

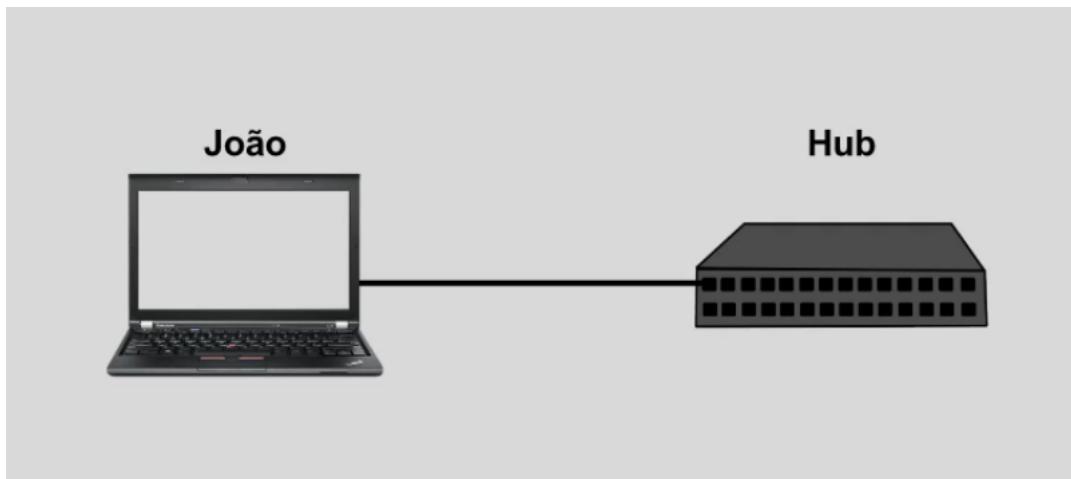
02

Hub wireshark

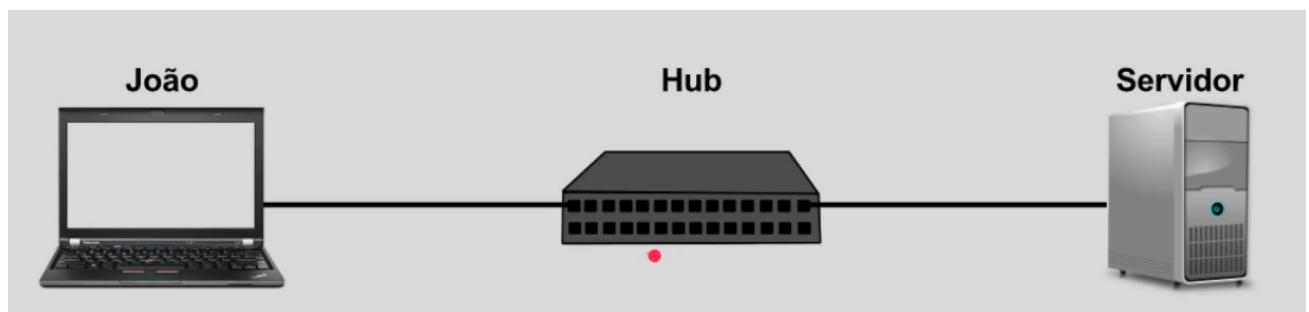
Transcrição

No curso de [redes](https://cursos.alura.com.br/course/redes-introducao) (<https://cursos.alura.com.br/course/redes-introducao>), havíamos conversado sobre o funcionamento de alguns equipamentos. Começamos pelo hub, que é usado para interconectar dispositivos finais. Vamos lembrar como ele trabalha?

