MASTER THESIS

Thesis submitted in partial fulfillment of the requirements for the degree of Master of Science in Engineering at the University of Applied Sciences Technikum Wien - Degree Program IT-Security

Development of an Optical Cavity Simulator as a Didactic Tool for IT Security Professionals

By: Benjamin Medicke, BSc

Student Number: 2010303027

Supervisors: Dr. Dr. Lukas Mairhofer
Moritz Kriegleder, BSc MSc

Vienna, September 18, 2022

Declaration

“As author and creator of this work to hand, I confirm with my signature knowledge of the relevant copyright regulations governed by higher education acts (see Urheberrechtsgesetz /Austrian copyright law as amended as well as the Statute on Studies Act Provisions / Examination Regulations of the UAS Technikum Wien as amended).

I hereby declare that I completed the present work independently and that any ideas, whether written by others or by myself, have been fully sourced and referenced. I am aware of any consequences I may face on the part of the degree program director if there should be evidence of missing autonomy and independence or evidence of any intent to fraudulently achieve a pass mark for this work (see Statute on Studies Act Provisions / Examination Regulations of the UAS Technikum Wien as amended).

I further declare that up to this date I have not published the work to hand nor have I presented it to another examination board in the same or similar form. I affirm that the version submitted matches the version in the upload tool.“

Vienna, September 18, 2022

Signature

Signiert von: Benjamin Medicke

Datum: 18.09.2022 12:34:56

Dieses mit einer qualifizierten elektronischen Signatur versehene Dokument hat gemäß Art. 25 Abs. 2 der Verordnung (EU) Nr 910/2014 vom 23. Juli 2014 ("eIDAS-VO") die gleiche Rechtswirkung wie ein handschriftlich unterschriebenes Dokument.

Dieses Dokument ist digital signiert!

Prüfinformation:
Informationen zur Prüfung der elektronischen Signatur finden Sie unter:
www.handy-signatur.at

www.a-trust.at

A TRUST
einfach sicher

Kurzfassung

Das Ziel dieser wissenschaftlichen Arbeit war die Konstruktion eines Simulators für optische Resonatoren, der als didaktisches Instrument zur Schulung von IT-Sicherheitsingenieuren in den Grundlagen der Quantenkryptografie eingesetzt werden soll. Quantenkryptografie verwendet probabilistische und nicht die bekannteren deterministischen Prozesse. Optische Resonatoren spielen eine wesentliche Rolle in mehreren vielversprechenden Ansätzen zur Entwicklung von Einzelphotonenquellen. Diese stellen einen wichtigen Schritt zur Umsetzung von Quantenkryptografieprotokollen wie BB84 und einem möglichen Bestandteil zur Entwicklung von Quantencomputern und damit zur Ausführung von Quantenkryptologie Algorithmen wie denen von Shor dar.

Das BB84-Schema beschreibt eine Methode zum Austausch eines Schlüssels über einen unsicheren, klassischen Kanal in Kombination mit einem Quantenkanal. Die Verschlüsselung auf der Grundlage von richtig gehandhabten One-Time-Pads ist nachweislich die sicherste Art der Kommunikation über einen unsicheren Kanal mit eventuell vorhandenen Angreifern, die heimlich mithören.

Shors Algorithmen bieten Lösungen in Polynomialzeit für das Problem des diskreten Logarithmus und die Primfaktorzerlegung ganzer Zahlen, die beide als schwierige Probleme für klassische Computer gelten. Diese Probleme bilden die Sicherheitsgrundlage für viele derzeit verwendete asymmetrische Kryptografiesysteme wie RSA.

Es wurde ein webbasierter Simulator für optische Resonatoren entwickelt, der Visualisierungen, Erklärungen und Formeln für alle Elemente der durchgeführten Berechnungen bietet. Darüber hinaus bietet dieser Optionen zur Automatisierung von Variablenmanipulation und Visualisierungen des sich im Inneren des Resonators aufbauenden Lichtfeldes.

Schlagworte: Quantenkryptographie, Optischer Resonator, BB84, Simulation

Abstract

The goal of this scientific work was constructing an optical cavity simulator to be used as a didactic tool to train IT security engineers in the fundamentals of quantum cryptography, which uses probabilistic processes, not the more familiar deterministic ones. Optical cavities play an essential part in several promising approaches to creating single-photon sources, an important step towards implementing quantum cryptography protocols, such as BB84, and a possible element for creating quantum computers and thus executing quantum cryptology algorithms such as Shor's.

The BB84 scheme describes a method for exchanging a key over an insecure, classical channel in combination with a quantum channel. Encryption based on properly handled one-time pads is evidentially the most secure way of communication over an insecure channel with potential eavesdroppers.

Shor's algorithms are polynomial-time solutions for the discrete logarithm problem and prime factoring integers, which are considered hard problems for classical computers. These problems are the security basis for many currently used asymmetric cryptography systems such as RSA.

We developed a web-based optical cavity simulator that provides visualizations, explanations, and formulas for all elements of the performed calculations. In addition, we include options to automate the process of manipulating variables and visualizations of the light field building up inside the cavity.

Keywords: quantum cryptography, optical cavity, BB84, simulation

Contents

1 Introduction

The past fifty years have seen a dramatic increase in application scenarios of cryptography. Before the 1970s, it was almost entirely limited to secure communication between governments, military agencies, and diplomats. Today, cryptography plays an essential role in many aspects of modern life: Governmental bodies, companies, and private individuals use it to secure their transmissions, guarantee the integrity of their messages and authenticate communication partners. In addition, industries, such as shopping, banking, and healthcare, provide services securely over the internet. Cryptography protects cryptocurrencies, Internet of things (IoT) devices talking to each other, and drones controlled from far away. Offline applications include radio-frequency identification (RFID) tags, biometrics, and car-to-car communication, to name a few. [74][48][28][121]

The field of quantum mechanics is in the unique position of both posing a problem in the form of quantum cryptanalysis and offering a potential solution, through quantum cryptography, to the IT security field [88]. McKinsey's latest research on quantum technologies [55], reports quantum computing as the sector with the most funding, while quantum communication is among the sectors with the highest increase in new funding. However, the report also identifies a global shortage of quantum technology talent, with the European Union having the highest concentration of specialists and the United States leading in the number of pertinent Master's degrees offered.

While today's quantum computers cannot break encryptions with commonly used parameters, attackers can record traffic today and decrypt it when the technology matures, requiring alternative solutions in preparation for fully functioning quantum computers. [74] Recent increases in funding for the field [21] heightened the need to prepare IT security practitioners for the threats and opportunities arising. Quantum key distribution, which uses physics to secure a key exchange, is one such opportunity. Depending on the implementation, it requires single-photon sources, which can rely on optical cavities.

The goal of this thesis is to create a tool that aids in the understanding of how light waves interact with optical cavities. For this purpose, the simulator should visualize this process and the involved variables to make the interdependencies more transparent to the user. The interaction between light waves and cavities plays a pivotal role in promising approaches to creating single-photon sources (particularly deterministic ones), allowing for quantum cryptography implementations and the construction of quantum gates and quantum bits (qubits) [18]. These components are necessary for quantum computing and thus executing quantum cryptanalysis algorithms such as Shor's or Grover's.

The emitter in an optical cavity is an interface between the classical computing model and the quantum world. IT security professionals are intimately familiar with the former strictly deterministic model based on bits and classical gates, especially so since to the increased presence of side-channel attacks in recent years, such as the hardware-based Meltdown [53] and Spectre [50] attacks that require a deep understanding of the low-level processes and hardware of classical computers. However, the arguably less intuitive probabilistic model of quantum computing requires an understanding of an entirely different set of physical phenomena and a new approach to calculating results. Quantum cryptography, too, is inherently different from classical cryptography. The, in the area of IT security, well established former category relies on mathematical complexity while the latter is based on the laws of physics.

1.1 Scientific Question

This thesis and the simulator developed in its course aim to address the following research question:

How to train IT security engineers in the foundations of quantum cryptography?

1.2 Methodology

Due to the interdisciplinary nature of the thesis (ranging from quantum mechanics over optics to all branches of cryptology) and the wide range of related topics, I performed extensive **literature research**. A particular focus was on the underlying physics that dictates the behavior of optical cavities in and out of resonance, the critical element of the developed simulator. Users of the simulator can use this thesis' Theory chapter in tandem with it.

The practical part of the project, the development of an optical cavity simulator that should aid in the didactic process, was a collaboration between Nikolai Benedikt and the author of this thesis. We identified the following **set of requirements** prior to the implementation stage:

1. The application should support a wide range of devices to facilitate a broad user base: it should support desktop and mobile devices.
2. Installation and update procedures should impose as few requirements as feasible: this encompasses operating system restrictions and access permissions.
3. Variables included in the calculations should be adjustable and provide visualizations that automatically update on changes. In addition, the chosen technology should offer support for both 2D and 3D graphics.

4. Performed calculations should be transparent to the user.
5. The simulator should be able to adjust a value without relying on constant user input.
6. The application should be extensible and accessible to new software engineers to simplify and foster further development.

We employed the **empirical method** of developing prototypes for our simulator. The initial prototype of the software, written in a different programming language (Python instead of JavaScript), lacking a user interface and using hard-coded values, iteratively improved until we reached the current version (using the method of evolutionary prototyping [92]). Once the first calculatory elements of the simulation were operational, we used pen-and-paper prototypes [92] for the visualizations, implemented variants that showed promise in a sandbox, and introduced our choices in the software prototype. From here on out, the selected visualizations took part in the standard evolutionary development process for future refinements.

1.3 Structure

The first chapter, **Introduction**, provides this thesis's motivation, scientific question, and relevance. It outlines the importance of cryptography in today's world, briefly lays out the current state of the quantum technology sector, and explains the impact the latter might have on the former and why IT security professionals should learn about it. Next, it proposes a tool to aid in this training and shows the methodology used to develop it. Finally, it concludes with this summary.

The second chapter, **Theory**, details the necessary background for understanding the thesis. The subsections are grouped by area of research, starting with Classical Cryptography to build a common foundation for the following chapters that build on explained concepts (Quantum Cryptanalysis, Quantum Cryptography and Post-Quantum Cryptography). The remaining chapter, Optics, covers concepts pertaining to light, concluding with a chapter about cavities, the focus of the optical cavity simulator.

The third chapter, **Single-Photon Sources**, takes a look at current technologies used to emit a single photon, Attenuation, Spontaneous Parametric Down Conversion and Heralding and Microcavity-Based. It explains the requirements for an ideal single-photon source and mentions alternative protocols for less-than-ideal sources and cavity metrics.

The fourth chapter, **Optical Cavity Simulator**, details the features of the programmed optical cavity simulator, discusses the advantages of web applications, and describes the technology stack and the reasons for the individual choices. Next, it demonstrates the user interface elements before it concludes with details about the architecture, implementation, and possibilities of extending the program.

The final chapter, **Summary**, discusses the produced optical cavity simulator, points out opportunities for future work, and draws a conclusion about the project.

2 Theory

The topic of the thesis encompasses multiple disciplines: IT security in general and cryptology in particular, quantum mechanics, and several branches of optics. This chapter introduces the necessary theory, context, and background information to understand the remaining thesis. The goal of the thesis is the development of an optical cavity simulator and to highlight the importance of quantum mechanics for the IT security field. The simulator should make it easier for IT security engineers to understand the basics of quantum cryptography.

2.1 Classical Cryptography

Cryptography originated at least four thousand years ago when Egyptians used non-standard hieroglyphs to disguise writings on the wall of the tomb of KHNUMHOTEP II in Beni Hasan with the presumed goal of evoking mystery and confusion in the observer. [45]

Many civilizations that developed written systems also had some form of cryptography. For example, ancient Greeks used scytales (from *skutálē* meaning *baton* or *cylinder*) to communicate during military campaigns. First, the sender wraps a strip of parchment around a cylinder with a secret diameter. Next, he or she writes the message on the parchment, unwraps the strip, and passes it along without the scytale. Finally, the recipient wraps the parchment around a cylinder with the same diameter to decrypt the message. See figure 1 on page 1.

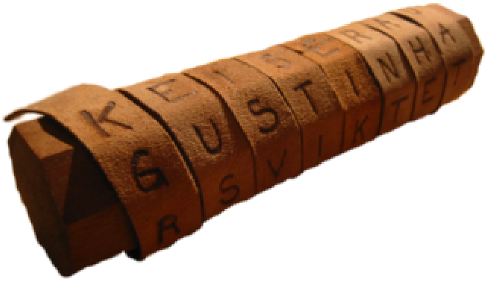


Figure 1: a scytale with its parchment [112]

Until the 1970s, cryptographic systems continued to be primarily used to protect diplomatic, governmental, and military communications. Then, in the 1980s, the banking industry followed suit. Finally, late in the same decade, the telecommunication industry deployed the first mass-marketed cryptographic system: the digital mobile phone system. [74]

2.1.1 Modern Usage of Cryptography

The etymology of the word cryptography stems from the greek *kryptós* (meaning *hidden* or *secret*) and *graphein* (meaning *to write*): Cryptography is the science of writing secrets. While privately writing and transmitting data is still a significant application scenario for cryptography, there are many more subjects that this area of research has influenced:
Today, the field of cryptography has extended the range of applications from secret communication to Digital Rights Management for digital media (such as music, games, and movies), the modern banking system, and secure storage of biometric data in passports, to name a few.

2.1.2 Classification of the Field of Cryptography

The rise of cryptography and the possibility of concealing information soon led to the desire to break encryptions. This resulted in the emergence of a related field of research: cryptanalysis, the science of breaking cryptosystems. Cryptography and cryptanalysis are both branches of the overarching area of research cryptology.

Cryptanalysis is not limited to breaking cryptographic systems but is also used to reason about the security of cryptographic protocols and algorithms and, as such, takes on an adversarial and supporting role.
Cryptanalysis contains the following subfields [74]:

- classical cryptanalysis
  - brute-force attacks
  - mathematical analysis
- implementation attacks
- social engineering

The field of cryptography can be subdivided into three branches:

- symmetric cryptography
- asymmetric cryptography
- protocols

2.1.3 Symmetric Cryptography

Symmetric Cryptography is the oldest branch of cryptography and has predated asymmetric cryptography for thousands of years. All cryptographic systems developed before 1970 belong to this category.

Symmetric cryptography schemes use the same shared secret (the key) for encryption and decryption. The relative simplicity of these schemes results in a lower overhead than asymmetric cryptography systems (that use multiple keys and different encryption and decryption functions), making them faster and easier to implement.

Terminology and definitions

Let us assume the following scenario: Alice (communication partner A) wants to talk to Bob (communication partner B) over an insecure channel while preventing a third party (Eve) from eavesdropping on them. A *channel* can be any medium that transports data. The use of an insecure channel allows Eve to listen to all data transported over it. See figure 2 on page 7.

The message to be exchanged (variable x) is the *plaintext* or *cleartext*. The encrypted message (variable y) is the *ciphertext* or *cipher*. Variable k represents the *shared secret* or *key*. The *keyspace* is the number of possible values for the key k.

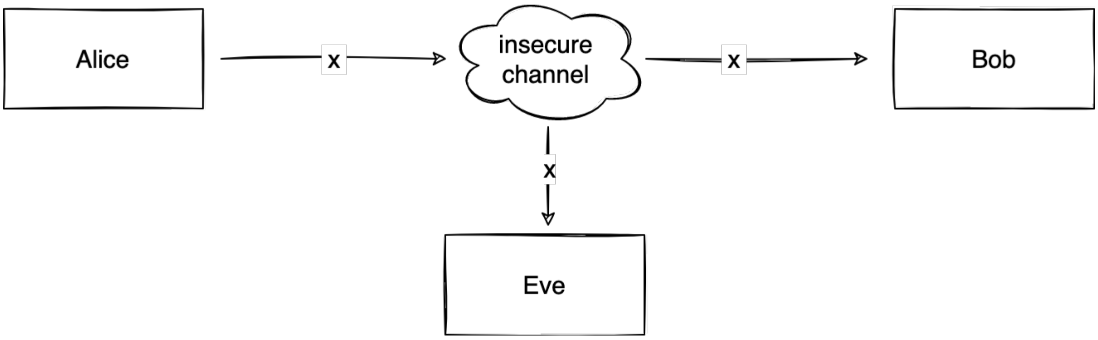


Figure 2: communication via an insecure channel, Alice and Bob are the communication partners, Eve is an eavesdropper that can read the plaintext (x) those two send to each other due to the use of an insecure channel, based on [74]

$e()$ for encryption is the function that takes the plaintext and the key as input and returns the ciphertext. The inverse operation, $d()$ for decryption, is the function that takes a key and a ciphertext as input and - provided the key is correct - returns the plaintext.

$$y = e(x, k)$$

$$x = d(y, k)$$

$$d(e(x, k), k) = x$$

The decryption function is the inverse function of the encryption function.

This system can not solve the problem of the key exchange. Alice can not send the key k over the insecure channel before the communication begins because Eve would be able to read both the key and the encrypted message, allowing her to decrypt it as well. See section

Rivest-Shamir-Adleman for a solution. The following section assumes that Alice and Bob have exchanged the key over a secure channel, and Eve is not privy to it. Such a key is called a *pre-shared* key. See figure 3 on page 3.

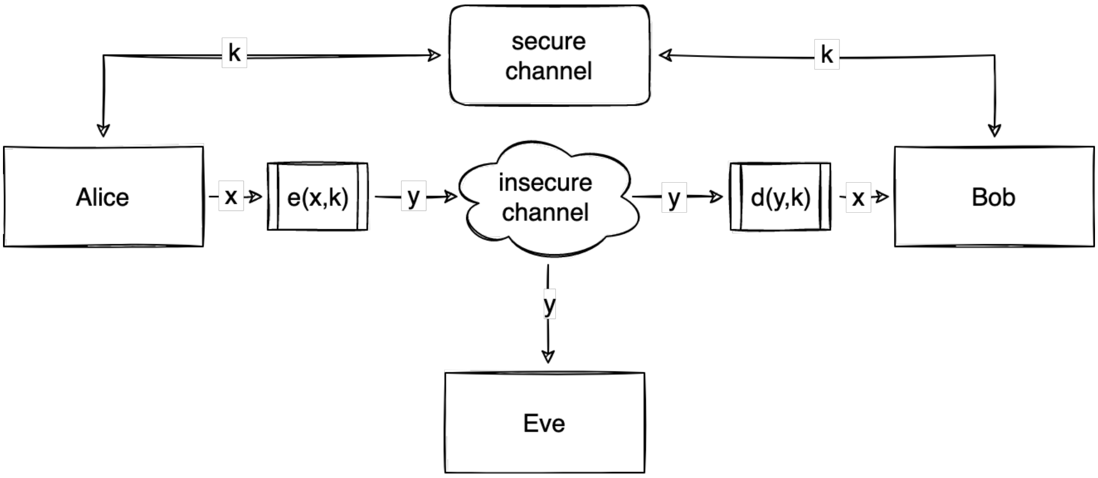


Figure 3: secure communication between Alice and Bob over an insecure channel using a pre-shared key (k), Eve can only read the ciphertext (y), which she cannot decrypt because she lacks the key, based on [74]

The security of this system does not depend on keeping the encryption and decryption functions a secret. The key is the only element that Alice and Bob must conceal from Eve. Kerckhoff's principle states that a cryptosystem's security should only depend on the secrecy of the key, not on the confidentiality of any other part of the system [47].
This principle runs counter to the idea of security by obscurity, which prevailed by thousands of years: keeping as much of a system secret as possible. While this makes sense intuitively, reality has shown that the more people inspect a cryptosystem, the more secure it becomes.

Substitution and transposition ciphers

A substitution cipher changes the identity of a unit of cleartext, while a transposition cipher changes the position of a unit of cleartext. A unit can be a single character, a group of characters, or in the case of most modern ciphers, a single bit or a group of bits. Substitution ciphers that operate on units consisting of multiple bits or characters are called polygraphic. Otherwise, they are called simple substitution ciphers.

The class of substitution ciphers contains two branches: monoalphabetic and polyalphabetic ciphers. The former uses the same substitution unit for each occurrence of a plaintext unit, while the latter uses more than one possible substitution unit to complicate statistic-based attacks.

Monoalphabetic example: Caesar cipher

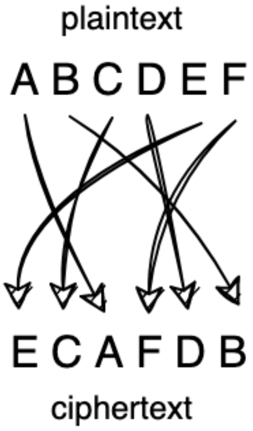


Figure 4: a **transposition scheme** changes the position of a unit of plaintext

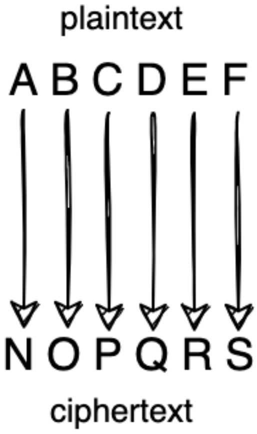


Figure 5: a **substitution scheme** (like ROT13) changes the identity of a unit of plaintext

The caesar cipher, also known as the shift cipher, is a simple, monoalphabetic substitution cipher first used in ancient Rome. Gaius Julius Ceasar is its namesake as, according to Suetonius, he used it to secure his military correspondence:

> "There are also letters of his to Cicero, as well as to his intimates on private affairs, and in the latter, if he had anything confidential to say, he wrote it in cipher, that is, by so changing the order of the letters of the alphabet, that not a word could be made out. If anyone wishes to decipher these, and get at their meaning, he must substitute the fourth letter of the alphabet, namely D, for A, and so with the others."
> — *Suetonius, Life of Julius Caesar 56 BCE.* [104, paragraph 56]

Using modular arithmetic, we can describe the encryption algorithm with:

$$e(x,k) \equiv x + k \; mod \; 26$$

and the decryption algorithm with:

$$d(y,k) \equiv y - k \; mod \; 26$$

While Caeser's decision to always use three as the key (shifting each letter by that amount to the right) would today be considered a severe lapse in security, the keyspace for this cipher is very limited, making it trivial to try each key no matter the choice. There are only 25 possible keys: the number of letters in the alphabet minus one (as a shift by 26 would result in the original plaintext):

$$\underbrace{0 \; mod \; 26}_{\text{no shift}} \equiv \underbrace{26 \; mod \; 26}_{\text{complete rotation}}$$

Breaking encryptions by an exhaustive search of the entire keyspace is called brute-forcing. This is the only attack that is always possible, no matter the encryption algorithm (though the size of the keyspace determines the feasibility of such an endeavor).

A special case of the Caesar cipher that is still sporadically used today is ROT13. As the name implies, the key (used for rotating the plaintext) is 13. This key evenly divides the alphabet into two parts, which results in the property $e() = d()$, making it straightforward to implement. While not providing useable cryptographic security, its simplicity and symmetricity have led to its use for obscuring text passages such as spoilers or solutions to puzzles [90].

Statistical analysis

A simple modification can expand the limited keyspace of the Caesar cipher: each plaintext character is substituted by another of the same alphabet while avoiding repeated mappings. This cipher has 26 possible characters for the first letter, 25 for the second, 24 for the third, etc., resulting in a keyspace of size $26!$ or more than 2^{88}. The size of this keyspace makes it less practical to brute force than the previous example.

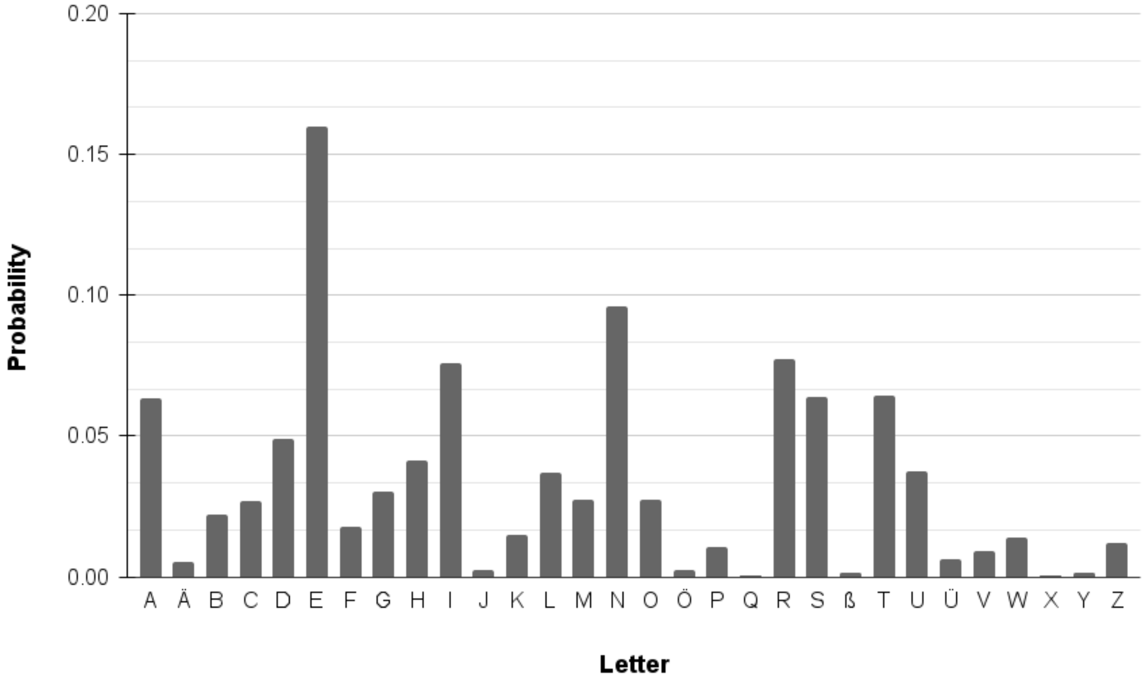


Figure 6: letter frequency (monograms) of the German language, knowing the frequency distribution of letters in the plaintext can help in breaking the ciphertext via frequency analysis attacks, despite a large keyspace (data for plot from [78])

However, the modified cipher is still insecure because it is vulnerable to statistical attacks, such as frequency analysis. This is because languages do not use each letter with equal frequency. Knowledge of the plaintext language and the frequency of each letter in that language enables an attacker to map likely ciphertext characters to plaintext characters. An attacker can improve this process by considering not only monograms (single characters) but also bigrams (groups of two) and trigrams (groups of three). See figure 6 on page 10.

Here is an example: the most common letter in German texts - `e` - has a frequency of about 16 percent. Therefore, if a ciphertext contains a similar percentage of `Q`s, it is likely that `e` → `Q`. An attacker can repeat this process for all monograms, bigrams, and trigrams until it produces the entire plaintext. See listing 1 on page 12 for an example attack.

Polyalphabetic Example: Vigenère Cipher

In 1553, Giovan Battista Bellaso developed a cipher on which Blaise de Vigenère would base his eponymous cipher. The Vigenère cipher is resistant to simple frequency analysis attacks. As a result, the cryptographic community thought it to be unbreakable for three centuries, earning it the moniker le chiffrage indéchiffrable (*the indecipherable cipher* in French). [45]

Bellaso's cipher switched the key for each plaintext letter, making it a polyalphabetic cipher. Like the Caesar cipher, the Vigenère cipher shifts each plaintext letter by a value; however, unlike it, the newer cipher does not keep this value constant. Instead, the key is a multicharacter string repeated when the plaintext is longer than the ciphertext.

For example: if the key is `HEY`, then the first letter is shifted by eight (the position of `H` in the alphabet), the second by five, and the third by 25. If the plaintext is longer than the key, then the fourth letter is again shifted by eight and so forth. For decryption, the receiver inverts the direction of the shift.

One-Time Pad

A special case of the Vigenère cipher is one where the key is at least as long as the plaintext, avoiding any repeated use of the key: the one-time pad (OTP).

From a mathematical standpoint, the OTP is perfectly secure, and the only possible attack on the scheme itself is exhausting the entire keyspace via a brute force attack. For this to be true, the following conditions have to be met:

- The key must be kept secret at all times.
- The key must be truly random.
- There can be no implementation errors.
- The key has to be at least as long as the ciphertext.
- No part of the key can ever be reused.

Modern versions of the OTP use bitwise instead of letter-based encryption. For encryption, the

```python
# this script performs a frequency analysis attack.
import string

# we assume knowledge of the ciphertext language (english):
prevailing_letter_alphabet = 'E' # most common letter in english.
cipher = 'MVYAFVULKLNYLLZHUKAOPYALLUTPUBALZUVYAOLHZAHUKIFUVYAO'
plain, prevailing_letter_cipher = '', '' # plaintext, most common cipher letter.
occurrences = 0 # store last high-score of occurrences, to find most common.

# find most common letter in ciphertext:
for char in string.ascii_uppercase:
    c = cipher.count(char) # count occurences for current letter.
    if c > occurrences: # found a more common occurence:
        occurrences = c # update high-score.
        prevailing_letter_cipher = char # update most common cipher letter.

# calculate frequency of most common cipher letter:
frequency = round(cipher.count(prevailing_letter_cipher)/len(cipher), 2)
# calculate offset between most common cipher and plaintext letter:
key = ord(prevailing_letter_alphabet) - ord(prevailing_letter_cipher)

for char in cipher: # iterate over cipher, one letter at a time:
    char = ord(char)+key # convert to integer (ASCII) from letter, add key.
    while char < 65: char += 26 # lower bound: "A" (mod 26)
    while char > 90: char -= 26 # upper bound: "Z" (mod 26)
    plain += chr(int(char)) # convert back to character, append to plaintext.

print(
f"""{prevailing_letter_cipher} is most common with a frequency of {frequency}
{prevailing_letter_cipher} shifted to {prevailing_letter_alphabet} by {key}
---\n{cipher}\n{plain.lower()}""")
```

```
L is most common with a frequency of 0.15
L shifted to E by -7
---
MVYAFVULKLNYLLZHUKAOPYALLUTPUBALZUVYAOLHZAHUKIFUVYAO
fortyonedegreesandthirteenminutesnortheastandbynorth
```

Listing 1: **the simplicity of a frequency analysis attack** on a Caesar cipher shows the **importance of choosing a good cipher** (and a non-repeating key when incorporated into more advanced attacks) (Python) [58]

key and the plaintext are acted on with exclusive OR (XOR)/($\oplus$) operator bit by bit: $y_i = x_i \; xor \; k_i$ where i is the position in the text. In modular arithmetic, this is equivalent to:

$$y_i \equiv x_i + k_i \; mod \; 2$$

x_1	x_2	$x_1 \oplus x_2$
0	0	0
0	1	1
1	0	1
1	1	0

Table 1: XOR truth table, x_1 and x_2 are the input bits, the last column is the corresponding result from the operation

Since the xor (exclusive-or) function is its own inverse function, the encryption and decryption functions are the same.

Kasiski Examination

The repeated use of the same key in the regular Vigenère cipher is what allowed Friedrich Kasiski to formulate a method [46] to break the Vigenère cipher, the Kasiski examination:

The first step is finding the length of the key used for the encryption. This is achieved by looking in the ciphertext for bigraphs and trigraphs with multiple occurrences and recording the distance between them. This set of distances is integer factored. The most common result is likely to be the key length (n) because this indicates that the same plaintext bigraphs and trigraphs were encrypted with the same part of the key.

The second step is subdividing the ciphertext into n new ciphertexts. The first group consists of all letters encrypted by the first letter of the keyword, the second group of all those for the second letter of the keyword continued up to the nth group. So, for example, if a key length of $n = 3$ is identified, then the first letter of the ciphertext would be placed in group one. The same goes for the fourth, seventh, eleventh letter, et cetera:

$$groupNumber = positionInCiphertext \; mod \; keyLength$$

The final step is performing the same statistical attack described in chapter Statistical analysis separately on each of the groups and merging the decoded groups back together, resulting in the plaintext. The relative simplicity of this attack shows the importance of not reusing keys.

