

**XVIII  
JORNADAS  
STIC  
CCN-CERT**

**VI  
JORNADAS  
DE CIBER\_  
DEFENSA  
ESPDEF-CERT**

# **Blast Radius**

## **Vulnerabilidades en el protocolo RADIUS y contramedidas**



**Dr. Alfonso Muñoz - Founder CriptoRed**



@mindcrypt



t.me/Criptored

<https://www.linkedin.com/in/alfonsomuñoz/>

<https://www.criptored.es>



alfonso@criptored.com

## Doctor Ingeniero de Telecomunicaciones (UPM)

+20 años atacando y protegiendo  
multinacionales/startups/gobiernos (EU/US/ASIA)

- Offensive security (sw/hw)
- Cryptography & covert channels/stego
- Cutting-edge research (defensive & offensive)
- Offensive IA

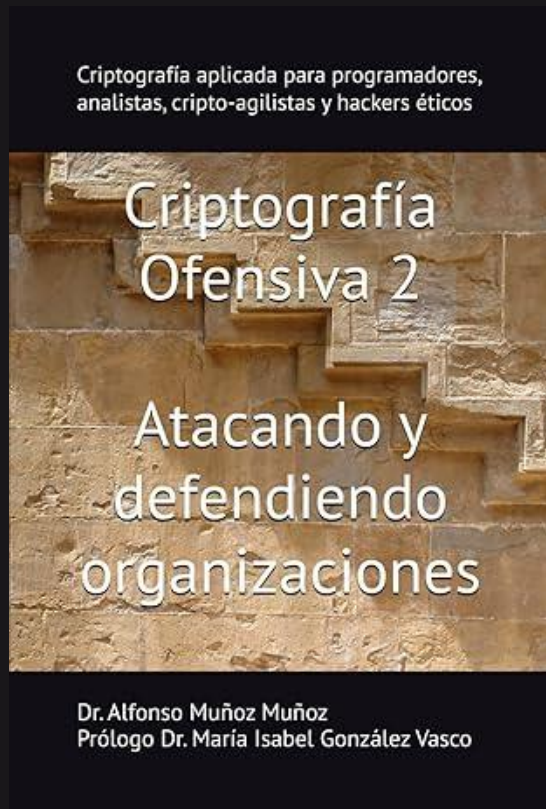
**Speaker:** RootedCon, Ekoparty, Blackhat USA-Europe-Asia, 8.8, BSIDES, VirusBulletin, HackInTheBox, DeepSec, STIC CCN/CERT...

4 premios SIC, premio al conocimiento (ISACA), premios Red Seguridad, premio Antonio Ropero/RootedCON, IDG 2020-Top 50 Blue Team.... Profesor de Máster de ciberseguridad (UEM, UCLM, UC3M, UPM, UCM, UNIR...)

**Bug hunter** - Security bulletins (Microsoft, Foxit, Google - Hall of fame, etc.).



Me gusta relaja “escribir”....

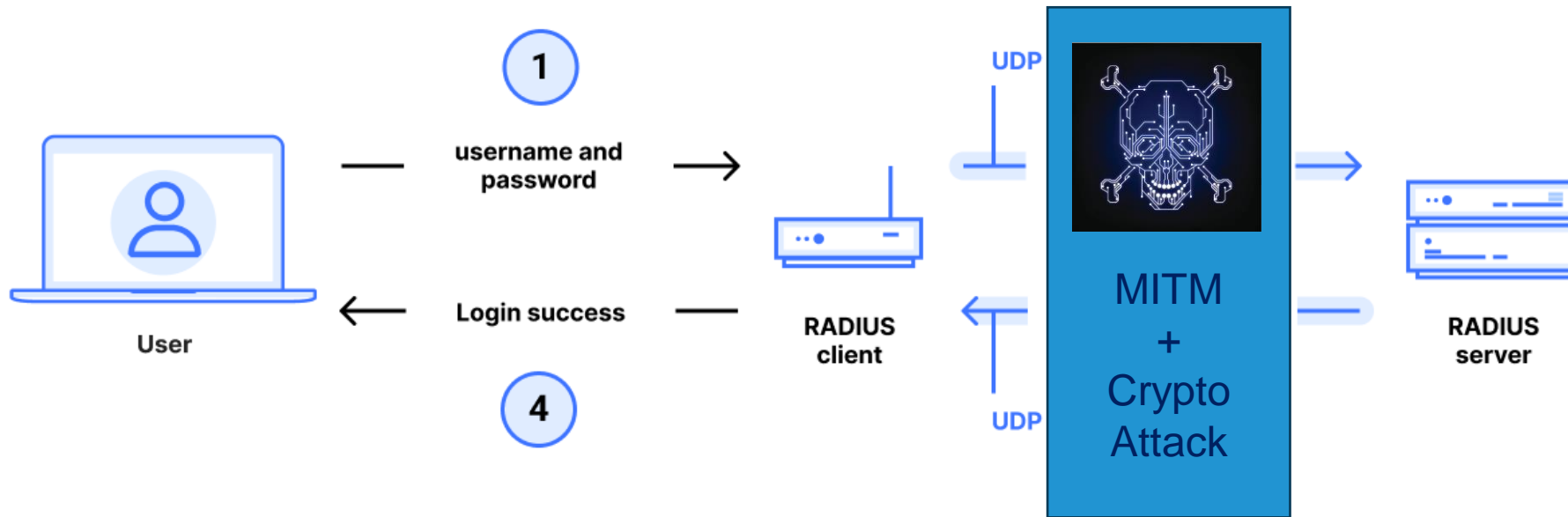


<https://www.amazon.es/Criptografía-Ofensiva-Atacando-defendiendo-organizaciones/dp/B0D8YPT322>