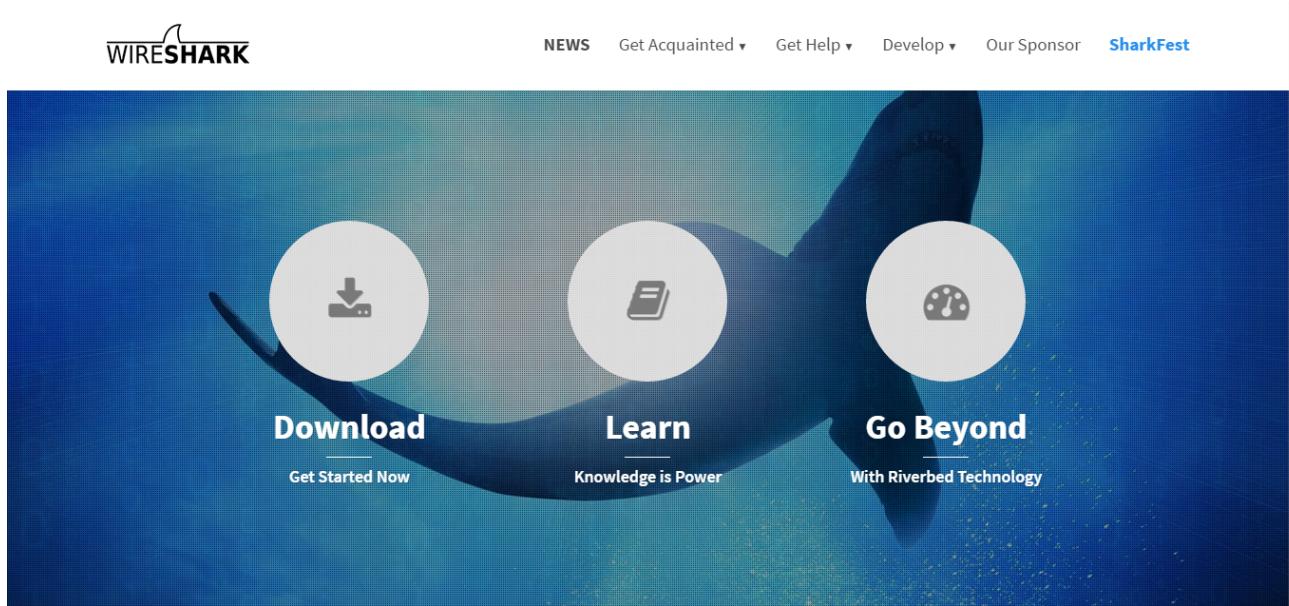
Suponha que tenhamos João com o seu computador, e que este computador está conectado a um hub.



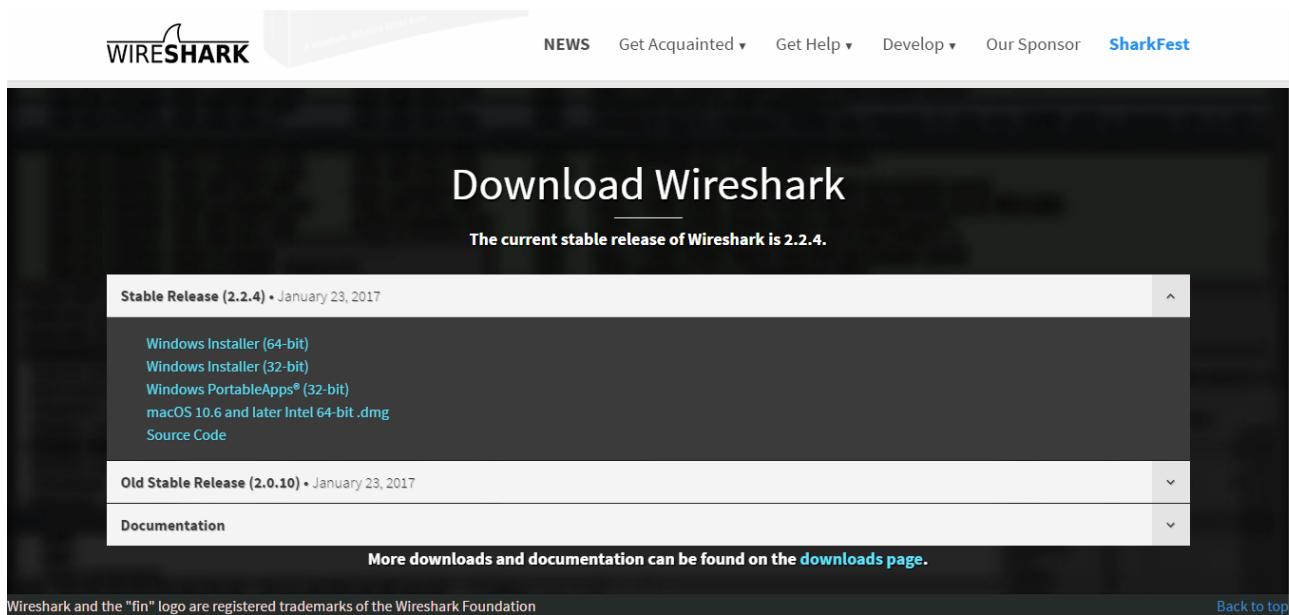
No curso de redes, fizemos algumas simulações de acesso a algumas páginas da internet, usando um servidor. No entanto, o hub tem uma limitação: ele não consegue identificar qual equipamento está conectado em cada porta.