2.1.4 Asymmetric Cryptography

Asymmetric Cryptography, also known as public key (PK) cryptography, departs from using the same key for encryption and decryption. Instead, it uses a private key for decryption, and a mathematically derived public key for encryption, resulting in an asymmetric process. Each communication party has its own set of keys. This provides several benefits over symmetric cryptography:

- It solves the key distribution problem over insecure channels because the private key stays with the owner.
- It rectifies the issue of the high number of different keys required: one for each unique pair of communication partners. For symmetric cryptography this is $(n * (n - 1))/2$ where n is the number of communication partners. It is only $n * 2$ for asymmetric cryptography, as each user can reuse their key pair.
- It allows for authentication of the communication partners and provides nonrepudiation, the proof of a message's authorship.

Public-key cryptography is generally slower than symmetric cryptography. However, developers can mitigate this disadvantage by implementing a hybrid scheme that uses asymmetric cryptography to exchange a key, followed by a switch to symmetric cryptography to employ that key.

Trapdoor Functions

Asymmetric cryptography systems rely on trapdoor functions, which are easy to compute in one but difficult in the opposite direction unless one knows a secret (the eponymous trapdoor).

There are three commonly used trapdoor functions for asymmetric cryptography:

- the multiplication of prime numbers,
- the discrete logarithm problem (DLP) in finite fields,
- and the DLP in elliptic curves.

The simplest example of a trapdoor function is the multiplication of two large prime numbers. $n = p * q$ where p and q are prime. This is a simple multiplication which is a trivial task. Computing the inverse (prime factoring n to obtain p and q) is a much more computationally intensive task. If either p or q is known, the problem becomes trivial again: a simple division of n by p or q results in the other prime number.

There is currently no known classical algorithm that can factor primes in polynomial time. The most efficient (currently known) classical algorithm is the general number field sieve (GNFS).

[38] A faster algorithm could exist. Schemes based on the difficulty of prime factorizations are thus not provably safe. A key difference between symmetric and asymmetric cryptography is that the former does not rely on the mathematical complexity of trapdoor functions to provide security.

Diffie-Hellman Key Exchange and the Discrete Logarithm Problem

The first asymmetric cryptography system was published in 1976 by Whitfield Diffie and Martin Hellman, who based it on prior work by Ralph Merkle. "New Directions in Cryptography" highlighted the problems of distributing pre-shared keys and authenticating communication partners [14]. One of the therein proposed solutions for establishing a shared secret came to be known as the Diffie-Hellman key exchange (DHKE).

The original implementation described in the paper uses the DLP in finite fields as a trapdoor function.

The scheme requires several parameters:

- p, a large public prime,
- α, a public integer,
- $\{a, A\}$ Alice's private/public key,
- $\{b, B\}$ Bob's private/public key,

The private keys are a random number from the set $a, b \in \{2, 3, ..., p-2\}$. A is computed by: $A \equiv \alpha^a \; mod \; p$, and B is calculated analogously.

Following their generation, Alice and Bob exchange public keys over the insecure channel. They can now compute the shared secret (k) by raising the other's public key to the power of their own private key in modulus p: $k_{AB} = A^b \; mod \; p$ and $k_{BA} = B^a \; mod \; p$. As both keys are the same, the key exchange is complete: $k_{AB} = k_{BA} = k$.

The proof for the equivalence of the keys is the following:

Alice computes:

$$B^a \equiv (\alpha^b)^a \equiv \alpha^{ba} \; mod \; p$$

Bob computes:

$$A^b \equiv (\alpha^a)^b \equiv \alpha^{ab} \; mod \; p$$

From which follows that: $\alpha^{ab} \equiv \alpha^{ba} \; mod \, p$ and thus $k_{AB} = k_{BA}$ due to the commutative property.

The security stems from the difficulty of computing the DLP modulo a prime number (assuming the involved numbers are large). A logarithm is the inverse function of the exponentiation function. A *discrete* logarithm is one that is defined for a cyclic group. The definition of a *group* is a set of elements and an associative, reversible operator that combines two elements of the group and returns an element of the group, plus a neutral element that produces the same element. A group is cyclical if it has a finite number of elements that can be obtained by repeatedly applying the defined operator to the generator of the group. Figure 7 on page 7 shows the cyclical nature of the discrete logarithm contrasted with the continuous property of the regular logarithm. The discussed Diffie-Hellman implementation uses the cyclical group $\mathbb{Z}_p^*$:

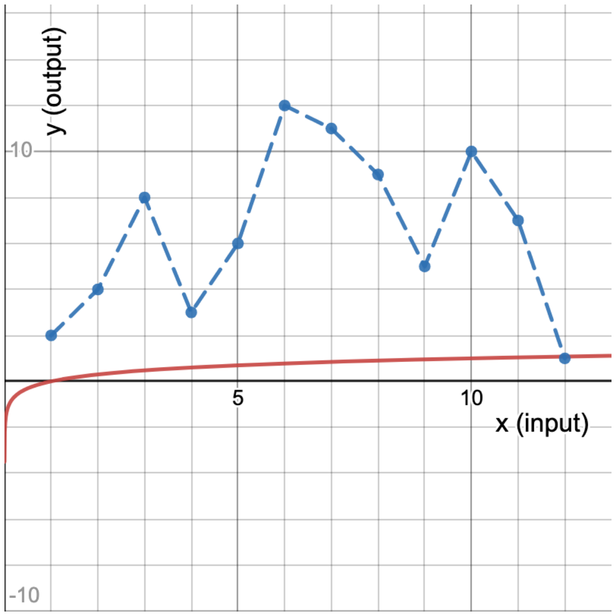


Figure 7: plot of ordinary logarithm (continuous-red) vs. discrete logarithm (dashed-blue) of $\mathbb{Z}_{13}^*$

- The group operation is multiplication.
- The neutral (or identity) element is one as $1 * n = n$
- The elements are positive integers modulo a prime (defined by the prime p).
- The cyclical and finite nature stems from the use of the modulo.

The DLP and the DHKE are not limited to groups of integers modulo a prime. A more modern variation uses points on an elliptic curve ($y^2 = x^3 + Ax + B$ where $4A^3 + 27B^2 \neq 0$) as elements. This elliptic-curve Diffie-Hellman (ECDH) is prominent in web browsers and messenger applications such as those based on the Signal protocol. [76]

Man-in-the-Middle

Diffie-Hellman (DH) does not facilitate participants' authentication, making it vulnerable to man-in-the-middle (MITM) attacks. As a result, an attacker (Mallory) with the power to alter traffic on the insecure channel can stay undetected while fully controlling the seemingly secured channel.

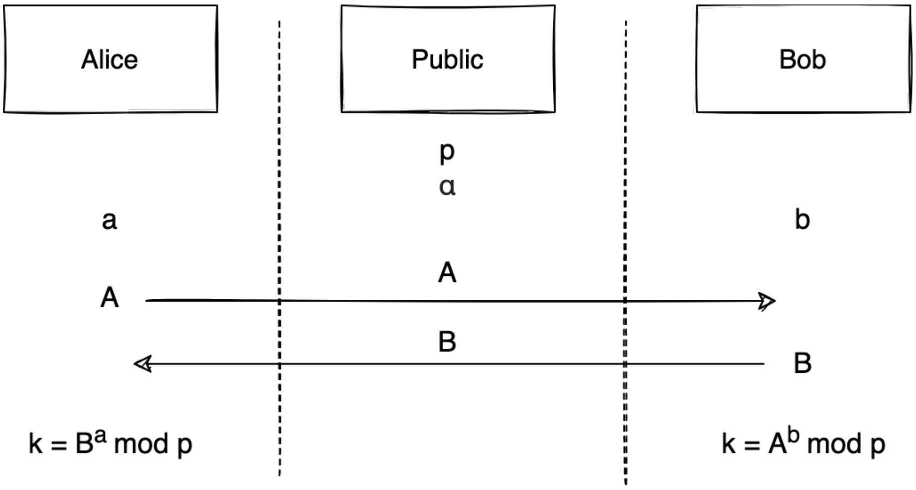


Figure 8: DHKE, Alice and Bob publicly agree on prime p and integer α, use it to calculate and exchange their public keys A and B and use the others public key to calculate a shared secret

Assume the following scenario: Alice and Bob have not established a shared secret yet. Unbeknownst to them, Mallory sits between them, resulting in a communication link between Alice and Mallory, plus one between Mallory and Bob. See figure 9 on page 9.

- Alice initiates the DHKE, which Mallory accepts on Bob's behalf to create a shared secret for Alice and Mallory.
- Mallory initiates a separate DHKE between herself and Bob to create another key for Bob and Mallory.
- Alice and Bob send messages to each other, which Mallory decrypts with one key and re-encrypts with another before passing it on. Optionally, Mallory can drop, modify or fabricate traffic.

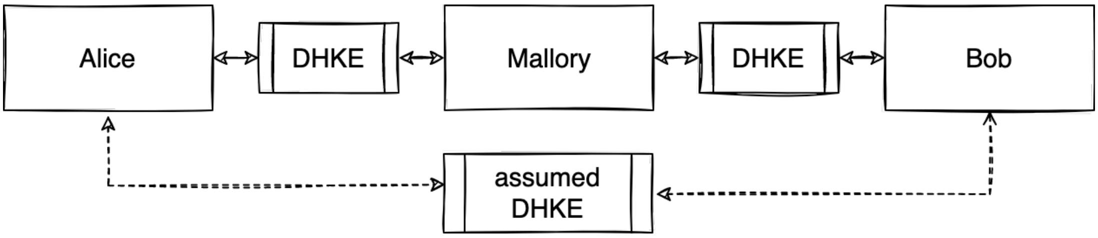


Figure 9: a MITM attack on a DHKE, Mallory lets Alice and Bob think they are communicating with each other by performing a DHKE with both

The only requirement for Mallory to stay undetected is always being present during communication as Bob's and Alice's keys are not the same, which would result in failed decryption should they ever communicate directly.

Rivest-Shamir-Adleman

In 1977 Ron Rivest, Adi Shamir, and Leonard Adleman published: Rivest–Shamir–Adleman (RSA), a PK cryptosystem based on the multiplication of prime numbers.[82]

Unlike DH, each communication partner independently computes private and public keys. Advantages are:

- Alice and Bob need not be simultaneously present on the channel.
- They can distribute key pairs to other communication partners.
- Keys enable authentication and signatures (assuming a public key infrastructure (PKI)).

However, static keys have a disadvantage: the lack of perfect forward secrecy (PFS). Systems with PFS (such as DHKE) use a new key for each communication session so that even if an attacker obtains the key, they are limited to decrypting a single session. RSA can be used for key transport (for a symmetric cipher), though this is not part of the scheme itself.

Another disadvantage of RSA specifically is the large key size required (2048 bits and more) to prevent factorization coupled with the slow algorithm, which presents a challenge for devices with limited computational power, such as embedded systems. In February of 2020, a team of scientists factored RSA-250 (250 decimal digits or 829 bits, a new factor record). The scientists used the open-source CADO-NFS software. This software can split the workload between multiple cores. [75]

As the cost of computation sinks, developers need to constantly increase key sizes, which places a growing burden on low-performance devices. This and the lack of PFS are the reasons why the transport layer security (TLS) protocol stopped using RSA for key exchanges, starting with version 1.3.[81]

The RSA cryptosystem describes four steps: key generation, key distribution, encryption, and decryption. There are several ways to accomplish key distribution, such as via a PKI or exchanging them in person. Encryption and decryption are straightforward: $y = x^e \ mod \ n$ and $x = y^d \ mod \ n$ where $\{e, n\}$ is the public and $\{d\}$ the private key. x and y are the binary values of the plain- and ciphertext.

Key generation is more involved than the other steps:

1. Choose two big primes: p and q.
2. Calculate the product n: $n = p * q$ with $p, q \geq 1024 \ bits$ so that $n \geq 2048 \ bits$ for RSA-2048.
3. Compute $\phi(n)$ with Euler's totient function: $\phi(n) = (p-1)*(q-1)$ (The original implementation used Euler's totient function. Carmichael's totient function can substitute it.)

4. Choose the public key e from the set $\{1, 2, ..., \phi(n) - 1\}$ so that e and $\phi(n)$ are relatively prime: $gcd(e, \phi(n)) = 1$. This assures that an inverse element exists.
5. The private key is the inverse element of the public key. Compute it so that $d * e \equiv 1 \; mod \; \phi(n)$ via the extended euclidian algorithm.

The difficulty of factoring very large numbers provides the security for RSA. If an adversary manages to factor n back into p and q, they can find the private key d by executing step three and iterating over steps four and five.

As previously mentioned, the current record for breaking RSA by factorization is 829 bits. The algorithm used and currently most efficient known method for large numbers is the GNFS. As a reference, it took about 2700 years of CPU time, employing an Intel Xeon Gold 6130 CPU with 2.1 GHz. [75]

2.2 Quantum Cryptanalysis

Quantum advantage or quantum supremacy is a quantum computer's ability to solve a problem that would take unfeasibly longer on a classical computer. [79] In August of 2022, a researcher disproved Google's 2019 claim of having bested this challenge. Google used their Sycamore quantum computer and a problem based on interferences, specifically designed to be difficult for classical computers. This algorithm has no known practical use besides demonstrating quantum advantage. [7] [8]

The first hint for a theoretical possibility of quantum supremacy in one case came years earlier, in 1992, from David Deutsch and Richard Jozsa. The Deutsch-Jozsa algorithm tests if a black box function is balanced or constant. Solving this problem on a quantum computer is faster than doing so on a classical one. [11]

This discovery led Daniel R. Simon to conceive a problem based on periodicity finding and collisions in the output of a black box function with a secret bit string. The quantum solution to this problem offers an exponential speedup. [96] Simon's algorithm, in turn, inspired Peter Shor to look for an algorithm that could solve similar problems based on periodicity finding and collisions. He managed to find two. [94]

2.2.1 Quantum Computers

Moore's law is an observation and prediction that the number of transistors on a dense integrated circuit (IC) doubles approximately every two years. However, as transistors become smaller with time, the insulating barrier approaches sizes that allow electrons to transfer through it via quantum tunneling. This effect starts to materialize at 5 to 7nm and will become more common and harder to mitigate as transistor designs shrink. [99]

While quantum effects might lead to the end of Moore's law and a reduced rate of expected performance increases for classical computers, they have also led to a new type of computation. Quantum computing represents a new paradigm for the method of calculating results.

Classical computers use registers of bits to store and logical gates to operate on data. The discussed XOR operation (table 1 on page 13) is performed by one such gate. Each bit can either be one, the presence of electrical current, or zero, the absence of the same. A classical register with a capacity of eight bits can be in exactly one of 2^8 (256) possible configurations at a specific point in time.
Quantum computers, on the other hand, use registers of qubits. A quantum register with eight qubits can be in all 2^8 configurations at the same time. This number grows exponentially: with each additional qubit, the number of states doubles. Quantum computers can calculate with each possible state simultaneously; however, each run only returns a single result. [70]

The qubit is the analog of the classical bit. As a generalization thereof, a qubit can be in either of the classical states: zero and one. These are known as the computational basis states, and the Dirac or Bra-Ket notation represents them by $|0\rangle$ (Ket-zero) and $|1\rangle$ (Ket-one). However, unlike bits, qubits can also be in combinations of these states, called superpositions:

$$|\psi\rangle = \alpha\,|0\rangle + \beta\,|1\rangle$$

α and β are complex numbers representing the wave equation's probability amplitudes. They can be used to calculate the probability that the state of the qubit (ψ) will be $|0\rangle$ ($P_{|0\rangle} = |\alpha|^2$) or $|1\rangle$ ($P_{|1\rangle} = |\beta|^2$) when measured. The Born rule demands that the sum of the probabilities of both states must be one:

$$|\alpha|^2 + |\beta|^2 = 1$$

A qubit's state is visualized by a vector in Hilbert space with:

$$|\psi\rangle = e^{i\gamma}(cos\frac{\theta}{2}|0\rangle + e^{i\varphi}sin\frac{\theta}{2}|1\rangle)$$

If one is only interested in the phase shift between the two angles $e^{i\gamma}$ can be 1. The formula can be simplified:

$$|\psi\rangle = cos\frac{\theta}{2}|0\rangle + e^{i\varphi}sin\frac{\theta}{2}|1\rangle$$

[70]

A Hilbert space is a vector space with an inner product operation ($\langle\;|\;\rangle$) which returns a scalar when applied to two vectors ($|\Phi\rangle, |\psi\rangle$). This operation uses the complex conjugate (*) that can be calculated by inverting the sign of the imaginary part of the complex number. The resulting scalar of the inner product represents how much the first vector lies along the second one:

$$\langle\Phi|\psi\rangle = (\Phi_1^*, \Phi_2^*, ..., \Phi_n^*) \begin{pmatrix} \psi_1 \\ \psi_2 \\ ... \\ \psi_n \end{pmatrix} = \Phi_1\psi_1 + \Phi_2\psi_2 + ... + \Phi_n\psi_n$$

The normalization condition for quantum systems requires $\langle\psi|\psi\rangle = 1$ to hold true because the probabilities have to add up to one (as each measurement has to result in one state).

See figure 10 on page 21 for a graphical depiction of the state of a qubit as a vector via a Bloch sphere. The infinite number of possible values for $|\psi\rangle$ map to the infinite number of points on the hull of the Bloch sphere. It is important to note that any measurement of a qubit collapses the probabilities into either $|0\rangle$ or $|1\rangle$, which is the reason for the limitation of only ever receiving a single result from a calculation. This makes it impossible to measure α or β via direct measurements in the computational basis.

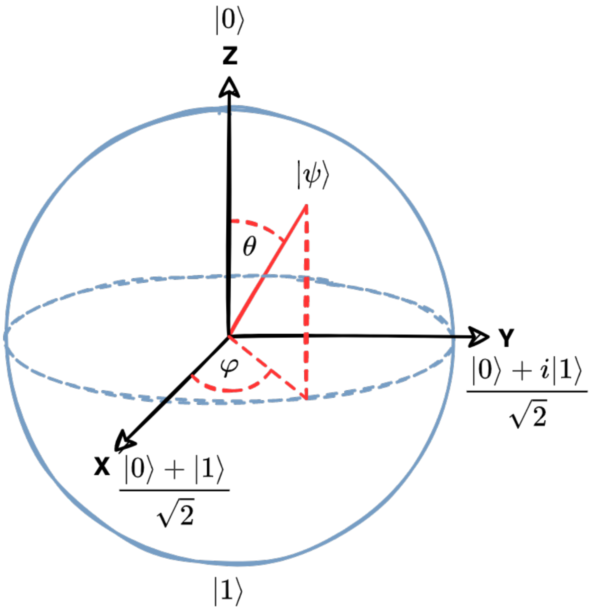

$|0\rangle$
Z
$|\psi\rangle$
θ
Y
$\frac{|0\rangle + i|1\rangle}{\sqrt{2}}$
φ
X
$\frac{|0\rangle + |1\rangle}{\sqrt{2}}$
$|1\rangle$

Figure 10: Bloch sphere representation of a qubit ($|\psi\rangle$) with $|\alpha|^2 > |\beta|^2$ (a higher probability of state $|0\rangle$ than state $|1\rangle$ and thus a qubit that is closer to state $|0\rangle$'s point on the sphere)

A quantum register with two qubits has four (2^2) computational basis states: $|00\rangle$, $|01\rangle$, $|10\rangle$, and $|11\rangle$. These can be in a superposition just like a single qubit: $\alpha|00\rangle + \beta|01\rangle + \gamma|10\rangle + \delta|11\rangle$ and as before the sum of the probabilities has to add up to one.

Quantum gates (used in the circuit-based quantum computation model) can manipulate the probability distributions of qubits without collapsing them. An analog of the classical XOR gate is the controlled NOT (CNOT) quantum gate that inverts the target-qubit if the control-qubit is $|1\rangle$. Compare table 1 on page 13 with table 2 on page 22. As vectors can visualize qubits, matrices operating on these vectors can act like gates. The CNOT matrix is:

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix}$$

Note that this is an identity matrix with the last two rows flipped. When using the matrix to operate on states, each basis state converts to a four-dimensional vector:

- $|00\rangle = [1\ 0\ 0\ 0]$
- $|01\rangle = [0\ 1\ 0\ 0]$
- $|10\rangle = [0\ 0\ 1\ 0]$
- $|11\rangle = [0\ 0\ 0\ 1]$

and then multiplied with the CNOT matrix. Applied to a superposition, this means:

$$\texttt{CNOT} \cdot |\psi\rangle \;=\; \texttt{CNOT} \cdot (\alpha|00\rangle + \beta|01\rangle + \gamma|10\rangle + \delta|11\rangle) \;=\; \alpha|00\rangle + \beta|01\rangle + \underbrace{\delta|10\rangle + \gamma|11\rangle}_{\substack{\text{flipped} \\ \text{probability amplitudes}}}$$

$control$	$target$	$control, target \oplus control$
$\lvert 0\rangle$	$\lvert 0\rangle$	$\lvert 0\rangle,\ \lvert 0\rangle$
$\lvert 0\rangle$	$\lvert 1\rangle$	$\lvert 0\rangle,\ \lvert 1\rangle$
$\lvert 1\rangle$	$\lvert 0\rangle$	$\lvert 1\rangle,\ \lvert 1\rangle$
$\lvert 1\rangle$	$\lvert 1\rangle$	$\lvert 1\rangle,\ \lvert 0\rangle$

Table 2: CNOT truth table

As is the case for classical gates, some quantum gates operate on a single qubit, for example, the Hadamard gate. This gate creates an equal superposition from basis states. See table 3 on page 23 for the truth table. The Hadamard matrix looks as follows:

$$H = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}$$

This matrix is its own inverse: $H^2 = I$ where I is the identity matrix. This makes chaining two Hadamard gates a no-operation (NOP).

$input$	$output$
$\|0\rangle$	$\frac{1}{\sqrt{2}}(\|0\rangle + \|1\rangle)$
$\|1\rangle$	$\frac{1}{\sqrt{2}}(\|0\rangle - \|1\rangle)$

Table 3: Hadamard truth table

Connecting a Hadamard gate to the input of the control-qubit of a CNOT gate creates a quantum circuit that entangles two qubits. From a mathematical viewpoint, entanglement is the unfactorable (into independent states) tensor product ($\otimes$) of two state vectors. If the measurement of one qubit affects a second qubit, they are entangled. In this example if one qubit is measured as $|1\rangle$ the second qubit will be too. The same holds true for $|0\rangle$. [35]

All operations that a quantum computer can perform can also be simulated on a classical computer (though the complexity class might differ). Listing 2 on page 24 shows the result of 10.000 runs of a simulation of this circuit. About fifty percent of the time, the result is $|00\rangle$, the other time $|11\rangle$. There is not a single $|01\rangle$ or $|10\rangle$, showing that the qubits are maximally entangled.

While the lack or presence of electrical current represents the state of bits in a classical register, a qubit can be any two-level quantum system. These are systems that can represent two states and their superposition. For example, possible systems are the spin of a particle, a photon's polarization, or an atom's energy levels, among others.

Quantum gates and their ability to manipulate probabilities instead of only ones and zeroes allow quantum computers to efficiently solve a subset of problems that classical computers can not efficiently solve. However, due to quantum computation's different nature (probabilistic, not deterministic), this requires specific algorithms.

2.2.2 Shor's Algorithm(s)

In 1994 Peter Shor published a paper [94] in which he described algorithms that, running on an at the time hypothetical quantum computer, could solve two real-world problems efficiently:

1. finding the prime factors of an integer
2. and the discrete logarithm problem

At the time of writing, no efficient classical algorithms for solving these problems were known. The discovery of these algorithms has special significance for the field of cryptology. The basis for the security of RSA is problem number one, as shown in section Rivest-Shamir-Adleman. More crucially, most other asymmetric cryptography relies on problem number two (which I covered in section Diffie-Hellman Key Exchange and the Discrete Logarithm Problem). A subset of

```python
import NumPy as np
from qiskit import QuantumCircuit
from qiskit.providers.aer import QasmSimulator
from qiskit.visualization import plot_histogram

circuit = QuantumCircuit(2, 2) # initialize 2 qubits and 2 classical bits.
circuit.h(0) # add a Hadamard gate to qubit 0.
circuit.cx(0, 1) # add CNOT gate with control qubit 0 and target qubit 1.
circuit.measure([0,1], [0,1]) # measure qubits, store in bits.

simulator = QasmSimulator() # create quantum simulator.
compiled_circuit = transpile(circuit, simulator)
runs = 10_000
job = simulator.run(compiled_circuit, shots=runs) # run simulations.
result = job.result()
counts = result.get_counts(circuit)

print(f'Measured 0,0: {counts["00"]} times and 1,1: {counts["11"]} times \
after {runs} simulations.')
circuit.draw()
```

```
Measured 0,0: 5061 times and 1,1: 4939 times after 10000 simulations.
```

```
     ┌───┐     ┌─┐   
q_0: ┤ H ├──■──┤M├───
     └───┘┌─┴─┐└╥┘┌─┐
q_1: ─────┤ X ├─╫─┤M├
          └───┘ ║ └╥┘
c: 2/═══════════╩══╩═
                0  1 
```

Listing 2: simulating the results of an entanglement circuit with IBM's Qiskit, a quantum computing software development kit (SDK) that can both simulate a quantum computer or run on quantum hardware (Python) [58]

affected schemes is: The Diffie-Hellman key exchange (both for finite fields and elliptic curves), the Digital Signature Algorithm (DSA), a part of the Digital Signature Standard (DSS), and El-Gamal encryption.

An attacker can break these in polynomial time if he or she has a quantum computer with a sufficient number of error free and stable qubits. This number is approximately:

$$2n \texttt{ qubits}$$

where n is the n-bit modulus for the DLP or the n-bit integer for the factorization problem. For elliptic curves, it is $10n$, but due to the smaller size of keys for the equivalent resistance against classical attacks, the number of qubits is about half for keys with the same strength. [24]

A single run of an algorithm will have a fixed probability of returning the correct solution, independent of the size of the problem, which simplifies scaling. However, the algorithm requires multiple runs to find the correct solution, which a classical computer checks in a post-processing step.

Logical Qubits

The estimations above are for logical qubits. Physical qubits suffer from three main problems that complicate the building of quantum computers:

- scalability,
- noise,
- and decoherence.

Scalability is an issue because each physical qubit requires a readout wire, a control wire, and connection wires to other qubits, which scale exponentially. Noise can be any form of energy that undesirably influences the state of a qubit such as cosmic rays or heat. Decoherence occurs when the environment of a qubit influences its state. Qubits can not be perfectly isolated because they have to be interacted with to be useful.
A logical qubit groups several entangled physical qubits together to create a more robust qubit that can hold its state for a longer time. This quantum error correction (QEC) was another discovery by Peter Shor [95]. In August of 2022, a team of researchers demonstrated for the first time instances where QEC implementations with logical qubits outperform the corresponding implementations using only physical qubits. [85]

2.2.3 Grover's Algorithm

Two years after Shor published his algorithms, Lov Grover followed up with his eponymous algorithm for searching unstructured lists of data. Grover's algorithm provides less of a speedup

than Shor's algorithms do. However, compared to the current classical algorithm for this task, it is still quadratically faster when running on a quantum computer with enough qubits: $O(n)$ versus Grover's $O(\sqrt{n})$ where n is the size of the domain to be searched. [29]
The relevance of this algorithm for quantum cryptanalysis stems from the fact that the keyspace of a symmetric cryptography algorithm is one such domain. As such, Grover's algorithm can brute-force symmetric encryptions. An exhaustive search of the keyspace would take $2^{\frac{n}{2}}$ steps, which provably is the optimal solution for quantum computers. [74]

2.3 Quantum Cryptography

The discovery of algorithms that threaten classical security guarantees (chief among them Shor's) and the looming presence of quantum supremacy have led to rising demand for alternative solutions.

Recent investments of major companies and several states have shown an increasing interest in quantum technologies. For example, between 2019 and 2021, China has already invested as much as $11 billion. In the same timeframe, the United States, Europe, and the U.K. have spent $3 billion, $5 billion, and $1.8 billion, respectively. [21]

These investments further increase the risk posed by quantum cryptanalysis. Quantum cryptography relies not on mathematical complexity for its security guarantees and thus presents one possible avenue for future-proof cryptography [80].

2.3.1 BB84 Quantum Key Distribution

The BB84 quantum key distribution (QKD) scheme by Charles Bennett and Gilles Brassard uses the uncertainty principle to distribute a key between two communication partners in a way that makes it possible to detect an eavesdropper with high probability. [5]

Section One-Time Pad includes five conditions required for a perfectly secure OTP implementation. If these conditions are adhered to, then a combination of both schemes yields a perfectly secure connection between two communication partners via an insecure classical and a quantum channel.

Heisenberg's uncertainty principle and the no-cloning theorem

Heisenberg's uncertainty principle states that it is impossible to know the exact values of all properties of a quantum system. For a particles position (x) and momentum (p) it is:

$$\Delta x \Delta p \geq \frac{h}{4\pi}$$

[33]
where Δx is the uncertainty of the position and Δp is the uncertainty of the momentum. h is Planck's constant ($6.62607015 * 10^{-34} JHz^{-1}$) that defines the relationship between the energy of a photon and the frequency.
From this follows that when measuring the position to a high degree of certainty (Δx decreases), the uncertainty of the momentum has to increase at some point (Δp increases) and vice versa. This correlation is true for all conjugate values of a quantum system.

Heisenberg's uncertainty principle can prove the no-cloning theorem. If perfectly cloning unknown states were possible, one could clone such a state multiple times and use the clones to perform as many measurements with arbitrary precision as desired. From this follows that if one could perfectly clone an unknown state, this would violate the uncertainty principle; thus, cloning unknown states is impossible.

The no-cloning theorem greatly complicates QECs when compared to classical error correction techniques (which have access to the precise values of states they are correcting). However, the consequences of the theorem also allow for the detection of eavesdroppers for BB84.

The BB84 Scheme

The basic order of events is the following:

1. Generate a shared key between Alice and Bob.
2. Check if anybody was eavesdropping.
3. If not, use the key; otherwise, discard it and try again.

The following detailed explanation of the scheme uses the polarization (the angle of the electromagnetic field in propagation direction) of single photons as states, as is the case in the original paper. However, implementations of BB84 with any two-level quantum system are possible. The suitability of polarization is described by Bennet and Brassard as follows:

> "[...] the state of a photon may be completely described as a linear combination of, for example, the two unit vectors r_1 = (1, 0) and r_2 = (0, 1), representing respectively horizontal and vertical polarization." [5, p. 8]

Figure 11 on page 28 shows three different linear polarizations of light waves.
Assuming the same scenario from section Man-in-the-Middle with the addition of a quantum channel to transmit photons, BB84 works as follows:

- Alice and Bob agree on how to encode bits for two polarization bases: $+$ and $\times$. See table 4 on page 29 and figure 12 on page 29.