# Blast Radius - <https://www.blastradius.fail/>

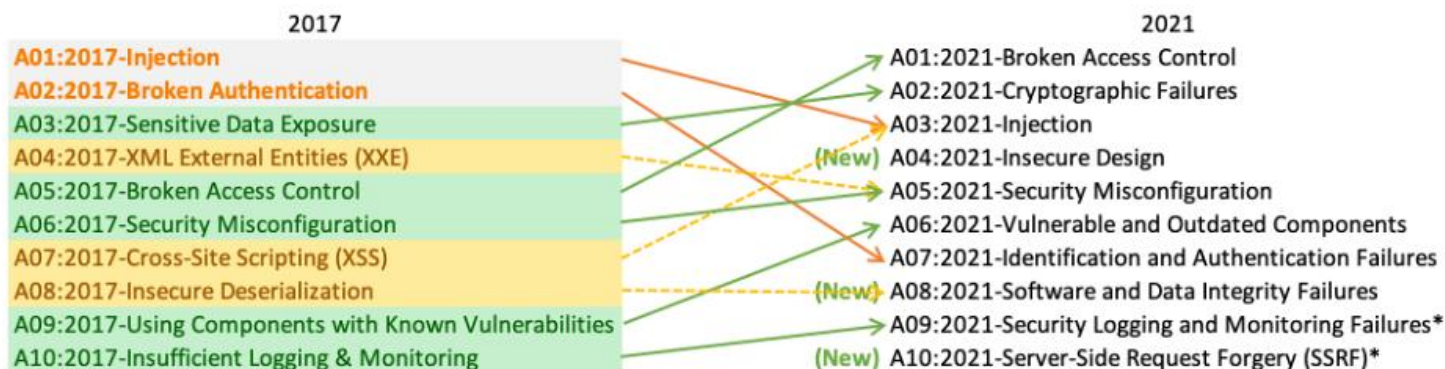
**Blast-RADIUS is a vulnerability that affects the RADIUS protocol.** RADIUS is a very common protocol used for authentication, authorization, and accounting (AAA) for networked devices on Enterprise and telecommunications networks. **Blast-RADIUS is a protocol vulnerability, and thus affects all RADIUS/UDP implementations using non-EAP authentication methods over UDP.**



The Blast-RADIUS attack allows a man-in-the-middle attacker between the RADIUS client and server to forge a valid protocol accept message in response to a failed authentication request. This forgery could give the attacker access to network devices and services without the attacker guessing or brute forcing passwords or shared secrets. The attacker does not learn user credentials.

# Practical Encryption Bypassing...

## OWASP Top 10 - Web Application Security Risks



[https://owasp.org/Top10/A02\\_2021-Cryptographic\\_Failures/](https://owasp.org/Top10/A02_2021-Cryptographic_Failures/)

- Is any data transmitted in clear text? This concerns protocols such as HTTP, SMTP, FTP also using TLS upgrades like STARTTLS. External internet traffic is hazardous. Verify all internal traffic, e.g., between load balancers, web servers, or back-end systems.
- Are any old or weak cryptographic algorithms or protocols used either by default or in older code?
- Are default crypto keys in use, weak crypto keys generated or re-used, or is proper key management or rotation missing? Are crypto keys checked into source code repositories?
- Is encryption not enforced, e.g., are any HTTP headers (browser) security directives or headers missing?
- Is the received server certificate and the trust chain properly validated?
- Are initialization vectors ignored, reused, or not generated sufficiently secure for the cryptographic mode of operation? Is an insecure mode of operation such as ECB in use? Is encryption used when authenticated encryption is more appropriate?
- Are passwords being used as cryptographic keys in absence of a password base key derivation function?
- Is randomness used for cryptographic purposes that was not designed to meet cryptographic requirements? Even if the correct function is chosen, does it need to be seeded by the developer, and if not, has the developer over-written the strong seeding functionality built into it with a seed that lacks sufficient entropy/unpredictability?
- Are deprecated hash functions such as MD5 or SHA1 in use, or are non-cryptographic hash functions used when cryptographic hash functions are needed?
- Are deprecated cryptographic padding methods such as PKCS number 1 v1.5 in use?
- Are cryptographic error messages or side channel information exploitable, for example in the form of padding oracle attacks?

### Debian OpenSSL Predictable Random Number Generator Vulnerability (CVE-2008-0166)

<https://www.youtube.com/watch?v=-SLr5FLUKmY>

### Reaping and breaking keys at scale – When crypto meets big data - Batch-GCD attack (2018)

<https://media.defcon.org/DEF%20CON%2026/DEF%20CON%2026%20presentations/DEFCON-26-Nils-Amiet-and-Yolan-Romailer-Reaping-and-breaking-keys-at-scale-when-crypto-meets-big-data-Updated.pdf>

...

# Cryptofails 2024

## Pwnie for Best Crypto Bug

Breach Extraction Attacks	The paper describes attacks leveraging leakage from cryptographic protocols and compromised credential-checking services, specifically Cloudflare's implementation, to reconstruct encrypted passwords on servers. It combines cryptographic insights and neural networks to guess encrypted credentials in an unprecedented manner.	@kornaropoulos
GoFetch: Breaking Constant-Time Cryptographic Implementations Using Data Memory-Dependent Prefetchers	A microarchitectural side-channel attack that exploits data memory-dependent prefetchers on the latest Apple processors to extract secret keys from constant-time cryptographic implementations.	boruchen
Blast Radius: RADIUS/UDP Considered Harmful	This significant attack targets a major protocol using a powerful, underused vector that exploits the legacy use of MD5, which lacks chosen-prefix collision resistance.	nadiah

<https://16years.secvuln.info/>  
<https://terrapin-attack.com>

...