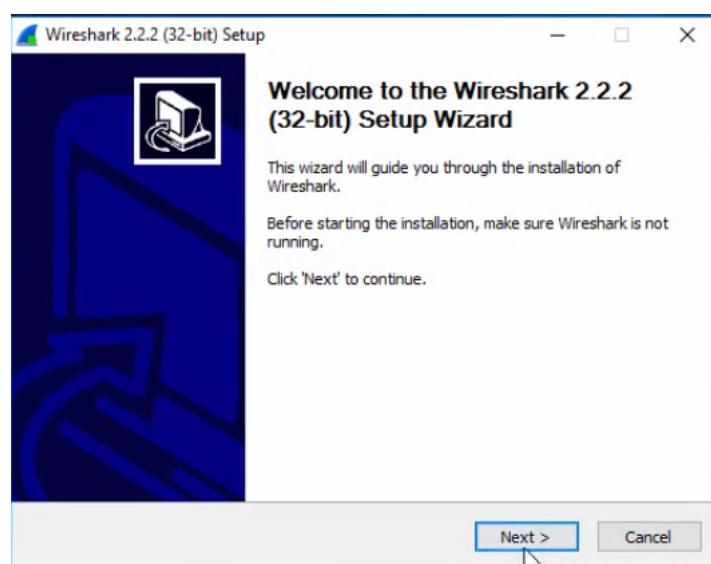
Ele não sabe que o computador do João está à esquerda e o servidor à direita. Sabendo como o hub trabalha, o hacker irá conectar o computador dele e usar programas que analisam tráfego e os protocolos que estão passando pela rede. O seu objetivo é descobrir o que João, que agora se tornou uma vítima, está acessando. O programa usado pelo hacker para colher esses dados é o [Wireshark](https://www.wireshark.org/) (<https://www.wireshark.org/>). Você pode baixá-lo e fazer uso do mesmo neste curso.



Clicando em **Download**, você é redirecionado para uma página que te permite escolher o seu sistema operacional.



Basta clicar no que corresponde a sua máquina e o download se iniciará automaticamente. A instalação desse programa é bem imediata, basta abrir o instalador e clicar em **Next** e **I Agree** até que a instalação peça mais autorizações.



Logo a seguir abre-se o Wizard de instalação do WinPcap, que é um programa que roda por trás para pegar os protocolos. Também iremos instalá-lo.



Depois de completar as instalações, abriremos o Wireshark.