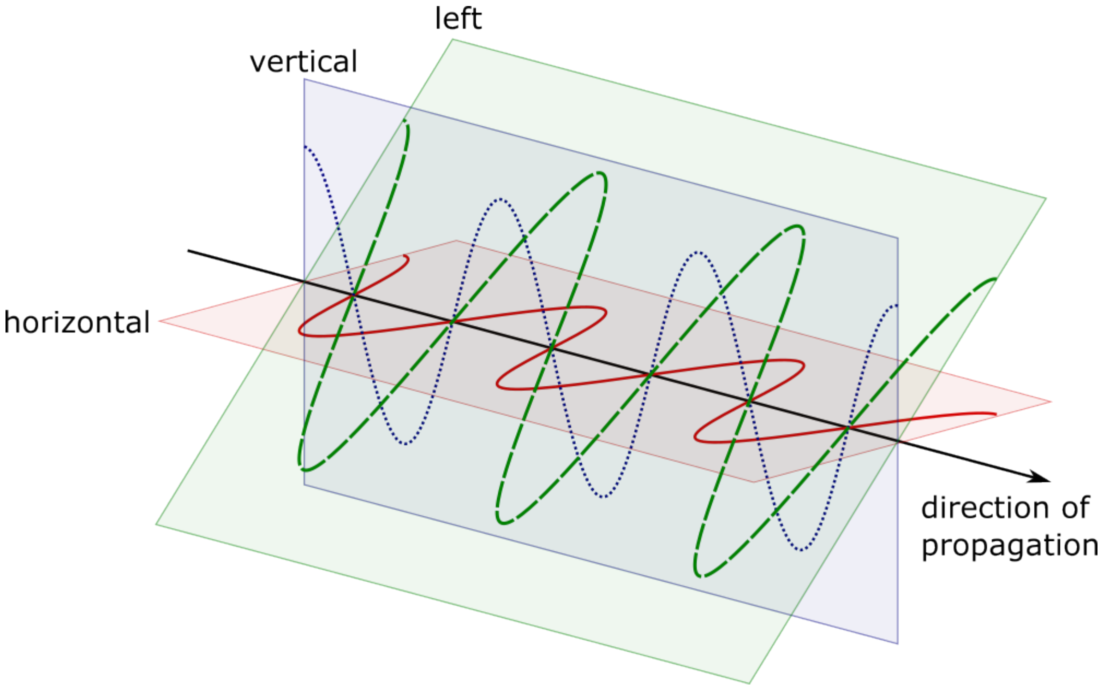


Figure 11: Horizontal (solid), vertical (dotted) and left (dashed) polarization of light corresponding to 0, -45 and 90 degrees respectively, based on [116]

- Alice can send a bit over the quantum channel via either: $|H\rangle/|V\rangle$ in the $+$-set or $|L\rangle/|R\rangle$ in the $\times$-set.
- Bob can measure each photon in the $+$ or $\times$ basis using a corresponding beam splitter.
- Measuring in the same basis returns the value sent (assuming neither transmission, nor measurement errors).
- Measuring in the wrong basis returns a random bit, which results in a 50% chance of it being wrong. For example: $\psi = \sqrt{\frac{1}{2}}|L\rangle\sqrt{\frac{1}{2}}|R\rangle$ when measuring $|H\rangle$ in $\times$ See figure 13 on page 13 for a Bloch sphere visualization.

The key distribution consists of the following steps:

1. Alice chooses a random bit and sends it via a random encoding.
    - She records both choices.
2. Bob chooses a random basis and measures the photon with it.
    - He records the basis choice and the value of the bit.
3. They repeat this process several times.
4. They publicly compare the basis choices over the insecure classical channel and throw away any record where the bases did not match up.
5. They publicly compare a subset of transmitted bits to test for eavesdroppers. Alice and Bob discard the key if an unexpectedly high number of bits mismatch despite matching basis choices.

The final step works because an active eavesdropper with the capability to intercept, measure and re-transmit photons has to guess the yet-to-be-published basis choices. Fifty percent of the time, the attacker will guess the wrong basis. In fifty percent of these cases, the superposition will collapse into a different polarization than what was sent by Alice. As measuring collapses the probabilities and perfectly cloning the photon before measuring it is prevented by the no-cloning theorem, this introduces an error rate of twenty-five percent. [26]

If no eavesdropper was detected, the remaining bits (set from correct basis choices minus subset used for eavesdropper-checking) are the shared secret.

polarization	bit encoding	polarization basis
$\|H\rangle = \|\,0°\rangle$	0	$HV, +$
$\|V\rangle = \|+90°\rangle$	1	$HV, +$
$\|L\rangle = \|-45°\rangle$	0	$LR, \times$
$\|R\rangle = \|+45°\rangle$	1	$LR, \times$

Table 4: BB84 bit-to-polarization encodings

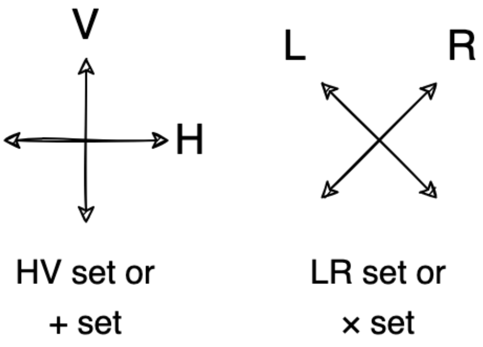


Figure 12: BB84's four polarizations grouped into the HV/$+$ and LR/$\times$ sets

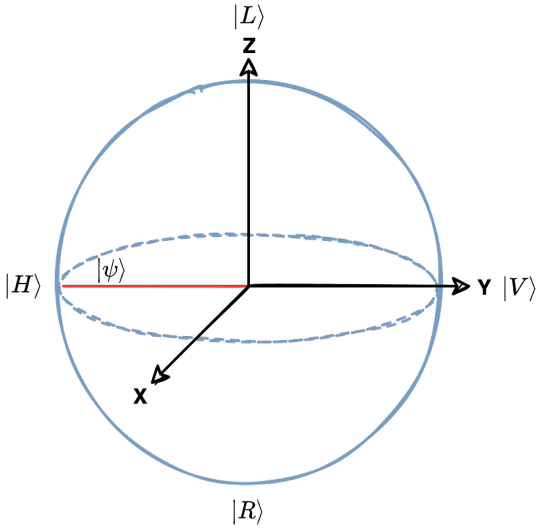


Figure 13: Bloch sphere of a horizontally polarized photon (equidistant to $|L\rangle$ and $|R\rangle$, resulting in an equal chance of measuring either in in the $\times$ basis)

2.4 Post-Quantum Cryptography

Post-quantum cryptography is an area of research concerned with algorithms that quantum computers can not break (or that are more resistant to them). Unlike quantum cryptography, these algorithms work with classical hardware. As quantum computing impacts symmetric cryptography much less, the main focus of post-quantum cryptography concentrates on asymmetric procedures avoiding approaches based on integer factorization and discrete logarithms. There are several promising candidates.

Hash-based

Hash functions are one-way functions that calculate a *hash value* or *digest* (z) from an input: $h(x) = z$. They have no inverse, do not use a key, and should be highly collision resistant ($h(x_1) \neq h(x_2)$ when $x_1 \neq x_2$). There are no hash-based procedures for key exchanges or encryption. However, signatures are a common usage scenario for hashes. Computer science does not limit hash functions to their use as cryptographic primitives; as such, they are widely used and well studied. [6].
The Internet Research Task Force (IRTF) has formalized two Request for Comments (RFCs) based on this technology: the eXtended Merkle Signature Scheme (XMSS) [36] and Leighton-Micali Hash-Based Signatures [57].

Code-based

Code-based systems are contingent on error correction codes. The first cryptosystem with this approach was McEliece's. [56] The candidacy status for a post-quantum cryptography standard by the National Institute of Standards and Technology (NIST) has led to renewed interest in this comparatively old category. [72]

Supersingular Isogeny

Supersingular isogeny works similar to the ECDH protocol. First, Alice and Bob start on the same node of a supersingular isogeny graph, a particular form of an elliptic curve. Then, both diverge, walking a random path along the graph for several steps. Next, they exchange their current positions, which are their public keys. Finally, they switch places on the graph and repeat their first walk (which is their private key), landing on a shared secret.

Multivariate

This approach uses the eponymous multivariate polynomials over a finite field. The public key is a set of multivariate polynomials $P(x_1, ..., x_n) = (p_1(x_1, ..., x_n), ..., p_m(x_1, ..., x_n))$ and the private key is the inverse P^{-1}. Hashes can be signed by applying P^{-1} and signatures verified with P.

The security builds on the fact that solving a set of multivariate equations of high degrees (with random coefficients) is a hard problem.

Lattice-based

A lattice (in the mathematical field of geometry) is a grid of points generated by scaling a pair of integer-vectors by another integer. This structure lends itself to creating easy-to-define problems that are hard to solve, such as the closest vector problem. Currently, no known quantum (or classical) algorithms can calculate the solutions to some of these problems in better than exponential time.

In July of 2022, after a process that took six years, NIST announced four of a total of eight planned, quantum-resistant cryptographic algorithms. [71]. The selections are three lattice-based and one hash-based algorithm (SPHINCS+) chosen by security, cost, algorithm, and implementation characteristics. NIST specified their use for general encryption (CRYSTALS-Kyber) and digital signatures (CRYSTALS-Dilithium, Falcon, and SPHINCS+). The yet-to-be-announced remaining four will use different technologies.
It should be noted that NIST has intentionally recommended a weak algorithm at the behest of the National Security Agency (NSA) in the past, as revealed in documents leaked by Edward Snowden. [69] However, all four chosen algorithms are open-source and were extensively discussed and reviewed in public.

The submitted reference implementations are: Falcon[1], Kyber[2], Dilithium[3], and SPHINCS+[4].

2.5 Optics

An implementation of the BB84 QKD scheme requires hardware that can emit single photons. Such a device is a *single photon source*. If more than one photon leaves the source for each bit, an attacker could split off and measure a subset without generating errors or violating the no-cloning theorem.
The following chapters will examine concepts from all branches of optics required for the construction of a single photon source.

[1] <https://falcon-sign.info/impl/falcon.h.html>
[2] <https://github.com/pq-crystals/kyber>
[3] <https://github.com/pq-crystals/dilithium>
[4] <https://github.com/sphincs/sphincsplus>

2.5.1 Classification of the Field of Optics

The field of optics, the study of light, can be categorized into three branches:

- geometrical optics or ray optics
- physical optics or wave optics
- and quantum optics.

Geometrical optics is an approximation concerned with how light rays propagate, reflect, and refract. Physical optics approaches light as an electromagnetic wave instead of as rays. Waves allow for the explanation of constructive and destructive interferences, diffraction, color, and polarization. Finally, quantum optics models light as quanta and examines how it interacts with atoms and molecules.

Particle-Wave Duality

The particle-wave duality is a concept of quantum mechanics that describes the apparent contradiction that single photons behave as if they were both particles and waves. For long stretches of history, the two possibilities seemed mutually exclusive. However, depending on the experiment performed, one or the other is suited to explain the results. [3]

2.5.2 The Photon - a Quantum of Light

In 1900 Max Planck formulated Planck's law, which describes the thermal radiation of a black body. It postulates that electromagnetic energy is emitted in discrete chunks proportional to the frequency, not continuously as previously believed. [77] These packets of energy were called *quanta*.

Five years later, Albert Einstein published his paper on the photoelectric effect [16] that extended the concept of discrete energy packets to light itself. At the time, Einstein termed such a packet a *light quanta*; today, this is known as a *photon*.

Properties

Photons are elemental particles that make up electromagnetic radiation. They have no mass but exert force (the electromagnetic force). [3]

The electromagnetic wave that a photon comprises has a specific *frequency* (ν) that defines how often the magnitudes of electric and magnetic fields sinusoidally oscillate in a given time span. The two fields sit perpendicular to each other. The fields are also perpendicular to the propagation direction of the wave, making it a *transverse wave*. [3]

Two waves with identical waveforms can be at different points in their cycle at a specific location, such as a sine and a cosine function that have the same frequency and amplitude, but the former starting at the origin and the latter $\frac{\pi}{2}$ radians later. This offset is called the *phase shift* (ϕ), and its unit is either degrees or radians. [102] A waveform repeats after 2π radians or 360 degrees. See figure 14 on page 33.

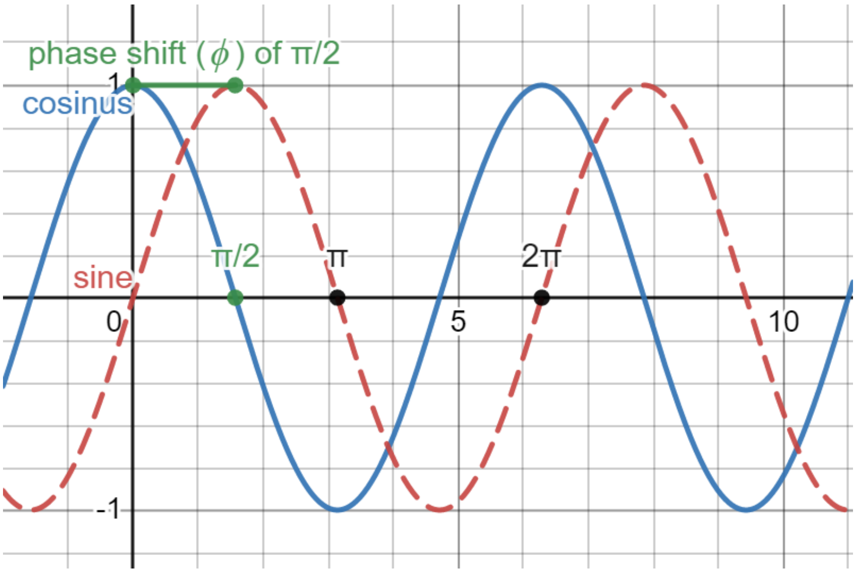


Figure 14: **phase shift** (ϕ, the horizontal offset in this plot) between a sine (dashed-red) and a cosine (solid-blue) wave (using the first visible crest of each wave as a point of reference)

The *polarization* of a transverse wave is the direction of oscillation. Circularly polarized light rotates the waves along the axis of propagation. Linearly polarized waves have no rotation. For electromagnetic waves, polarization is the direction of either the electric or the magnetic field vectors (because they perpendicularly relate to each other). [3] See figure 11 on page 28 for a visualization (of no particular field).

The energy of a photon is: $E = h\nu$ where ν is the frequency and h Planck's constant. [31]

The Electromagnetic Spectrum

The frequency also determines where a photon falls on the electromagnetic spectrum. Visible light is only a small spectrum section ranging approximately from 400nm (red) to 700nm (violet). [103] See figure 15 on page 34.

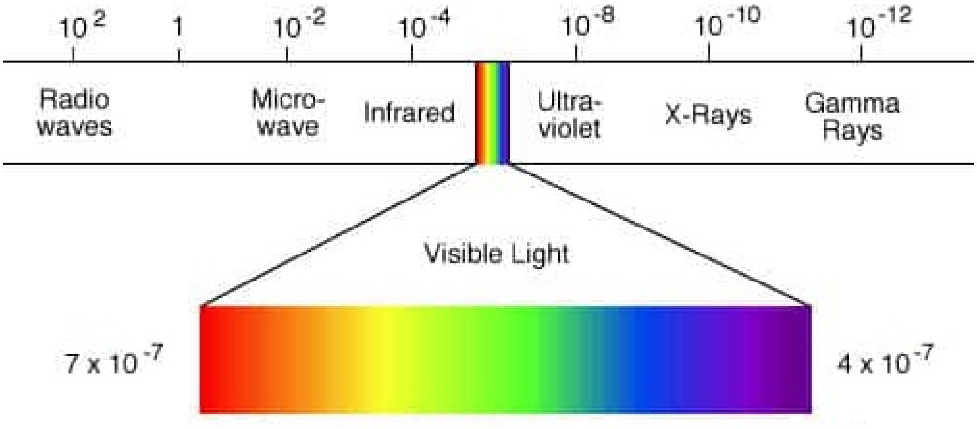


Figure 15: the electromagnetic spectrum in meters [119]

2.5.3 Laser as a Device and Process

The term *laser* stems from the acronym for microwave amplification by stimulated emission of radiation (maser). When the technology advanced to allow for the more complicated construction of devices that use visible light instead of microwave radiation, *laser* was termed. Who conceived of the idea first sparked a thirty-year-long legal battle involving Charles Townes and Gordon Gould, from which the former emerged with a Nobel prize and the latter with a number of patents. [105]. "Laser" describes both a process and a device.
The process can be broken down into two parts, the "stimulated emission of radiation" that gives rise to "light amplification". [102]

Absorption and Spontaneous Emission

For an electron in an atom to move from a low energy (E_1) *ground state* ($|1\rangle$) to a high energy (E_2) *exited state* ($|2\rangle$) it has to absorb energy. If the source of energy is a photon this is $h\nu_{12} = E_2 - E_1$ where ν_{12} is the frequency of the photon and h is Planck's constant. This process is called *absorption of light*. [13]

$|2\rangle$ is finite and after its *lifetime* (τ_2) the electron returns to $|1\rangle$, emitting a photon of the same frequency but with a random direction in the process ($\nu_{12} = \nu_{21}$). This process is called *spontaneous emission of light*. Such a system is called an *oscillator*. [13] See figure 16 on page 35.

Stimulated Emission

Albert Einstein realized that spontaneous emission could not be the only decay channel, leading him to discover a second process later termed *stimulated emission*. [17] If a radiation field is present, a second photon can trigger the transition from $|2\rangle$ to $|1\rangle$ before τ_2. This process

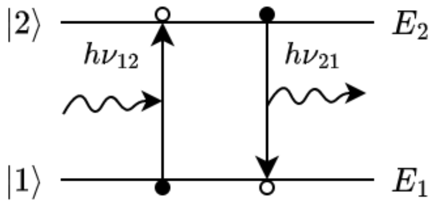


Figure 16: absorption followed by spontaneous emission in a 2-level system, based on [13]

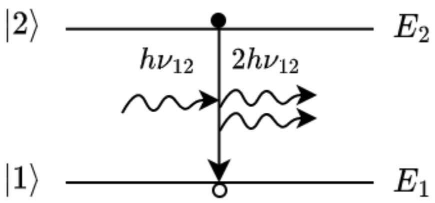


Figure 17: stimulated emission in a 2-level system, based on [13]

releases two photons (one from the state transition and the trigger photon). Both photons have the same energy, the same direction, and the same phase. [102] See figure 17 on page 35.

Rates of Optical Transition and Population Inversion

The amount of electrons in a particular state is called the population (N_1 for $|1\rangle$ and N_2 for $|2\rangle$). The *density of the radiation field* of a particular frequency is $\rho(h\nu)$. The *rates of optical transition* are:

$$\text{rate of absorption} = B_{12}N_1\rho(h\nu_{12})$$

$$\text{rate of spontaneous emission} = A_{21}N_2$$

$$\text{rate of stimulated emission} = B_{21}N_2\rho(h\nu_{12})$$

[13][102]

where B_{12}, A_{21} and B_{21} are the *rate constants* (or Einstein coefficients) for absorption, spontaneous emission and stimulated emission. The *principle of detailed balance* states that at equilibrium, the rate of absorption is equal to the rate of emissions. [13]

The goal of a laser is amplification. For this, stimulated emission needs to be high, which requires decreasing spontaneous emission. The ratio $\frac{A_{21}}{B_{21}}$ needs to be minimized. This relationship is why amplification gets harder the higher ν (a part of the divisor) gets and why masers came before lasers. [20][89]

Additionally, the ratio of the stimulated emission rate to the absorption rate requires maximization. This ratio can be reduced to $\frac{N_2}{N_1}$ when $B_{12} = B_{21}$. Laser action happens when stimulated emission dominates:

$$\frac{N_2}{N_1} > 1$$

[13]

This condition is called *population inversion*, which results in positive *gain*. However, this is not possible for two-level systems, which are limited to $\frac{N_2}{N_1} < \frac{1}{2}$. [13][103]

Higher Level Systems

Systems of a higher level than two are required to achieve population inversion. These introduce one to several *metastable levels* between $|1\rangle$ and $|2\rangle$, that have a higher τ than τ_2 and the ability to emit light. In these systems, the transition from a metastable level to $|1\rangle$ (or a lower metastable level) emits light. $|2\rangle$ is only used as a stepping stone to increase $N_{metastable}$. Adding energy to the system to transition electrons from $|1\rangle$ to $|2\rangle$ is called *pumping*. [20][102]

When $N_{metastable} > N_1$ population inversion occurs. After $\tau_{metastable}$ a photon with $h\nu_{metastable\ 1}$ energy will be emitted by spontaneous emission. This process triggers a chain reaction of stimulated emissions of this frequency: a laser beam. [102]

Three-level systems can work as pulsed lasers, such as the very first, ruby-based one. Four-level systems allow for continuous lasing as is possible with the typical Helium-Neon (HeNe) laser. The latter system requires less pumping and is more efficient since the lower-energy metastable level is empty at the start, resulting in an immediate population inversion between it and the higher-energy metastable level. [102][32]

Lasers have several properties that lend themselves to unique applications (ranging from spectroscopy over military and medical applications to astronomy) [13][1]:

- They emit **monochromatic radiation**, that is, light with a very narrow *linewidth* (the width of the optical spectrum).
- The produced light is **coherent** (all photons have the same constant phase relationship).
- Laser beams are **collimated** (the rays are parallel).
- They have a high potential of **power** (constant or peak).
- Emitted light is highly **directional** (so diffusion is minimal). [102][13]

2.5.4 Optical Components

Lenses

Optical *lenses* are constructed from transparent materials and focus or disperse light by refraction. *Refraction* is a deviation of the path light takes when it travels through materials with different densities. The cause is a change in the velocity of the light, which occurs at the boundary between materials. The angle of refraction depends on the densities of materials involved in the optical path and the frequency of light (which explains chromatic aberrations, one type of colored fringes in photographs). [3]

Positive lenses are thicker at the center and converge light rays, while negative lenses are thinner and disperse them. The surfaces of a lens can be plane, concave, convex, or an optically equivalent form. [3]

A positive lens's *focal point* (F) is where the light rays entering the lens parallel to the optical axis converge. The *focal length* (f) of a thin lens is the distance from the center of the lens to the focal point. The shorter the focal length, the stronger light is bent. [3] The lensmaker's equation allows the calculation of the focal length:

$$\frac{1}{f} = (n_l - n_m)\left(\frac{1}{R_1}\frac{1}{R_2}\right)$$

where R_1 and R_2 are the radii of the lens surfaces, while n_l and n_m are the refractive indices of the lens and the medium in which it resides. For a biconvex lens, one radius is positive and one negative to mark the contrasting orientations of the surfaces. [102] See figure 18 on page 38.

Mirrors

Mirrors change the optical path by reflecting light. *Reflectance* r is the property of a material that describes how much of the incident light (P_{inc}) reflects off of the surface. The remaining power is absorbed by or transmitted through the material (described by *absorptance* a and *transmittance* t respectively) [102][3]:

$$r = \frac{P_{refl}}{P_{inc}} \qquad t = \frac{P_{trans}}{P_{inc}} \qquad a = \frac{P_{abs}}{P_{inc}}$$

One parameter that influences reflectance is the *angle of incidence*, the angle between an incident ray of light and the surface normal. When the angle of incidence is zero, the *angle of reflectance* will be too: the light ray returns on the same path. [13]

Assuming incidental rays parallel to the optical axis, a formula similar to the lensmaker's equation (but with one surface and no refraction), allows for calculating the focal length of a concave mirror: $\frac{1}{f} = \frac{2}{R}$ or the reciprocal $f = \frac{R}{2}$. See figure 19 on page 38. Flat mirrors focus at infinity (when $R = \infty$ then $f = \frac{\infty}{2} = \infty$). [102]

Polarizers

Polarizers are optical elements that interact with electromagnetic waves depending on their polarizations. [3] The following categories exist:

- linear polarizers
  - absorptive polarizers
  - beam-splitting polarizers

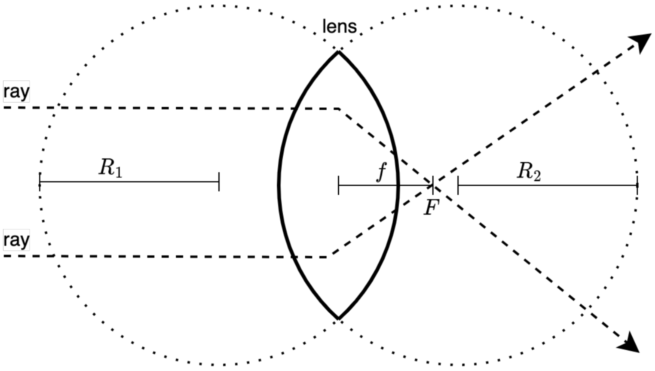


Figure 18: a biconvex lens focusing incidental rays, based on [3]

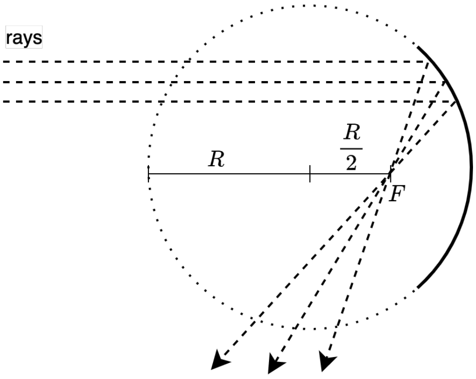


Figure 19: a concave mirror focusing incidental rays, based on [3]

- circular polarizers

Absorptive polarizers absorb all but one polarization, resulting in a linearly polarized output. Beam-splitting polarizers split one beam into two beams of orthogonal polarization states. Circular polarizers intromit either left- or righthand polarization (where the handedness specifies the rotation direction of the electromagnetic field vectors). [3]

Wave Plates

Wave plates rotate the polarization of linearly polarized light that passes through them —sending a horizontally polarized beam through a half-wave plate ($\lambda/2$-plate) results in a vertically polarized beam. [3]

The mechanism behind wave plates and one type of polarizer is birefringence, a property of some materials to have variable refractive indices based on the polarization of the incident waves. Minerals with this property were supposedly already utilized in medieval times by Vikings in the form of sunstones. Birefringence can aid in determining the sun's azimuth and thus navigation during partially cloudy skies, though the compass made this application scenario obsolete. [83]

2.5.5 Optical Cavities

Optical cavities, also known as optical resonators, are arrangements of optical components that capture light. In its simplest form, this configuration consists of as little as two highly reflective opposing mirrors and air as a carrier medium.

The pumped light can be introduced externally or, like when constructing lasers, from inside the cavity. Figure 20 on page 39 shows a diagram of a ruby laser. A flash tube serves as a *pump* around a rod of ruby crystal (the active medium). Two mirrors, one less reflective to let the laser beam escape, cap the rod. The cavity facilitates the final characteristic of the laser: amplification.

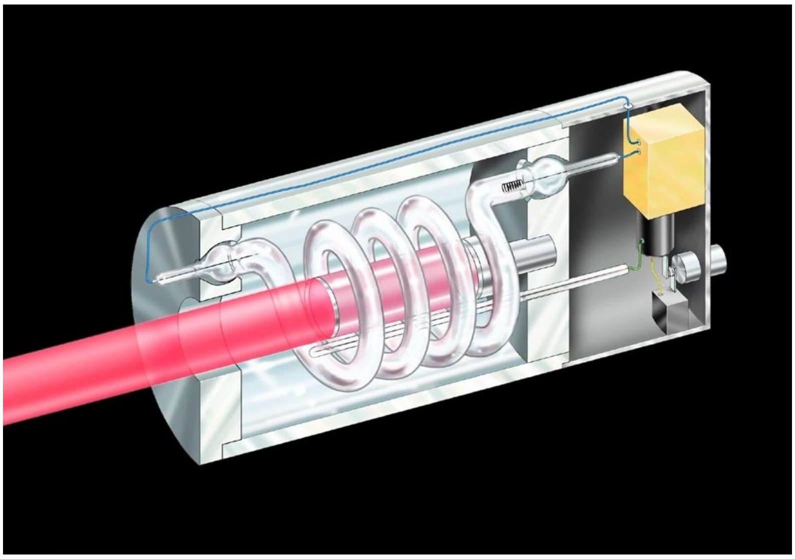


Figure 20: diagram of a ruby laser, the red rod is the ruby crystal (the emitter), the spiraling glass tube is a flash tube (the pump), the silver end-caps terminating the end of the crystal are the mirrors [98]

Classification of Optical Cavities

One classification for optical cavities is whether the light escapes after several reflections (unstable) or not (stable). Another classification is by the type and arrangement of mirrors used. Major configurations are [97]:

- **Plane-parallel** cavities consist of two flat mirrors whose normals point at each other.
- **Concentric** cavities use two concave mirrors with a shared focal point: $L = R$.
- **Confocal** cavities use two concave mirrors with each focal point positioned on the opposing mirror: $L = \frac{R}{2}$.

L is the *length of the cavity*, the distance between the mirror surfaces at the optical axis, and R the radius of a mirror, assuming both mirrors use the same one. As flat mirrors focus at infinity, no length for plane-parallel cavities is provided.

Fabry-Pérot-Cavity

The Fabry-Pérot-interferometer (FPI) was developed in 1897 by the french physicists Charles Fabry and Alfred Perot (who enjoyed adding an accent to his family name when publishing). It is an optical cavity with an external source of light that consists of two facing mirrors, which can be concave or flat. As the phrase "interferometer" indicates, the original purpose of the cavity was to gauge interferences; however, the same device enables several usage scenarios, including constructing single-photon sources. [52] For this purpose, the angle of incident is perpendicular to the mirror.

The pump is a laser (which is itself a cavity) that shines a monochromatic, collimated beam through the back of one mirror (with the amount depending on transmittance). This process introduces coherent electromagnetic waves to the cavity that largely reflect between the mirrors (the extent of which depends on the corresponding reflectance). When a wave reflects, it reverses direction. [12]

Constructive and Destructive Interferences

When waves that are in phase with each other meet, *constructive interference* leads to an enhancement of the intensity of light. The opposite (*destructive interference*) happens when two waves are perfectly out of phase. Summing the amplitudes of two sine waves (in or out of phase) and looking at the resulting plot is a good way to visualize this process in two dimensions. [3] See figure 21 and 22 on page 40. Interferences are not limited to the two extremes of $\pi\ rad$ phase shift and $0\ rad$ phase shift.

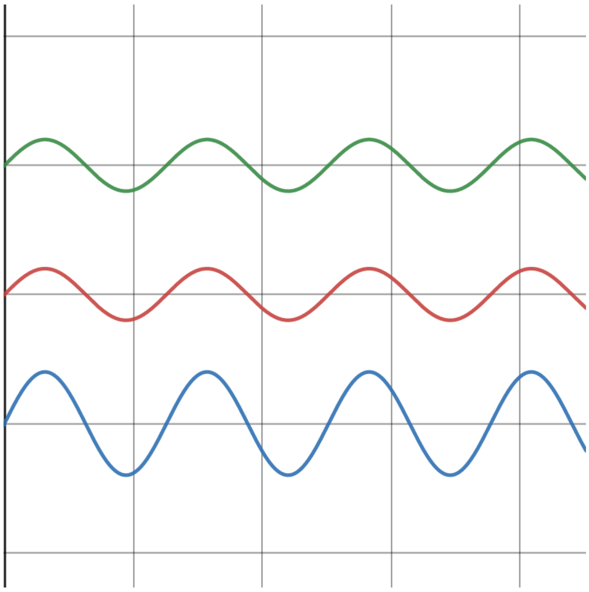


Figure 21: constructive interference (bottom) from two waves in phase, adapted from [3]

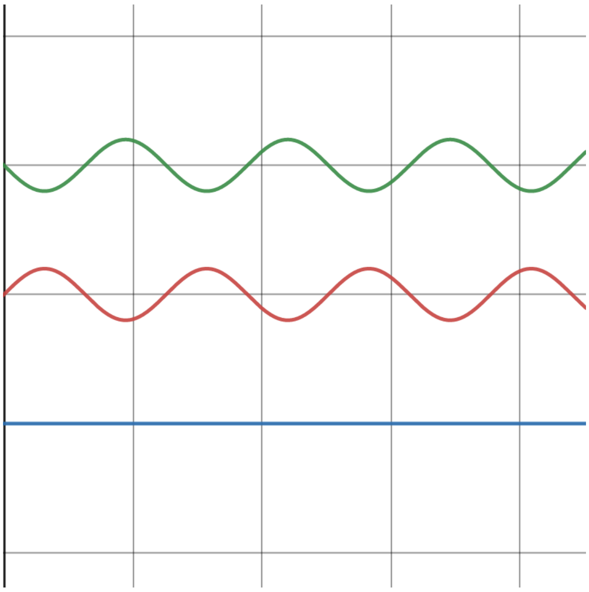


Figure 22: destructive interference (bottom) from two waves out of phase, adapted from [3]

Electromagnetic waves are three-dimensional, but the same concept applies. A more tangible example of three-dimensional transverse wave interferences is the effect that occurs when two pebbles drop into a still lake. The waves radiate outwards until they overlap and influence each other. Where two troughs meet, the water sinks even deeper; where two crests meet, it rises higher. [12]

Resonance and Modes

When waves in a cavity maximally constructively interfere, the cavity is in *resonance*. The *resonance condition* is:

$$L = \frac{\lambda}{2}m,\ m \in \mathbb{N}^{+}$$

where L is the cavity length and λ the wavelength. The *wavelength*, the spatial distance after which the wave pattern repeats, is related to the frequency by the speed of light: $\lambda = \frac{c}{\nu}$ where the speed of light is dependent on the medium through which it travels. m is a multiple of half the wavelength so that a *node* (any point with zero amplitude) sits at each turning point. This condition creates a *standing wave*: the amplitude at the anti-nodes (the locations of the minima/maxima) oscillates in place. Possible configurations that result in resonance are called *longitudinal modes* and m specifies the order of the mode. [102][31]

See figure 23 on page 41 for the second longitudinal mode of a plane-parallel cavity along with the corresponding nodes and anti-nodes. The wave's amplitudes oscillate between the solid and the dotted sines, forming a standing wave.

