often indicate an attacker attempting to penetrate a site.	Triggered 0 times	ON
OWASP Bad Robots	Detection of bad web robots that are not from search engines but perform malicious searching and spidering of web sites.	Triggered 0 times	ON
OWASP Protocol Anomalies	Detection of unusual use of the HTTP protocol that may indicate an attack, but that may also be legitimate.	Triggered 0 times	ON
OWASP Sir Et PhpBB Attacks	Rules to detect attacks on PHPBB.	Triggered 0 times	OFF
OWASP Request Limits	Detection of excessively large numbers of HTTP headers, HTTP arguments or files.	Triggered 0 times	ON
OWASP HTTP Policy	Enforcement of policies around the HTTP protocol such as methods that are supported and headers that are allowed.	Triggered 0 times	ON

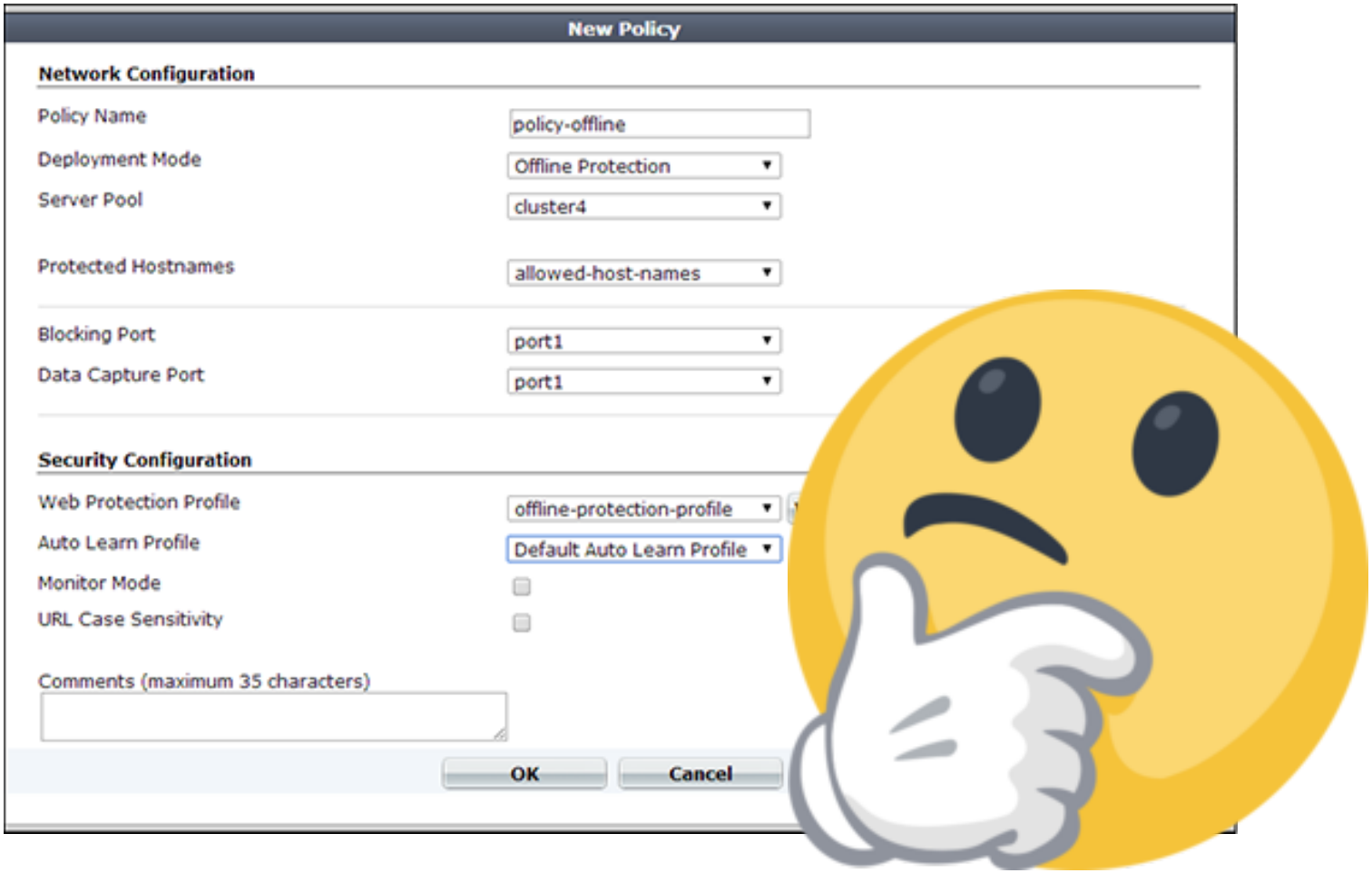


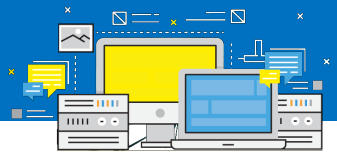
COOL DASHBOARDS



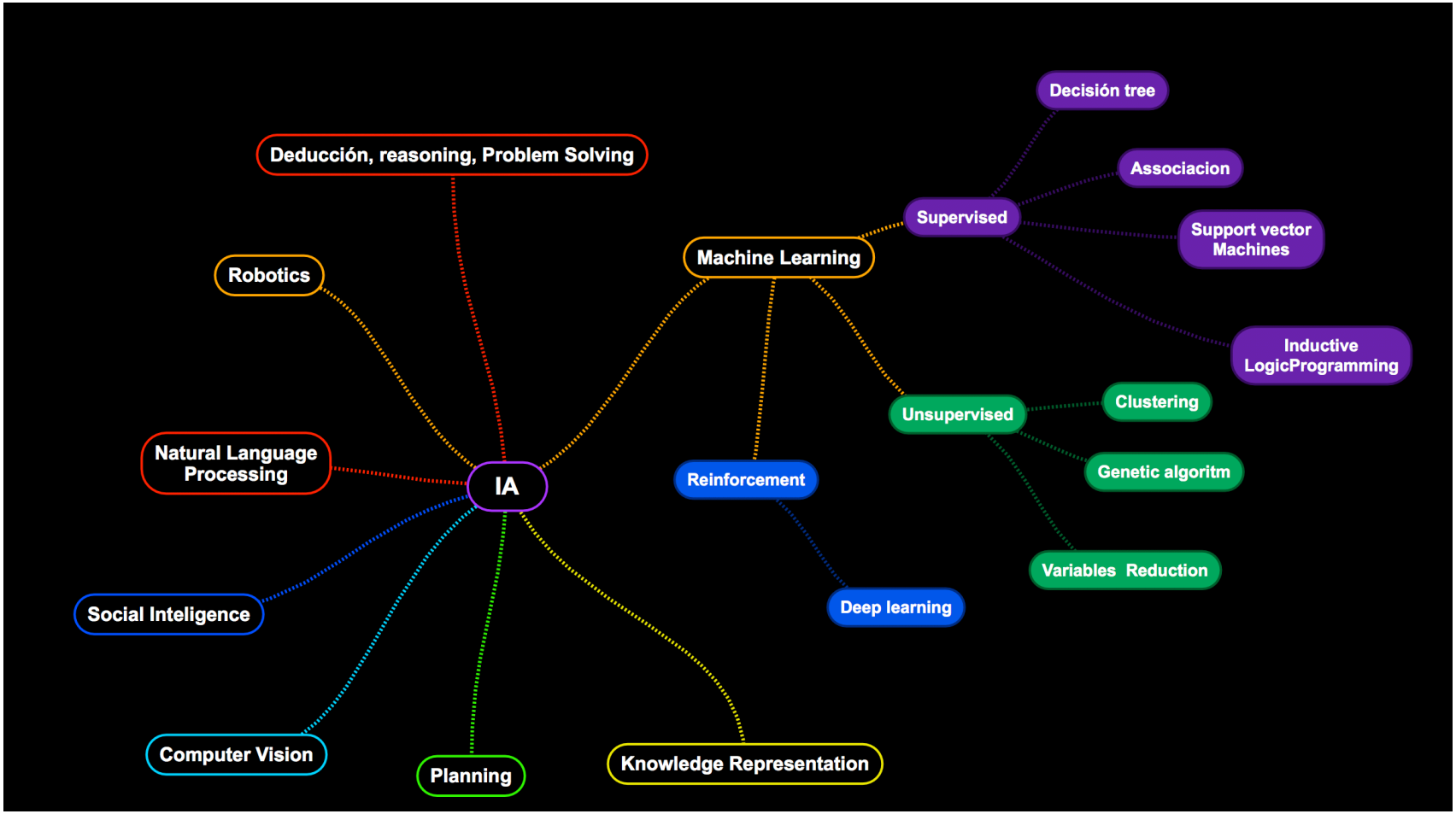


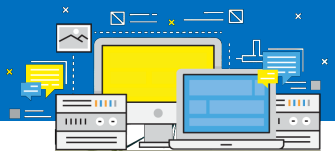
LEARNING MODE



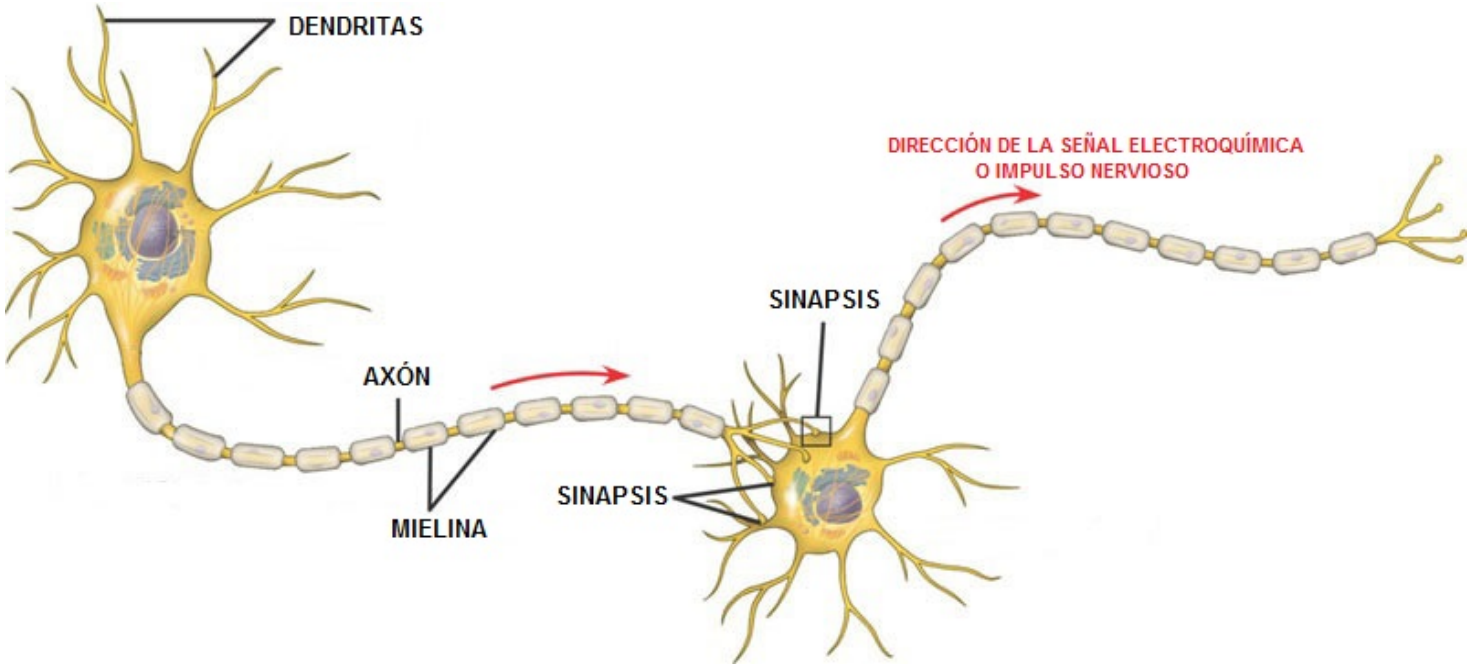


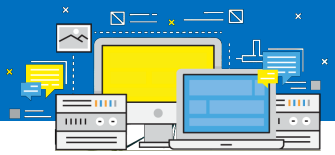
WHAT IS MACHINE LEARNING?