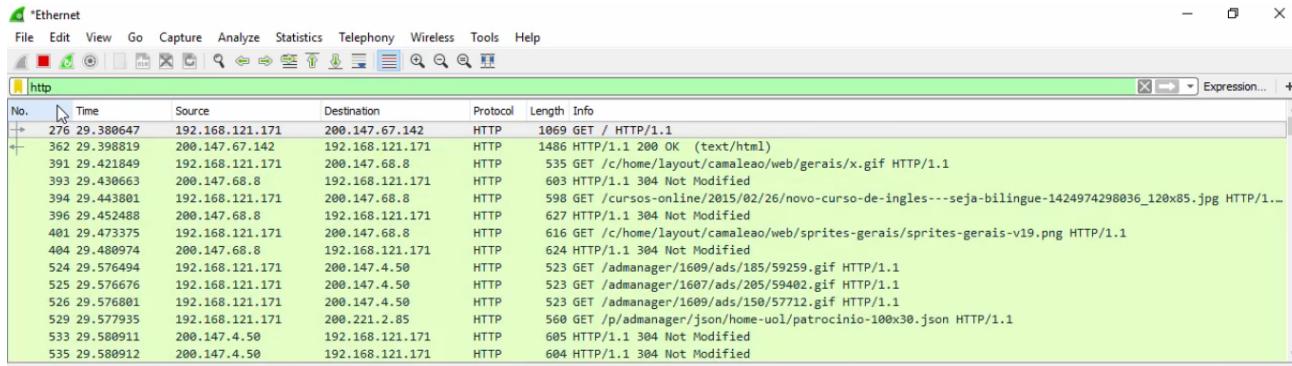
Estou usando conexão cabeada aqui, por isso o programa mostra a tecnologia `Ethernet`. Na sua casa você terá outras placas de rede, e essa entrada estará de acordo com elas. Ao clicar duas vezes sobre essa placa de rede, o programa começará a nos mostrar uma série de protocolos que estão passando pela rede.



O João, a nossa vítima, quer ler algumas notícias na internet. Para isso, vai acessar o site da Uol.

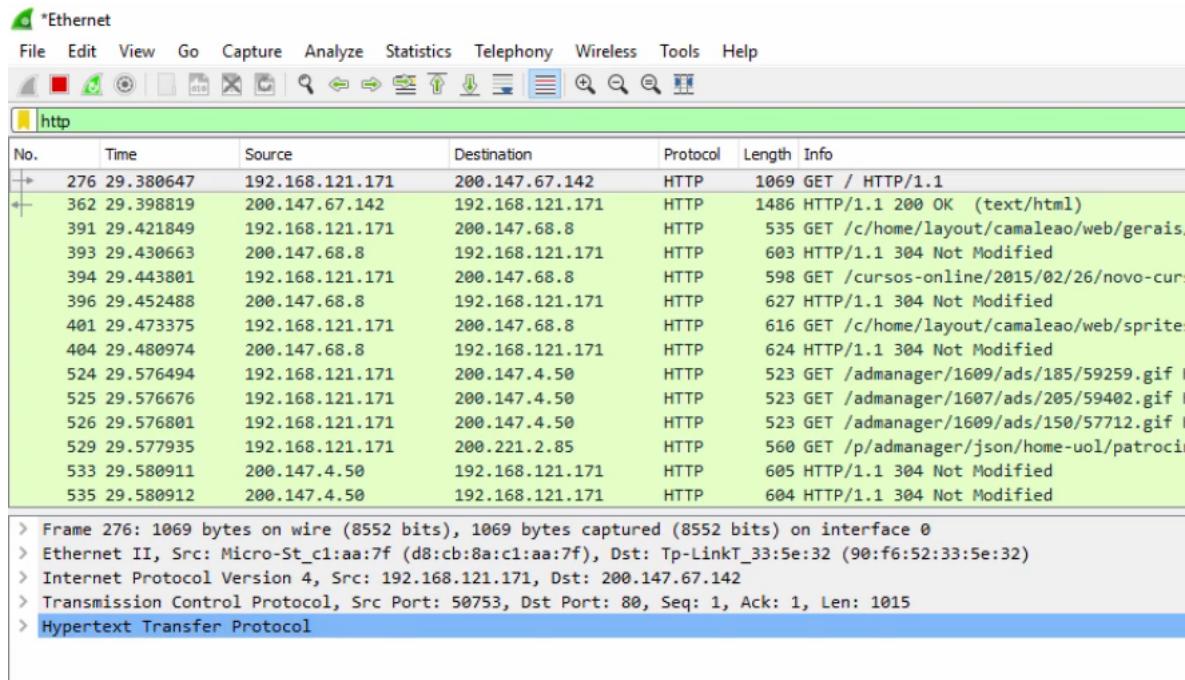
Enquanto isso, o hub está passando a informação para todas as portas e o hacker está colhendo as informações de tudo o que o João acessa.

