**EPQ Presentation Script**

**Slide 1:** In 499 BC, Histiaeus, tyrant ruler of the ancient Greek city Miletus, calls forth a slave and shaves the hair from his head. Upon the scalp he tattoos orders for his son-in-law Aristagoras to spark a revolt in a nearby city. Once the slave’s hair grows back he is dispatched, and no one questions him as he passes, allowing his message to be safely transported to Aristagoras. This is one of the first recorded cases of an information hiding technique called steganography.

I first heard about this subject at a computer science course last year, where the lecturer gave an overview of steganography, the science of hiding information in plain sight. This includes techniques like invisible ink, shrinking messages until they fit inside a full stop, and many digital methods, one being Least Significant Bit modification. In summary, every colour on a computer screen has a binary representation, such as this purple square. If I were to invert the first digits (called bits) in each set, the colour would change, to this sea green. But inverting the last bits has barely any effect – a potential hiding space. The lecture left me wanting to explore the topic further, so I settled on the title for my EPQ: ‘What is steganography, and how can I use it to conceal a message in a digital image, and then retrieve it?’

This title provided a lot of scope for me to further investigate steganography, including: a definition of the concept; its history and non-digital methods; modern applications of the technique, as well as who may be using it and why; and a section on digital images, which is where I explained how Least Significant Bit (or LSB) modification actually works. Then I was able to implement the technique for myself in an artefact, a Python program that hides a message inside an image, and then retrieves it. This provided an interesting combination of cyber security and software development, and allowed me to test my skills as a programmer.

**Slide 2:** After creating a plan of the project, I began to research, collecting an assortment of facts from digital articles and using them to collate a paragraph for each subheading. These provided a template for me to write my first draft, which allowed me to identify which areas were in need of more evidence. I filled in these gaps using a variety of sources, including online articles, websites, and videos. An issue in the research was that many of the sources were extremely technical, and I had to work around intricate mathematical explanations to find the information I needed. Also not all of the sources provided enough contextual information, and in one case I found facts written identically to another source without acknowledgement. However, this allowed me to practice critiquing my sources and determining which were more reliable when I found contradictions.

**Slide 3:** Here are some of my most interesting findings. Steganography played a role in the conviction and execution of Mary Queen of Scots, where messages smuggled in beer barrels provided enough evidence of her plots against Queen Elizabeth for her to be put on trial. Nowadays steganography is commonly used by cyber criminals to hide malicious software, though there have been reported cases of use by terrorists and spies too. LSB modification can be countered using a technique called steganalysis: On the left is a picture of a house. Each colour is represented by a different code. On the right is the same house, but with only the least significant bits of each code. While the detail has been lost, this is still clearly a house, but if the LSBs had been altered to store a message, this image would be completely random, making it obvious that it has been tampered with.

**Slide 4:** The artefact allowed me to put this research to use. My goal was to hide an extract from Lewis Carroll’s Alice in Wonderland inside this image. This snapshot shows part of the code used to embed the message. The issue was that as the change is supposed to be unnoticeable, when testing the program I had no way to see if it was actually working.

**Slide 5:** My solution was to alter the pixels I was modifying to a bright pink, like this. Once that was working, I restored the original code and tested the outputs. The new image was a success, appearing identical; however the extracted message contained a few minor errors. This was due to the way Python saved the image, and I was unable to counter it. But I was able to reduce their frequency by using an image that was especially large.

**Slide 6:** Through this project, I have gained experience in time management and carrying out concise but thorough research, and the artefact allowed me to consolidate my understanding of steganography in a practical way. If I were to attempt it again I would spend more time looking for reputable sources, and perhaps investigate some already existing steganography software. Overall, the EPQ has deepened my interest in cyber security and ambition to study computer science at a university level. Any questions?