Soren DeHaan and Carl Tankersley

(For future reference, is this too weird? In our defense, you did say to tell a story.)

Perhaps you have heard the rumors. Rumors of a place that does not exist in our physical world, yet it exists all the same. This particular place, though existing entirely within the Thin World, is run by a being of the Thick World, and is held secret, known only to a small and devout circle in proximity to this being. The locale, named after its caretaker, is guarded by a daemon of the Thin World, designed to ward off any unworthy interlopers, be they of the Thin World, or the Thick.

But, alas, this daemon is easily outwitted, as we shall later see.

```
GET /basicauth/ HTTP/1.1
Host: cs231.jeffondich.com
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:78.0) Gecko/20100101 Firefox/78.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Connection: keep-alive
Upgrade-Insecure-Requests: 1
```

Here is our first attempt at contact. A simple request to the daemon- may we be let in? The daemon has yet to decide, as we are unidentified.

```
HTTP/1.1 401 Unauthorized
Server: nginx/1.14.0 (Ubuntu)
Date: Fri, 09 Apr 2021 16:29:40 GMT
Content-Type: text/html
Content-Length: 204
Connection: keep-alive
WWW-Authenticate: Basic realm="Protected Area"
```

“Unworthy,” it says, “until proven worthy. Who are you?”

```
<html>
<head><title>401 Authorization Required</title></head>
<body bgcolor="white">
<center><h1>401 Authorization Required</h1></center>
<hr><center>nginx/1.14.0 (Ubuntu)</center>
</body>
</html>
```

Or rather:

Authorization Required

But we understand its meaning.

```
GET /basicauth/ HTTP/1.1
Host: cs231.jeffondich.com
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:78.0) Gecko/20100101 Firefox/78.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Connection: keep-alive
Upgrade-Insecure-Requests: 1
Authorization: Basic Y3MyMzE6cGFzc3dvcmQ=
```

“Accept this offering,” we say, “in the name of *Y3MyMzE6cGFzc3dvcmQ=*.”

Note: in our native language, this translates to “cs231:password”, but we ought to address the daemon in its own tongue.

```
HTTP/1.1 200 OK
Server: nginx/1.14.0 (Ubuntu)
Date: Fri, 09 Apr 2021 16:29:43 GMT
Content-Type: text/html
Transfer-Encoding: chunked
Connection: keep-alive
Content-Encoding: gzip
```

“You have proven yourselves worthy; you may enter.” The daemon stands aside.

```
<html>
<head><title>Index of /basicauth/</title></head>
<body bgcolor="white">
<h1>Index of /basicauth/</h1><hr><pre><a href="../">../</a>
<a href="amateurs.txt">amateurs.txt</a>                                    30-Mar-2021 16:58                  75
<a href="concrete.txt">concrete.txt</a>                                    30-Mar-2021 16:58                 161
<a href="pigs.txt">pigs.txt</a>                                          30-Mar-2021 17:09                 227
</pre><hr></body>
</html>
```

Here, we find the sacred texts. Amateurs, concrete, pigs. This is the trove of knowledge which the daemon strives to protect.

Yet… it does not sit right. What would happen if someone unworthy were to gain access to the sanctum through underhanded means? The caretaker has entrusted a great deal to the daemon, so we must be certain of its efficacy.

Let us reexamine our journey through a different lens. For while the daemon has done an admirable job in allowing us worthy access, it must also keep out the unworthy. Consider this, dear reader: each of these stages was performed in plain view of anyone who desired to look.

That we attempted to contact the daemon, and that it was wary- these are all fine for an unworthy observer. But recall, the test of worthiness is merely the knowledge of the name of *Y3MyMzE6cGFzc3dvcmQ=*! While this may seem foreign to a native of the Thick World, it is a simple matter of translation to one learned in the tongues of the Thin World. By exposing the secret method of entry so incautiously, every access to the daemon-warded sanctum would surely enable any nearby to follow in kind, worthy or otherwise.

But this is not all. For not only have we bared the sacred name of *Y3MyMzE6cGFzc3dvcmQ=* in an unguarded place, but the daemon opens wide the door to let us in. And as we peer inside, so too may anyone who has taken an interest in our activity. The sacred texts, on full display! This will not do; if we are to keep the contents of the sanctum secure, we must ensure not only that the name of *Y3MyMzE6cGFzc3dvcmQ=* is not spoken so indiscreetly, but also that the contents are not visible to those who have not spoken it.

Our daemon is nought but a deterrent for errant passers-by. Perhaps we ought to bring word to the caretaker.

On a more serious note, what is actually going on here is relatively simple to explain. The connection between our computer and the server starts off like any other connection to a website with the three part TCP handshake. Once the handshake has been completed successfully, we then send a GET request to the server asking for the /basicauth/ directory, to which the server responds, after acknowledging that it's received our request, saying that we need to enter our login information to access this part of the site. We then acknowledge receipt of this message and resend our GET request, but this time we send along our username and password with it. This information is slightly obfuscated, though we wouldn't call it encrypted, by converting it from ASCII characters to a number in base 64. This number is then sent in plain text with our second GET request, which the server then acknowledges and gives us access to the directory. Once we acknowledge that we've received access to the directory, we are free to explore its contents.

The primary issue with this authentication scheme is that our login credentials are sent in a very slightly obfuscated format over the network with no encryption of any sort. This information isn't even hidden; it's clearly labelled "Authorization: Basic" so that anyone who intercepts these packets knows to look at that specific part to find the chunk of the data that is being used to authenticate us as authorized users. The base 64 encoding is trivial to break; an attacker with a pencil and a piece of paper, an ASCII table, and about five minutes can decode it, or they can do it almost instantaneously with any number of base 64 to ASCII conversion tools, which can be found online and run in a browser. Consequently, it would not be wise to use basic authentication to protect any data that is even slightly sensitive, as the login prompt serves more as a deterrent to prying eyes without much of a tech background than as an actual security measure. Any knowledgeable attacker can access it with almost trivial ease.

However, this excessive transparency and lack of security or encryption does allow us to see how the actual exchange conforms to the standards set in the specification for the protocol. For example, we can see in the intercepted HTTP packet containing the login information that it is indeed stored as a base 64 string, which, when decoded back into ASCII text, is of the form "username:password", as is specified in the standards. We see that the field for the user agent is also present, as is required by the specifications.

Even the highlighted problems above seem to follow the specifications:

"The Basic authentication scheme is not a secure method of user authentication, nor does it in any way protect the entity, which is transmitted in cleartext across the physical network used as the carrier... it results in the cleartext transmission of the user's password over the physical network."