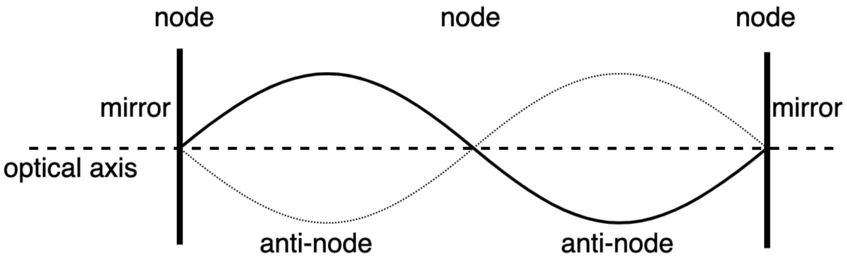


Figure 23: second order longitudinal mode ($L = \frac{\lambda}{2}2 = \lambda$) with nodes and anti-nodes, when a node (zero amplitude) sits at the each mirror (as depicted), the cavity is in resonance

Practical Construction Considerations

It is impossible to manufacture perfectly plane mirrors or to position them exactly in parallel. Minor imperfections will result in *loss*, the leakage and absorption of light from the cavity. Overcoming this loss is required to to achieve strong resonance. For this reason, curved mirrors in a concentric or confocal configuration focus the beam in the cavity and capture rays that travel off-axis slightly. The *beam waist* w is the location where the circumference of the beam is at its minimum. This point is located at $\frac{L}{2}$ for confocal and concentric cavities. Concentric cavities have a smaller beam circumference at this point than confocal cavities.

If the rays from the pump-laser fan out too much, a focusing lens can be positioned before the cavity. A further improved setup protects the laser from rays that reflect from the back side of the first cavity mirror. This setup adds a beamsplitter followed by a quarter-plate before the cavity: The linearly polarized beam from the laser passes undisturbed through the beamsplitter, is rotated 45 degrees by the quarter plate, (partially) reflects back towards the quarter plate which rotates the polarization another 45 degrees. The beamsplitter perpendicularly deflects the beam (with at this point 90 degrees polarization) away from the laser.

Two parameters can be tuned to achieve resonance: the wavelength λ or the cavity length L. Assuming no change of medium (and thus no change in the speed of light), frequency can substitute the wavelength.

3 Single-Photon Sources

Single-photon sources play an essential part in both quantum cryptography, where photons are physical information carriers, and quantum cryptanalysis, through their role in the construction of quantum gates and qubits, the fundamental components for some quantum computing models. Such an emitter has several other applications, which will impact cryptology through the increased research interest and resulting developments.

There are several requirements that an ideal single-photon source for quantum cryptology would fulfill:

- The *efficiency* of the single-photon source should be 100 percent, meaning that each electrical or optical control pulse should result in one photon leaving the emitter.
- The source should strictly exhibit *antibunching*, never emitting more than one photon simultaneously. For implementations of the BB84 QKD scheme, the source must emit a maximum of one photon to avoid photon number splitting attacks.
- Two produced photons should be *indistinguishable* from each other, a quantum optics term meaning having the same quantum state (frequency, polarization, et cetera).
- The source should be fast and small to facilitate scaling. [52]

No current model fulfills all requirements, though the last years have shown rapid improvements in every category.

The first single-photon source used *heralding*, predicting a photon based on an earlier photon in a chain of state transitions of a mercury atom [9]. The predictable chain of transitions allows the measurement of the first photon and indirect gain of information about the one that will follow.
Today, the most common technique for commercial purposes is using a laser beam and *attenuating* it with a filter so that the probability of releasing only a single photon is high. It is an inexpensive but fundamentally stochastic process that does not provide antibunching.
A third promising approach is coupling an atom (or a similar, small structure) to an optical microcavity so that a photon resulting from spontaneous emission of the atom is likely to match the mode of the cavity, allowing for some control over the directionality. [18]

3.1 Attenuation

Attenuation-based single photon sources use an optical filter with a high absorptance to reduce the intensity of transmitted light to such a level that for a single laser pulse, the likelihood is high of receiving one photon at the output. However, a significant disadvantage of this method is that there is a non-zero chance that no photon will pass through the filter (which is inconvenient) or that multiple photons will pass through for a single pulse (enabling attackers to split off and measure a subset when used for BB84). Likewise, the random distribution of these possibilities in time makes this an inherently non-deterministic source, unsuitable for unmodified BB84 implementations. [70]

3.1.1 Alternative QKD Schemes

There are QKD schemes that have less strict requirements than BB84 regarding the properties of a single photon source, such as those relying on decoy techniques or privacy amplification. These methods allow for the use of attenuated lasers for practical QKD applications.

The first *decoy technique* was proposed as a solution for using BB84 with a non-ideal photon source over surface-to-satellite free space. This channel introduces high losses, which empowers an eavesdropper. Eve can discard single-photon pulses (hiding this loss among the noise) and split multi-photon pulses. The solution is based on Alice sending *decoy pulses* among the regular transmission. The final step entails Alice and Bob comparing regular transmission losses with decoy pulses' losses to identify the presence of an eavesdropper [37]. Other decoy techniques require the ability to send and receive light with multiple levels of power to create *decoy states* [54].

Privacy amplification is the reduction of an initial key that might be partially known to an eavesdropper to a shorter key, about which the attacker has no usable amount of information. Applying a hash function to the initial key is one way to achieve this. [70] As mentioned in chapter 2.4, hash functions are a category of post-quantum cryptography.

Some protocols use other quantum phenomena (with different requirements). For example, Artur Ekert's E91 QKD uses pairs of quantum entangled photons to securely supply the same state to Alice and Bob. [19]

The existence of different protocols does not preclude the usefulness of deterministic single-photon sources for quantum cryptography, nor the applicability of the original BB84 scheme once reliable single-photon sources are available. Protocols that are less complicated present a smaller chance of implementation errors and offer a smaller attack surface. [84] Simplicity also has a financial impact: simple designs are cheaper to realize and scale than more elaborate schemes. Finally, long-distance QKD requires using quantum repeaters, and some quantum repeater protocols rely on single-photon sources [87].

3.2 Spontaneous Parametric Down Conversion and Heralding

Spontaneous parametric down conversion (SPDC), also known as parametric fluorescence, is a process that uses a high energy input photon to trigger the spontaneous emission of two lower energy output photons. This process can happen when light enters a nonlinear optical medium, often a crystal. Conservation laws stipulate that the energy sum of the output equals the input; the same is true for the momentum and polarization. SPDC can produce entangled pairs of photons. This technique combines with heralding to create a nondeterministic single-photon source.

Traditionally this method has suffered from very low efficiency as most light pulses transmit undisturbed through the nonlinear material and trigger light pulses have to be weak to avoid stimulated emission. In 2021 scientists published a process that relies on metasurfaces (nanoscale layers and patterns) with comparatively high efficiency (1400 $\frac{Hz}{mm^2}$ compared to 72 $\frac{Hz}{mm^2}$ for previous metasurfaces) at room temperature. [42] The major disadvantage of the process being stochastic in nature (due to the probability of absorption, not the deterministic spontaneous emissions) remains. Figure 24 on page 45 shows the conservation of energy ($h\nu_{12} = h\nu_{2m} + h\nu_{m1}$) of SPDC in a three-level system.

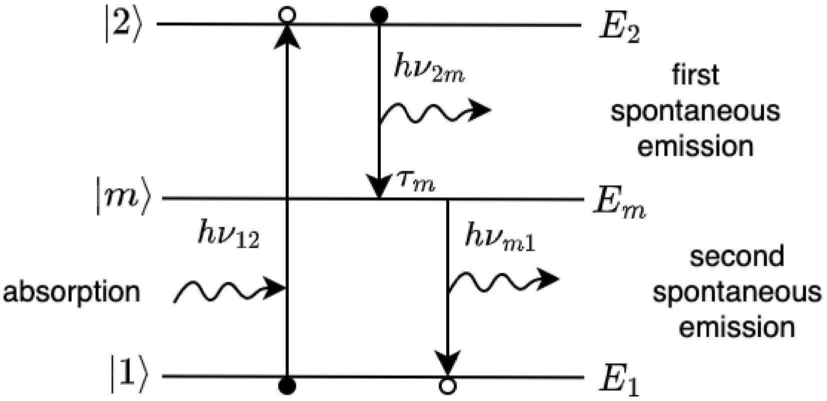


Figure 24: spontaneous parametric down conversion, the energy sum of the emissions equals the absorption's, based on [41]

Heralding relies on processes that emit photons in pairs, such as SPDC. The first photon is the *heralding photon*. Its detection predicts the second photon: when the first photon is measured, a second photon, the *heralded photon*, must exist. Heralding mitigates some disadvantages of probabilistic single-photon sources, though it also introduces additional efficiency disadvantages and components. [63] The technique allows the testing of the sensitivity of single-photon detectors by measuring both photons with a detector each and calculating the percentage of missed photons.

Some other disadvantages of SPDC are resolvable as well. Suppose τ_m (in figure 24 on page 45), the time between the two spontaneous emissions, is too short for the desired application scenario. In that case, the heralded photon can be redirected into a delay line to increase the time between the arrival of the first and the second photon at the cost of increasing losses. The issue of sometimes missing the heralding photon (due to losses in the system or imperfect single-photon detectors) can lead to unforeseen photons on the heralded channel. Blocking the heralded output and only opening it when a photon on the other channel is detected can mitigate this. [49].

3.3 Microcavity-Based

As opposed to attenuation- or SPDC-based techniques, microcavities allow for creating non-probabilistic single-photon sources. Deterministic methods bypass the problems introduced by the probability of emitting more than one photon per pulse, which limits applications and poses a security risk for BB84 implementations. Furthermore, these methods also do not require heralding and hence avoid its disadvantages.
Microcavity-based single-photon sources have other disadvantages, chief among them the prevalent requirement to operate at cryogenic temperatures. [26]

There a several possible designs though the basic principle, ignoring any losses, is as follows. The setup consists of an emitter placed in a small cavity. First, the emitter is excited by a triggering pulse, which might be radiant energy or an applied charge. The pulse does not couple to the optical cavity, whereas the photon created by spontaneous emission from the emitter does. Finally, after bouncing between the two reflective surfaces, the photon exits through the less reflective end of the cavity. The average duration a photon remains inside the cavity is called the lifetime and it depends both on the time a photon takes for one round-trip and how many of those happen on average. [102][52]

The use of spontaneous emission (which, as discussed, produces a photon with a random direction) instead of stimulated emission (which results in two photons with the original heading) allows the pumping axis to be orthogonal to the optical axis of the cavity. This positioning avoids coupling the pumping energy to the cavity while maintaining a chance for the photon to couple to it. [52] See figure 25 on page 47.

The name "microcavity" stems from the fact that its size is in the range of micrometers ($10^{-6}\ m$), sometimes even smaller. For reference, the wavelength of visible light, one band used for long-distance QKD, approximately starts at 400 and ends at about 700 nanometers ($10^{-9}\ m$). These parameters mean that microcavities have low mode orders, and compared to macro optical cavities, large reflective surfaces in relation to the cavity length. Some designs use one curved and one flat reflective surface, where a surface can be a mirror or other structure with reflective properties. [52]

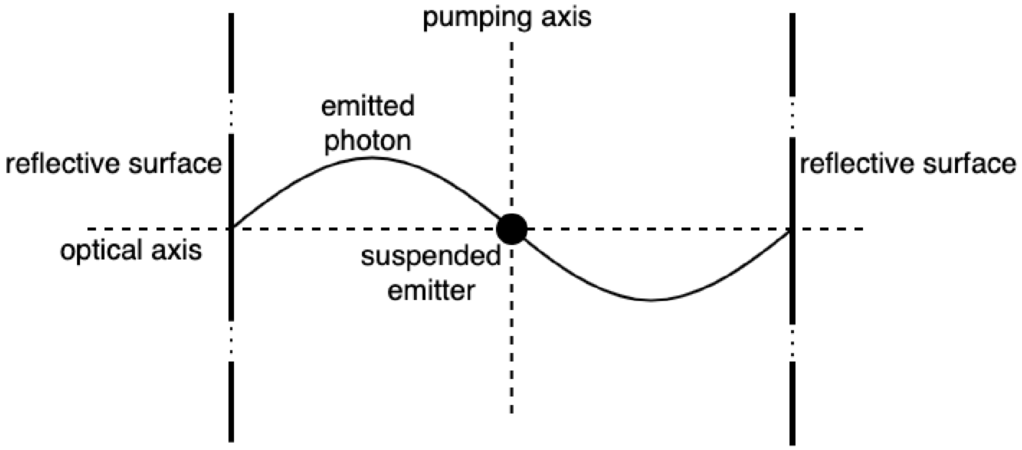


Figure 25: a microcavity with an emitter and orthogonal pumping axis, this allows exciting the emitter, without interfacing with the reflective surfaces, based on multiple graphics from [52]

3.3.1 Emitters: Atoms, Ions, Quantum Dots

Previous sections have not precisely specified the phrase "emitter"; this chapter remedies this by covering some standard options. However, this enumeration does not claim to be comprehensive.

The role of the emitter for a QKD scheme is to act as an interface between the classical control side that operates with bits and the quantum channel that uses photons to transport information securely. The requirements concerning an emitter for a single-photon source include:

- The emitter must consist of a two-level (or higher) quantum system with one radiant emission transition.
- The energy differential between that transition's higher and lower states must correspond to the desired wavelength.

  - The wavelength is specified by the required transmission characteristics of the emitted photons. For example, which frequencies does the atmosphere not readily absorb for satellite-to-ground free space? Alternatively, which frequencies provide high transmittance through fiber-optic cables?
  - The relation of length and wavelength constitute the minimum device size: $L \geq \frac{\lambda}{2}$.
- The emitter must be excitable by an optical or electrical triggering pulse but should avoid any other excitation.

Possible choices for emitters include single neutral atoms, single ions (positive with missing and negative with extra electrons), and quantum dots. [52]

Atoms and Ions

Atoms contain a nucleus (positively charged protons plus neutrons) and electrons. [31] The outdated Bohr model [51] of an atom consists of electrons circularly orbiting in shells around the nucleus. The discrete distances between the shells correspond to the discussed energy levels of the atom. A newer model, the based on the Schrödinger equation [91], which, in keeping with quantum mechanics, only allows for probabilities of the electron's position, producing a cloud of possible locations of varying size and density for the different energy levels. [31] Exciting an atom, be it via electromagnetic radiation, heat, or otherwise, increases the radius of that cloud, while photon emission reduces it.

Both atoms and ions require electromagnetic traps (contraptions that can hold particles) to confine them to the cavity after they have been cooled and dropped from above. The former requires an optical trap, which uses focused beams of light, and the latter a radio-frequency ion trap, which allows for more precise positioning of the trapped particle. Current experimental setups for both types use macrocavities. [18]

Quantum Dots

Quantum dots are tiny structures made of multiple molecules (groups of atoms joined by chemical bonds). Contrary to the previously discussed emitters, quantum dots do not require traps as their substrate for growing them holds them in place [108]. A 2010 review of cavity-based single-photon sources by Axel Kuhn and Daniel Ljunggren described the applicability of these structures succinctly:

> "Quantum dots are often considered as artificial atoms par excellence, as they usually possess several discrete energy levels for electron-hole pairs, with optical transitions between these levels comparable to electronic transitions in atoms." [52]

The makeup of a quantum dot is semiconductor particles, usually indium arsenide (InAs) with a structure that results in a photon with 900 to 950 nm wavelength when electrically excited.

InAs requires low temperatures for emission, though alternative materials for optical excitation at room temperatures exist. [93]

3.3.2 Heterostructure Semiconductor Photonics

A heterostructure is a combination of two or more different semiconductor materials. It is similar to a regular p-n junction in a diode where the materials are identical, but the doping differs. A p-i-n diode of different materials containing a quantum dot can create an electrically pumped single-photon source. [93] p-doped materials have missing electrons, n-doped extra electrons, and i-type (intrinsic) materials are undoped or equally doped. Pumping this setup by applying a charge between the p-doped and n-doped material produces a single photon.

The authors of a 2021 paper [108] used this technique combined with a Fabry-Perot microcavity to create a small single-photon source with very low dampening. From the bottom to the top of the vertical cavity, the structure is as follows:

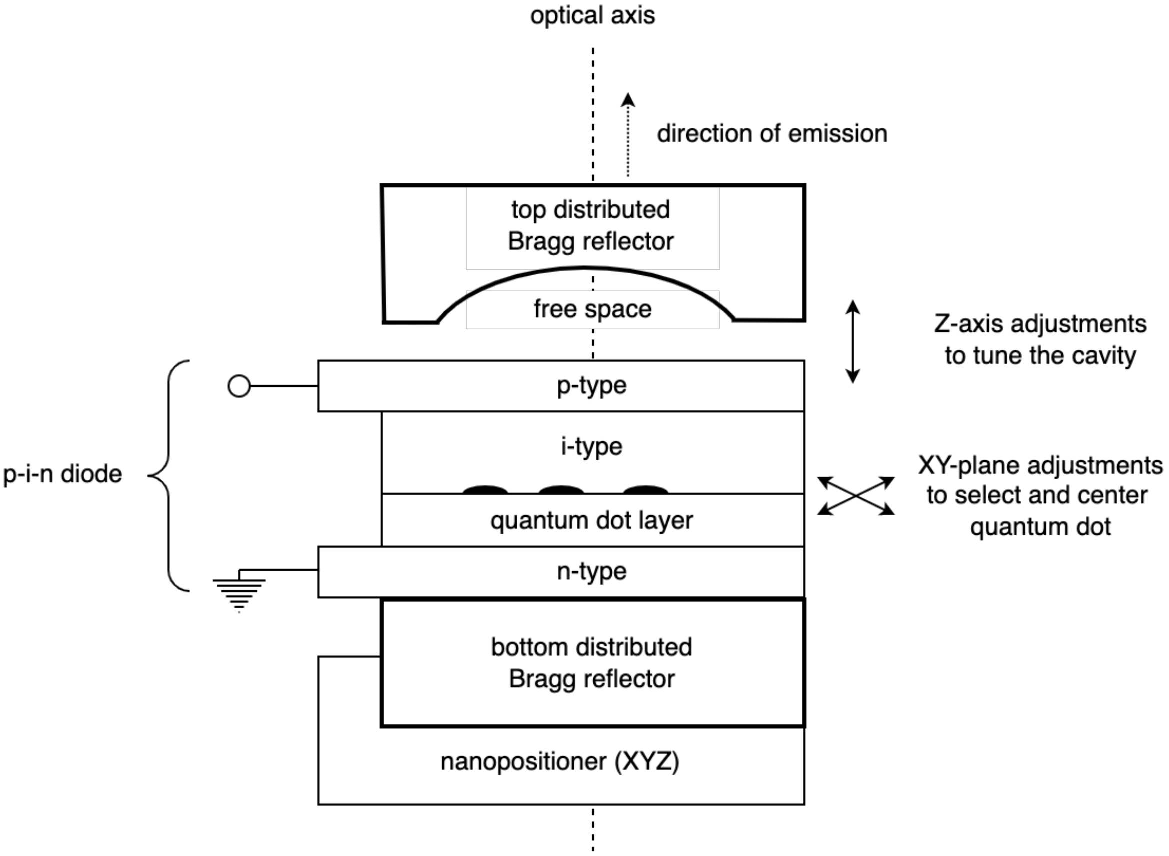
1. An n-doped material connects to a flat mirror at the bottom of the optical stack. A layer of InAs quantum dots is grown on top of the n-type layer.
2. Next, a layer of i-type material tracked by one of p-type follows the quantum dots.
3. Finally, the cavity ends with a concave dielectric mirror after a small space of about 1 micrometer. [108]

The concave top mirror has a fixed place in the assembly and a radius of 12 micrometers. The surface's reflectivity stems from a silica substrate coated with several alternating layers of silicon dioxide and tantalum pentoxide, forming a dielectric *distributed Bragg reflector*.
The adjustable bottom mirror consists of a semiconductor substrate (gallium arsenide) followed by alternating arsenide layers (gallium and aluminum). The layer of InAs quantum dots tops this distributed Bragg reflector. [107]

A nanopositioner moves the lower part of the cavity (consisting of the p-i-n diode with the embedded quantum dots and the flat mirror) in three axes. The process of growing the layer of quantum dots does not allow for the exact determination of their location. Thus, lateral XY-plane adjustments enable positioning one of the randomly grown quantum dots in line with the optical axis. Unlike atoms and ions, quantum dots are not indistinguishable, meaning their energy states and, consequently, the frequencies they emit can differ slightly. The lateral adjustments of the nanopositioner can select a frequency from the available quantum dots. Finally, movements along the Z-axis (the optical axis) bring the cavity into resonance. [108] For a depiction of the cavity stack see figure 26 on page 50.

The i-type layer reduces the current that flows between the p-type and the n-type layer. This setup assures that only a single quantum dot enters the excited state. Applying forward bias (meaning the p-type connects to the positive and the n-type to the negative terminal of the power source) to the p-i-n diode excites the quantum dot that is in line with the optical axis. A single photon effuses via spontaneous emission when the emitter moves from the high-energy state to the low-energy state.

The cavity can improve (in resonance) or hinder (out of resonance) the photon-matter interaction in regards to creating a single-photon source due to the *Purcell effect*. Andrew Shields describes this effect and the *Purcell factor* that quantifies the effect as follows:



optical axis
direction of emission
top distributed Bragg reflector
free space
Z-axis adjustments to tune the cavity
p-type
i-type
p-i-n diode
XY-plane adjustments to select and center quantum dot
quantum dot layer
n-type
bottom distributed Bragg reflector
nanopositioner (XYZ)

Figure 26: microcavity-based single-photon source using an InAs quantum dot layer in a p-i-n diode (this depiction is based on multiple diagrams from [107])

> "Purcell predicted enhanced spontaneous emission from a source in a cavity when its energy coincides with that of the cavity mode, due to the greater density of optical states to emit into. For an ideal cavity, in which the emitter is located at the maximum of the electric field with its dipole aligned with the local electric field, the enhancement in decay rate is given by $F_p = (3/4\pi^2)\ (\lambda/n)^3\ Q/V$, where Q is the quality factor, a measure of the time a photon is trapped in the cavity, and V is the effective mode volume." [93, p. 223]

As Shields mentions, the *quality factor (Q factor)* describes how high or low the losses of such a system are. A high Q factor means low dampening and vice versa. The Q factor is the temporal confinement of the light in the resonator (how long it stays inside). The *mode volume* is a measure of how spatially confined the light in a resonator is. A small mode volume means light is highly confined, as in a microcavity.

The probability of a single photon spontaneously emitting into the cavity (P_{SE}) can be calculated with:

$$P_{SE} = \frac{F_p}{F_p + 1}$$

This shows that the Purcell factor should be as high as possible to increase the chance of spontaneous emission. Increasing the Q-factor while decreasing the mode volume to the fullest extent accomplishes this.

One experimental setup with two equally reflective high-reflectivity surfaces (to minimize outcoupling) built by the authors of the 2021 paper mentioned above has a Q factor of $4.5 \pm 0.5 * 10^5$ at about 915 nm wavelength and a Purcell factor of 10. This means that the probability to emit a photon into the cavity is close to 91% ($P_{SE} = \frac{10}{11} \approx 0.91$). Replacing the top mirror with a slightly lower reflectivity surface allows photons to exit through the top while reducing the Q factor. The end-to-end efficiency of this single-photon source is 55% at a repetition rate of 76.3 MHz. This percentage is the probability of a photon exiting the optical fiber at the end of the device after a single trigger pulse and includes losses along the entire chain. [108]

4 Optical Cavity Simulator

Quantum computing and cryptology demand new skills from IT security professionals. However, the familiar, strictly deterministic computational models, bits, and classical gates are not enough to understand quantum cryptography, nor quantum algorithms like Shor's and Grover's that will impact the IT security field.

The authors of this optical cavity simulator set out to create a tool that could help with taking the first steps toward understanding the required concepts. The resulting simulator was a joint project of Nikolai Benedikt[1] and the author of this thesis[2]. The project is open-source, and the latest version's source code is always available from our GitHub repository[3], where issues, feedback, and pull requests are welcome. Additionally, I host a live version of the simulator on my homepage[4] that I update on significant changes to the codebase.

We have licensed the project under the GNU General Public License v3.0 (GPLv3). So, in a nutshell, any published derivative based on the code of this simulator needs to be open-sourced under the same license [22]. The choice of a viral (copyleft) license in the form of the GPLv3 is meant to ensure that any potential future improvements to the simulator codebase will always be available to as many users and developers as possible.

The structure of this chapter is as follows. First, it starts with a breakdown of the significant features of the optical cavity simulator that should get most users up and running quickly. Next follows an overview of the utilized technologies and the decisions behind their selections. The final three sections offer a deep dive into the user interface (UI) elements, the main formula behind the simulation, and the code of the simulator, including pointers on how to extend the application.

4.1 Features

This section broadly covers the major features of the optical cavity simulator. A more detailed breakdown of the UI elements including all variables and their visualizations follows in chapter 4.3 User Interface Elements of the Simulator.

[1] <https://github.com/nikobenedikt>

[2] <https://github.com/bmedicke>

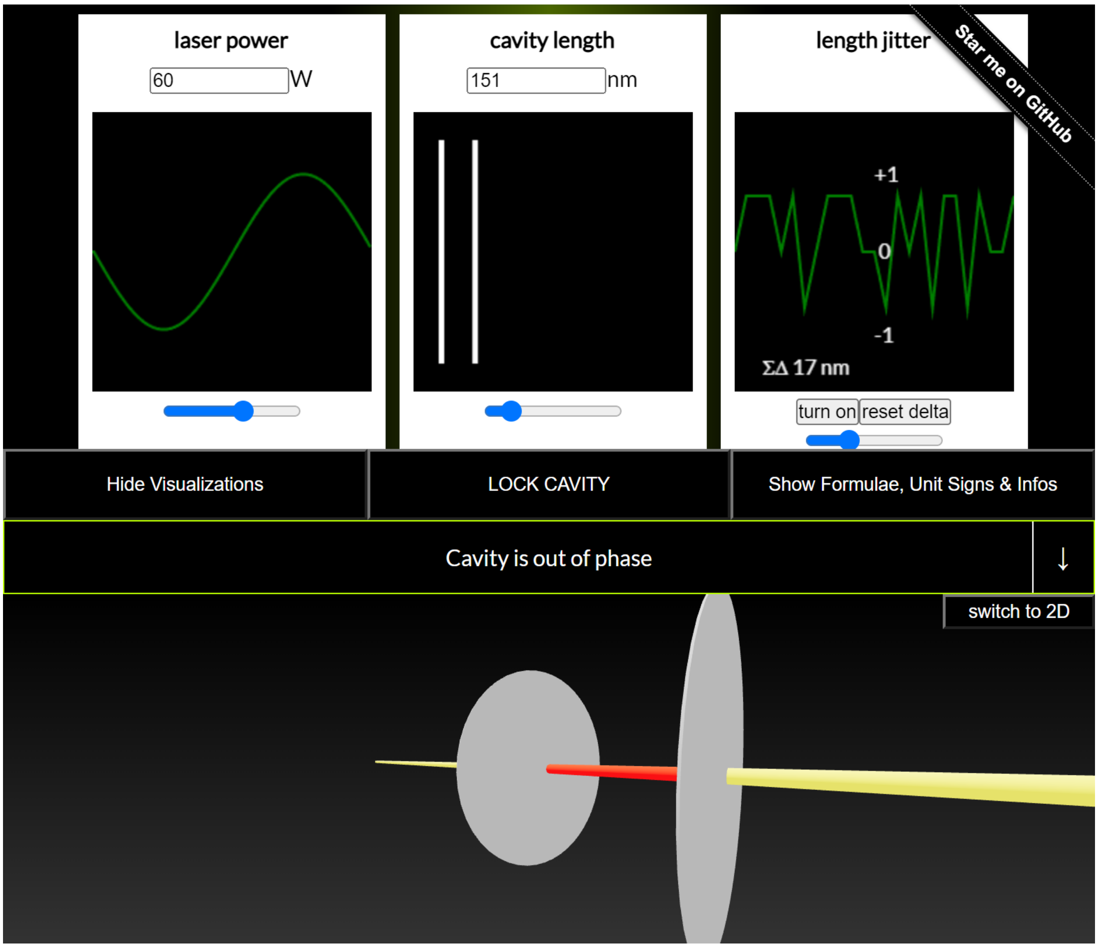
[3] <https://github.com/bmedicke/optical-cavity>

[4] <https://benmedicke.com/simulator>

The optical cavity simulator has the following features:

- It is based on modern **web technologies** to be future-proof and to support as wide a range of devices as feasible. Section 4.2 Technology Stack covers this in more detail.
- Additionally, it is **mobile-friendly**. This means it behaves well and consistently on reasonably up-to-date small-screen devices ranging from iPhones and Android mobiles to tablets, not just large-screen devices.
- The simulator is built as a **single-page application (SPA)**, which reduces the required communication between the server and the client. Once the simulator loads from the server, no further communication must occur. All calculations and renderings happen on the client side; the server is only used to distribute the application. A cached version of the simulator is fully functional, and so is a version saved for offline use.
- Input and output variables are adjustable from independent box components. These components (with two exceptions) provide **visualizations for the corresponding variables** to aid in understanding the impact of changes to variables.
- Visualizations and calculations **update on the fly**. Dragging a slider (or entering a new number into an input field) to change a value immediately triggers an update for all related variables and visualizations. This process ensures that the displayed results are always up to date.
- The **box components' design is adjustable**.
  - Users of low-end devices or those that prefer a more condensed layout can **toggle visualizations** off.
  - The simulator can **display relevant formulas** (where applicable) and **units**. See figure 28 on page 55 for an example.
  - Box components have buttons that open **info overlays**. These overlays give the user more information about the component, the corresponding calculation of the variable, and its visualization.
- The **cavity view** can be set to 2D mode, 3D mode, or hidden.
  - **3D mode** interactively displays a typical Fabry-Pérot cavity setup while
  - **2D mode** visualizes the current light inside of the cavity.

