STPA-Sec Applied to Path Planning: Quantum-Safe Autonomous Vehicles

by

David Jepson

A thesis
presented to the University of Waterloo
in fulfillment of the
thesis requirement for the degree of
Master of Mathematics
in
Combinatorics and Optimization (Quantum Information)

Waterloo, Ontario, Canada, 2022

Abstract

Autonomous vehicles and quantum computers are two emerging technologies that will transform our world in the not-too-distant future. This thesis examines the safety and security of autonomous vehicles in a world where adversaries have access to large-scale quantum computers. Large-scale quantum computers are relevant to automotive security because they can defeat the cryptographic foundation underlying critical safety systems such as path planning, perceptual unit, braking, steering, and engine electronic control units (ECUs). Peter Shor discovered a quantum computer algorithm in 1994 that can defeat modern-day public-key cryptography, including digital signatures (e.g., RSA, Ed-DSA), due to the algorithm's ability to factor large numbers and find discrete logarithms efficiently [23]. According to existing mathematical theory, classical computers cannot factor large numbers or find discrete logarithms efficiently. The critical insight derived from this thesis is that an adversary can defeat an autonomous vehicle's security of safety-critical systems with a large-scale quantum computer. In particular, the digital signatures used for authentication of over-the-air (OTA) software updates can be forged by an adversary with a large-scale quantum computer which, in the worst-case scenario, could enable a fleet-wide hack of an autonomous vehicle system potentially compromising a million vehicles simultaneously. The thesis explicitly identifies Tesla as a significant risk through their use of Ed25519, a discrete logarithm-based digital signature for OTA software updates [77], [78], [79]. Likely, most automotive manufacturers are at risk, but Tesla was the only company whose digital signature protocols were found to be publicly available on the internet. The analysis was completed using STPA-Sec (System-Theoretic Process Analysis for Security), an engineering risk management framework for identifying safety issues caused by security breaches. Overviews of quantum computing and quantum-safe cryptography are given. In addition, a Monte Carlo simulation framework is proposed to estimate the probability and severity of a large-scale quantum computer attack on autonomous vehicles. In addition to outlining the attack, countermeasures are provided to mitigate the risk, such as automotive companies upgrading to quantum-safe cryptography that NIST is currently standardizing. The NIST standardization is scheduled for completion in 2024. If automotive companies upgrade to quantum-safe cryptography, the risk against known attacks is eliminated, but there is a residual risk regarding currently unknown attacks.

There is a reasonable amount of time to mitigate this risk as large-scale quantum computers are not expected to exist until the end of the decade. However, the section on quantum cyber risk analytics focuses on estimating the risk in the worst 1 in 1,000 chance scenario. Based on a model that estimates quantum risk, whose details including assumptions are outlined in Chapter 11, the central insight from the analytics is that there is an approximate 99 in 100 chance the RSA-2048 will be broken in 24 hours within the next 15 years in the worst 1 in 1,000 chance scenario. A vision of a quantum-safe and quantum-enhanced autonomous vehicle future is painted where quantum computers and quantum sensors may significantly enhance many aspects of autonomous vehicles. Recommendations to improve STPA-Sec are provided. The main contributions of this work are identifying a worst-case scenario where a million cars could be compromised by an adversary with access to a large-scale quantum computer, conducting a formal STPA-Sec analysis on the path planning control loop of an autonomous vehicle in the presence of an adversary with a large-scale quantum computer, providing suggestions on how to improve STPA-Sec, and the section on quantum risk management. In particular, conducting the first known quantum stress test by estimating the risk of the worst 1 in 1,000 chance scenario for RSA-2048 to be broken in 24 hours within 15, 20, and 30 years completes the contributions of this thesis.

Acknowledgements

I want to thank my supervisor Michele Mosca for providing excellent support and always believing in me throughout this non-trivial part-time Master's degree.

Thank you to Sebastian Fischmeister for guiding me on my journey through analyzing automotive security, autonomous vehicles, STPA-Sec, for suggesting using Monte Carlo simulation to estimate cyber risk, and for joining my MMath thesis committee.