<https://pwnies.com/>

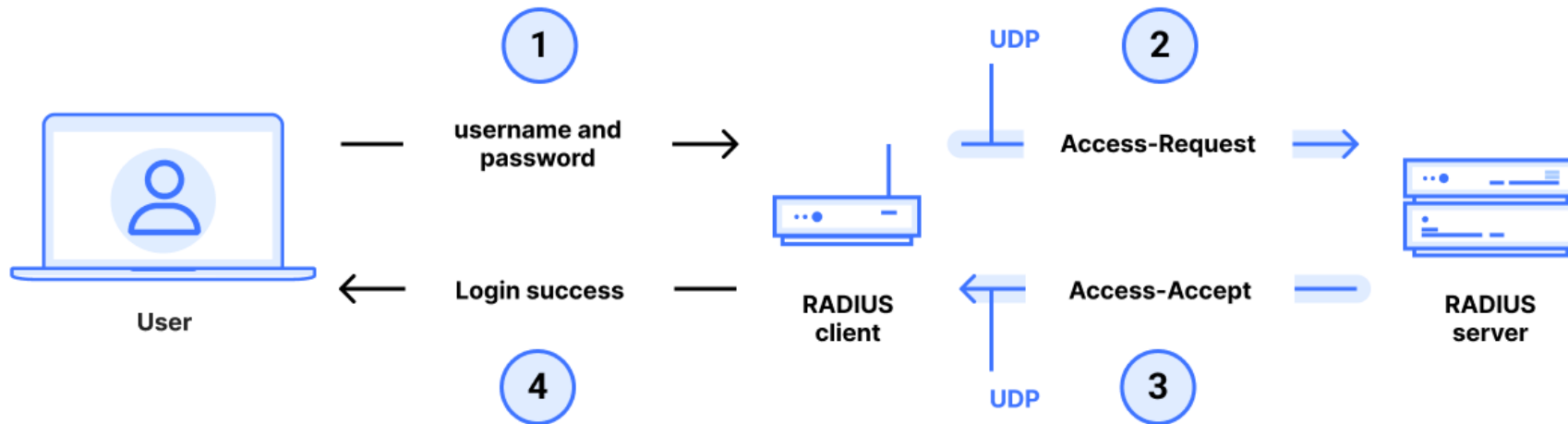


# Radius – rfc2865

<https://datatracker.ietf.org/doc/html/rfc2865>

“The RADIUS (Remote Authentication Dial-In User Service, 1991) protocol is at the core of today’s network infrastructure - de facto standard lightweight authentication protocol used for remote access for users and administrators to networked devices...

RADIUS is used in a wide variety of applications, including in enterprise networks to authenticate access to switches and other routing infrastructure, for VPN access, by ISPs for DSL and FTTH (Fiber to the Home), in 802.1X and Wi-Fi authentication, 2G and 3G cellular roaming and 5G DNN (Data Network Name) authentication, mobile Wi-Fi offload with SIM card-based authentication, private APN authentication, to authenticate access to critical infrastructure, and in the Eduroam and OpenRoaming wifi consortia - <https://www.blastradius.fail>



## Notes

RADIUS traffic is still “**commonly transported over UDP**” in the clear, protected only by outdated cryptographic constructions based on MD5

- <https://datatracker.ietf.org/doc/draft-ietf-radext-deprecating-radius/>

RADIUS-TLS in process...

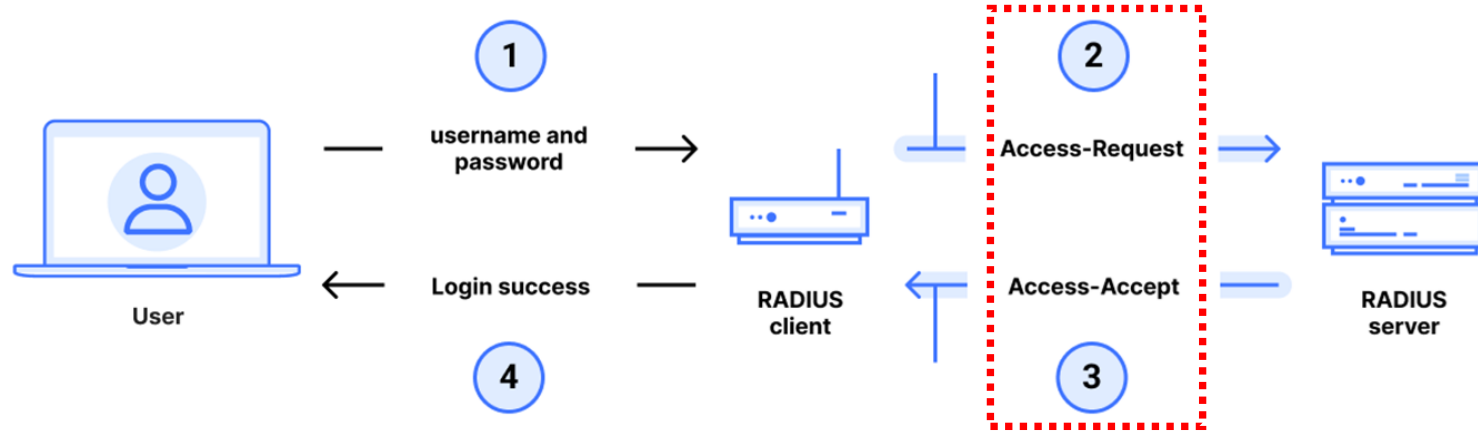
<https://datatracker.ietf.org/doc/draft-ietf-radext-radiusdtls-bis/03/>

## Authentication Modes

- **PAP (Password Authentication Protocol)** - The client sends credentials in plain text to the RADIUS server.
- **CHAP (Challenge Handshake Authentication Protocol)** - Uses a hashed challenge-response mechanism
- **EAP (Extensible Authentication Protocol)** - EAP-TLS (requires certificates on both the client and server), PEAP (Protected EAP- uses a TLS tunnel to protect credentials) and EAP-TTLS (Tunneled TLS - similar to PEAP but allows more authentication methods within the tunnel).