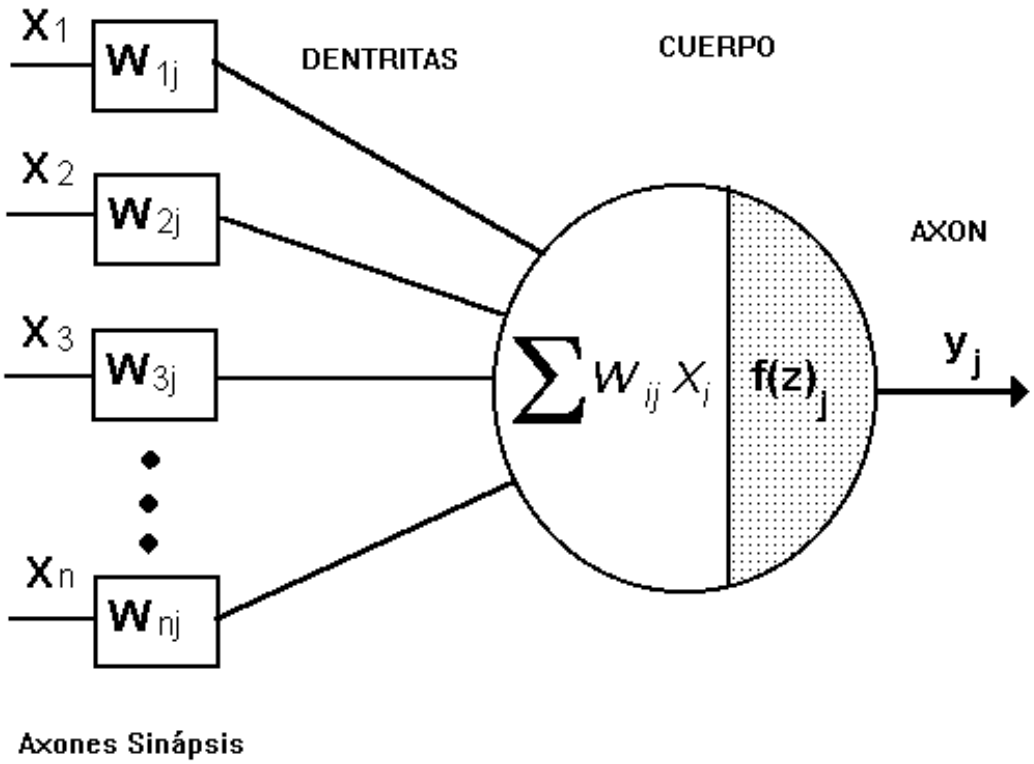


DEEP LEARNING

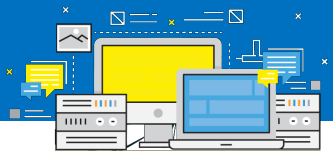




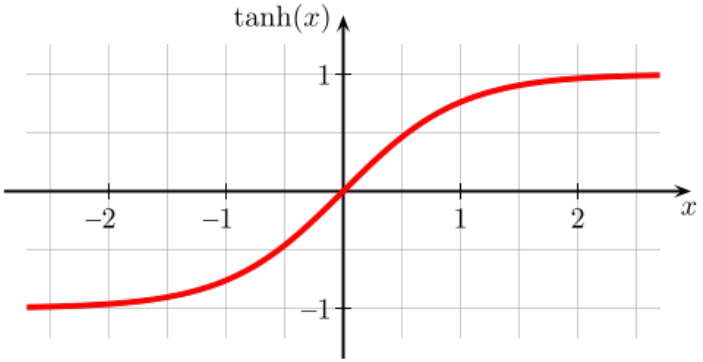
DEEP LEARNING



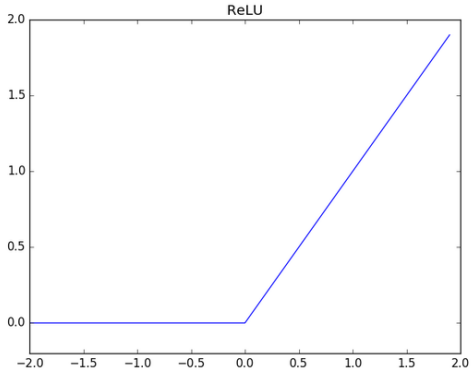
$$Y=f(SUM(W*X*)-U)$$



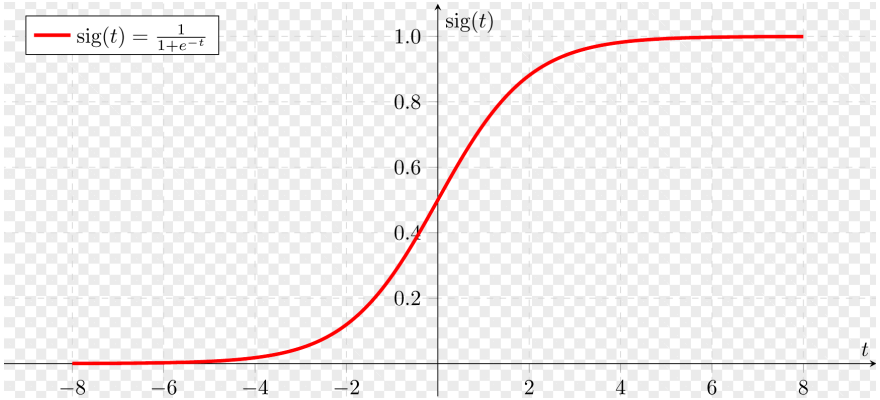
ACTIVATION FUNCTIONS



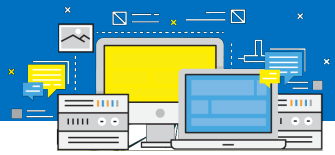
Tanh



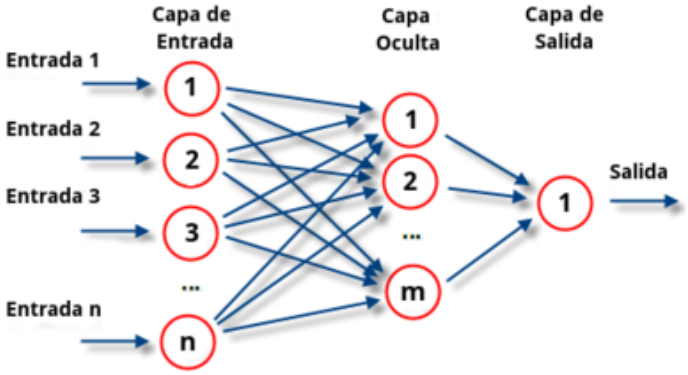
RelU



Sigmoid



ARTIFICIAL NEURAL NETWORK



INPUT LAYER:

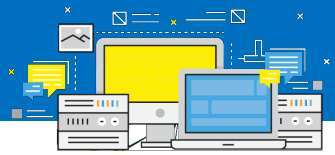
The number of perceptrons in input layer depends of the features of our problem.