O hacker fará uma análise de protocolos no Wireshark. Faremos uma busca por protocolos HTTP, aplicando um filtro na parte superior.



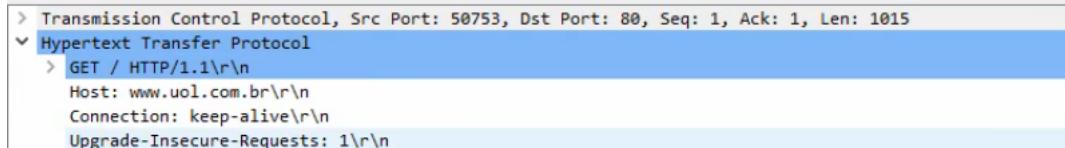
No.	Time	Source	Destination	Protocol	Length	Info
276	29.380647	192.168.121.171	200.147.67.142	HTTP	1069	1069 GET / HTTP/1.1
362	29.398819	200.147.67.142	192.168.121.171	HTTP	1486	1486 HTTP/1.1 200 OK (text/html)
391	29.421849	192.168.121.171	200.147.68.8	HTTP	535	535 GET /c/home/layout/camaleao/web/gerais/x.gif HTTP/1.1
393	29.430663	200.147.68.8	192.168.121.171	HTTP	603	603 HTTP/1.1 304 Not Modified
394	29.443801	192.168.121.171	200.147.68.8	HTTP	598	598 GET /cursos-online/2015/02/26/novo-curso-de-ingles---seja-bilingue-1424974298036_120x85.jpg HTTP/1.1
396	29.452488	200.147.68.8	192.168.121.171	HTTP	627	627 HTTP/1.1 304 Not Modified
401	29.473375	192.168.121.171	200.147.68.8	HTTP	616	616 GET /c/home/layout/camaleao/web/sprites-gerais/sprites-gerais-v19.png HTTP/1.1
404	29.480974	200.147.68.8	192.168.121.171	HTTP	624	624 HTTP/1.1 304 Not Modified
524	29.576494	192.168.121.171	200.147.4.50	HTTP	523	523 GET /admanager/1609/ads/185/59259.gif HTTP/1.1
525	29.576676	192.168.121.171	200.147.4.50	HTTP	523	523 GET /admanager/1607/ads/205/59402.gif HTTP/1.1
526	29.576801	192.168.121.171	200.147.4.50	HTTP	523	523 GET /admanager/1609/ads/150/57712.gif HTTP/1.1
529	29.577935	192.168.121.171	200.221.2.85	HTTP	560	560 GET /p/admanager/json/home-uol/patrocino-100x30.json HTTP/1.1
533	29.580911	200.147.4.50	192.168.121.171	HTTP	605	605 HTTP/1.1 304 Not Modified
535	29.580912	200.147.4.50	192.168.121.171	HTTP	604	604 HTTP/1.1 304 Not Modified

Agora vemos apenas os protocolos HTTP que foram transmitidos pelo hub. Perceba que o primeiro deles já é uma requisição GET. Sabemos então que algum usuário fez essa requisição para alguma página. Se olharmos embaixo, veremos que há o campo Hypertext Transfer Protocol.