# Radius – PAP

## Password Authentication Protocol mode

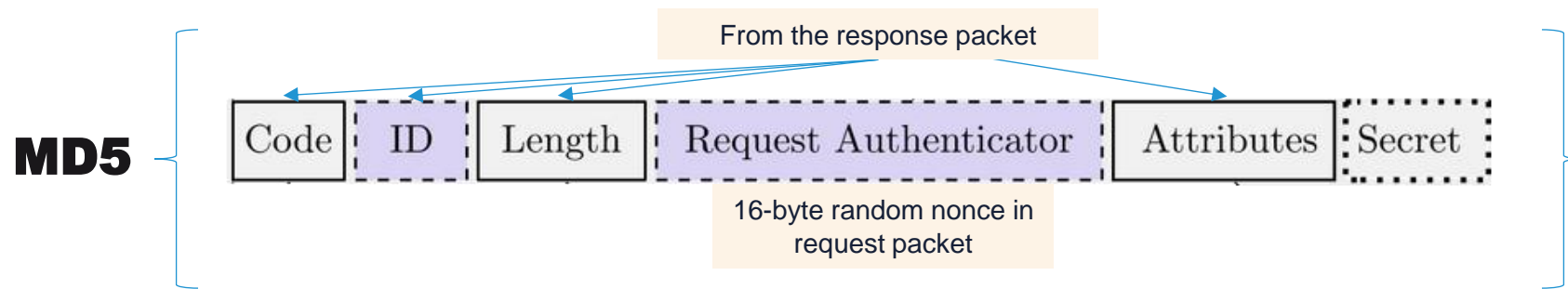


With RADIUS/UDP PAP authentication, the RADIUS client sends a username and password in an *Access-Request* packet to the RADIUS server over UDP. The server drops the packet if its source IP address does not match a known client, but otherwise the *Access-Request* is entirely unauthenticated. This makes it vulnerable to modifications by a MitM.

The RADIUS server responds with either an *Access-Reject*, *Access-Accept* (or possibly also an *Access-Challenge*) packet sent to the RADIUS client over UDP. These response packets are “authenticated” with an ad hoc “message authentication code (MAC)” to prevent modifications by an MitM. This “MAC” is based on MD5 and is called the *Response Authenticator*.

Source: <https://blog.cloudflare.com/radius-udp-vulnerable-md5-attack/>

**The RADIUS *Response Authenticator* “authenticates” RADIUS responses** via an ad hoc MD5 construction that involves concatenating several fields in the RADIUS request and response packets, appending **a *Secret shared between RADIUS client and RADIUS server***, and then hashing the result with MD5.





# MD5 attacks

How to break MD5 and other hash functions (2004) – Xiaoyun Wand and Hongbo Yu

Advances in Cryptology Eurocrypt 2005 - <https://iacr.org/archive/eurocrypt2005/34940019/34940019.pdf>

*“identical prefix collisions” of the form  $MD5(\text{Message } P \parallel \text{Gibberish1} \parallel \text{Message } S) = MD5(\text{Message } P \parallel \text{Gibberish2} \parallel \text{Message } S)$ . This attack can run on a regular consumer laptop in seconds. While this attack is a devastating blow for any cryptographic hash function, it’s still pretty difficult to use gibberish messages (with identical prefixes) to create practical attacks on real protocols like RADIUS*

Chosen-prefix Collisions for MD5 and Colliding X.509 Certificates for Different Identities (2007) - Marc Stevens, Arjen Lenstra and Benne de Weger - <https://www.iacr.org/archive/eurocrypt2007/45150001/45150001.pdf>

*the “chosen-prefix collision attack”. This attack is slower and more costly, but allows the prefixes in the collision to be different, making it valuable for [practical attacks on real protocols](#). The collision is  $MD5(\text{Message } P1 \parallel \text{Gibberish1} \parallel \text{Message } S) = MD5(\text{Message } P2 \parallel \text{Gibberish2} \parallel \text{Message } S)$ , where  $P1$  and  $P2$  are different freely-chosen meaningful messages,  $G1$  and  $G2$  are meaningless gibberish and  $S$  is a meaningful message.*

MD5 considered harmful today. Creating a rogue CA certificate (2008/2009) – A. Sotirov, Marc Stevens et al.

A **chosen prefix attack on MD5** allowed them to create a rogue certificate authority that could generate TLS certificates that would be trusted by all major browsers. A key ingredient for the attack is software named **hashclash**

<https://eprint.iacr.org/2009/111> <https://marc-stevens.nl/p/hashclash/> <https://github.com/cr-marcstevens/hashclash>

“Deprecation of MD5 didn’t start in earnest until **2012** after **malware known as Flame**, reportedly created jointly by the governments of Israel and the US, was found to have **used a chosen prefix attack to spoof MD5-based code signing by Microsoft’s Windows update mechanism**. Flame used the collision-enabled spoofing to [hijack the update mechanism](#) so the **malware could spread from device to device inside an infected network...**” - <https://arstechnica.com/security/2024/07/new-blast-radius-attack-breaks-30-year-old-protocol-used-in-networks-everywhere/>



