# Elf Web Access--Encryption Gone Wrong Part 6, Lessons Learned

Some of you may ask, “If all we had to do was follow Pepper’s hint and change the length of the ciphertext to 16 bytes (before base64 encoding), why did we mess with all that code stuff?” That’s a valid question. A “fuzzer” that just submits random input would have come across the vulnerability eventually. However, we learned a lot about how the vulnerability really works by examining the code.

## A Note from an expert

I asked the Counterhack team if the EWA cryptography challenge was based on an implementation they had come across in the real world, or specially written for the challenge. This is the answer from Chris Davis, who created the EWA challenge.

To answer your question, the Elf Web Access uses a mixture of technologies that are used in practice (NodeJS, AES Modules, and other libraries) etc... then under the hood we used post fix and send mail to create a Linux email server. On the other hand, the web server itself was mostly custom coded made for just this specific scenario. I would say 90% of the web front end and NodeJS server-side code were custom crafted for this specific instance to create web access to the email services. In doing so, we hoped to create a challenge that’s answer was not easily “Googled” or that contained public exploit code. In this way, participants would be challenged to come up with a solution themselves and would also have to understand the fundamental technologies at work. As such, you won’t see this specific interface in the wild.

With that said, Fortune 500 companies and even many small to medium sized businesses are notorious for creating custom web applications to interface with their business applications. As such, secure coding practices, code reviews, fuzzing and security assessments for such applications are critical. In fact, the primary flaws found in this application are common security flaws and can be found in a the OWASP top 10 vulnerabilities.

I personally have partaken in Penetration tests or real-world organizations in which I found custom code similar to the EWA challenge that totally broke the authentication process. In fact, I’ve seen even worse authentication mechanisms than the EWA challenge used in the real world.

## Question 1

At the simplest level, Alabaster could improve his cookie scheme by checking the plaintext that is returned in his cookie. If the plaintext is an empty string, the cookie\_checker function should return a failure (false). However, there are other problems. Alabaster sould study the suggestions at the [Open Web Application Security Project](https://www.owasp.org/index.php/About_The_Open_Web_Application_Security_Project) (OWASP) about [session management](https://www.owasp.org/index.php/Session_Management_Cheat_Sheet), and then rebuild his cookie scheme from the beginning. At a minimum, the user name should be encrypted in the ciphertext, and not left as a separate value in the cookie where it can be changed by the user. Any user that can log in can see other people’s email just by editing the name value of the cookie. Also, sending plaintext and ciphertext together is bad form, even if the AES algorithm is strong.

## Question 2

The answer to this question is mostly a repeat of the final question from the previous challenges. Santa, Alabaster, and the North Pole security team should ask themselves some questions.

1. Is the security of North Pole servers checked by a knowledgeable security team different from those who develop the servers? Do the development and security teams work together during the development process?
2. Is SSH access to the Letters to Santa server from the Internet necessary? If so, how is it controlled and monitored? Can it be configured to use multi-factor authentication?
3. How is access to the internet facing servers monitored? Is there an IPS, IDS, or other network device? Does anyone look at the logs?
4. When servers exposed to the Internet access internal computers, how is that controlled and monitored? Is there a reason that the Letters to Santa server should be logging in to the email server, the file server, or other internal assets? Shouldn’t that access set off alarms?
5. What is the procedure for testing server security before servers are moved into production? Are there final checks to be sure that only tested code that is necessary for the server to do its job is included in the server?