Thank you to David Jao for teaching me about cryptography throughout my Master's degree and for joining my MMath thesis committee.

Thank you to Brett Lindsay and Paul Check for their feedback on the quantum cyber risk analytics approach.

Dedication

This thesis is dedicated to my family and friends. Thank you for your love and support while I went on multiple academic journeys.

Table of Contents

List of Figures

List of Tables

"It always seems impossible until it's done"
Nelson Mandela

Chapter 1

Overview of Quantum-Safe Autonomous Vehicles

"I think one of the biggest risks for autonomous vehicles is somebody achieving a fleet-wide hack. You know in principle, if somebody was able to hack say all of the autonomous Teslas, they could say, I mean just as a prank, send them all to Rhode Island, from across the United States, and I'd be like well, OK, that would be the end of Tesla."

Elon Musk

Modern cars are complex containing at least 100 electronic control units (ECUs), 5 internal networks, 2 miles of cables, and 100 million lines of code [38]. The security of the vehicle's cyber-physical systems is critical to the safety of passengers, pedestrians, and other road participants. Compromise of the vehicle's security and safety systems can lead to immeasurable suffering and loss of life as well as a material impact on an automotive company's shareholder value. For example, in 2015, Charlie Miller and Chris Valasek bought a Jeep Cherokee [43], [66] for automotive security research purposes. Over the Sprint network, they discovered that they could identify the IP address of 1.4 million automobiles made by Chrysler Fiat [43], [66]. Any computer with access to the Sprint network could send unsafe control actions to these vehicles [43], [66]. There was the potential to shut off the engines on 1.4 million vehicles with the push of a button from the comfort of their home [43], [66]. Following the announcement of a recall due to the vulnerability they identified, the share price of Chrysler Fiat dropped by approximately 6% [66].

Fortunately, Miller and Valasek were white hat hackers [94] trying to protect society, and the result was only a loss of approximately a billion dollars of market capitalization. If two security researchers could discover this vulnerablity, then it is possible that adversarial nation-states, terrorist organizations, or other black hat hackers could also, and their motives could be malignant.

There are many ways to attack automotive security systems leading to safety issues; refer to Section 5 on automotive security below for details. In addition to the work by Miller and Valasek, the automotive security research conducted by Keen Security Labs from Tencent is of particular interest.

In 2016, Keen Security Lab from Tencent hacked a Tesla over Wi-Fi and could overwrite the firmware on ECUs, enabling them to send unsafe control actions over the CAN bus through Wi-Fi [13]. The attack prompted Tesla to do an over-the-air (OTA) update requiring code signing on future ECU firmware updates to help mitigate against the attack [6]. In 2017, Keen Security Lab's discovered a way to bypass the code signing control added by Tesla in 2016 and upload custom firmware to the gateway ECU [77], [78]. In addition, the digital signature scheme used by Tesla is the Edwards-curve Digital Signature Algorithm (EdDSA) using the Ed25519 implementation, which is not a quantum-safe digital signature. Ed25519 is not quantum-safe because it is based on the discrete log problem, which can be defeated by a large-scale quantum computer running an algorithm discovered by Peter Shor used for quantum cryptanalysis [23], [77], [78], [79].

If a Master's researcher can discover that the cryptographic underpinning of Tesla's security systems for safety critical ECUs can be compromised then surely adversarial nation-states, terrorist organizations, and other black hat hackers could also discover this since the information is publicly available on the internet.