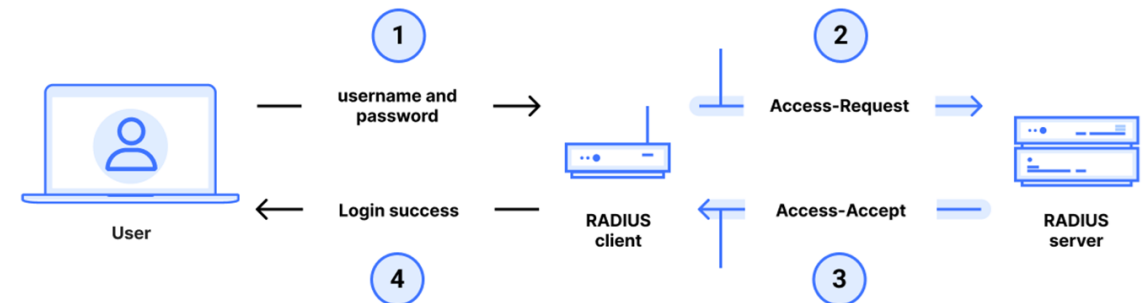
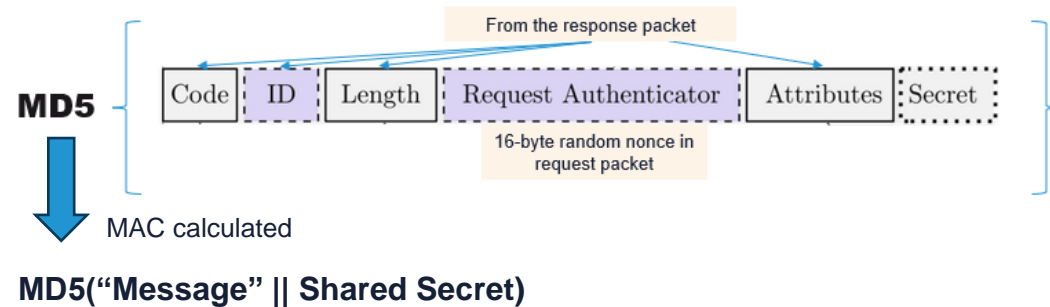
Creating a rogue CA certificate using Sony PlayStation 3 - <https://marc-stevens.nl/research/hashclash/rogue-ca/>

# Blast Radius – Details

<https://www.blastradius.fail/>

The Blast-RADIUS attack allows a **man-in-the-middle** attacker between the **RADIUS client and server** to **forge a valid protocol accept message in response to a failed authentication request**. This forgery could give the attacker access to network devices and services without the attacker guessing or brute forcing passwords or shared secrets. *The attacker does not learn user credentials...*

RADIUS Response Authenticator – Message authentication code (rfc2869, HMAC-MD5)



Blast-RADIUS is a protocol vulnerability, and thus affects all RADIUS/UDP implementations using non-EAP authentication methods over UDP.

**Length extension attacks → Given MD5(X) for an unknown X, then anyone who knows Y can compute MD5(X || Y)**

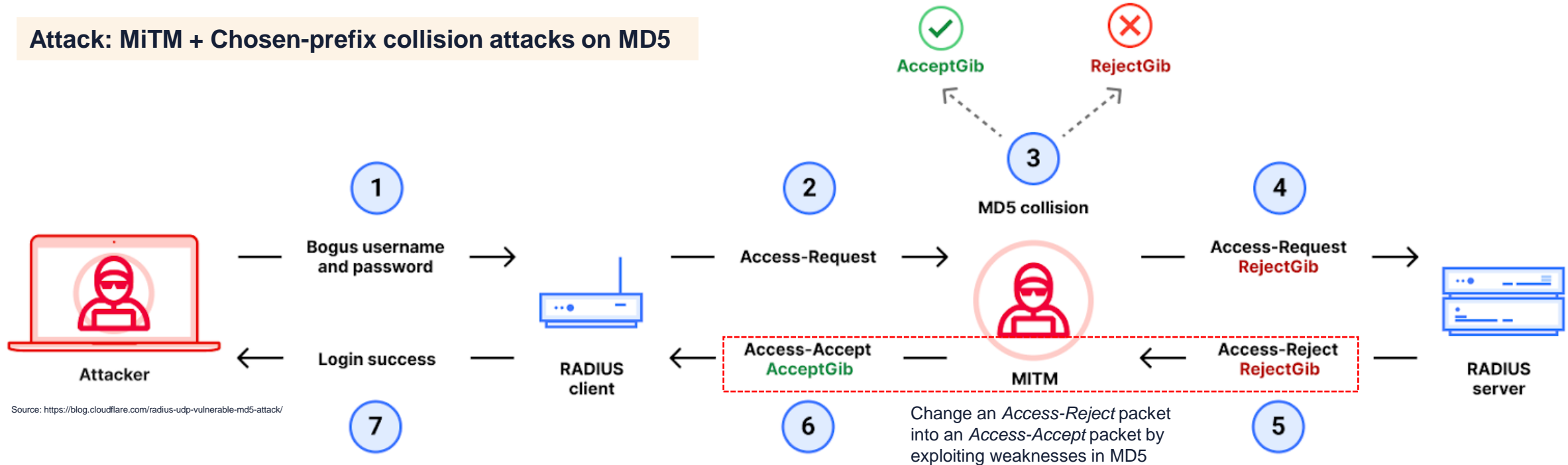
This block-wise processing is also an issue for the **Response Authenticator of RADIUS**. If someone finds an **MD5 collision**, namely two different messages Message1 and Message2 such that  $\text{MD5}(\text{Message1}) = \text{MD5}(\text{Message2})$ , then it follows that  $\text{MD5}(\text{Message1} \parallel \text{Secret}) = \text{MD5}(\text{Message2} \parallel \text{Secret})$

The attacker then learns the "MAC" on Message1, which is  $\text{MD5}(\text{Message1} \parallel \text{Secret})$ . **Now the attacker can forge the "MAC" on Message2 without ever needing to know the Secret, simply by reusing the "MAC" on Message1**

# Blast Radius – Details

<https://www.blastradius.fail/>

## Attack: MiTM + Chosen-prefix collision attacks on MD5

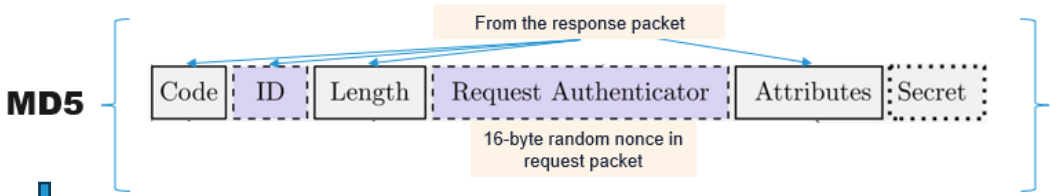


Is it possible  $\text{MD5}(\text{Access-Reject}||\text{RejectGibberish}) = \text{MD5}(\text{Access-Accept}||\text{AcceptGibberish})$ ?