HIDDEN LAYER

A greater number of intermediate layers more complex were the characteristics that we are able to analyze.

EXIT LAYER

The number of output layers depends on the possible answers that our model has.



GRADIENT DESCENT

F(w)=(w-1)^2

Target: Minimize f(w)

Classic method:

F'(w)=2(w-1)

2(w-1)=0

w=1

Gradient descent:

W	f(w)
2	1
2,1	1,21
1,9	0,81

For modify W = -f'(w)

W(n+1)=Wn-f'(Wn)

Ex:

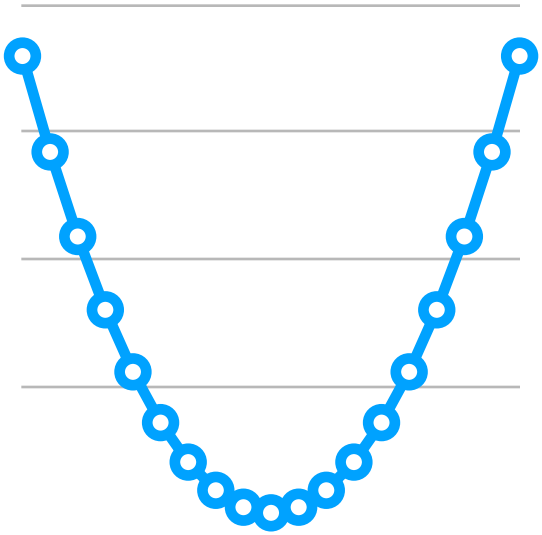
W0=2

W1=2-2(2-1)

W1=0

W2=0-2(0-1)

W2=2



W(n+1)=Wn-Tf'(Wn)

Ex:

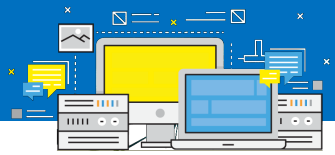
W0=2

W1=2-0,1*2(2-1)

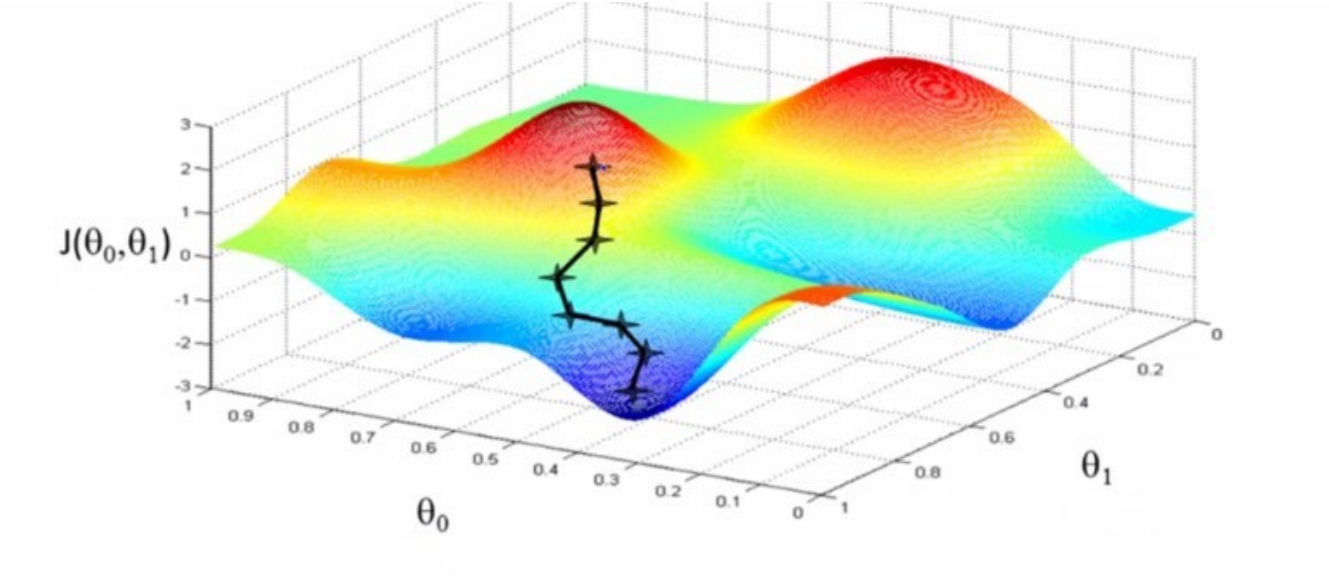
W1=1,8

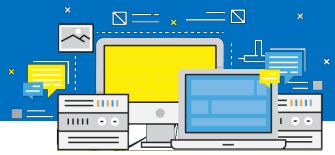
W2=1,8-0.1*2(1,8-1)