Figure 27 on page 54 shows the optical cavity simulator with active visualizations and the cavity view (the lower half of the screenshot) set to 3D mode. This mode presents the user with an adjustable three-dimensional view of the cavity. For example, changing the cavity length in the corresponding box component moves the mirrors while changing the wavelength inside the visible spectrum updates the color of the pumped and emitted beam. In addition, dragging inside the 3D visualization changes the camera's perspective by pivoting around the cavity and holding the shift key while dragging or using two fingers to drag pans the cavity setup.



laser power 60 W

cavity length 151 nm

length jitter

ΣΔ 17 nm

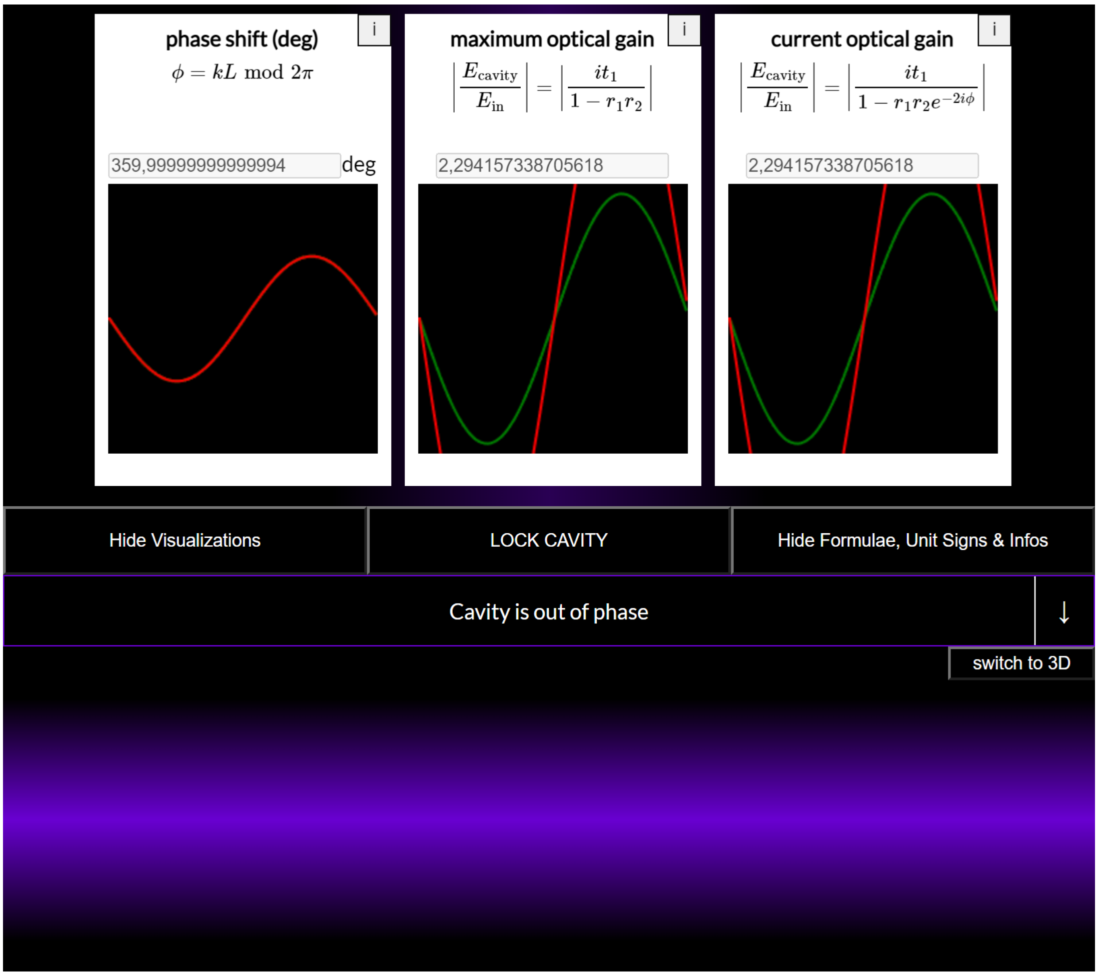
Cavity is out of phase

Figure 27: the cavity simulator with the cavity view (bottom part) in 3D mode (showing the arrangement of the mirrors) and active visualizations of parameters and control components (top part), UI elements adjust their color to match the selected wavelength (greenish-yellow in this instance)

Switching the cavity view to 2D mode shows the Gaussian beam that builds inside of the cavity (see the lower half of figure 28 on page 55). This can be thought of as taking a closer look at the red part of the beam from the 3D visualization. A **Gaussian beam** is a good approximation of most lasers. Such a beam is brightest at its center. The exact intensity gradient follows the normal (or Laplace-Gauss) distribution. Increasing the power or bringing the cavity further into resonance (by adjusting options that influence the phase shift) leads to a brighter longitudinal beam profile in the 2D cavity view.

Note that figure 27 and figure 28 use different wavelengths, resulting in different colors for both a selection of UI elements and the beams. For example, the simulator's configuration in the first image uses about 570nm wavelength (a yellowish/greenish beam), while the second uses around 400nm wavelength (a violet beam). Selecting a wavelength that lies outside the visible spectrum of the human eye, such as one in the ultraviolet or x-ray bands, switches the color to white. This was an obvious choice as white light is inherently non-monochromatic and, as such, not a part of the visible spectrum, avoiding any possibility of confusion while being subjectively easier on the eyes than other non-spectral colors such as brown or magenta at the same time.

Figure 28 also shows the cavity simulator with enabled formulas and units. The same option enables rendering the info buttons in the upper right corner of the box components that open the info overlay when clicked on. The simulator renders and displays the formulas dynamically, which means they do not rely on static images. The formulas and units can be copied in either TeX (as used in LaTeX) or MathML (XML-based) formats via the right-click menu.



phase shift (deg)

$\phi = kL \bmod 2\pi$

359,99999999999994 deg

maximum optical gain

$\left|\frac{E_{\text{cavity}}}{E_{\text{in}}}\right| = \left|\frac{it_1}{1-r_1r_2}\right|$

2,294157338705618

current optical gain

$\left|\frac{E_{\text{cavity}}}{E_{\text{in}}}\right| = \left|\frac{it_1}{1-r_1r_2e^{-2i\phi}}\right|$

2,294157338705618

Cavity is out of phase

Figure 28: the cavity simulator with enabled formulas, visualizations, and the cavity view (bottom) set to Gaussian beam 2D mode

The upper part of the optical cavity simulator is scrollable (or draggable on touch-enabled devices) to switch between the list of possible box components on small screens.

4.2 Technology Stack

We decided to implement the optical cavity simulator in the form of a web application (web app) for the following reasons:

- Unlike desktop applications, web apps do not require installation or update procedures when fetched directly from the server. Applications that require installations can be a problem when user accounts lack the necessary permissions. Additionally, feature rollout becomes trivial and does not rely on the application user. [101]
- Implementing the application as a SPA still allows the user to use a cached version of the simulator or to save it for offline use. [15]
- Web apps automatically support many devices, including prevalent mobile phones, without any operating system (OS) restrictions. This fact means that we only need to maintain a single codebase. [101]
- Many non-jailbroken mobile phones obtain applications from an app store that requires developers to submit their programs for review. This approval process can involve non-free developer accounts and requires repetition for each update. Web apps sidestep these disadvantages. [4]
- For most developers, the burden of entry for web apps is lower than for native applications due to the popularity of the used technology stack. [100] We hope this choice increases the likelihood of someone interested in the simulator diving in further.
- Modern browsers use hardware acceleration. This use of the graphics processing unit (GPU) for rendering makes frequently updating 2D and 3D graphics a non-issue. [34]

The technology stack that we chose for the optical cavity simulator can be subdivided into two main parts. The first part of the stack concerns the deployed application, as seen in figure 29 on page 57. The second part is the more comprehensive technology stack that includes build and DevOps tools, the used web server, and the base OS. The following chapters cover the former stack plus the build toolchain (Node.js, npm, and npx). The remaining components of the more comprehensive technology stack are generally interchangeable. Therefore, interested developers can choose technologies depending on their requirements and preferences.

Personally, I have chosen GitHub pages[5] as a hosting platform (and resulting from that choice, a git-based deployment workflow) in combination with the static site generator Jekyll[6] to embed the simulator into my existing homepage.

4.2.1 Fundamental Front-End Web Technologies

The core technologies of every web app's front-end are: the HyperText Markup Language (HTML), Cascading Style Sheets (CSS) and JavaScript [67]. HTML creates the building blocks of web apps displayed in the browser. These elements, such as input fields, sliders, and objects holding visualizations, become nodes of the Document Object Model (DOM), an XML-

[5] <https://pages.github.com/>

[6] <https://jekyllrb.com/>

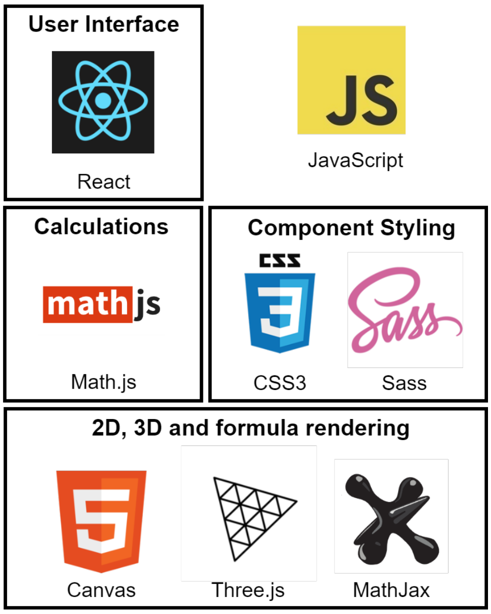


Figure 29: **technology stack of the deployed application**, we use JavaScript [111] as the overarching programming language, React [110] for the UI and CSS3/Sass [117, 43] for its styling, the simulator's calculations utilize Math.js [43], and rendering relies on HTML5's Canvas API [118] (2D), Three.js [113] (3D), plus MathJax [114] (formulas)

based tree graph of the UI. [65] CSS styles these DOM elements. Finally, JavaScript (JS) adds functionality to the client side, including, in the case of SPAs, manipulating the DOM without performing additional HTTP requests to the server or navigating to a new URL [66]. It is possible to program client functionality in languages other than JS, such as TypeScript or CoffeeScript. [10, 62] However, before execution by the browser, the code must be transpiled to JS (ignoring solutions based on WebAssembly [39]).

4.2.2 The Build Toolchain: Node.js, npm, and npx

Node.js (or Node) is an open-source JS run-time environment based on Google's V8 JS engine that facilitates running JS outside of browsers [73]. It expands the role of JS from a strictly client-side programming language to a server-side language. While creating application programming interfaces (APIs) is a common use case for Node, it is also the basis for the **Node package manager (npm)**. The `npm` command-line interface (CLI) provides access to the npm registry of open-source tools and libraries. **`npx`**, npm's package runner tool, can execute programs from the npm registry directly without prior installation [27]:

The shell command `npx create-react-app simulator` bootstraps a new React app named `simulator` from a boilerplate (a starting template). The `create-react-app` package generates the project structure, dependency plus configuration files, and build scripts. Executing `npm start` calls the **build scripts** that produce the final application from the source code and media files. [60]

4.2.3 React

React is a declarative JS library by Meta for creating web app UIs. It uses the concept of composition: **components**, isolated and reusable UI elements, combine into a fast and interactive web app. Like DOM elements, components are nodes in a tree graph with a single root element. Similarly, components can pool multiple DOM elements but also further child components (which are part of hierarchically higher components). [61]

Modern React uses **functional components**, which are components based on JS functions instead of classes. These functions receive external data via arguments called **props** (from *properties*) and return **JavaScript Syntax Extension (JSX)** syntax, React's JS extension that aims to be a facsimile of the HTML syntax. React's preprocessor converts JSX to JS which returns HTML that ultimately ends up in the DOM of the user's browser as part of the UI. [61]

One of React's significant technical advantages is the speedup stemming from its use of a **virtual DOM**. Usually, when any DOM element changes programmatically, the browser has to re-render the entire DOM-tree, which is a very slow operation. Even accessing the DOM via a read-operation is slow. React remediates this issue by keeping a representational JS copy of the real DOM in memory, the virtual DOM. When the state of a component changes, React duplicates the virtual DOM, applies the pursuant changes to the new copy, and then diffs it with the unchanged version. The resulting delta lets the browser know which DOM elements changed and thus require re-rendering. [61]

In the context of the simulator, this results in a fast re-render of only the required page elements when the user changes the cavity setup's properties. For example, if the user drags the cavity length slider, only these components redraw:

- the cavity length component itself,
- the phase shift components (radians and degrees),
- three of the four gain components (all except maximum gain),
- and the cavity view.

The other components remain untouched by the re-render. The DOM update ignores hidden components as well. [61]

4.2.4 HTML5 Canvas

HTML5's <**canvas**> element has a JS API that allows for the drawing of paths, shapes, and other two-dimensional graphics (with some limited support for drawing three-dimensional objects). In addition, the rendering of the Canvas element is hardware-accelerated in modern browsers, profiting from possibly available GPUs. [64]

The 2D visualizations of the optical cavity simulator's box components use the Canvas element to render graphics with high performance to provide visual feedback about variable changes to the simulator user that feels instantaneous.

4.2.5 WebGL and Three.js

Web Graphics Library (WebGL) is a hardware-accelerated 3D and 2D graphics web standard and API initiated by the Mozilla Foundation. As a low-level library, it requires developers to interact directly with vertices (corners of geometry), edges (connections of two vertices), and polygons (faces between three vertices). [68]

Three.js is a JS 3D graphics library for the browser that builds on top of WebGL to provide a high-level API, though recent versions also include the option to select a 2D Canvas-based or scalable vector graphics (SVG) rendering engine. It supports objects (geometric meshes and textures), lights, cameras, and shaders. We use Three.js for the 3D cavity view of the simulator. [23]

4.2.6 Math.js

Math.js is an open-source JS library that provides, among features we do not use, common constants and support for calculating with complex numbers, a feature missing from vanilla JS that we rely on for some of the calculations. [44]

4.2.7 MathJax

MathJax is a JS library that renders mathematical formulas in browsers. The input can be either MathJS markup or the math-specific subset of the LaTeX syntax. The default output is rendered with available math fonts, HTML, and CSS. Alternatively, it can produce SVG graphics. [106]

We use this library to avoid having to create, store and transfer an image (or possibly multiple images due to varying resolutions and sizes of display devices) for each formula we display in the simulator.

4.3 User Interface Elements of the Simulator

This section offers a more in-depth look at the UI of the optical cavity simulator. Figure 30 on page 61 shows the application with hidden visualizations and a collapsed cavity view to display all box components in a compact space. The highlighted elements are:

a.) A **box component**. These are UI elements that can contain information, control elements, and visualizations concerning one specific formula, variable or function. These come in one of three different types:

- **Input variable components** allow the user to manipulate a variable directly, either by adjusting the slider or typing a value into the input field. On a change, all corresponding UI elements update as well.
- **Automation components** constantly manipulate a variable without further input once a user activates them. Changes from this box type trigger the same update procedure as those from input variable boxes. Users can start or stop the automation and adjust some properties, such as the delay between automated manipulations.
- **Output variable components** contain automatically calculated values using related variables. Users can not set these values directly. These components disable their text fields (greyed out), and there is no slider, but most offer a visualization.

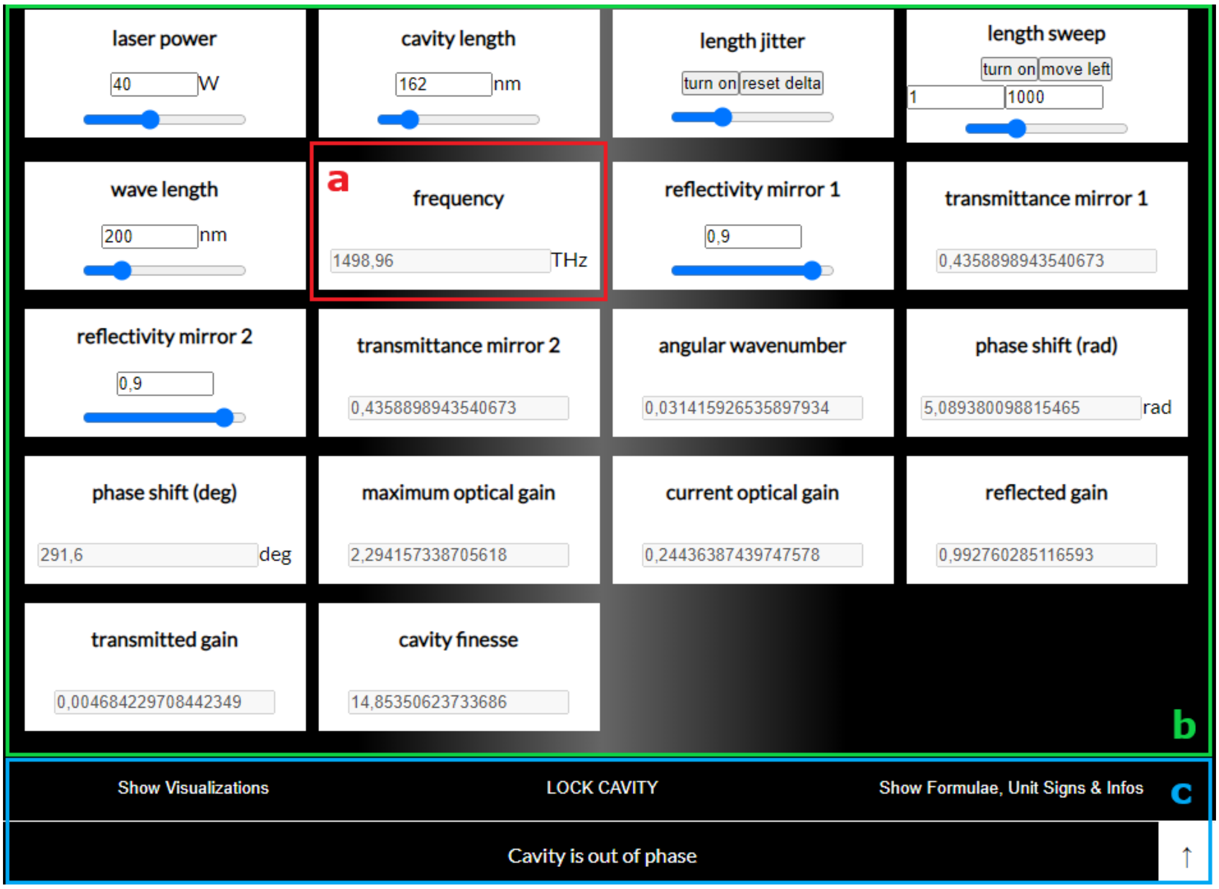
b.) The **control and results area** of the cavity houses all types of box components.

c.) Two rows in the lower part of the simulator hold buttons that control the UI or bring the cavity into resonance, and the **status bar** with the **locking indicator**.

4.3.1 Locking Indicator

The locking indicator is the fastest way to check if the current configuration represents an optical cavity in resonance. See figure 31 on page 61 for all possible values. In addition to showing if the cavity is locked (c.) or out of phase (b.), it also displays a message if the cavity is maximally out of phase (a.). This state can be thought of as the opposite of a cavity in resonance. When constructive interferences in a cavity are at their maximum, a cavity is said to be locked or in resonance. However, if destructive interferences are at their maximum, the cavity is maximally out of phase. For example, in a Fabry-Pérot cavity, this happens at 90 degrees phase shift between the waves that travel from left to right and those that travel in the opposite direction. After another 90 degrees, the cavity is in resonance again. This pattern continues in perpetuity (unless one calculates in modulo 360).

This behavior makes sense intuitively as a cavity is in resonance when $L = \frac{\lambda}{2}\, m$ where m is a positive natural number. A cavity with a pumping laser wavelength of 200 nm and a cavity length of 100 nm is in resonance. The same is true for a cavity length of 200 nm, 300 nm, and so forth (all 180-degree phase shift steps). The points where the cavity is maximally out of phase have to lie between those values. Checking it with the simulator confirms this: 150 nm, 250 nm, and so on are indeed values where destructive interference dominates. These are 180-degree phase shift steps again, but with an initial offset of 90 degrees.



laser power 40 W

cavity length 162 nm

length jitter

length sweep 1 1000

wave length 200 nm

frequency 1498,96 THz

reflectivity mirror 1 0,9

transmittance mirror 1 0,4358898943540673

reflectivity mirror 2 0,9

transmittance mirror 2 0,4358898943540673

angular wavenumber 0,031415926535897934

phase shift (rad) 5,089380098815465 rad

phase shift (deg) 291,6 deg

maximum optical gain 2,294157338705618

current optical gain 0,24436387439747578

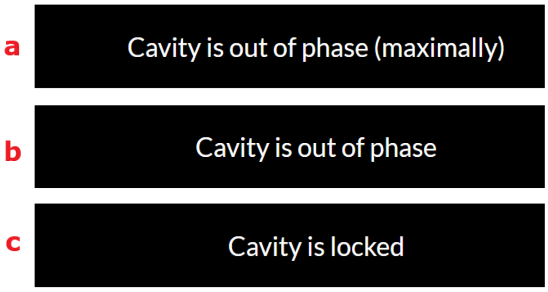
reflected gain 0,992760285116593

transmitted gain 0,004684229708442349

cavity finesse 14,85350623733686

Cavity is out of phase

Figure 30: **major UI elements** of the cavity simulator: **a.)** a box component, **b.)** the control and results area, **c.)** the UI and cavity lock buttons (above) with the status bar and cavity view toggle button (below)



a Cavity is out of phase (maximally)

b Cavity is out of phase

c Cavity is locked

Figure 31: possible values of the **cavity locking indicator** in the status bar: **a.)** the nodes of the inbound wave overlap with the anti-nodes of the reflected wave **c.)** nodes overlap between inbound and reflected waves, **b.)** any state in between

4.3.2 Input Variable Components

Input variable components are the primary way users interact with the optical cavity simulator. These components tune all pertinent properties of the simulated cavity and represent the decisions one must make when constructing an actual cavity, such as which wavelength the pumping laser should emit.

Laser Power and Cavity Setup

The laser power component sets the power output (P) of the pumping laser, which is located on the left side of the cavity setup and points its beam towards the back side of the first (fixed) mirror. See figure 32 on page 62 for the simulated setup. Note that the curvature of the mirrors does not impact this model of simulation; they were chosen in this graphic to represent the mirrors' orientation clearly. One advantage of a simulation over an actual cavity is that the simulator is not limited to measuring the out-coupled light hitting the detector past the second, adjustable mirror, allowing for the simultaneous display of the gain in all stages of the beam's travel.

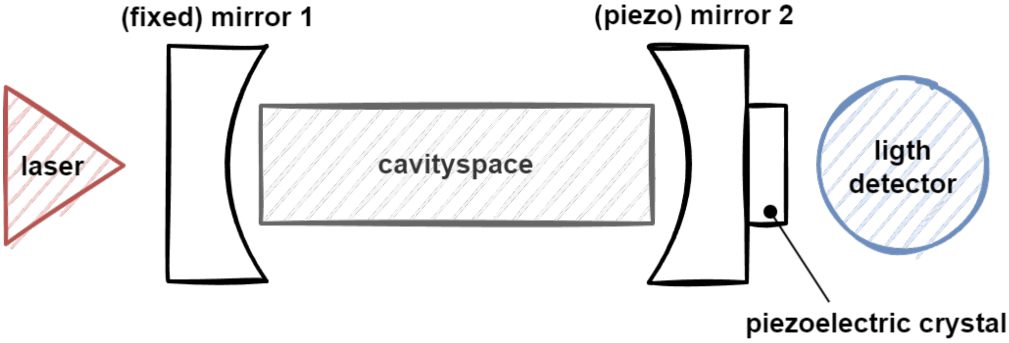


Figure 32: the setup of the simulated cavity, the piezoelectric crystal moves mirror 2 to adjust the spacing between the two mirrors, this distance influences how much light from the laser reaches the light detector

Possible values for the laser power range from zero to 100 Watt (W), a purely representative range that doubles as a percentage of the maximum power output. The output wattage of actual lasers can reach significantly higher levels, but this does not influence the gains for an idealized cavity.

The visualization shows a green, single wavelength of a sine wave. We chose green to represent the initial wave, in this case, the light which the laser emits towards the cavity. If the user increases the laser power, the sine wave's amplitude follows. See figure 33 on page 63 for the difference in visualization between two divergent values.

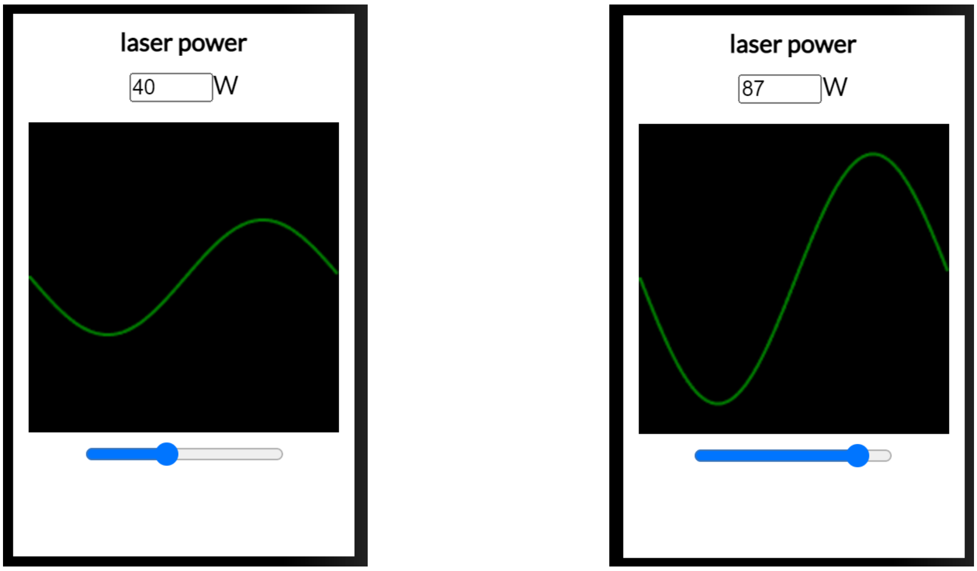


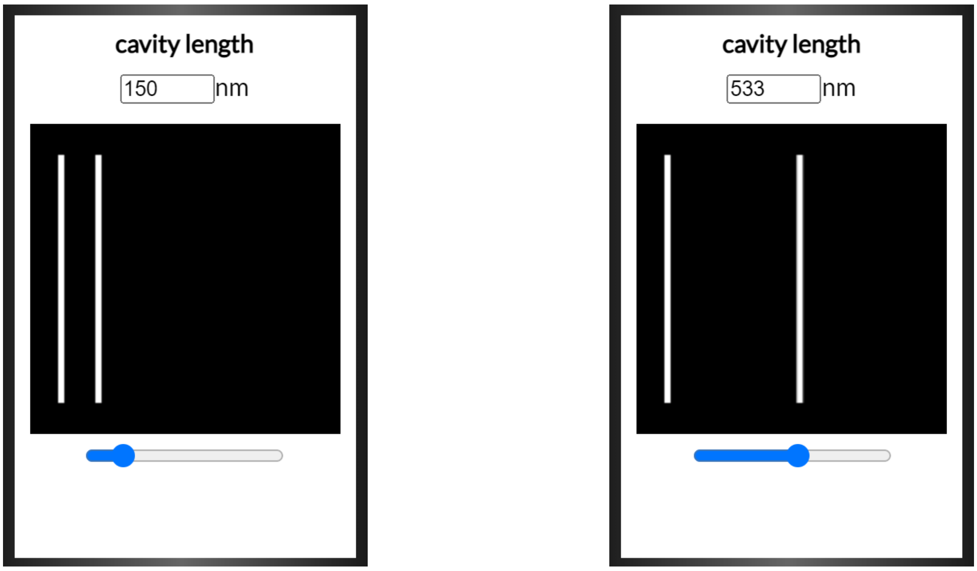
Figure 33: **laser power component** with low (left) and high (right) pumping laser output, the height of the sine wave's amplitude visualizes the power level

Cavity Length

The cavity length (L) is the distance between the left fixed mirror and the right movable mirror. The simulator measures this distance in nanometers. The corresponding component's visualization uses two white, vertical lines to stand in for these mirrors. As in a real-world setup, only one mirror adjusts (the right one in this case). Real experimental setups often use a piezoelectric crystal affixed to the mirror for adjustments, as shown in figure 32 on page 62. When a power source applies a charge to it, a piezoelectric material slightly changes its shape. The amount of distention increases with rising voltage. This property lends itself to creating precise actuators (devices that create movement). [25] For this reason, we call this mirror the **piezo mirror** and the other one the **fixed mirror**. See figure 34 on page 64 for screenshots.

Wavelength

Like the laser power component, the wavelength component adjusts a property of the light the laser emits. The used unit for the wavelength (λ) is nanometers. The possible wavelength values in the simulator range from one to 1000 nanometers, the same as for the cavity length. However, unlike the wavelength, the cavity length can be significantly higher in real-world implementations of optical cavities than in the simulator. The exact range span depends on the construction (microcavities versus macrocavities). We decided to limit the range to increase the slider's sensitivity, especially for tablets and mobile phones, where entering exact numbers is more cumbersome. This does not limit the applicability of the simulator for macrocavities, as a user can subtract distances above 1000 nm according to the resonance condition without altering the simulation results.



cavity length

150 nm

cavity length

533 nm

Figure 34: **cavity length component**, the two vertical lines represent the mirrors that encompass the optical cavity

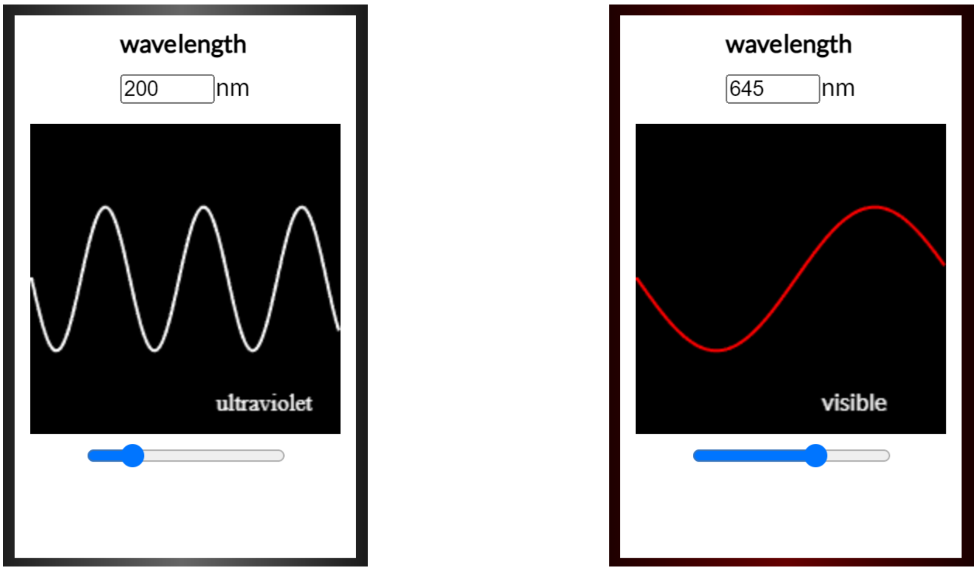
The color and number of waveforms (with heavily scaled-up sizes) of the sine wave in the visualization depend on the selected wavelength. In the visible spectrum, the color matches the perception of the human eye. Outside this spectrum, the sine wave is white. See figure 35 on page 65 for two examples, one outside and one inside the visible spectrum. The name of the current band of the spectrum sits in the lower right. The ranges are:

- $< \ 11 \ \ nm$ **x-ray** (from leftmost slider position)
- $< \ 380 \ nm$ **ultraviolet**
- $< \ 750 \ nm$ **visible**
- $\geq \ 750 \ nm$ **infrared** (up to the rightmost slider position)

Mirror Reflectivity for Mirror 1 and Mirror 2

The reflectivity component is the first kind with more than one proxy, namely, mirror one (the fixed one on the left) and mirror two (the adjustable one on the right). The visualization requires some explanation. As seen in figure 36 on page 66 there are three visual components to it:

1. The already familiar vertical, **white line** represents a **mirror** (either number one or two, depending on the component's title).
2. A **green line** that represents the **incident beam** starts at the top left and ends at the mirror's center.
3. Finally, a **red line** that represents the **reflected beam**. This angles from the mirror's



wavelength

200 nm

ultraviolet

wavelength

645 nm

visible

Figure 35: **wavelength component**, the range goes from x-ray (slider left) to infrared (slider right), the color of this sine wave and select UI elements correspond to the wavelengths in the visible spectrum (white otherwise)

center to the lower left.