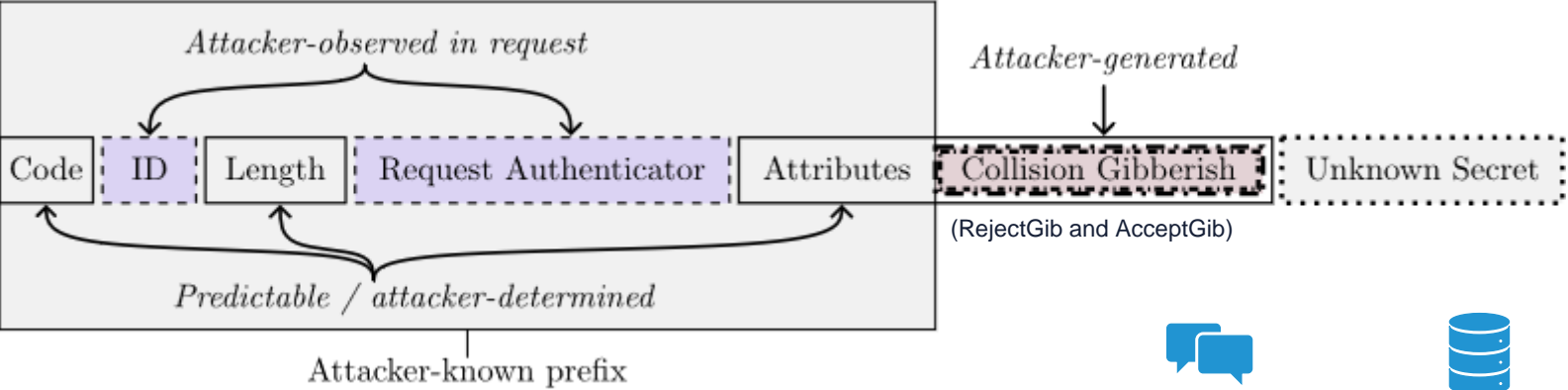
Is it possible to add “RejectGibberish” to the Access-Request (fake) packet?  
Could “RejectGibberish” affect the Response Authenticator message?

# Blast Radius – details

<https://www.blastradius.fail/pdf/radius.pdf>



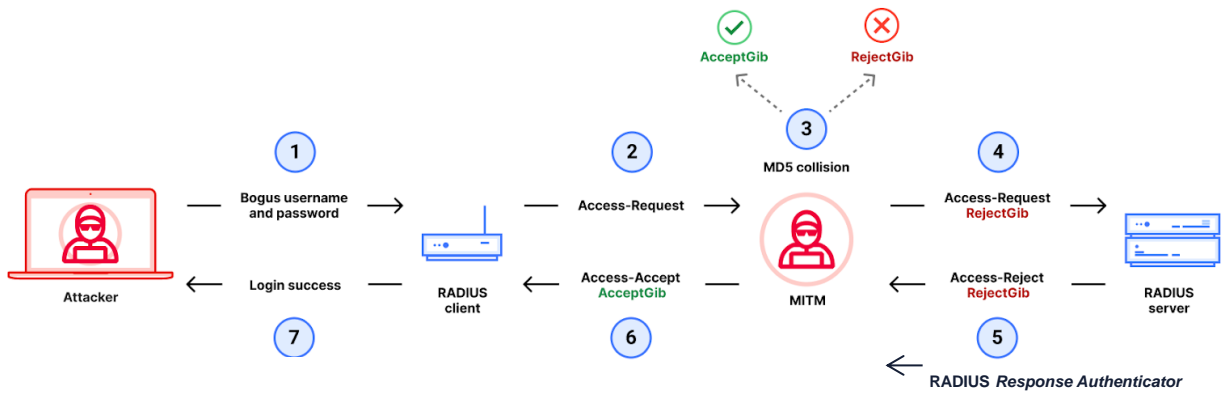
## RADIUS Response Authenticator



An optional RADIUS/UDP attribute called the *Proxy-State*. The *Proxy-State* is an ideal place to stuff this gibberish because a **RADIUS server must echo back any information it receives in a *Proxy-State* attribute from the RADIUS client**. The *Proxy-State* must also be hashed by MD5 in the corresponding response's *Response Authenticator*.



If (collision)  
$$\text{MD5}(\text{Access-Reject}||\text{RejectGibberish}) = \text{MD5}(\text{Access-Accept}||\text{AcceptGibberish})$$
  
then  
$$\text{MD5}(\text{Access-Reject}||\text{RejectGibberish}||\text{Shared secret}) = \text{MD5}(\text{Access-Accept}||\text{AcceptGibberish}||\text{Shared secret})$$