W2=1,64



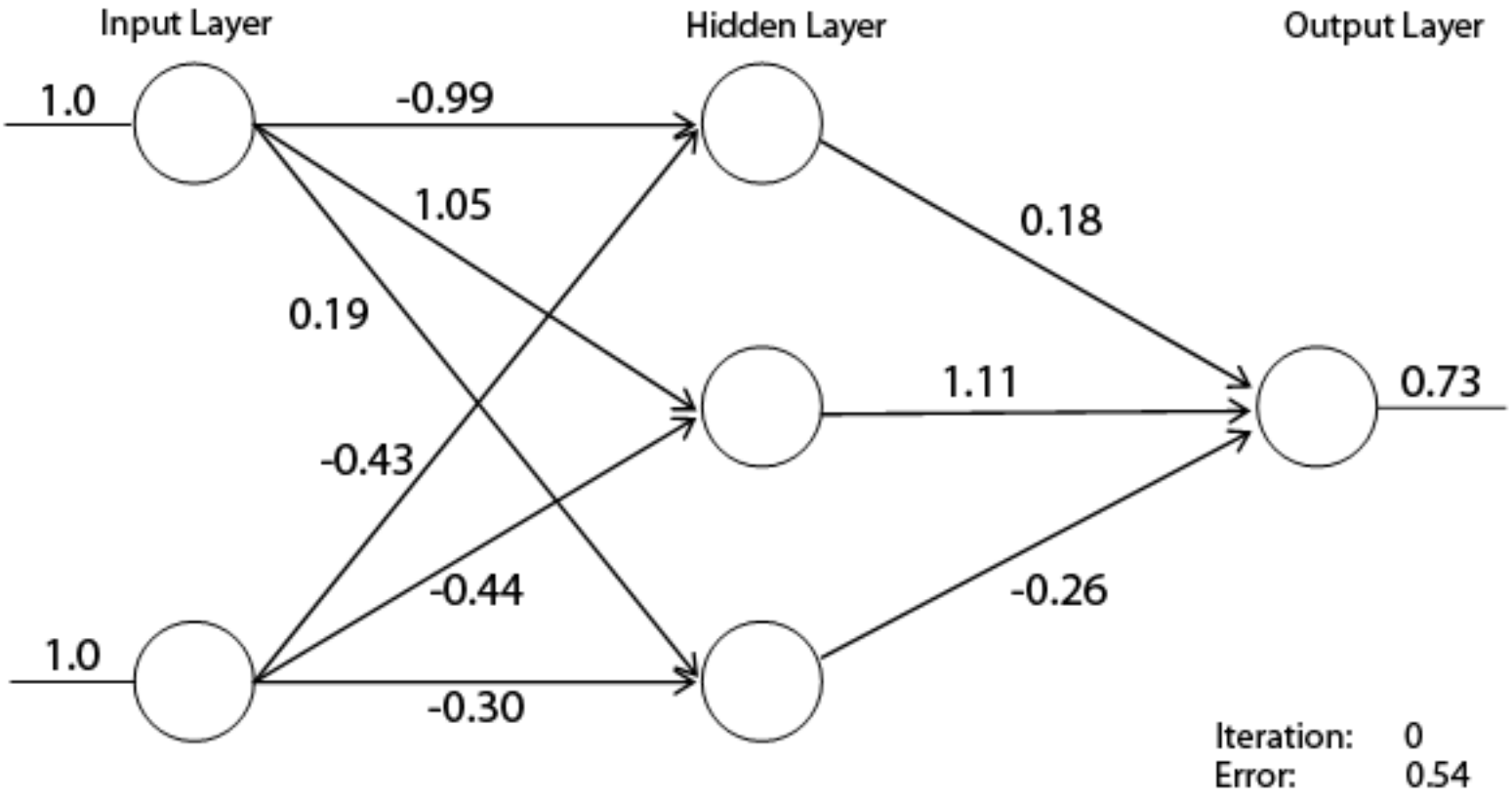
GRADIENT DESCENT

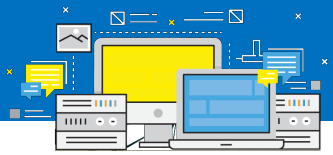




HOW TO LEARN A NEURAL NETWORK?

BACKPROPAGATION





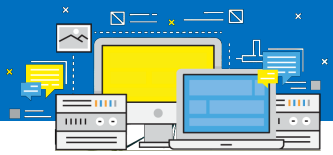
WHAT IS OUR PROBLEM?