The thickness of a beam signifies its intensity. For example, compare the left and right images in the figure. The green beam has a static thickness (independent of the selected laser power), while the red one's boldness depends on the chosen value.

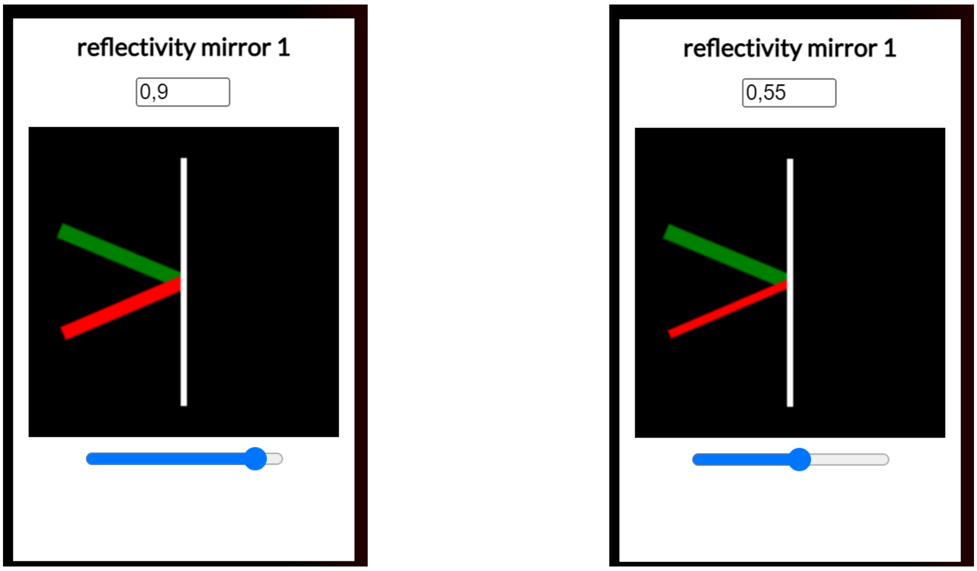
It is important to note that the angles of the beams are purely for visual purposes! This decision allows us to not draw the two beams on top of each other, making the visualization easier to parse. The simulated FPI does not use angled mirrors or incidental rays that are non-parallel to the axis of the mirrors, as visible in figure 32 on page 62.

4.3.3 Automation Components: Sweep and Jitter

This component category automatically and independently manipulates variables of the simulator once started. The **slider** at the bottom sets the **delay** between manipulations. A short delay (with the slider towards the left side) results in more frequent updates. The slider covers a range from one to 100 milliseconds delay.

Length Sweep

When active, this component continuously increases the length of the cavity until it hits the up-



reflectivity mirror 1

0,9

reflectivity mirror 1

0,55

Figure 36: **reflectivity component**, specifies the amount of reflected light for the corresponding mirror, the angle of the rays in the visualization is for demonstrational purposes only

per limit, the value in the right text field. At this point, the direction reverses, and the component continuously decreases the length to the lower limit, the left text field's value. A user can toggle this sweep on or off by clicking the first button while the second button reverses the direction (as if one of the limits were hit). See figure 37 on page 67 for a component with disabled (left) and enabled (right) sweep. The arrow visualizes the direction and state of the sweep (greyed out when disabled, white otherwise).

This component helps create animations or get a quick feel of how a cavity with specific settings behaves without requiring the user to drag the cavity length slider by hand (which might be off-screen on small-screen devices). In a real-world cavity, this can be achieved by applying a constantly increasing and decreasing voltage to the piezoelectric actuator.

Length Jitter

Once a user activates this component, it applies a random jitter to the length of the cavity. For each tick, this jitter has a random chance to apply one of the following three modifications with an equal chance:

- Increase the length of the cavity by one nanometer.
- Do nothing.
- Decrease the length of the cavity by one nanometer.

This emulates a noise source acting on the cavity, for example, a temperature drift. The visu-

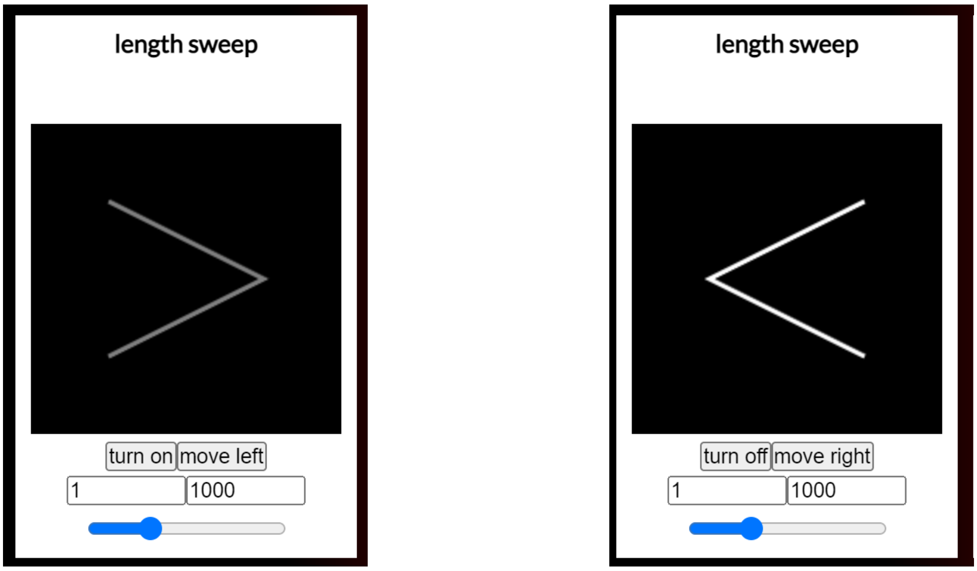


Figure 37: **length sweep component**, a component that sweeps the movable mirror back and forth between set values to automatically adjust the cavity length, the arrow points in the direction of the current movement, a grey arrow means it is inactive (left image) while a white one means it is active (right image)

alization plots the last couple of random choices and the (resettable) total cavity length delta stemming from the jitter component. See figure 38 on page 67. The source of randomness is a function that generates non-coherent noise.

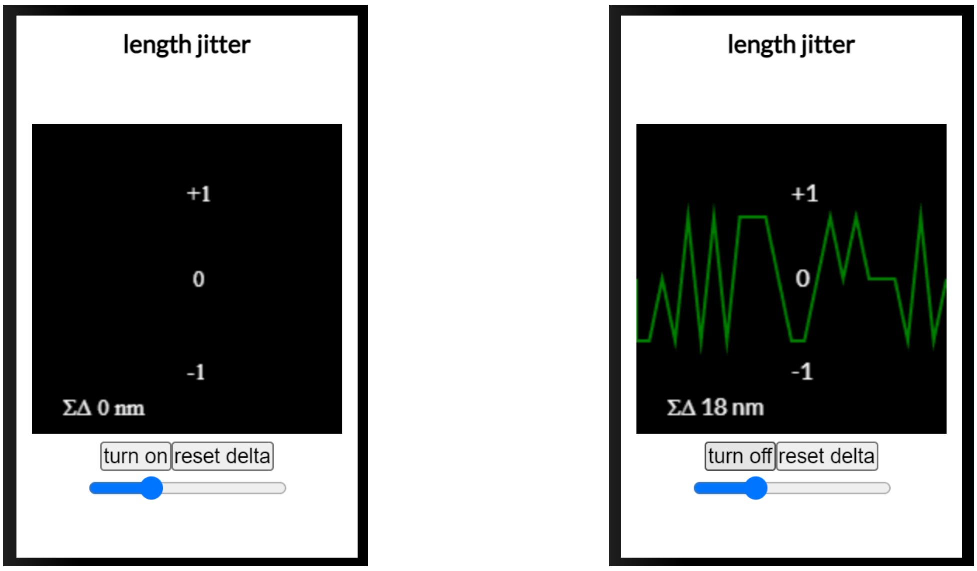


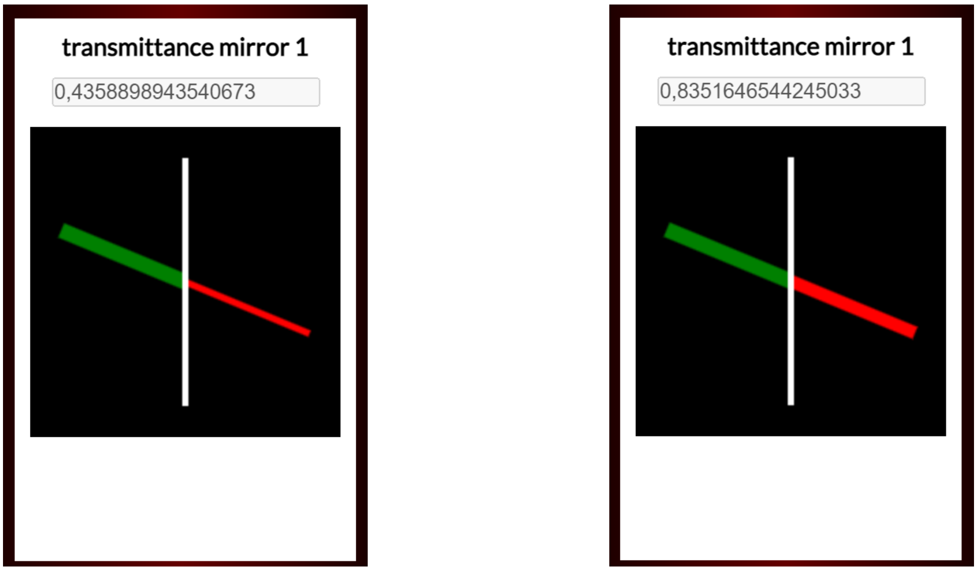
Figure 38: **length jitter component**, a component that randomly adjusts the movable mirror when active (right) to simulate noise, keeps track of the total delta of the position

4.3.4 Output Variable Components

Output variable components calculate and display values that rely on input or output variables. Users can not change these values directly but must find and modify all related components. For this purpose, it is advisable to display formulas by clicking on the "Show Formulae, Unit Signs & Infos" button as shown in figure 28 on page 55. As an example, the formula for the current optical gain inside the cavity is $\left|\frac{it_1}{1-r_1r_2e^{-2i\phi}}\right|$. e is Euler's constant, and i is the imaginary unit. This means one would have to look for the t_1, r_1, r_2, and ϕ variables, which might in turn depend on other variables and so forth.

Mirror Transmittance for Mirror 1 and Mirror 2

These components use the reflectivities of the corresponding mirrors to calculate the coefficient of transmission ($t_n = \sqrt{1-r_n^2}$). The visualization is similar to the reflectivity component, and the same caveat about the angle of incident applies. See figure 39 on page 68.



transmittance mirror 1

0,4358898943540673

transmittance mirror 1

0,8351646544245033

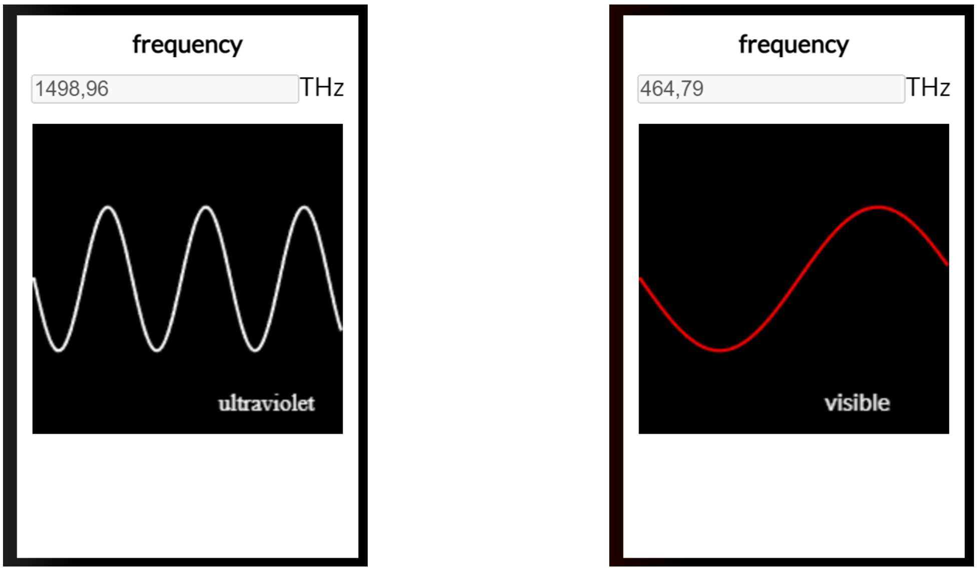
Figure 39: **transmittance component**, an output component that calculates the transmittance from the reflectivity of the corresponding mirror, the angle of the rays in the visualization is for demonstrational purposes only

Frequency

The frequency (ν) of a wave relates to the wavelength (λ) by the speed of light (c) in the medium that it travels through: $\nu = \frac{c}{\lambda}$. The optical cavity simulator uses the **speed of light in a vacuum** for calculations, which is exactly $299,792,458\ \frac{m}{s}$ (because the meter's definition is the distance light travels in $\frac{1}{299,792,458}\ s$). Due to the air's refractive index of about 1.0003 versus vacuum's 1.0, light slows down in this medium to $\frac{1.0003}{299,792,458}\ \frac{m}{s} = 299,702,547\ \frac{m}{s}$. While a difference of more than $90\ \frac{km}{s}$ might seem significant, it barely impacts the calculation in our case (a $\sim 0.03\%$

change). Additionally, the refractive index of air is not constant, as it varies with temperature, which led us to choose the speed c for a vacuum. The unit of the frequency is the Hertz ($1\ Hz = \frac{1}{s}$, meaning one cycle or wavelength per second).

The frequency component reuses the same visualization as the one for the wavelength. See figure 40 on page 69. The component displays the frequency in terahertz ($1\ THz = 10^{12}\ Hz$) with two decimal places precision. We chose to include this component even though it finds no direct use in further calculations because the unit of Hz is well-known to IT professionals. This familiarity is due to its use as a measure of the clock speed for processing units. However, these values range in the gigahertz ($1\ GHz = 10^{9}\ Hz$) or megahertz ($1\ MHz = 10^{6}\ Hz$) and describe operations per second (and not wavelengths).



frequency

1498,96 THz

frequency

464,79 THz

Figure 40: **frequency component**, a component that uses the speed of light in a vacuum and the set wavelength to calculate the corresponding frequency

Angular Wavenumber

The wavelength is the distance encompassing one wave pattern (for example, from crest to crest). A related quantity, the **angular** wavenumber (k), describes how many wave patterns fit into 2π distance: $k = \frac{2\pi}{\lambda}$. In the background, the optical cavity simulator uses nanometers as a unit for the distance to match the unit of the cavity length and simplify subsequent calculations. As an example, when a user sets the wavelength to $200\pi\ nm$ ($\approx\ 628.319\ nm$), the angular wavenumber will be 0.01, as only $\frac{1}{100}$th of the wave pattern fits into $2\pi\ nm$.

This component does not have a visualization because the wavenumbers of the visible spectrum range from only ~ 0.016 near the ultraviolet to ~ 0.008 near the infrared, which would be an imperceptible change in relation to 2π. However, figure 41 on page 70 displays the relationship between the wavelength and the angular wavenumber. For an example, in the x-ray band, when assuming nanometers as a unit. The plot shows a cosine wave with a wavelength of ~ 4.18879 that corresponds to a wavenumber of 1.5 as $\frac{2\pi}{4.18879} \approx 1.5$. The waveform fits into 2π one and a half times (one time from crest to crest plus one-half times from crest to trough).

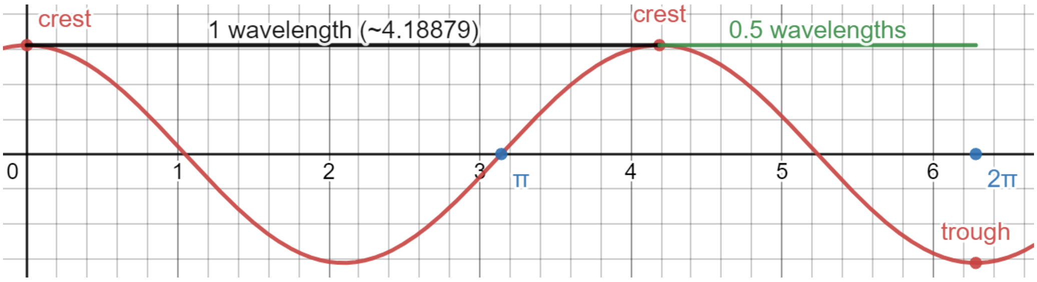
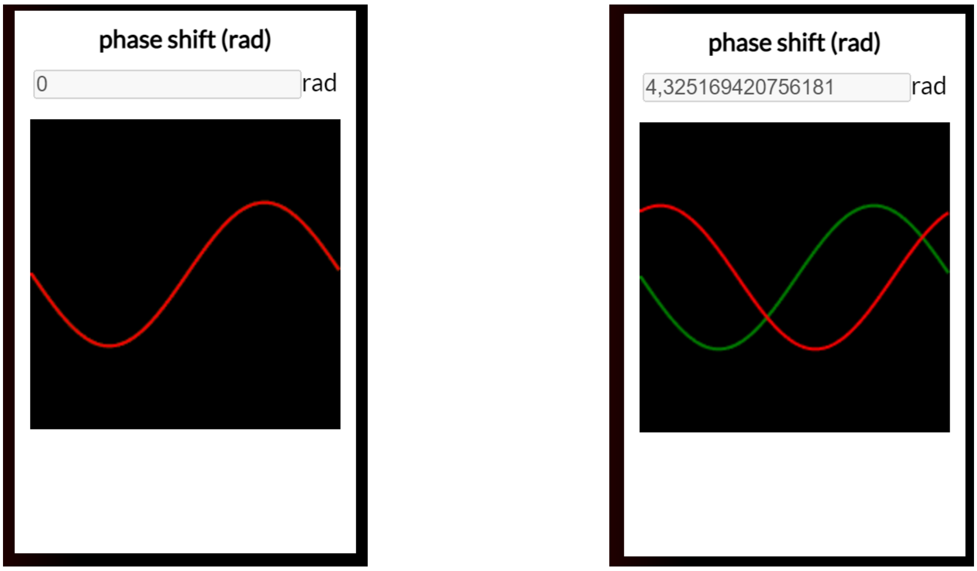


Figure 41: a plot showing a cosine wave with an **angular wavenumber** of 1.5, this number describes the spatial frequency of the wave (specifically, how many times the waveform, crest to crest, fits into 2π)

Phase Shift (Radians and Degrees)

The phase shift components contain values representing the offset between the incident and reflected wave, either in radians or degrees. The formula is: $\phi = kL \bmod 2\pi$ and the result is in radians. $\bmod 2\pi$ clamps the range to $< 2\pi$ radians (or < 360 degrees) because this is the point where the waveforms repeat. See figure 42 on page 70 for visualizations of cavity setups in and out of phase. The visualization for the radian and degree versions are identical.



phase shift (rad)

0 rad

phase shift (rad)

4,325169420756181 rad

Figure 42: **phase shift component**, visualizes the offset between the incident (green) and reflected (red) wave, this version displays the phase shift in radians

Gains in all Stages of the Optical Path

Gain or amplification is the change in the amplitude of a wave when comparing the output power with the input power (the ratio $\frac{P_{out}}{P_{in}}$). A gain of 1 means no change, a gain below that, a reduction of, and a gain above it, an increase in power. The simulator calculates three different types of gain, depending on the progress or route of light through the optical path, one before, one inside, and one after the cavity:

1. the reflected gain,
2. the gain inside the cavity,
3. and the transmitted gain.

The first and last ones have one component with according names. In contrast, the second one has two: the **current optical gain** is the amplification of the incidental light in the cavity with the actual settings, while the **maximum optical gain** shows what the cavity can achieve if it is in resonance. The **reflected gain** refers to the beam that reflects off the backside of the first mirror, thus never entering the cavity, plus the beam leaving through the first mirror, and the **transmitted gain** treats the beam leaving the cavity through the second mirror.

Both the reflected and the transmitted gain have a theoretical upper limit of 1 (during constant output), though, in practice, it is lower than that. See figures 43 and 44 on page 72 and note the scientific notation in the latter. The gain inside the cavity can be larger than 1 due to constructive interferences. See figures 45 and 46 on page 72. As with the previous components, the green wave represents the input and the red one the output.

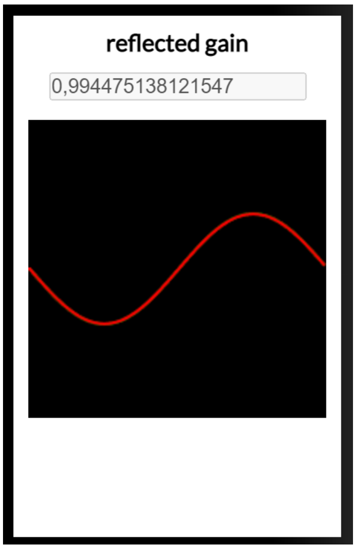
Unlike in other components, the amplitudes in the visualization of gains scale with the laser power to demonstrate the effect of the pumping power on the final output. A detailed examination of the formulas follows in chapter 4.4 Airy Distribution. **Sweeping the length of the cavity in the simulator while observing the gains shows that the transmission through the cavity is highest when the cavity is locked.**

Finesse

This component has no visualization, as the finesse does not serve as an input for further calculations.

Fabry-Pérot resonators can have a variable or fixed length. As shown in chapter 2.5.5 Optical Cavities, the resonance condition is:

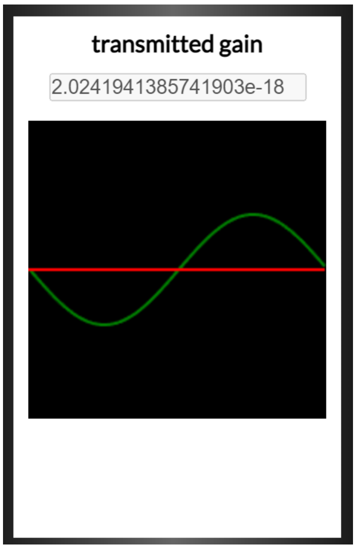
$$L = \frac{\lambda}{2} m, \; m \in \mathbb{N}^+$$



reflected gain

0,994475138121547

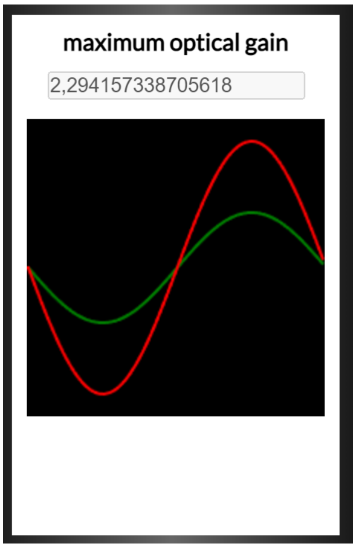
Figure 43: **reflected gain component**, visualizes how much light moves towards the laser



transmitted gain

2.0241941385741903e-18

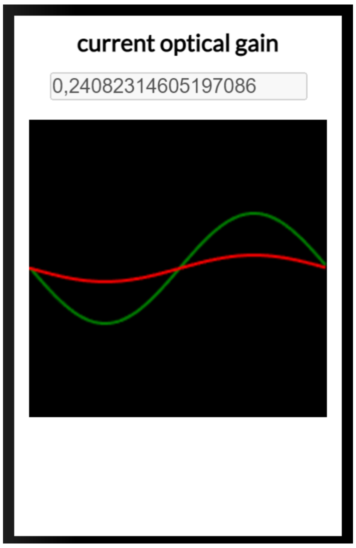
Figure 44: **transmitted gain component**, visualizes the input amplitude (green) and the transmitted one (red)



maximum optical gain

2,294157338705618

Figure 45: **maximum optical gain component**, visualizes the maximum strength of the light building in the cavity



current optical gain

0,24082314605197086

Figure 46: **current optical gain component**, visualizes the strength of the light building in the cavity with the current settings

This equation shows, that if the length L is fixed, the wavelength λ is the only variable left to bring the cavity in and out of resonance. When the refractive index of the medium, that light travels through, does not change, the frequency ν can stand in for the wavelength. It follows that the measurements of the output from such a resonator can be used to inspect both the cavity length or the light's frequency. Plotting the frequency and the corresponding transmitted power shows peaks where the cavity is in resonance. The distance between these high points is called the **free spectral range (FSR)** (or $\boldsymbol{\Delta}\nu_{FSR}$ because it denotes the **difference** between two frequencies).

The **full width at half maximum (FWHM)** ($\Delta\nu_{Airy}$) is another delta of two frequencies, those that encompass the resonance peak at the wave's half maximum, located at fifty percent of its maximum amplitude. [40] See figures 47 and figure 48 on page 73.

The definition of the **Finesse of an FPI Airy distribution** is as follows:

$$\mathcal{F}_{Airy} = \frac{\Delta\nu_{FSR}}{\Delta\nu_{Airy}} = \frac{\pi}{2}\frac{1}{\arcsin\left(\frac{1-\sqrt{r_1 r_2}}{2\sqrt[4]{r_1 r_2}}\right)}$$

$\mathcal{F}_{Airy}$ specifies the number of Airy distributions that are distinguishable from each other in the span of one FSR. The wider the FWHM (in the denominator) is, the worse the resolving power will be. As shown in the formula this number depends on $r1$ and $r2$, the reflectivities of the two mirrors: the more reflective the mirrors are, the narrower the FWHM, the higher the finesse. Note that this definition of the finesse does not work for low reflectivities. [40]

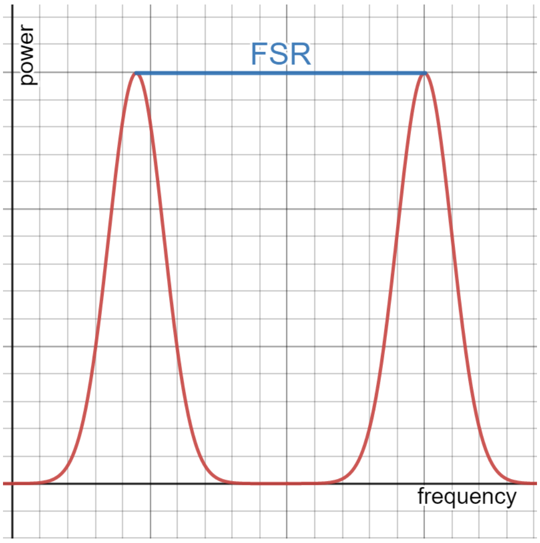


Figure 47: **free spectral range**, the distance between two transmission peaks

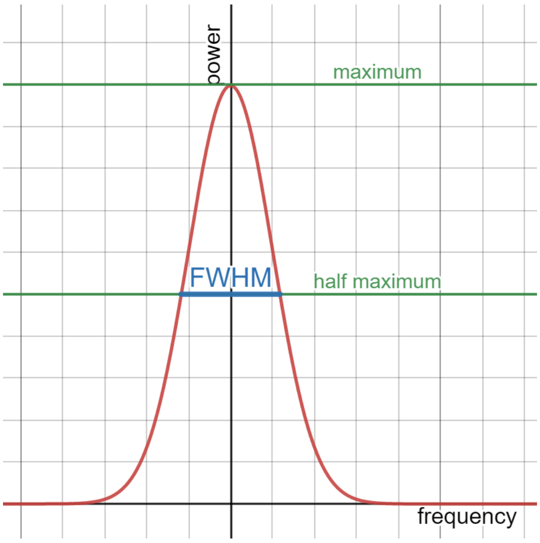


Figure 48: **full width at half maximum**, the linewidth at 50% of the maximum amplitude, based on [115]

4.4 Airy Distribution

In 1838 Astronomer Royal Sir George Biddell Airy, who was instrumental in establishing the prime meridian at Greenwich, published a paper [2] about the intensity of light near a caustic (an envelope of a light pattern formed by curved surfaces, such as at the bottom of a pool). A reformulation of the therein introduced Airy formula can calculate the intensities of light bouncing off, circulating in, and exiting an FPI.

Figure 49 on page 74 shows five different electric fields (E) in and around an FPI:

1. The laser emits E_{laser} towards mirror one.
2. The fraction of the field that makes it into the resonator (E_{in}) depends on the transmittance of that mirror.
3. $E_{reflected}$ is the sum of two fields:
   - The field that reflects off mirror one's backside and never reaches the cavity space. (The remaining energy field ($E_{laser} - E_{reflected}$) transmits through the mirror into the cavity.)
   - The field that transmits through mirror one from the cavity.
4. E_{cavity} is the field that circulates inside the cavity in the forward (outgoing) propagation direction.
5. $E_{transmitted}$ is the field that transmits through mirror number two (towards the physical light detector) and leaves the cavity. Again, the fraction that makes it through the mirror depends on the transmittance, this time of mirror two.

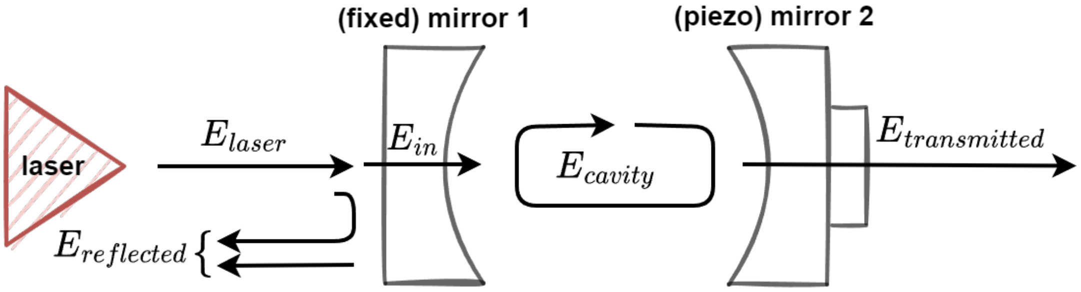


Figure 49: the simulator's **electric fields**, the laser emits the field E_laser towards the cavity, from that field E_in reaches the cavity, E_cavity circulates between the mirrors and either transmits through ($E\_transmitted$) or joins the reflected energy from E_laser to result in $E\_reflected$, based on [40]

The simplest of the gains to calculate is the **maximum optical gain**, the amplification that the current cavity configuration can achieve if it is in resonance. The field that enters the cavity (via t_1) circulates between the two mirrors (via r_1 and r_2).

$$G_{max} = \left|\frac{E_{\text{cavity}}}{E_{\text{in}}}\right| = \left|it_1 \frac{1}{1 - r_1 r_2}\right| = \left|\frac{it_1}{1 - r_1 r_2}\right|$$

This is a simplification of the formula for the **current optical gain** that introduces the effects of constructive and destructive interferences from phase shift via the angular wavenumber (k) and the cavity length (L):

$$G_{cur} = \left|\frac{E_{\text{cavity}}}{E_{\text{in}}}\right| = \left|\frac{it_1}{1 - r_1 r_2 e^{-2i\phi}}\right|$$

[40]

Using the new term $e^{-2i\phi}$, that adds the influence of the phase shift, with values of a locked cavity, for example $\lambda = 200$, $k = \frac{2\pi}{\lambda} \approx 0,0314$ and $L = 100$, we get the result 1, which explains the previous simplification. Repeating the calculation several times while increasing L to bring the cavity more and more out of phase results in decreasing (real) values and thus a reduction of the current gain. For a maximally out-of-phase cavity, the term returns -1. See figure 50 on page 75 for a plot of the current optical gain versus the cavity length. The following Python script generated the data for the plot: listing 3 on page 76.
The plot shows gain peaks with an interval of 50 nanometers ($\lambda/2$) between them. The first peak happens at the location of the first mode at 50nm. The troughs, where the cavity is maximally out of phase, start at 25 nm and repeat with the same interval as above. The first data point is at 1 nm, as $L \leq 0$ does not make sense for a physical cavity.