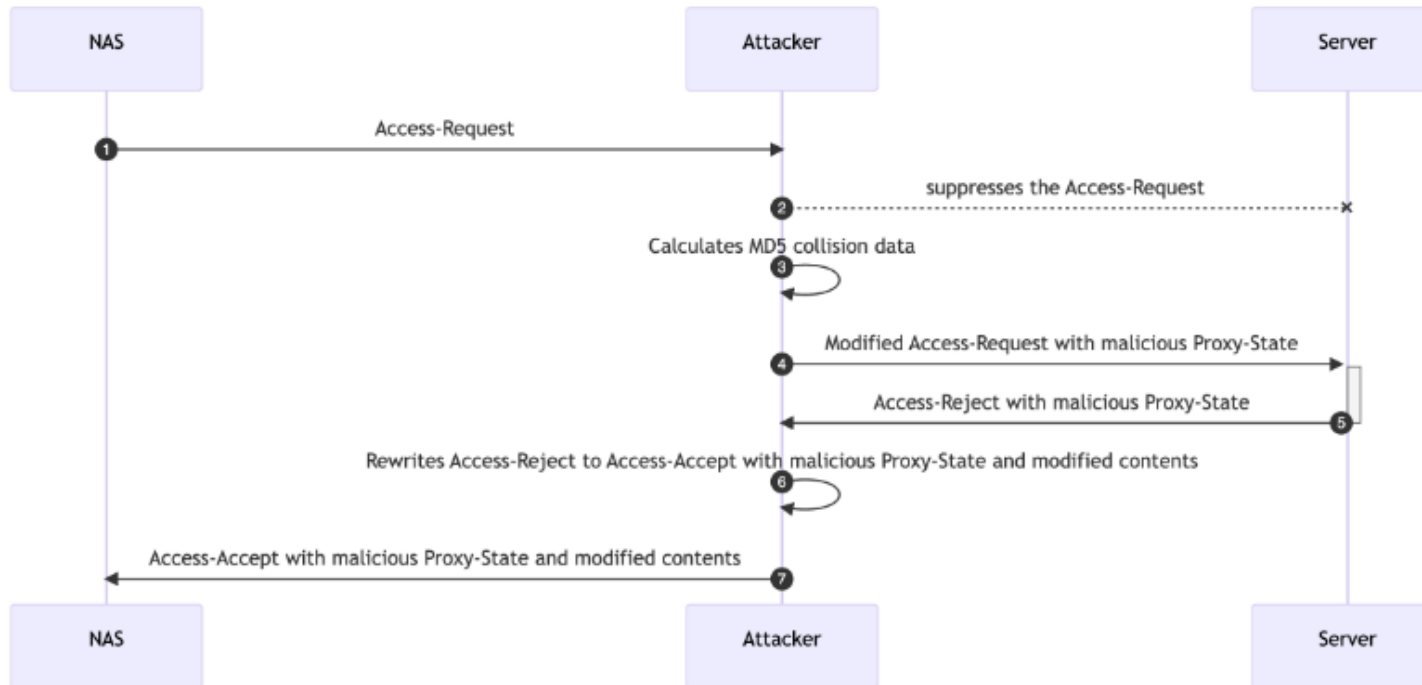


- Capture Access-Request message
- MITM & MD5 collision-> Fake Access-Request message -> Add Gibberish RejectGib -> Proxy-State attribute
- Radius Server -> Access-Reject packet -> includes Proxy-State attribute/RejectGib AND Response Authenticator MD5(..||RejectGib|| Shared Secret)
- MITM -> Access-Reject packet->Access-Accept packet+MD5 collision (Response Authenticator msg)
- Radius-Client accepts the forged Access-accept packet



# Blast Radius – details

<https://www.blastradius.fail/pdf/radius.pdf>



This vulnerability exists for many common uses of Access-Request, including packets containing PAP, CHAP, MS-CHAP.

# Blast Radius – MD5 collision

## Hashclash tool – MD5 Chosen-prefix collision attacks

$\text{MD5}(\text{RADIUS Access-Reject}||\text{RejectGibberish}||\text{Shared secret}) = \text{MD5}(\text{RADIUS Access-Accept}||\text{AcceptGibberish}||\text{Shared secret})$



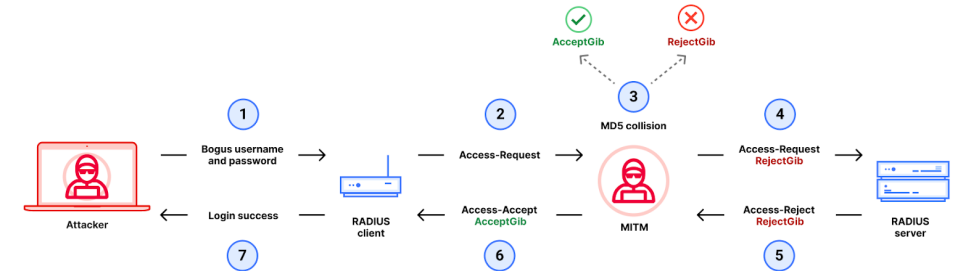
**Problem:** MD5 collision attack had to be fast! – Client could time out while waiting for a response packet (30sec to 60sec) - RFC 5080 Section 2.2.1 suggests a time limit of thirty (30) seconds and some equipment follows that recommendation



**Limitations:** a 16-byte random nonce included in the request packet. The MitM cannot predict (precompute) the *Request Authenticator* without intercepting the request packet in flight. To find MD5 chosen-prefix collisions take hours...



**Improved attack and hashclash:** MD5 considered harmful today. Creating a rogue CA certificate (2008) – A. Sotirov, Marc Stevens et al. - <https://marc-stevens.nl/research/hashclash/rogue-ca/> & <https://github.com/cr-marcstevens/hashclash/pull/37>



### 4.6 Comparison to offline brute force

For the six-block collision we chose, the computational cost of our online collision attack is dominated by the birthday phase, with a complexity of around  $2^{43.2}$ , and is mostly done using GPUs. Comparing this cost to the cost of the offline brute force attack to recover shared secrets discussed in Section 2.2, this is somewhat less than the cost to brute force a 7-character shared secret. Current recommendations prescribe a minimum shared secret length of at least 24 octets in order to protect against offline brute force attacks [18]. Unfortunately, while it is straightforward to mitigate brute force attacks against the shared secret by generating longer shared secrets uniformly at random, this has no effect on the cost of our collision attack.