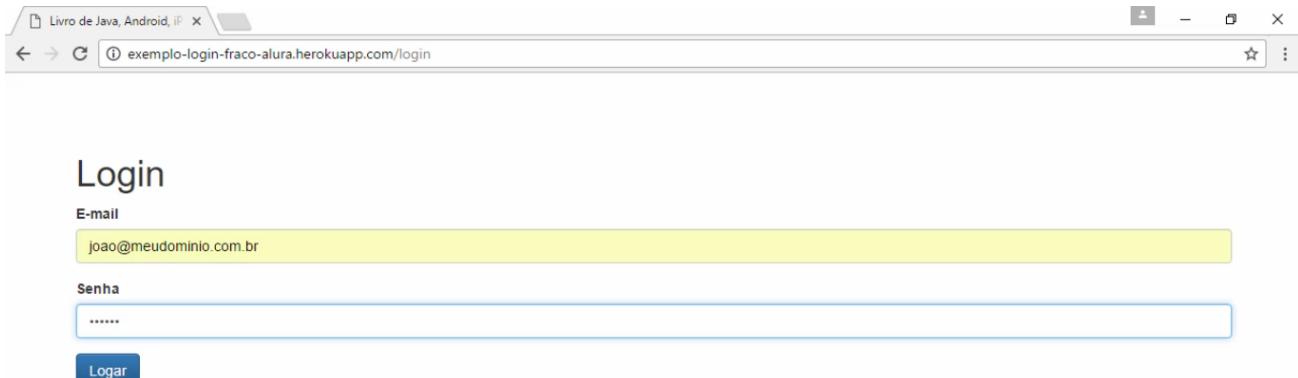
No.	Time	Source	Destination	Protocol	Length	Info
276	29.380647	192.168.121.171	200.147.67.142	HTTP	1069	1069 GET / HTTP/1.1
362	29.398819	200.147.67.142	192.168.121.171	HTTP	1486	1486 HTTP/1.1 200 OK (text/html)
391	29.421849	192.168.121.171	200.147.68.8	HTTP	535	535 GET /c/home/layout/camaleao/web/gerais/x.gif HTTP/1.1
393	29.430663	200.147.68.8	192.168.121.171	HTTP	603	603 HTTP/1.1 304 Not Modified
394	29.443801	192.168.121.171	200.147.68.8	HTTP	598	598 GET /cursos-online/2015/02/26/novo-curso-de-ingles---seja-bilingue-1424974298036_120x85.jpg HTTP/1.1
396	29.452488	200.147.68.8	192.168.121.171	HTTP	627	627 HTTP/1.1 304 Not Modified
401	29.473375	192.168.121.171	200.147.68.8	HTTP	616	616 GET /c/home/layout/camaleao/web/sprites-gerais/sprites-gerais-v19.png HTTP/1.1
404	29.480974	200.147.68.8	192.168.121.171	HTTP	624	624 HTTP/1.1 304 Not Modified
524	29.576494	192.168.121.171	200.147.4.50	HTTP	523	523 GET /admanager/1609/ads/185/59259.gif HTTP/1.1
525	29.576676	192.168.121.171	200.147.4.50	HTTP	523	523 GET /admanager/1607/ads/205/59402.gif HTTP/1.1
526	29.576801	192.168.121.171	200.147.4.50	HTTP	523	523 GET /admanager/1609/ads/150/57712.gif HTTP/1.1
529	29.577935	192.168.121.171	200.221.2.85	HTTP	560	560 GET /p/admanager/json/home-uol/patrocino-100x30.json HTTP/1.1
533	29.580911	200.147.4.50	192.168.121.171	HTTP	605	605 HTTP/1.1 304 Not Modified
535	29.580912	200.147.4.50	192.168.121.171	HTTP	604	604 HTTP/1.1 304 Not Modified

Esse é o nome oficial do HTTP, e, clicando sobre ele vemos que houve uma requisição GET, e que ela foi feita para o site da Uol. Assim o hacker sabe que o João acessou esse site.