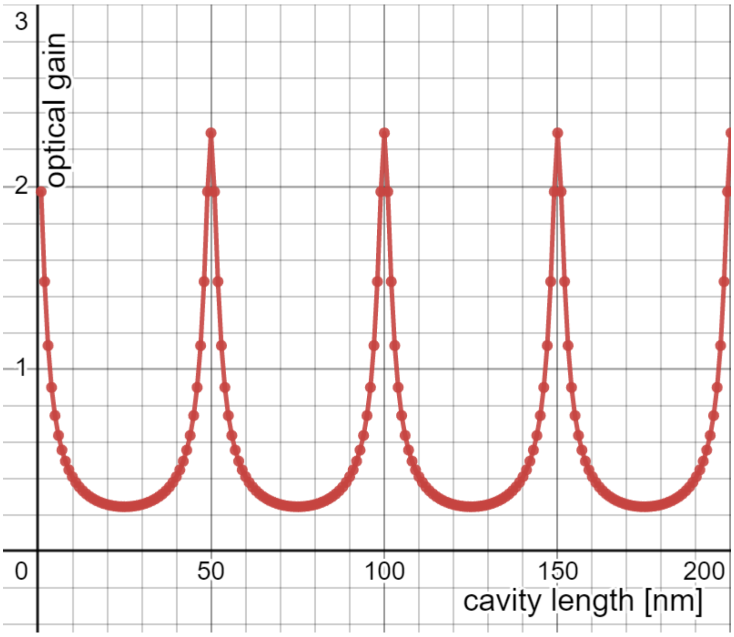


Figure 50: plot of current gain values in relation to the cavity length ($\lambda = 100$, $r_1 = r_2 = 0.9$)

```python
#!/usr/bin/env python3
# output csv format: ./sweep.py > data.csv
from cmath import sqrt, exp, pi  # support imaginary numbers.

j = complex(0, 1)  # avoid using "i" (common loop variable).
r1 = r2 = .9  # use identical mirrors.
t1 = sqrt(1-r1**2)  # calculate transmittance coefficient.
lambd = 100  # Python reserves lambda as a keyword.
k = 2*pi/lambd  # calculate angular wavenumber.
print('cavity length [nm], optical gain')  # output the csv header.
for L in range(1, lambd*3+1):  # sweep over the cavity length (wavelength x3).
    rho = (k*L)%(2*pi) # calculate phase shift in radians, clamp to 360 deg.
    denominator = 1-r1*r2 * exp(-2*j*rho)  # split calculation for clarity.
    # drop j from numerator (has no effect):
    G_cur = abs((t1)/denominator)  # calculate the gain with current L.
    print(L, G_cur, sep=',')  # output current data row
```

Listing 3: script that generates current optical gain data as used for figure 50 (Python)

The **reflected gain**'s formula is:

$$G_{refl} = \left|\frac{E_{\text{reflected}}}{E_{\text{in}}}\right| = \left|\frac{-r_1 + r_2 e^{-2i\phi}}{1 - r_1 r_2 e^{-2i\phi}}\right|$$

As mentioned at the start of the chapter, $E_{\text{reflected}}$ is a sum of two fields, the one that reflects off the backside of mirror one (via $-r1$) and the one that is transmitted to the left through mirror one bounced from mirror two (via $r2$ and the phase shift term). G_{refl} is at its maximum when the cavity is maximally out of phase and at its minimum when it is locked.

The formula for the **transmitted gain** is:

$$G_{trans} = \left|\frac{E_{\text{transmitted}}}{E_{\text{in}}}\right| = \left|\frac{-t_1 t_2 e^{-i\phi}}{1 - r_1 r_2 e^{-2i\phi}}\right|$$

[40] Note that the term in the exponent is $-i\phi$, not $-2i\phi$, and the use of the transmittance coefficients instead of reflectivities. This gain behaves diametrically to G_{refl}: when the cavity is maximally out of phase, G_{trans} is minimal, and when the cavity is in resonance, G_{trans} is at its greatest. See figure 51 on page 77 for a zoomed-in plot of all three gains.

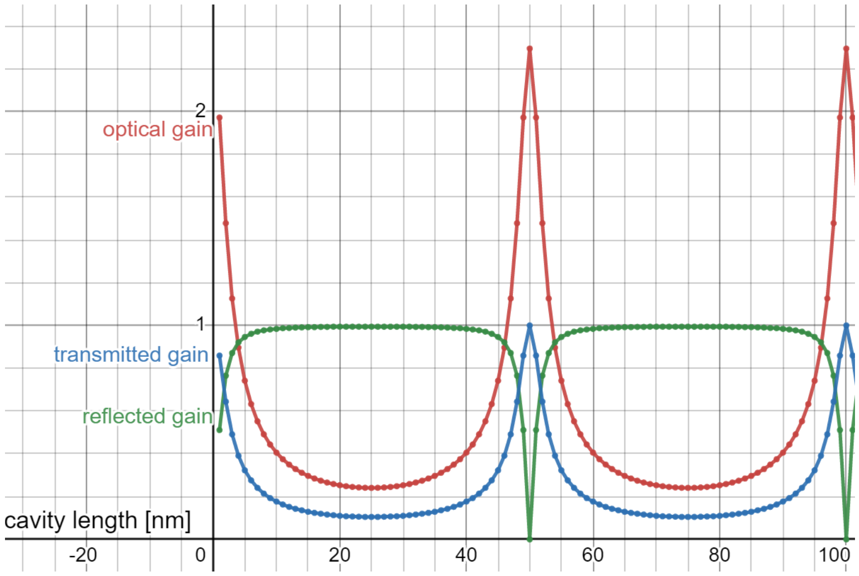


Figure 51: plot of **current optical (red), transmitted (blue) and reflected (green) gains** in relation to the cavity length using identical settings to the previous plot ($\lambda = 100$, r*1* = *r*$2 = 0.9$)

4.5 Source Code Breakdown

This chapter's first section documents the architecture of the optical cavity simulator. It concludes with a selection of implementation details and ideas how to extend the application by programming new components based on the `Box()` abstraction.

4.5.1 Project Structure

As a React-based application, the simulator's project structure contains two folders playing a part in the build process. These directories live at the root of the project:

- `src`, holds the program's source code. See figure 52 on page 79.
  - Executing `npm run build` creates an optimized production build from the contained files that is ready for deployment to a web server.
  - `npm run start` produces a development build and spins up a local web server used for the development process.
  - Additionally, we have added a script that connects the local development server to the internet via a tunnel. This script establishes a Secure Shell Protocol (SSH) connection to localhost.run[7] and assists in remote pair-programming workflows. The script lives in the `package.json` configuration file. To start it execute `npm run serve`.
- `build` contains the generated build files of the project, be it a production or development version. See figure 53 on page 79.

[7] <https://localhost.run/docs/http-tunnels>

  - Production builds are minified (stripped of comments and superfluous whitespace) and use joined files to improve load times.
  - Development builds contain debug symbols and support for hot-reloading (refreshing the application in the browser on changes to the codebase).

The `src` folder contains the `index.js` file, the starting point of the project. This script creates the React root that renders a DOM element housing the UI in the user's browser. It also initializes a context (that holds variables used by multiple components) and incloses the `<App/>` component, imported from `App.js`. See figure 54 on page 79, that visualizes this and further decencies.

The **context** allows components to use the same variables and subscribe to state changes. Using a context avoids passing data as props through the tree of components, whether each node on the way needs it or not. [59] This process is colloquially known as **prop-drilling**.

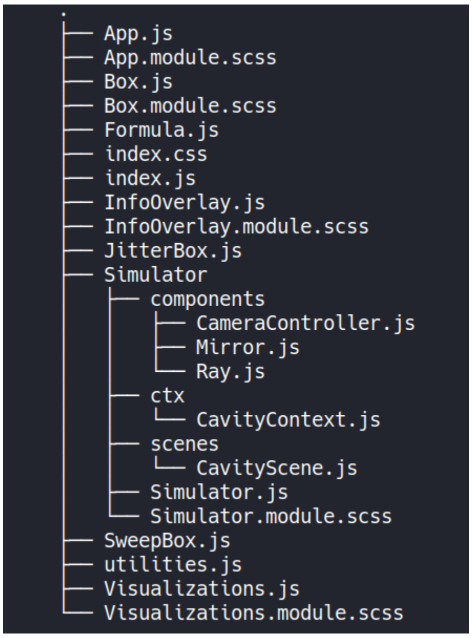
An additional context for adding LaTeX formula rendering support via MathJax wraps around the `<div/>` element that contains the UI. The `<App/>` component is the central part of the simulator. It stores user-defined and calculated variables in the shared context, creates the UI, and populates it with elements such as the box components or the status bar. The major components that `<App/>` adds are:

- The `<InfoOverlay/>` component, that is drawn on top of the other UI elements while `isOverlayHidden` is `false`.
- A container for the automation components (`<SweepBox/>` and `<JitterBox/>`) and the `<Box/>`-based input and output variable components. Depending on the current UI settings, these components can contain:
  - Canvas-based visualizations from `Visualizations.js`
  - LaTeX style formulas via `Formula.js`
- The `<Simulator/>` component and nested sub-components to render the 3D cavity view. We used conditionally rendered, CSS-based graphics to implement the 2D cavity view directly in the `<App/>` component.

The final JS file is `utilities.js`. It contains helper functions, for example `deg2rad()` that converts degrees to radians or `useInterval()` which repeatedly executes a supplied callback followed by a delay for the automation components.

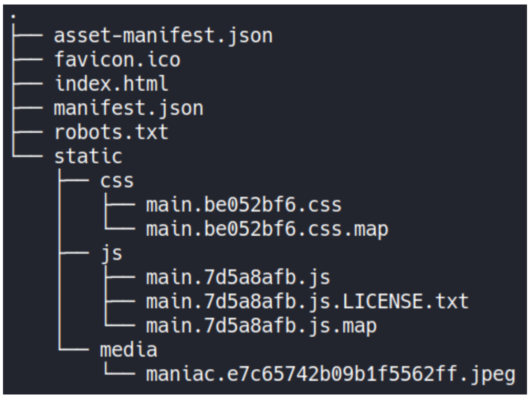
`.scss` files contain Sass[8], a CSS extension that is compiled to plain CSS in the deployment step.

[8] <https://sass-lang.com/>



```
.
├── App.js
├── App.module.scss
├── Box.js
├── Box.module.scss
├── Formula.js
├── index.css
├── index.js
├── InfoOverlay.js
├── InfoOverlay.module.scss
├── JitterBox.js
├── Simulator
│   ├── components
│   │   ├── CameraController.js
│   │   ├── Mirror.js
│   │   └── Ray.js
│   ├── ctx
│   │   └── CavityContext.js
│   ├── scenes
│   │   └── CavityScene.js
│   ├── Simulator.js
│   └── Simulator.module.scss
├── SweepBox.js
├── utilities.js
├── Visualizations.js
└── Visualizations.module.scss
```

Figure 52: `tree` of the `src` (source) folder, the project's structure



```
.
├── asset-manifest.json
├── favicon.ico
├── index.html
├── manifest.json
├── robots.txt
└── static
    ├── css
    │   ├── main.be052bf6.css
    │   └── main.be052bf6.css.map
    ├── js
    │   ├── main.7d5a8afb.js
    │   ├── main.7d5a8afb.js.LICENSE.txt
    │   └── main.7d5a8afb.js.map
    └── media
        └── maniac.e7c65742b09b1f5562ff.jpeg
```

Figure 53: `tree` of the `build` folder, the output from the compilation step

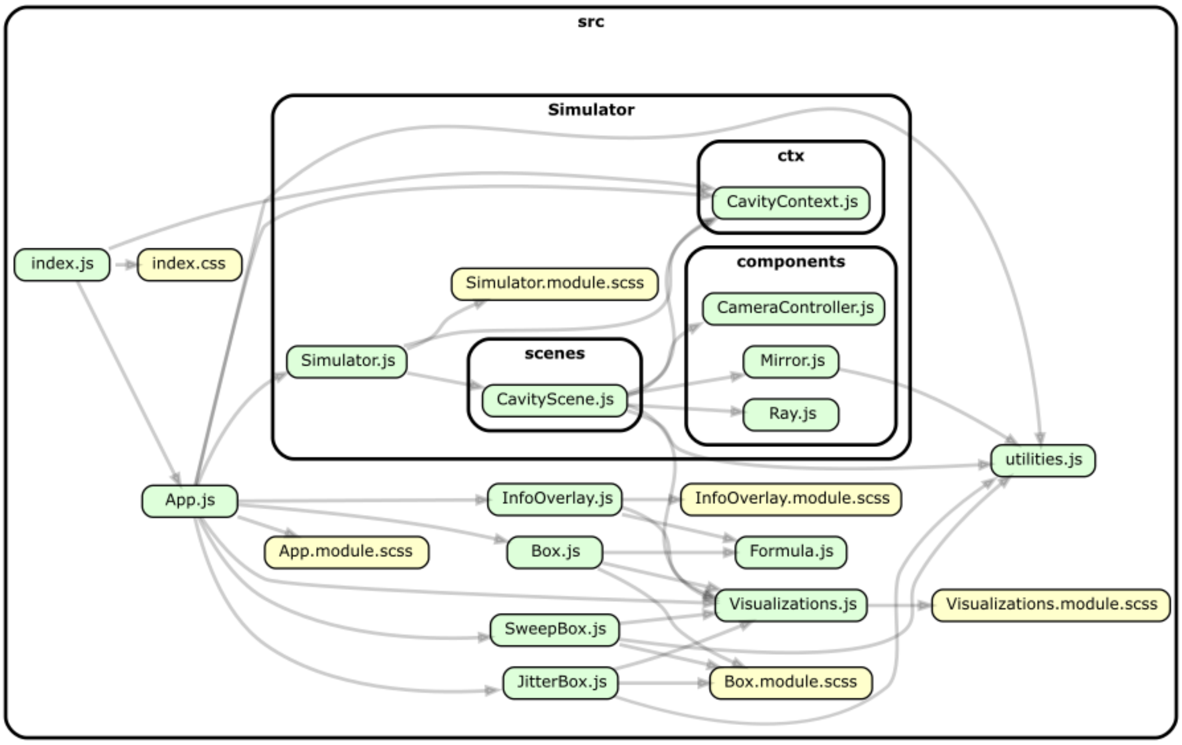


Figure 54: the simulator's **dependency graph**, nodes with green background (`.js`) depict JS files, other nodes (`.css`, `.scss`) represent stylesheets (figure based on graph generated by dependency-cruiser [86])

4.5.2 Implementation Details and Extensibility

This chapter outlines the life cycle of one exemplary component, the **frequency component**. See figure 55 on page 80 for a depiction of the visualization through a range of sample values. This section additionally includes hints about extending the app.

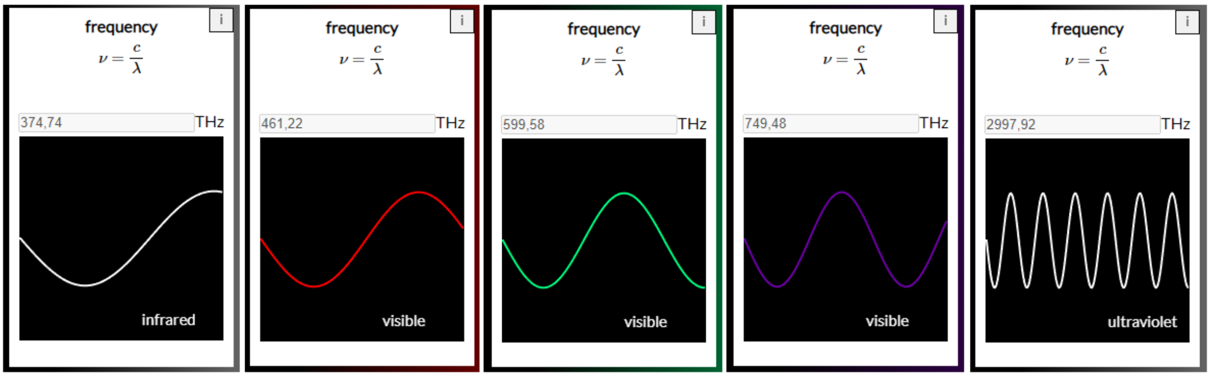


Figure 55: **frequency visualization range** (wavelengths from left to right: 800, 650, 500, 400, and 100 nanometers)

The call sequence starts at `index.js` (listing 4) where the <**App**/> component is imported (not shown) and the root element created:

```
// stripped imports.
// ...
const root = ReactDOM.createRoot(document.getElementById('root'))
root.render(
  <React.StrictMode>
    <CavityProvider>
      <App />
    </CavityProvider>
  </React.StrictMode>
)
```

Listing 4: `index.js` excerpt

`StrictMode` enables additional checks during development. The <**CavityProvider**/> from `Simulator/ctx/CavityContext.js` holds the shared variables including those used for the frequency and wavelength (listing 5):

Next, we create a context and a provider (lines 1-2). Components subscribe to a context and read values from the next provider up the tree. [59] The `useState()` hook returns an array that we destructure into a variable and its setter (lines 4-5). The `wavelength` is specified as

```
1  export const CavityContext = createContext({})
2  export const CavityProvider = (props) => {
3    // configurable cavity parameters:
4    const [wavelength, setWavelength] = useState(200) // in nm.
5    const [wavelengthColor, setWavelengthColor] = useState({})
6    // ...
7    // calculated variables:
8    const [frequency, setFrequency] = useState(0)
9    // ...
10   const value = useMemo(
11     () => ({
12       wavelengthColor,
13       setWavelengthColor,
14       // ...
15 return <CavityContext.Provider value={value} {...props} />
16 }
17 export default CavityProvider
```

Listing 5: `CavityContext.js` excerpt

200 nm by default. The app initializes it to this when it loads. The `wavelengthColor` will be calculated in a later step. Our provider **memoizes** its stored values (line 10). A memoized function stores all the values it calculates to speed up future executions with the same input. Finally, line 15 attaches the values to the provider, and line 16 exports it so `index.js` can use it.

With a prepared provider, the `root.render()` function in `index.js` creates the `<App/>` component from `App.js` (listing 6):

Like all our components, we implemented `<App/>` as a functional component (line 2). Unfortunately, Math.js does not include the speed of light in its list of constants, so we define it ourselves (line 3). Next, we fetch the values and the appertaining setters from the context (lines 5-8).

Lines 10 through 13 contain a `useEffect()` hook. We pass two arguments to it: an anonymous function (whose body spans lines 11-12) and an array of variables (containing only `wavelength`) to watch for (line 13). Each time `wavelength` changes, the anonymous function is called. This function will in turn call two further functions: `setWavelengthColor()` and `changeFavicon()`.

```
1  // stripped imports and the dynamic favicon function.
2  function App() {
3    const c = 299792458 // speed of light in vacuum, m/s.
4    // configurable variables:
5    const { wavelength, setWavelength } = useContext(CavityContext) // in nm.
6    const { wavelengthColor, setWavelengthColor } = useContext(CavityContext)
7     // calculated variables:
8    const { frequency, setFrequency } = useContext(CavityContext)
9    // ...
10   useEffect(() => {
11     setWavelengthColor(wavelength2rgb(wavelength))
12     changeFavicon(wavelength)
13   }, [wavelength])
14   // ...
15     useEffect(() => {
16     setWavenumber((2 * Math.PI) / wavelength)
17     setFrequency(Math.round((c / wavelength) * 1e9))
18   }, [wavelength])
19   // ...
```

Listing 6: `App.js` excerpt, part 1/2, `useEffect`s

The latter function updates the favicon (the app's bookmark/tab icon) to show the current wavelength's color, a little easter egg I have just ruined. The former function updates the `wavelengthColor` with its setter; both are stored in the context. To do this it uses the helper function `wavelength2rgb()` from `Visualizations.js`, that takes a wavelength and returns a JS object with red, green and blue values of that wavelength.

Another `useEffect()` (lines 15-18) finally calculates the frequency from the wavelength using the constant for the speed of air in vacuum and the formula $\nu = \frac{c}{\lambda}$. Additionally, it calculates the angular wavenumber, which relies on the wavelength too and thus does not warrant a separate `useEffect()`. Again, every time the variable in the dependency array updates (`wavelength`), these functions are called. However, calls with the same input will return memoized (cached) results.

Any functional component returns a React element. In the case of `<App/>` this is the UI in the form of JSX. See line 3 in listing 7 on page 84. Before we include any part of the UI, we add the `<MathJaxContext>` as an outer wrapper (opens in line 4 and closes in line 38). This context enables the use of the `<MathJax/>` component to render formulas in child elements of the context.

Line 6 and 7 open elements that hold the app and the list of box components. Line 10 adds the frequency component, and line 21 the wavelength component. Both are boxes. Each `<Box>` can have several props:

- `label` shows up in the title of the component
- `isResult` (line 14) specifies that a box is an output value, as is the case for frequency.
- `setF` is the setter (required for input components) and `value` the value of the input field
  - The `value` of the frequency component (line 17) converts Hz to THz and rounds it to two decimal places.
- If `hideCanvas` is **`false`**, the component from `canvasplot` is drawn. Similarly, `showDetails` defines if formulas and info overlay buttons are visible. If the plot uses the color of the wavelength, it requires `rgb`
- `formula` is for the displayed LaTeX, which needs escaped backslashes.
- `unit`'s value shows up to the right of the input field.
- `min`, `max` and `step` set the lower and upper bound of the slider and the step size.

Everything covered so far should be enough for getting started with creating custom input/output components or modifying existing ones. Here is a quick summary:

1. Add new variables to the context. (`CavityContext.js`)
2. Fetch the variable and its setter from the provider. (`App.js`)
3. Write a new `useEffect()` or extend an existing one. (`App.js`)
4. Add a `<Box>` component and configure it. (`App.js`)

Writing a custom **visualization** is a bit more involved. Listing 8 on page 86 show the first part of the `<Wavelength>` Visualization component. Line 2 creates the reference `ref` with the `useRef()` hook. The value returned from this hook is attachable to the component's output, the JSX. Doing this lets the component fetch its DOM element at runtime to access the canvas' and the 2D drawing context (lines 4-5).

The component has a `useEffect()` (lines 3-39) that executes when the wavelength changes. After obtaining the 2D drawing `context`, it is cleared (line 7). We set the color for drawing the text to white and the font size to a fraction of the canvas width so that it remains readable independently of the canvas scale. This pattern repeatedly finds application in the discussed source code. Line 10 initializes the `text` variable, which we set according to the frequency band the current wavelength fits into (lines 12 through 20). Finally, we draw the text onto the canvas (line 22).

The remaining lines are all about drawing the sine wave. We set the width and color of the

```jsx
// ...

return (
    <MathJaxContext>
      // stripped GitHub ribbon and info overlay.
      <div className={`${styles.App}`}>
        <div style={containerStyle}
             className={`variable-wrapper ${styles.container}`} >
        // ...
          <Box
            label="frequency"
            rgb={wavelengthColor}
            hideCanvas={!showvisualizations}
            isResult
            formula={`\\(\\nu = \\dfrac{c}{\\lambda}\\)`}
            unit="THz"
            value={Math.round((frequency / 1e12) * 100) / 100}
            canvasplot={<Wavelength wavelength={wavelength} />}
            showDetails={showdetails}
          />
          <Box
            label="wavelength"
            rgb={wavelengthColor}
            hideCanvas={!showvisualizations}
            min="1"
            formula={`\\(\\lambda\\)`}
            max="1000"
            step="1"
            unit="nm"
            value={wavelength}
            canvasplot={<Wavelength wavelength={wavelength} />}
            setF={(e) => setWavelength(e.target.value)}
            showDetails={showdetails}
          />
        // ...
      </div>
      // stripped buttons, status bar, and cavity view.
    </MathJaxContext>
  )
}
export default App
```

Listing 7: `App.js` excerpt, part 2/2, JSX

line, calculate a frequency (with a scale that fits nicely into the drawing area) and start our drawing path (lines 22-26). A path consists of one to several line segments we can draw onto the canvas in a separate step. The **for** loop's code block executes once for each pixel on the horizontal axis of the canvas, adding a new line segment each iteration:

- First we calculate `y` (lines 30-34), the next line segment's height.
  - The current `x` value divided by the width of the canvas element in pixels is the variable part of the input for the sine wave.
  - The `frequency` variable introduces the scaling factor `scaleWavelength`. It has a value of $632nm$, the wavelength of a HeNe laser. When a user selects this wavelength, the simulator will draw exactly one waveform in this component.
  - The result of the sine function is scaled to a maximum height of 50% of the canvas height (adjusting for the width of the drawn line) because it offers a nice visual balance with a bit of headroom (lines 32-34).
  - The final term, the addition of half the canvas' height, centers the sine wave vertically (line 34).
- Finally, we update `x` and append the line segment (line 35-36).

Line 38 draws the path we started before the **for** loop. This step completes the calculations for the current frame of the visualization. When the wavelength changes the process repeats.

```
const Wavelength = (props) => {
  const ref = useRef(null)
  useEffect(() => {
    var canvas = ref.current
    var context = canvas.getContext('2d')
    const scaleWavelength = 632
    context.clearRect(0, 0, canvas.width, canvas.height)
    context.fillStyle = 'white'
    context.font = `${Math.round(canvas.width / 13)}px Lato`
    var text = ''

    if (props.wavelength < 11) {
      text = 'x-ray'
    } else if (props.wavelength < 380) {
      text = 'ultraviolet'
    } else if (props.wavelength < 750) {
      text = 'visible'
    } else {
      text = 'infrared'
    }

    context.fillText(text, (canvas.width / 5) * 3, (canvas.height / 13) * 12)
    context.lineWidth = 2
    context.strokeStyle = rgb2string(wavelength2rgb(props.wavelength))
    var frequency = (1 / props.wavelength) * scaleWavelength
    context.beginPath()

    var x = 0
    for (var i = 0; i < canvas.width; i++) {
      const y =
        ((Math.sin((x / canvas.width) * frequency * 2 * Math.PI) *
          (canvas.height / 2)) /
          (100 + context.lineWidth * 4)) *
         50 + canvas.height / 2
      x = i
      context.lineTo(x, y)
    }
    context.stroke()
  }, [props.wavelength])
// ...
```

Listing 8: `Visualizations.js` `Wavelength` excerpt, part 1/2, `useEffect`

The <**Wavelength**> functional component returns JSX: a wrapping <**div**> containing the HTML <**canvas**> element with the discussed `ref` attribute (line 7 in listing 9). `className` is equivalent to HTML's `class` attribute, which is a reserved keyword in JS and is thus barred from JSX.

```
1  // ...
2    return (
3      <div>
4        <canvas
5          width={200}
6          height={200}
7          ref={ref}
8          className={styles.small_visualization}
9        ></canvas>
10     </div>
11   )
12 }
13 // ...
```

Listing 9: `Visualizations.js` `Wavelength` excerpt, part 2/2, JSX

5 Summary

This chapter discusses the produced optical cavity simulator, points out the potential for future work, and draws a conclusion on the potential impact of quantum cryptology on the IT security landscape.

5.1 Discussion

This master's thesis provides two contributions with the aim of improving the understanding of the fundamentals of quantum cryptology. The first part is a theoretical introduction for IT security engineers to quantum cryptology concepts and the context and relevance of the field for IT security. These sections include related research fields and state of the art of single-photon sources. The second part is the developed optical cavity simulator.

We set out to meet a catalog of requirements for the simulator. What follows is an itemization of the requirements and their compliance states:

1. *The application should support a wide range of devices to facilitate a broad user base: it should support desktop and mobile devices.*
   - We achieved this requirement by basing the optical cavity simulator on modern web technologies. This choice enables the simulator to run on devices with modern browsers, including desktop computers, tablets, and mobile phones. In addition, the ability to disable visualizations further increases the range of viable devices for those not supporting hardware acceleration (for 2D and 3D graphics) or those with limited screen real estate.
2. *Installation and update procedures should impose as few requirements as feasible: this encompasses operating system restrictions and access permissions.*
   - Similarly, the simulator passes the second requirement due to its implementation as an SPA. A web server hosts the application to facilitate easy deployment with minimal permission requirements on the client side. The client does not need to communicate further after the initial fetching of the SPA. Both offline uses via a saved version or through the version in the browser's cache enable the use of all features. The update process is identical; the user refreshes the page to retrieve the latest version from the server. The simulator distributes updated values to corresponding components on the fly.

3. *Variables included in the calculations should be adjustable and provide visualizations that automatically update on changes. In addition, the chosen technology should offer support for both 2D and 3D graphics.*
    - We partially fulfill this requirement since the simulator does not provide visualizations for each component. The two components with missing visualizations are the one for the angular wavenumber and the one for the cavity finesse. The remaining visualizations employ a combination of the fundamental web technologies (such as the canvas API for 2D graphics) and modern libraries (including React for creating the UI plus the state management and Three.js for displaying a 3D view of the cavity).
4. *Performed calculations should be transparent to the user.*
    - The simulator provides the option to display info overlays, implemented formulas, and the corresponding unit signs for each component that sets or calculates a value (input and output variable components). Additionally, we have selected visualizations that we personally found helpful in grasping the related concept. Unfortunately, due to practical constraints, this paper cannot comprehensively review whether other IT security professionals find the final product helpful in learning about quantum cryptography and the underlying concepts. Determining the didactic impact of the tool is outside of our area of expertise.
5. *The simulator should be able to adjust a value without relying on constant user input.*
    - The simulator fulfills this requirement. Automation components can adjust a variable without user input besides the initial configuration and initiation. Multiple automation components can run simultaneously.
6. *The application should be extensible and accessible to new software engineers to simplify and foster further development*
    - We took great care in the design, architecture, and the chosen tech stack of the optical cavity simulator to foster extensibility and a low entry burden. Additionally, this thesis includes notes about how to add new components to the simulator. Finally, we chose a license that guarantees that any modifications must remain open to the public. It is beyond the scope of this study to produce precise metrics about the extensibility of the project.

5.2 Future Work

There are several improvements and further developments one could make to the current version of the optical cavity simulator:

- Neither the cavity finesse component nor the angular wavenumber component currently has a visualization.
    - The angular wavenumber component lacks this feature because of the tiny variation

of the values in relation to the distance 2π (an entire waveform) spans. On the one hand, this would be an imperceptible change unless greatly exaggerated, at least when looking at the visible spectrum. However, on the other hand, a not-to-scale visualization could result in confusion about the true extent of the range.
  - A visualization for the finesse would either require the splitting of the input values for the finesse into two other components, the free spectral range ($\delta\nu_{FSR}$) and the full width at half maximum ($\delta\nu_{Airy}$), with corresponding visualizations each, or a set of static plots that serve as comparisons. As the cavity finesse does not serve as an input for further calculations, we focused our attention elsewhere.
- The visualization's main plot colors (green for input or primary value and red for output or secondary value), while easy to understand and readily distinguishable for most people, are also two colors affected by protanopia and deuteranopia [120], the most common forms of inherited colorblindness. There should be an option to adjust these colors (or plot the waves in a different style, such as one dotted and one solid).
  - There is ample room for further configuration options, be it colors, fonts, canvas sizes, or a customizable layout.
- Different speeds of the light in the cavity space influence the wavelength to frequency conversion. A drop-down menu containing common materials (air, vacuum, et cetera) in the frequency component or an input field for custom values could be helpful to simulate a broader range of optical cavity setups.
- The resolutions of the canvas elements are fixed. This simplification is less of an issue for most desktop devices, but mobile phones trend towards more pixels per inch, resulting in blurry canvas content.
- A mode where the light source is in the cavity could be helpful to illustrate a setup closer to those used in microcavity-based constructions. This model could also display the states of the emitter and the spontaneous emission following the absorption, perhaps with an emitter component with an "excite" button or triggered by another component representing the classical interface. A single-photon detector component could inform the user about photons coupled to the cavity via the Purcell effect.
- The 3D cavity view could use ray-casting and curved mirrors to illustrate the optical path differences through plane-parallel, concentric and confocal cavities.

5.3 Conclusion

The potential impact of quantum cryptography and quantum cryptanalysis on the IT security field manifests twofold: they can pose significant risks that require mitigation and offer opportunities for novel solutions simultaneously.