<https://www.blastradius.fail/pdf/radius.pdf>

**Improved attack** can run **under five minutes** on popular commercial RADIUS implementations  
2000 CPU cores ranging from 7 to 10 years old -> Not enough but highly parallelizable → [CVE-2024-3596](#) and [VU#456537](#)


# Specific countermeasures


Deprecating Insecure Practices in RADIUS - <https://datatracker.ietf.org/doc/draft-ietf-radext-deprecating-radius/>


Source: BLASTRADIUS VENDOR GUIDE - VU#456537 – InkBridge Networks - by Alan DeKok of FreeRADIUS.

<https://datatracker.ietf.org/doc/draft-ietf-radext-radiusdtls-bis/03/>

 Unaffected systems - systems which only perform EAP authentication, such as with 802.1X / WPA Enterprise.


 Physically secure all networking equipment.


 All RADIUS traffic should use a management VLAN (no user traffic VLAN).


 All RADIUS traffic sent (over the Internet) should be secured with TLS or IPSec. **Alert** – “hop by hop” - the weakest link (rfc 7585)

Short-term mitigations if TLS no possible (FreeRadius 3.2.4, Radiator 4.29, ...):

 **RADIUS/UDP traffic should never be sent over the Internet && It is technically possible to implement the attack for RADIUS/TCP**

 Decreasing timeouts (can negatively affect normal traffic)

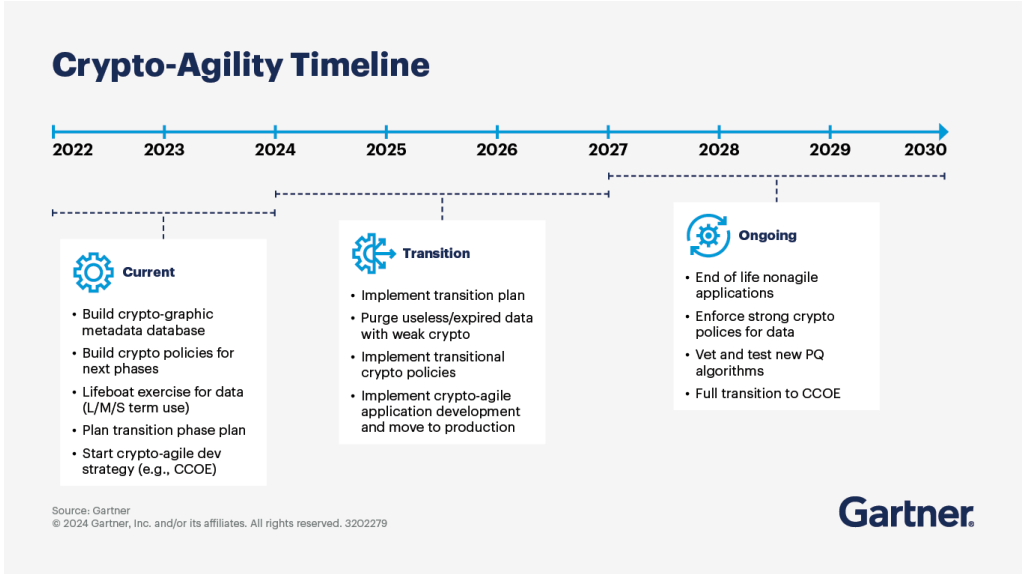
  **Clients and servers may include Message-Authenticator in all Access-Request packets and in responses to those requests (suggestion of Section 2.2.2 of RFC5080).**

 **Minimum all RADIUS servers must be updated (ideally clients too)**

Message-Authenticator attribute present in Access-Request packets is not sufficient. The server must require that the attribute is present, and discard packets where it is missing. Similarly, the client should also require that the attribute is present, and discard packets where it is missing

# Food for thought...

**CRYPTOAGILITY** - Process that allows organizations to monitor and inventory the use of cryptography in their technological infrastructures (IT/OT) and remedy the problems detected by relying on new cryptographic algorithms. It is a strategic pillar in the face of the quantum threat and compliance with cybersecurity standards (NIST, MITRE, ISO, ENS, NIS2, eIDAS, etc.).

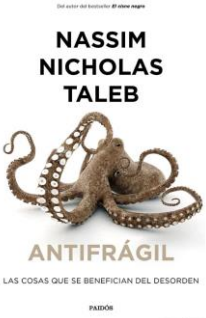


<https://www.gartner.com/en/articles/post-quantum-cryptography>

Cryptographic inventory (active and recurrent process)  
CBOM (Cryptography Bill of Materials)/ SBOM  
<https://github.com/IBM/CBOM>

This is why the Santander CyberSecurity Research (CSR) team started two great #opensource projects that will democratize cryptography discovery:

- 👉 CryptoBOM-Forge ([https://lnkd.in/dwrEXc\\_k](https://lnkd.in/dwrEXc_k)) will create a Cryptographic Bill of Materials from your code, including imported code.
- 👉 CryptoMon (<https://lnkd.in/dwwhAKJf>) will discover the cryptography in your network traffic.



**Anti-fragile organizations**  
**Zero trust: segmentation and behavioral analysis (AI)**  
<https://dodcio.defense.gov/Portals/0/Documents/Library/DoD-ZTStrategy.pdf>






**XVIII  
JORNADAS  
STIC  
CCN-CERT**

**VI  
JORNADAS  
DE CIBER\_  
DEFENSA  
ESPDEF-CERT**



Dr. Alfonso Muñoz - Founder CriptoRed

  @mindcrypt  t.me/Criptored

<https://www.linkedin.com/in/alfonsomuñoz/>

<https://www.criptored.es>

[alfonso@criptored.com](mailto:alfonso@criptored.com)

# Gracias



/RootedCON®

**CIBERDEFENSA ACTIVA  
PARA UN MUNDO DIGITAL**