- We try to detect web attacks as XSS or SQLi
- Good request

http://localhost/index.php?user=admin&pass=<SHA1>

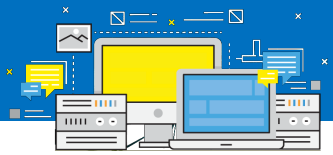
- Bad request:

http://localhost/index.php?user=admin&pass=' or '1'='1



HOW TO SELECT FEATURES?

- Len of the value per parameter
- Number of alphabet chars
- Number of numeric chars
- Number of special characters



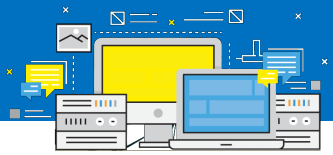
HOW TO SELECT FEATURES?

- Parameter 'user':

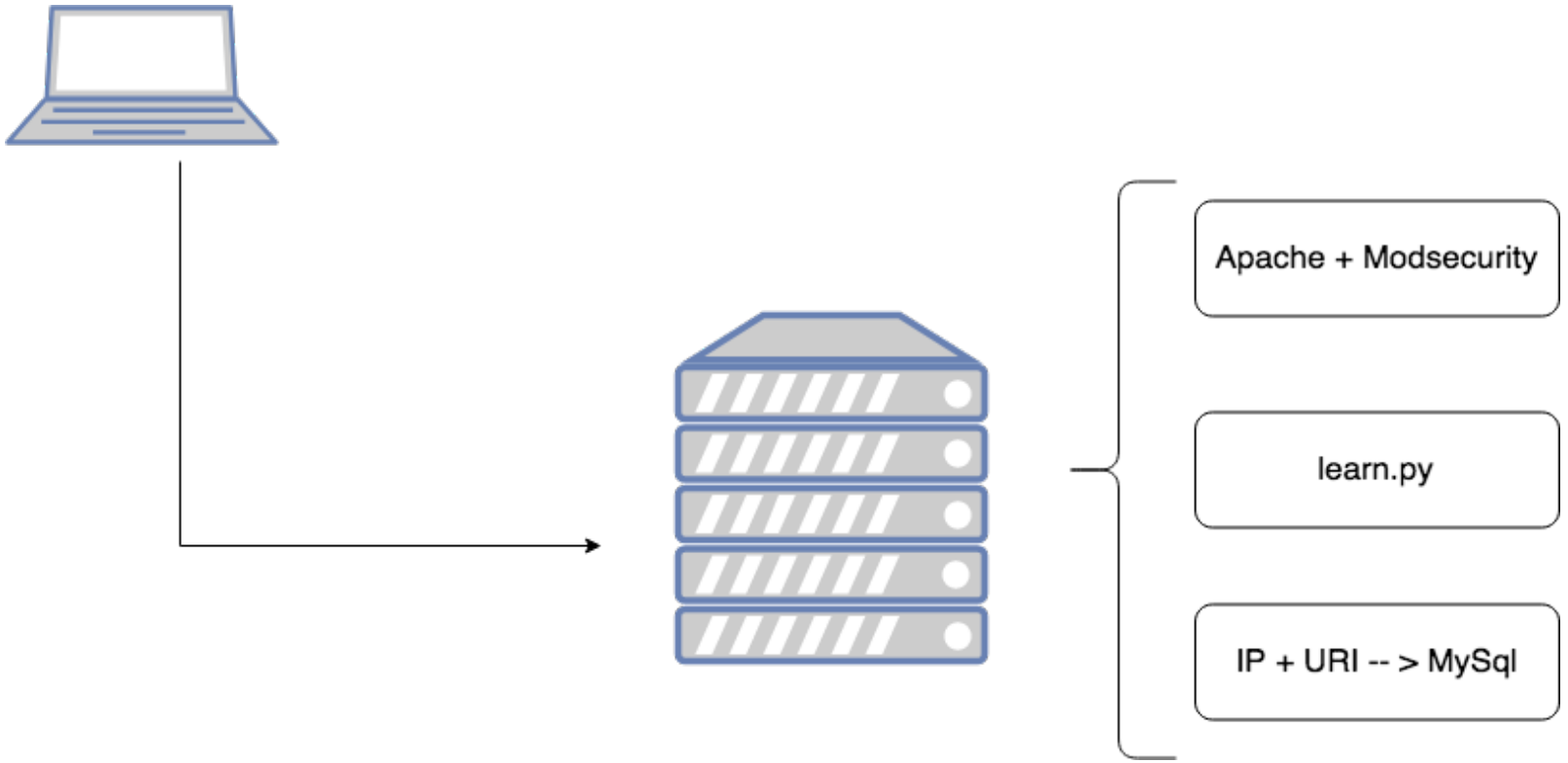
"admin"	"jortiz"	"hpotter10"
len=5	len=6	len=9
entropy=	entropy=	entropy=
num_alph=5	num_alph=6	num_alph=7
num_num=0	num_num=0	num_num=2
num_special_char=0	num_special_char=0	num_special_char=0