Foremost, risks include quantum algorithms that operate in entirely different complexity classes than classical algorithms solving the same problems. This risk is especially apparent in the branch of asymmetric cryptography that relies on trapdoor functions, primarily the DLP and factorization, both solvable in polynomial time by quantum algorithm, assuming a sufficiently advanced quantum computer. Recent advancements coupled with current funding [21] by major entities have made this scenario increasingly likely [109, 30].

Opportunities include quantum key distribution algorithms such as BB84 that, barring implementation errors, offer mathematically proven secure communication over insecure channels. In addition, with the miniaturization of single-photon sources, the possibility of scaling and, thus, widespread adoption increases.

Improving the outcome of both risks and opportunities requires IT security personnel with a fundamental understanding of the underlying quantum mechanical principles. These principles are probabilistic, unlike the (for IT professionals) familiar deterministic classical computing model. Should quantum technologies continue their pace of improvement, they will impact the field of IT security, which will require suitably trained specialists. However, even before quantum supremacy for cryptanalysis happens, the field must prepare for the possibility of an attacker already recording encrypted traffic to break the cryptography once technology permits.

The main goal of the thesis and the accompanying project was to develop an optical cavity simulator with the potential to be used as part of the didactic process for IT security engineers. The created simulator fulfills a majority of the specified requirements, including supporting a wide range of devices useful for classroom settings. However, if the need arises, measures taken to improve extensibility should facilitate further prototype development. To answer the remaining requirements and the scientific question (*How to train IT security engineers in the foundations of quantum cryptography?*), future studies on the current topic are therefore recommended. The developed optical cavity can serve as a base for future studies involving quantum technology experts, for example, as part of a seminar for IT security professionals undergoing relevant training.

Bibliography

[1] Syed Affan Ahmed, Mujahid Mohsin, and Syed Muhammad Zubair Ali. “Survey and technological analysis of laser and its defense applications”. In: *Defence Technology* 17.2 (Apr. 2021), pp. 583–592. ISSN: 2214-9147. DOI: 10.1016/J.DT.2020.02.012.

[2] George Biddell Airy. “On the Intensity of Light in the neighbourhood of a Caustic”. In: *Transactions of the Cambridge Philosophical Society* 6 (1838), p. 379. URL: https://ui.adsabs.harvard.edu/abs/1838TCaPS...6..379A/abstract.

[3] Elizabeth Allen and Sophie Triantaphillidou. *The Manual of Photography and Digital Imaging*. en. 10th ed. Oxford, England: Focal Press, 2010. ISBN: 978-0240520377.

[4] Apple. “App Review - App Store - Apple Developer”. URL: https://developer.apple.com/app-store/review/. *(visited on 16-09-2022)*.

[5] Charles H. Bennett and Gilles Brassard. “Quantum cryptography: Public key distribution and coin tossing”. In: *Theoretical Computer Science* 560.P1 (Mar. 2020), pp. 7–11. DOI: 10.1016/j.tcs.2014.05.025. URL: https://arxiv.org/abs/2003.06557v1.

[6] BSI. “BSI - Post-Quanten-Kryptografie”. URL: https://www.bsi.bund.de/DE/Themen/Unternehmen-und-Organisationen/Informationen-und-Empfehlungen/Quantentechnologien-und-Post-Quanten-Kryptografie/Post-Quanten-Kryptografie/post-quanten-kryptografie_node.html. *(visited on 10-04-2022)*.

[7] Adrian Cho. “Google claims quantum computing milestone”. In: *Science* 365.6460 (Sept. 2019), p. 1364. ISSN: 10959203. DOI: 10.1126/SCIENCE.365.6460.1364.

[8] Adrian Cho. “Ordinary computer matches Google’s quantum computer”. In: *Science* 377.6606 (Aug. 2022), pp. 563–564. ISSN: 10959203. DOI: 10.1126/SCIENCE.ADE2360.

[9] John F. Clauser. “Experimental distinction between the quantum and classical field-theoretic predictions for the photoelectric effect”. In: *Physical Review D* 9.4 (Feb. 1974), p. 853. ISSN: 05562821. DOI: 10.1103/PhysRevD.9.853. URL: https://journals.aps.org/prd/abstract/10.1103/PhysRevD.9.853.

[10] “CoffeeScript”. URL: https://coffeescript.org/. *(visited on 15-09-2022)*.

[11] David Deutsch and Richard Jozsa. “Rapid solution of problems by quantum computation”. In: *Proceedings of the Royal Society of London. Series A: Mathematical and Physical Sciences* 439.1907 (Dec. 1992), pp. 553–558. ISSN: 0962-8444. DOI: 10.1098/RSPA.1992.0167. URL: https://royalsocietypublishing.org/doi/10.1098/rspa.1992.0167.

[12] Wolfgang Demtröder. “Experimentalphysik 1 Mechanik und Wärme”. In: *Experimentalphysik 1* (2008). ISSN: 0937-7433. DOI: https://doi.org/10.1007/978-3-662-54847-9.

[13] Wolfgang Demtröder. *Laserspektroskopie 1 Grundlagen*. 2011. DOI: 10.1007/978-3-642-21306-9.

[14] Whitfield Diffie and Martin E Hellman. “New Directions in Cryptography”. In: *IEEE Transactions on Information Theory* 22.6 (1976), pp. 644–655.

[15] Ben Eaton. “Single-Page Applications: A Comprehensive Guide - KeyCDN”. URL: https://www.keycdn.com/blog/single-page-application. *(visited on 13-09-2022)*.

[16] A. Einstein. “Über einen die Erzeugung und Verwandlung des Lichtes betreffenden heuristischen Gesichtspunkt”. In: *Annalen der Physik* 322.6 (Jan. 1905), pp. 132–148. ISSN: 1521-3889. DOI: 10.1002/ANDP.19053220607. URL: https://onlinelibrary.wiley.com/doi/full/10.1002/andp.19053220607.

[17] A. Einstein. “Zur Quantentheorie der Strahlung”. In: *Physikalische Gesellschaft Zürich. Mitteilungen.* 16 (Mar. 1916), pp. 47–62. DOI: https://doi.org/10.1515/9783112596609-016.

[18] M. D. Eisaman et al. “Invited Review Article: Single-photon sources and detectors”. In: *Review of Scientific Instruments* 82.7 (July 2011), p. 071101. ISSN: 0034-6748. DOI: 10.1063/1.3610677. URL: https://aip.scitation.org/doi/abs/10.1063/1.3610677.

[19] Artur K. Ekert. “Quantum cryptography based on Bell’s theorem”. In: *Physical Review Letters* 67.6 (Aug. 1991), p. 661. ISSN: 00319007. DOI: 10.1103/PhysRevLett.67.661. URL: https://journals.aps.org/prl/abstract/10.1103/PhysRevLett.67.661.

[20] C J Foot. *Atomic physics*. Oxford New York: Oxford University Press, 2005. ISBN: 978-0198506966.

[21] François Candelon et al. “Here’s what it will take to win the quantum computing arms race | Fortune”. URL: https://fortune.com/2022/09/02/quantum-computing-cryptography-companies-arms-race/. *(visited on 15-09-2022)*.

[22] Free Software Foundation. “The GNU General Public License v3.0 - GNU Project - Free Software Foundation”. URL: https://www.gnu.org/licenses/gpl-3.0.en.html. *(visited on 12-04-2022)*.

[23] “Fundamentals - three.js manual”. URL: https://threejs.org/manual/#en/fundamentals. *(visited on 12-09-2022)*.

[24] Tommaso Gagliardoni. “Quantum Attack Resource Estimate: Using Shor’s Algorithm to Break RSA vs DH/DSA VS ECC – Kudelski Security Research”. URL: https://research.kudelskisecurity.com/2021/08/24/quantum-attack-resource-estimate-using-shors-algorithm-to-break-rsa-vs-dh-dsa-vs-ecc/. *(visited on 08-07-2022)*.

[25] Xiangyu Gao et al. “Piezoelectric Actuators and Motors: Materials, Designs, and Applications”. In: *Advanced Materials Technologies* 5.1 (Jan. 2020), p. 1900716. ISSN: 2365-709X. DOI: 10.1002/ADMT.201900716. URL: https://onlinelibrary.wiley.com/doi/full/10.1002/admt.201900716.

[26] Nicolas Gisin et al. “Quantum cryptography”. In: *Reviews of modern physics* 74.1 (2002), p. 145.

[27] GitHub. “About npm | npm Docs”. URL: https://docs.npmjs.com/about-npm. *(visited on 12-09-2022)*.

[28] Steven Goldfeder. *Bitcoin and Cryptocurrency Technologies*. Princeton, NJ: Princeton University Press, July 2016. ISBN: 978-0691171692.

[29] Lov K. Grover. “A fast quantum mechanical algorithm for database search”. In: *Proceedings of the Annual ACM Symposium on Theory of Computing* Part F1294 (July 1996), pp. 212–219. ISSN: 07378017. DOI: 10.1145/237814.237866.

[30] Helena Handschuh. “Security Implications Of Quantum Computing”. URL: https://semiengineering.com/security-implications-of-quantum-computing/. *(visited on 17-09-2022)*.

[31] Ulrich Harten. “Physik, Eine Einführung für Ingenieure und Naturwissenschaftler”. In: Springer-Lehrbuch (2012). DOI: 10.1007/978-3-642-19979-0. URL: http://link.springer.com/10.1007/978-3-642-19979-0.

[32] Jeff Hecht. “Beam: The race to make the laser”. en. In: *Opt. Photonics News* 16.7 (2005), p. 24. DOI: http://dx.doi.org/10.1364/OPN.16.7.000024.

[33] W. Heisenberg. “Über den anschaulichen Inhalt der quantentheoretischen Kinematik und Mechanik”. In: *Zeitschrift für Physik 1927 43:3* 43.3 (Mar. 1927), pp. 172–198. ISSN: 14346001. DOI: 10.1007/BF01397280. URL: https://link.springer.com/article/10.1007/BF01397280.

[34] Helge Klein. “Impact of GPU Acceleration on Browser CPU Usage • Helge Klein”. URL: https://helgeklein.com/blog/impact-gpu-acceleration-browser-cpu-usage/. *(visited on 16-09-2022)*.

[35] Matthias Homeister. *Quantum Computing verstehen*. Computational Intelligence. Wiesbaden: Springer Fachmedien Wiesbaden, 2018. ISBN: 978-3-658-22883-5. DOI: 10.1007/978-3-658-22884-2. URL: http://link.springer.com/10.1007/978-3-658-22884-2.

[36] A. Huelsing et al. “XMSS: eXtended Merkle Signature Scheme”. In: (May 2018). ISSN: 2070-1721. DOI: 10.17487/RFC8391. URL: https://www.rfc-editor.org/info/rfc8391.

[37] Won Young Hwang. “Quantum Key Distribution with High Loss: Toward Global Secure Communication”. In: *Physical Review Letters* 91.5 (Aug. 2003), p. 057901. ISSN: 10797114. DOI: 10.1103/PhysRevLett.91.057901. URL: https://journals.aps.org/prl/abstract/10.1103/PhysRevLett.91.057901.

[38] IBM. “Shor’s algorithm - IBM Quantum”. URL: https://quantum-computing.ibm.com/composer/docs/iqx/guide/shors-algorithm. *(visited on 13-08-2022)*.

[39] “Introduction — WebAssembly 2.0 (Draft 2022-09-01)”. URL: https://webassembly.github.io/spec/core/intro/introduction.html#design-goals. *(visited on 15-09-2022)*.

[40] Nur Ismail et al. “Fabry-Pérot resonator: spectral line shapes, generic and related Airy distributions, linewidths, finesses, and performance at low or frequency-dependent reflectivity”. In: *Optics Express* 24.15 (July 2016), p. 16366. ISSN: 10944087. DOI: 10.1364/OE.24.016366.

[41] J S Lundeen. “File:Spontaneous Parametric Downconversion.png - Wikimedia Commons”. URL: https://commons.wikimedia.org/wiki/File:Spontaneous_Parametric_Downconversion.png. *(visited on 08-14-2022)*.

[42] Boyuan Jin, Dhananjay Mishra, and Christos Argyropoulos. “Efficient single-photon pair generation by spontaneous parametric down-conversion in nonlinear plasmonic metasurfaces”. In: *Nanoscale* 13.47 (Dec. 2021), pp. 19903–19914. ISSN: 2040-3372. DOI: 10.1039/D1NR05379E. URL: https://pubs.rsc.org/en/content/articlehtml/2021/nr/d1nr05379e.

[43] Jos De Jong. “math.js | an extensive math library for JavaScript and Node.js”. URL: https://mathjs.org/. *(visited on 16-09-2022)*.

[44] Jos De Jong. “math.js | an extensive math library for JavaScript and Node.js”. URL: https://mathjs.org/docs/index.html. *(visited on 13-09-2022)*.

[45] David Kahn. *The Codebreakers: The Comprehensive History of Secret Communication from Ancient Times to the Internet*. 2nd ed. Simon & Schuster, 1996, pp. 93–94. ISBN: 978-0-684-83130-5.

[46] F.W. Kasiski. *Die Geheimschriften und die Dechiffrir-Kunst*. E. S. Mittler und Sohn, 1863.

[47] Auguste Kerckhoffs. “La cryptographie militaire”. In: *Journal des sciences militaires* IX (1883), pp. 5–38. URL: http://petitcolas.net/kerckhoffs/crypto_militaire_1.pdf.

[48] Manju Khari et al. “Securing Data in Internet of Things (IoT) Using Cryptography and Steganography Techniques”. In: *IEEE Transactions on Systems, Man, and Cybernetics: Systems* 50.1 (Jan. 2020), pp. 73–80. ISSN: 21682232. DOI: 10.1109/TSMC.2019.2903785.

[49] D N Klyshko. “Utilization of vacuum fluctuations as an optical brightness standard”. In: *Soviet Journal of Quantum Electronics* 7.5 (May 1977), pp. 591–595. ISSN: 0049-1748. DOI: 10.1070/QE1977V007N05ABEH012567.

[50] Paul Kocher et al. “Spectre Attacks: Exploiting Speculative Execution”. In: *Communications of the ACM* 63.7 (Jan. 2018), pp. 93–101. ISSN: 15577317. DOI: 10.48550/arxiv.1801.01203. URL: https://arxiv.org/abs/1801.01203v1.

[51] Helge Kragh. “Niels Bohr and the Quantum Atom: The Bohr Model of Atomic Structure 1913–1925”. In: *Niels Bohr and the Quantum Atom: The Bohr Model of Atomic Structure 1913-1925* 9780199654987 (May 2012), pp. 1–416. DOI: 10.1093/ACPROF:OSO/9780199654987.001.0001. URL: https://academic.oup.com/book/5807.

[52] Axel Kuhn and Daniel Ljunggren. “Cavity-based single-photon sources”. In: *Contemporary Physics* 51.4 (July 2010), pp. 289–313. ISSN: 00107514. DOI: 10.1080/00107511003602990. URL: https://www.tandfonline.com/doi/abs/10.1080/00107511003602990.

[53] Moritz Lipp et al. “Meltdown”. In: *World Watch* 9.3 (Jan. 2018), pp. 23–31. ISSN: 08960615. DOI: 10.48550/arxiv.1801.01207. URL: https://arxiv.org/abs/1801.01207v1.

[54] Xiongfeng Ma et al. “Practical decoy state for quantum key distribution”. In: *Physical Review A - Atomic, Molecular, and Optical Physics* 72.1 (July 2005), p. 012326. ISSN: 10502947. DOI: https://doi.org/10.1103/PhysRevA.72.012326. URL: https://journals.aps.org/pra/abstract/10.1103/PhysRevA.72.012326.

[55] Mateusz Masiowski et al. “Quantum computing talent not on pace with funding | McKinsey”. URL: https://www.mckinsey.com/business-functions/mckinsey-digital/our-insights/quantum-computing-funding-remains-strong-but-talent-gap-raises-concern. *(visited on 10-09-2022)*.

[56] Robert J. McEliece. “A Public-Key Cryptosystem Based On Algebraic Coding Theory - NASA/ADS”. In: *DSN Progress Report* 44 (1978), pp. 114–116. URL: https://ui.adsabs.harvard.edu/abs/1978DSNPR..44..114M/abstract.

[57] D. McGrew, M. Curcio, and S. Fluhrer. “Leighton-Micali Hash-Based Signatures”. In: (Apr. 2019). ISSN: 2070-1721. DOI: 10.17487/RFC8554. URL: https://www.rfc-editor.org/info/rfc8554.

[58] Benjamin Médicke. “bmedicke/amalthea: collection of Jupyter Notebooks”. URL: https://github.com/bmedicke/amalthea. *(visited on 08-04-2022)*.

[59] Meta Platforms. “Context – React”. URL: https://reactjs.org/docs/context.html. *(visited on 07-09-2022)*.

[60] Meta Platforms. “Create React App”. URL: https://create-react-app.dev/. *(visited on 12-09-2022)*.

[61] Meta Platforms. “Getting Started – React”. URL: https://reactjs.org/docs/getting-started.html. *(visited on 17-04-2022)*.

[62] Microsoft. “TypeScript”. URL: https://www.typescriptlang.org/. *(visited on 15-09-2022)*.

[63] Peter J. Mosley et al. “Heralded generation of ultrafast single photons in pure quantum states”. In: *Physical Review Letters* 100.13 (Apr. 2008), p. 133601. ISSN: 00319007. DOI: https://doi.org/10.1103/PhysRevLett.100.133601. URL: https://journals.aps.org/prl/abstract/10.1103/PhysRevLett.100.133601.

[64] Mozilla Corporation. “Canvas API - Web APIs | MDN”. URL: https://developer.mozilla.org/en-US/docs/Web/API/Canvas_API. *(visited on 12-09-2022)*.

[65] Mozilla Corporation. “Document Object Model (DOM) - Web APIs | MDN”. URL: https://developer.mozilla.org/en-US/docs/Web/API/Document_Object_Model. *(visited on 12-09-2022)*.

[66] Mozilla Corporation. “SPA (Single-page application) - MDN Web Docs Glossary: Definitions of Web-related terms | MDN”. URL: https://developer.mozilla.org/en-US/docs/Glossary/SPA. *(visited on 15-09-2022)*.

[67] Mozilla Corporation. “The web and web standards - Learn web development | MDN”. URL: https://developer.mozilla.org/en-US/docs/Learn/Getting_started_with_the_web/The_web_and_web_standards#html_css_and_javascript. *(visited on 16-09-2022)*.

[68] Mozilla Corporation. “WebGL: 2D and 3D graphics for the web - Web APIs | MDN”. URL: https://developer.mozilla.org/en-US/docs/Web/API/WebGL_API. *(visited on 16-09-2022)*.

[69] Lily Hay Newman. “Can You Trust NIST? - IEEE Spectrum”. URL: https://spectrum.ieee.org/can-you-trust-nist. *(visited on 08-04-2022)*.

[70] Michael A. Nielsen and Isaac L. Chuang. *Quantum Computation and Quantum Information: 10th Anniversary Edition*. Cambridge University Press, Dec. 2010. ISBN: 9780511976667. DOI: 10.1017/CBO9780511976667.

[71] NIST. “NIST Announces First Four Quantum-Resistant Cryptographic Algorithms | NIST”. URL: https://www.nist.gov/news-events/news/2022/07/nist-announces-first-four-quantum-resistant-cryptographic-algorithms. *(visited on 10-04-2022)*.

[72] NIST. “Post-Quantum Cryptography | CSRC”. URL: https://csrc.nist.gov/projects/post-quantum-cryptography/post-quantum-cryptography-standardization. *(visited on 08-04-2022)*.

[73] OpenJS Foundation and Node.js contributors. “About | Node.js”. URL: https://nodejs.org/en/about/. *(visited on 12-04-2022)*.

[74] Christof Paar and Jan Pelzl. *Understanding Cryptography*. Springer Berlin Heidelberg, 2010. ISBN: 978-3-642-04101-3. DOI: 10.1007/978-3-642-04101-3.

[75] Ioana Patringenaru. “New record set for cryptographic challenge”. URL: https://phys.org/news/2020-03-cryptographic.html. *PhysOrg (visited on 08-04-2022)*.

[76] Trevor Perrin. “The XEdDSA and VXEdDSA Signature Schemes”. URL: https://signal.org/docs/specifications/xeddsa/. *(visited on 14-08-2022)*.

[77] M Planck. *The Theory of Heat Radiation*. Blakiston, 1914.

[78] “Practical Cryptography: German Letter Frequencies”. URL: http://practicalcryptography.com/cryptanalysis/letter-frequencies-various-languages/german-letter-frequencies/. *(visited on 08-04-2022)*.

[79] John Preskill. “Quantum computing and the entanglement frontier”. In: *arXiv* (Mar. 2012). DOI: 10.48550/arxiv.1203.5813. URL: https://arxiv.org/abs/1203.5813v3.

[80] Renato Renner and Ramona Wolf. “Quantum Advantage in Cryptography”. In: *arXiv* (June 2022). DOI: 10.48550/arxiv.2206.04078. URL: https://arxiv.org/abs/2206.04078v1.

[81] E. Rescorla. “The Transport Layer Security (TLS) Protocol Version 1.3”. URL: https://www.rfc-editor.org/info/rfc8446. *(visited on 17-08-2022)*. ISSN: 2070-1721. DOI: 10.17487/RFC8446.

[82] R L Rivest, A Shamir, and L Adleman. “A Method for Obtaining Digital Signatures and Public-Key Cryptosystems”. In: *Communications of the ACM* 21.2 (1987), pp. 120–126.

[83] Guy Ropars et al. “A depolarizer as a possible precise sunstone for Viking navigation by polarized skylight”. In: *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences* 468.2139 (Mar. 2012), pp. 671–684. ISSN: 14712946. DOI: 10.1098/RSPA.2011.0369. URL: https://royalsocietypublishing.org/doi/10.1098/rspa.2011.0369.

[84] John D. Roth, Corey E. Lutton, and James Bret Michael. “Security Through Simplicity: A Case Study in Logical Segmentation Inference”. In: *Computer* 52.7 (July 2019), pp. 76–79. ISSN: 15580814. DOI: 10.1109/MC.2019.2906443.

[85] C. Ryan-Anderson et al. “Implementing Fault-tolerant Entangling Gates on the Five-qubit Code and the Color Code”. In: (Aug. 2022). DOI: 10.48550/arxiv.2208.01863. URL: https://arxiv.org/abs/2208.01863v1.

[86] Sander Verweij. “sverweij/dependency-cruiser”. URL: https://github.com/sverweij/dependency-cruiser. *(visited on 15-09-2022)*.

[87] Nicolas Sangouard et al. “Quantum repeaters based on atomic ensembles and linear optics”. In: *Reviews of Modern Physics* 83.1 (Mar. 2011), pp. 33–80. ISSN: 00346861. DOI: https://doi.org/10.1038/nphoton.2007.46. URL: https://journals.aps.org/rmp/abstract/10.1103/RevModPhys.83.33.

[88] Valerio Scarani et al. “The security of practical quantum key distribution”. In: *Reviews of modern physics* 81.3 (2009), p. 1301.

[89] A L Schawlow and C H Townes. “Infrared and Optical Masers”. In: *Phys. Rev.* 112.6 (1958), pp. 1940–1949.

[90] Bruce Schneier. *Applied cryptography: Protocols, algorithms, and source code in C*. en. 2nd ed. Nashville, TN: John Wiley & Sons, 1995.

[91] E. Schrödinger. “Quantisierung als Eigenwertproblem”. In: *Annalen der Physik* 384.4 (Jan. 1926), pp. 361–376. ISSN: 1521-3889. DOI: 10.1002/ANDP.19263840404.

[92] Andrew. Sears and Julie A. Jacko. *Human-computer interaction. Development process*. CRC Press, 2009, p. 337. ISBN: 9781420088908. URL: https://www.routledge.com/Human-Computer-Interaction-Development-Process/Sears-Jacko/p/book/9781420088908.

[93] Andrew J. Shields. “Semiconductor quantum light sources”. In: *Nanoscience and Technology: A Collection of Reviews from Nature Journals* (Jan. 2009), pp. 221–229. DOI: https://doi.org/10.1142/9789814287005.

[94] Peter W. Shor. “Algorithms for quantum computation: discrete logarithms and factoring”. In: *Proceedings 35th Annual Symposium on Foundations of Computer Science*. Nov. 1994, pp. 124–134. DOI: 10.1109/SFCS.1994.365700.

[95] Peter W. Shor. “Scheme for reducing decoherence in quantum computer memory”. In: *Physical Review A* 52.4 (Oct. 1995), R2493. ISSN: 10502947. DOI: 10.1103/PhysRevA.52.R2493. URL: https://journals.aps.org/pra/abstract/10.1103/PhysRevA.52.R2493.

[96] Daniel R. Simon. “On the Power of Quantum Computation”. In: *SIAM Journal on Computing* 26.5 (July 2006), pp. 1474–1483. ISSN: 00975397. DOI: https://doi.org/10.1137/S0097539796298637. URL: https://epubs.siam.org/doi/10.1137/S0097539796298637.

[97] Douglas C. Sinclair. “Choice of Mirror Curvatures for Gas Laser Cavities”. In: *Applied Optics, Vol. 3, Issue 9, pp. 1067-1071* 3.9 (Sept. 1964), pp. 1067–1071. ISSN: 2155-3165. DOI: 10.1364/AO.3.001067. URL: https://opg.optica.org/viewmedia.cfm?uri=ao-3-9-1067&seq=0&html=true.

[98] Source: US gov. “File:Ruby laser.jpg - Wikimedia Commons”. URL: https://commons.wikimedia.org/wiki/File:Ruby_laser.jpg. *Wikimedia Commons. (visited on 14-06-2022)*.

[99] Ed Sperling. “Quantum Effects At 7/5nm And Beyond”. URL: https://semiengineering.com/quantum-effects-at-7-5nm/. *(visited on 08-05-2022)*.

[100] Stack Exchange Inc. “Stack Overflow Developer Survey 2021”. URL: https://insights.stackoverflow.com/survey/2021#overview. *(visited on 16-09-2022)*.

[101] StackPath. “What is a Web Application?” URL: https://www.stackpath.com/edge-academy/what-is-a-web-application/. *(visited on 17-09-2022)*.

[102] Daniel A Steck. *Classical and Modern Optics*. Vol. Oregon Uni. 2006.

[103] Daniel A Steck. *Quantum and Atom Optics*. Vol. Oregon Uni. 2007.

[104] Suetonius. “Suetonius • Life of Julius Caesar”. URL: https://penelope.uchicago.edu/Thayer/L/Roman/Texts/Suetonius/12Caesars/Julius*.html#1. *(visited on 08-04-2022)*.

[105] Nick Taylor. *Laser: The Inventor, the Nobel Laureate, and the Thirty-Year Patent War*. iUniverse, Inc, 2007. ISBN: 0595465285.

[106] The MathJax Consortium. “MathJax Documentation — MathJax 3.2 documentation”. URL: https://docs.mathjax.org/en/latest/. *(visited on 13-09-2022)*.

[107] Natasha Tomm. “A quantum dot in a microcavity as a bright source of coherent single photons”. PhD thesis. 2021. DOI: 10.5451/UNIBAS-EP84050. URL: https://edoc.unibas.ch/84050/.

[108] Natasha Tomm et al. “A bright and fast source of coherent single photons”. In: *Nature Nanotechnology 2021 16:4 (plus supplemental)* 16.4 (Jan. 2021), pp. 399–403. ISSN: 1748-3395. DOI: 10.1038/s41565-020-00831-x. URL: https://www.nature.com/articles/s41565-020-00831-x.

[109] Kevin Townsend. “Quantum Computing Is for Tomorrow, But Quantum-Related Risk Is Here Today | SecurityWeek.Com”. URL: https://www.securityweek.com/quantum-computing-tomorrow-quantum-related-risk-here-today. *(visited on 16-09-2022)*.

[110] User: Facebook. “File:React-icon.svg - Wikimedia Commons”. URL: https://commons.wikimedia.org/wiki/File:React-icon.svg. *(visited on 16-09-2022)*.

[111] User: JavaScript Corp. “File:Javascript Logo.png - Wikimedia Commons”. URL: https://commons.wikimedia.org/wiki/File:Javascript_Logo.png. *(visited on 16-09-2022)*.

[112] User: Luringen. “File: Skytale.png”. URL: https://commons.wikimedia.org/wiki/File:Skytale.png. *Wikimedia Commons. (visited on 08-04-2022)*.

[113] User: Mstrdoob. “File:Three.js Icon.svg - Wikimedia Commons”. URL: https://commons.wikimedia.org/wiki/File:Three.js_Icon.svg. *(visited on 16-09-2022)*.

[114] User: Nageh. “File:MathJax.svg - Wikipedia”. URL: https://en.wikipedia.org/wiki/File:MathJax.svg. *(visited on 16-09-2022)*.

[115] User: norro. “File:Full width at half maximum.png - Wikimedia Commons”. URL: https://commons.wikimedia.org/wiki/File:Full_width_at_half_maximum.png. *(visited on 08-14-2022)*.

[116] User: RJB1. “File:Linearly Polarized Wave.svg - Wikimedia Commons”. URL: https://commons.wikimedia.org/wiki/File:Linearly_Polarized_Wave.svg. *Wikimedia Commons. (visited on 13-04-2022)*.

[117] User: Rudloff. “File:CSS3 logo and wordmark.svg - Wikimedia Commons”. URL: https://commons.wikimedia.org/wiki/File:CSS3_logo_and_wordmark.svg. *(visited on 16-09-2022)*.

[118] User: W3C. “File:HTML5 logo and wordmark.svg - Wikimedia Commons”. URL: https://commons.wikimedia.org/wiki/File:HTML5_logo_and_wordmark.svg. *(visited on 16-09-2022)*.

[119] William Anderson. “File:Electromagnetic spectrum 2.jpg - Wikimedia Commons”. URL: https://commons.wikimedia.org/wiki/File:Electromagnetic_spectrum_2.jpg. *Wikimedia Commons. (visited on 14-06-2022)*.

[120] Bang Wong. “Color blindness”. In: *Nature Publishing Group* (2011). DOI: 10.1038/nmeth.1618.

[121] Jean Paul Yaacoub et al. “Security analysis of drones systems: Attacks, limitations, and recommendations”. In: *Internet of Things* 11 (Sept. 2020), p. 21. ISSN: 2542-6605. DOI: 10.1016/J.IOT.2020.100218.

List of Figures

List of Tables

List of source codes

List of Abbreviations

API	application programming interface
CLI	command-line interface
CNOT	controlled NOT
CSS	Cascading Style Sheets
DHKE	Diffie-Hellman key exchange
DH	Diffie-Hellman
DLP	discrete logarithm problem
DOM	Document Object Model
DSA	Digital Signature Algorithm
DSS	Digital Signature Standard
ECDH	elliptic-curve Diffie-Hellman
FPI	Fabry-Pérot-interferometer
FSR	free spectral range
FWHM	full width at half maximum
GNFS	general number field sieve
GPLv3	GNU General Public License v3.0
GPU	graphics processing unit
HTML	HyperText Markup Language
HeNe	Helium-Neon
IC	integrated circuit
IRTF	Internet Research Task Force
InAs	indium arsenide

IoT	Internet of things
JSX	JavaScript Syntax Extension
JS	JavaScript
MITM	man-in-the-middle
NIST	National Institute of Standards and Technology
NOP	no-operation
NSA	National Security Agency
OS	operating system
OTP	one-time pad
PFS	perfect forward secrecy
PKI	public key infrastructure
PK	public key
Q factor	quality factor
QEC	quantum error correction
QKD	quantum key distribution
RFC	Request for Comment
RFID	radio-frequency identification
RSA	Rivest–Shamir–Adleman
SDK	software development kit
SPA	single-page application
SPDC	Spontaneous parametric down conversion
SSH	Secure Shell Protocol
SVG	scalable vector graphics
TLS	transport layer security
UI	user interface
WebGL	Web Graphics Library

XMSS	eXtended Merkle Signature Scheme
XOR	exclusive OR
maser	microwave amplification by stimulated emission of radiation
npm	Node package manager
qubit	quantum bit
web app	web application