> Transmission Control Protocol, Src Port: 50753, Dst Port: 80, Seq: 1, Ack: 1, Len: 1015
<-- Hypertext Transfer Protocol
> GET / HTTP/1.1\r\n
Host: www.uol.com.br\r\n
Connection: keep-alive\r\n
Upgrade-Insecure-Requests: 1\r\n

E o hacker pode saber ainda mais. Suponha que o João precisa fazer cadastro em um site e se logar na parte administrativa deste site.



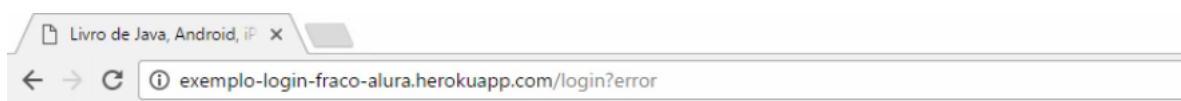
Login

E-mail

Senha

Logar

Quando o João se loga, o hacker está obtendo as informações. Vamos inspecionar a página para ver como está estruturado esse formulário.



Livro de Java, Android, IP

exemplo-login-fraco-alura.herokuapp.com/login?error

Login

E-mail

Senha

Logar

Voltar	Alt+Seta para a esquerda
Avançar	Alt+Seta para a direita
Recarregar	Ctrl+R
Salvar como...	Ctrl+S
Imprimir...	Ctrl+P
Transmitir...	
Traduzir para o português	
Exibir código fonte da página	Ctrl+U
Inspecionar	Ctrl+Shift+I

O que veremos, abrindo cada nível, é o seguinte código:

```
...<html>==</head>
<head>...</head>
<body>
<div class="container">
  ::before
  <h1>Login</h1>
  <form id="command" action="/login" method="POST">
    <div class="form-group">...</div>
    <div class="form-group">...</div>
    <button type="submit" class="btn btn-primary">Logar</button>
    <div>...</div>
  </form>
  ::after
</div>
</body>
</html>
```

Observe que fizemos uma requisição POST, e é por meio dela que o usuário envia as informações para o servidor.

Vamos ver se no Wireshark nós também encontramos essa requisição.

Como é uma requisição recente, vamos descer a barra de rolagem para encontrar essa requisição.

Wireshark interface showing a list of network captures. The 'http' tab is selected. A specific POST request (No. 3720) is highlighted in green. The details pane shows the request details for this POST /login HTTP/1.1. The bottom pane shows the expanded Hypertext Transfer Protocol details for this request.

Clicando sobre ele, veremos mais detalhes no campo inferior.

Hipertext Transfer Protocol

```
POST /login HTTP/1.1\r\n
Host: exemplo-login-fraco-alura.herokuapp.com\r\n
Connection: keep-alive\r\n
Content-Length: 92\r\n
Cache-Control: max-age=0\r\n
Origin: http://exemplo-login-fraco-alura.herokuapp.com\r\n
Upgrade-Insecure-Requests 1\r\n
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) \r\n
Content-Type: application/x-www-form-urlencoded\r\n
Accept: text/html, application/xhtml+xml, application/xml;q=0.9, image/webp,*/*;q=0.8\r\n
Referer: http://exemplo-login-fraco-alura.herokuapp.com/login\r\n
Accept-Encoding: gzip, deflate\r\n
Accept-Language: pt-BR, pt;q=0.8,en-US;q=0.6,en;q=0.4\r\n
Cookie: JSESSIONID=DDE16B48A56B0C958F2FE8413BC16F8F\r\n
\r\n
```

Vemos que houve uma requisição POST e a página em que o João a fez era exemplo-login-fraco-

alura.herokuapp.com\r\n. Mas não há nenhuma informação tão consistente. Será que o hacker teria acesso à senha ou ao login do João?