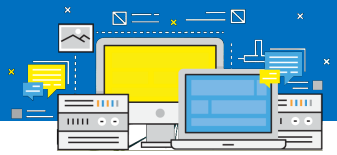
- Parameter 'pass':

len=40	len=40	len=40
entropy=	entropy=	entropy=
num_alph=18	num_alph=20	num_alph=15
num_num=22	num_num=20	num_num=25
num_special_char=0	num_special_char=0	num_special_char=0

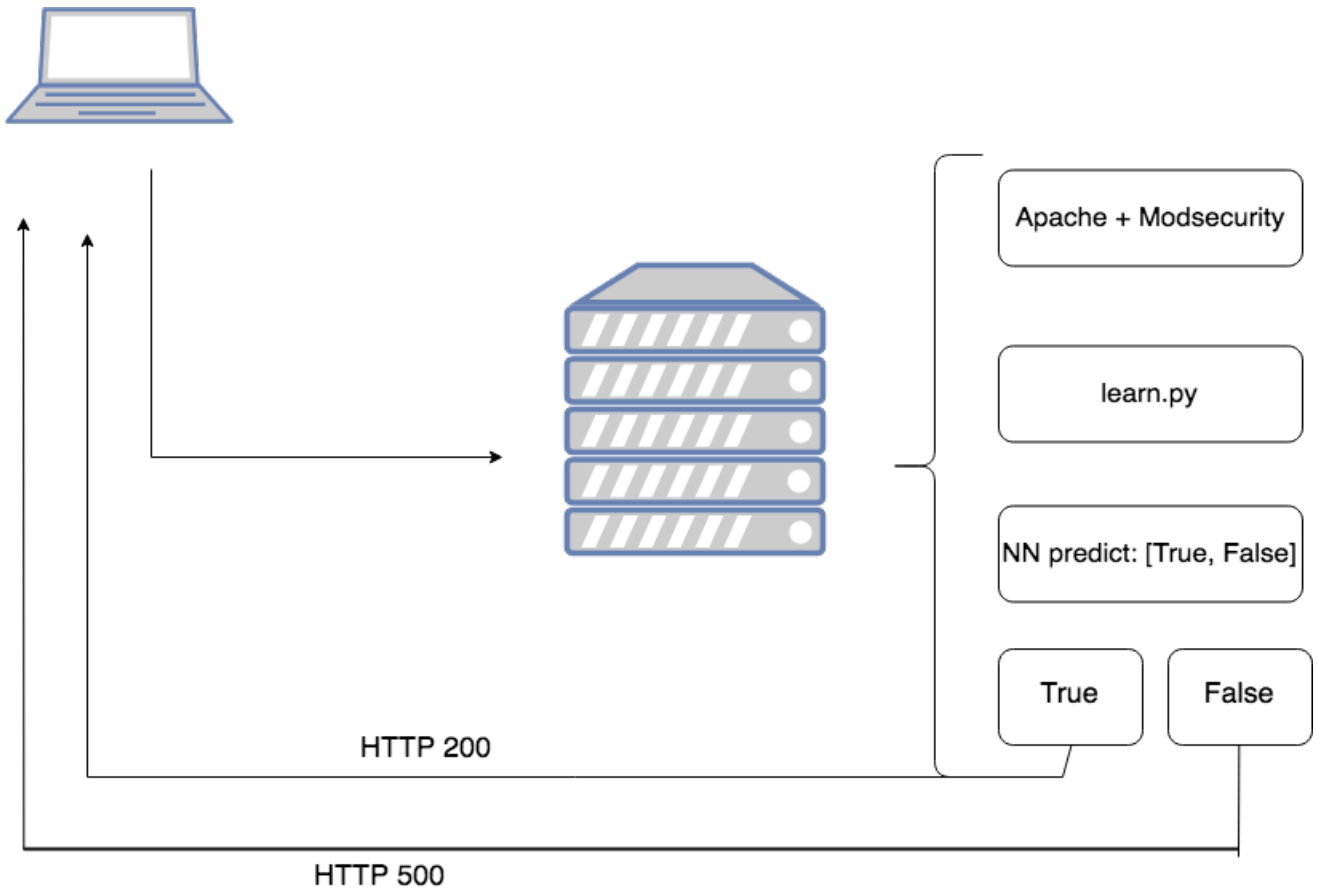


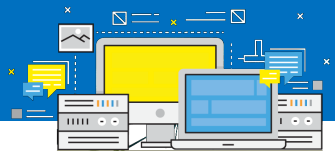
GENERATE DATA TO TRAIN





HOW TO WORK?


Apache + Modsecurity
learn.py
NN predict: [True, False]
True
False
HTTP 200
HTTP 500



https://github.com/dsfau/TecDay18