Embaixo do campo Hypertext Transfer Protocol, há o campo HTML Form URL Encoded: application/x-www-form-urlencoded .

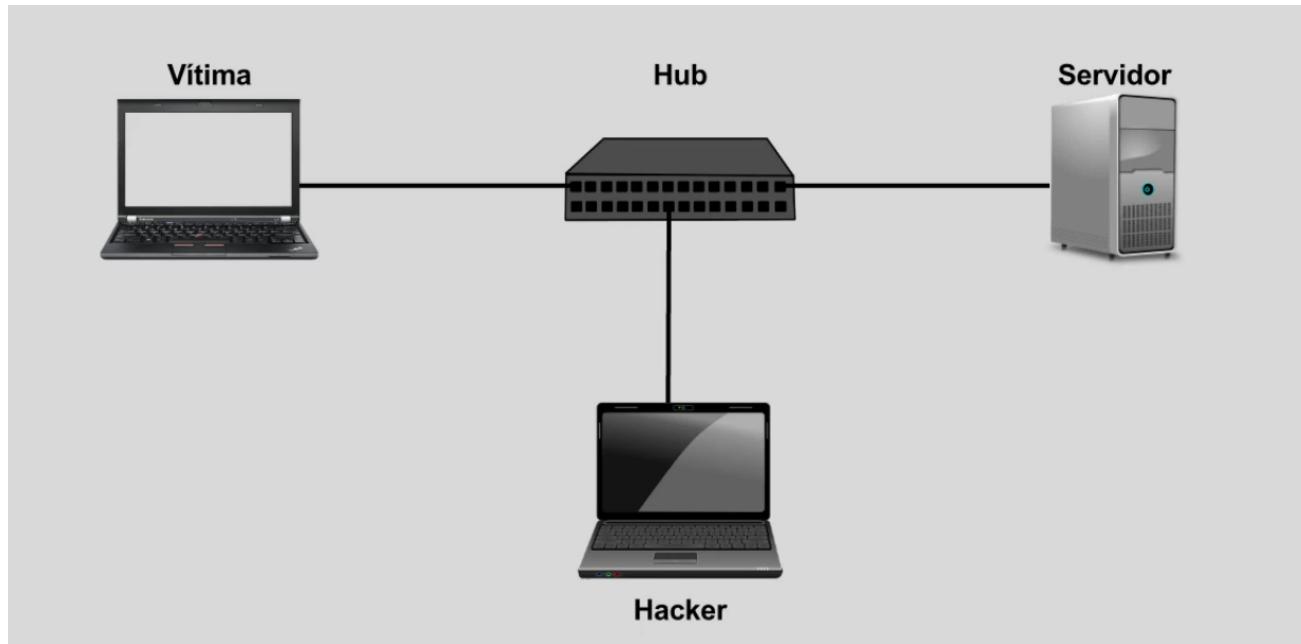
Wireshark interface showing the details of the POST request. The bottom pane shows the expanded HTML Form URL Encoded details for this request.

Quando o abrirmos, teremos:

```
Form item: "username" = "joao@meudominio.com.br"
Form item: "password" = "123456"
Form item: "_csrf" = "469bde26-87c6-4e9c-82d3-ae2c61a41fe3"
```

Conseguimos ver o email e a senha que o João digitou no cadastro do site.

Percebemos uma limitação do hub, que é essa vulnerabilidade do usuário. O hacker consegue fazer essa análise dos protocolos e ver as informações que os outros usuários estão transmitindo. Além disso, há também um problema de lentidão.



Como o hub não consegue identificar que máquina está em cada porta, ele passa essas informações para todas as portas. E por isso, temos a lentidão.

Existe um equipamento que é praticamente uma evolução do hub, e que veio suprir as limitações que mencionamos. Essa tecnologia é o switch. Em breve veremos como ele trabalha. Até lá!