There are many creative ways to hack a car's security systems, leading to safety issues described in Section 5 on automotive security below. In the case of the Keen Security Labs work mentioned above, Tesla has been playing a game of chess with their adversaries, and Keen Security Lab put Tesla in a state of "Check". Tesla counteracted Keen's vulnerability discovery with the implementation of code signing. Unfortunately, Tesla's solution is vulnerable to quantum cryptanalysis in the future. They are using Ed25519 to authentic their over-the-air (OTA) software updates. Public-key cryptography is not a universal defence system that can block any attack, but it is fundamentally vital to confidentiality, integrity, and authenticity (CIA). Public key cryptography such as Rivest-Shamir-Adelman (RSA), Elliptic-curve cryptography (ECC), and Diffie-Hellman (DH) have never been broken before when implemented correctly based on publicly available information. The above-mentioned public-key cryptography has never been broken before because its security is based upon the high level of computational horsepower required to factor or find discrete logarithms, which is infeasible using classical computers. Peter Shor changed that when he discovered an algorithm to efficiently solve the factoring and discrete logarithm problems on a quantum computer [23]. As mentioned above, Keen Security Lab counteracted Tesla's move of applying code signing and uploaded custom firmware to the gateway ECU [77], [78]. This highlights a significant issue, code signing can only offer protection when the security system is implemented correctly, and in the case above, they were able to bypass it. The reason why code signing is important is that in a properly implemented system it acts as a critical layer of defence to authenticate the origin and validate the integrity of the software that is being installed into the path planner, perceptual unit, and safety critical ECUs such as steering, braking, and the engine. Malicious artificial intelligence could be installed into the path planner and perceptual unit, such as a rogue "Full Self Driving" update from an adversarial nation-state that forged Tesla's digital signature using a large-scale quantum computer. Using a broken digital signature is like pouring gasoline on an already burning fire at the fireworks factory. If Tesla does not upgrade their digital signatures to quantum-safe versions as outlined in Section 3 on quantum-safe cryptography before quantum computers arrive, it could be "Checkmate".

In the worst-case scenario, broken digital signatures could lead to a fleet-wide compromise, which according to Elon Musk, would be "... the end of Tesla" [95].

In the worst-case scenario, an adversary could install malicious AI into a fleet of Tesla's using a large-scale quantum computer and turn them from autonomous vehicles into autonomous weapons. A coordinated attack could be launched on a Friday during afternoon rush hour when the malicious AI becomes activated across the fleet. Numerous unsafe control actions could be executed as listed in Section 7 on System-Theoretic Process Analysis for Security (STPA-Sec) below. To illustrate the severity of the issue, imagine a scenario with a million cars compromised. The vehicles could act as autonomous weapons travelling at high speeds, intentionally driving into as many pedestrians as possible before arriving at their final destination of a collision with a brick wall with a car full of passengers. Imagine this happening to a million vehicles simultaneously. This worst-case scenario attack would likely require that the same private key is used across the entire fleet, which may not be the case. The same private key could be used across the fleet intentionally by the company's decision, oversight, implementation error, or intentionally by actions taken by an employee with malicious intent.

The severity of the attack outlined above is extreme, but how likely is something like this to occur? It is prudent to manage the risk for a realistic worst-case scenario for risk management purposes. The Basel II accord states that a worst 1 in 1,000 chance scenario should be used for capital risk management in the financial services industry [21]. Quantum cyber risk analytics was conducted on survey data collected and published by Mosca and Piani, which estimates the probability that RSA-2048 will be broken in 24 hours within the next 5, 10, 15, 20, and 30 years [5].

From an experimentalist point of view, the mean probability that RSA-2048 will be broken in 5 years, which is the most conservative, is approximately 2%. However, in the worst 1 in 1,000 chance scenario, the probability could be significantly greater than 2%.

Over a longer time horizon, the chance of RSA-2048 being broken in 24 hours within 15 years is approximately 99 in 100 in the worst 1 in 1,000 probability scenario based on the minimum values of the experimentalists' points of view. The worst 1 in 1,000 probability scenario risk was calculated by fitting a statistical distribution to survey results from quantum computing experts and calculating the 99.9th percentile. See Section 11.1 on "Quantum Cyber Risk Analytics" below for details about the model and assumptions.

An approximate 99 in 100 chance that RSA-2048 will be broken in 24 hours within 15 years in the worst 1 in 1,000 probability scenario is alarmingly high. As a result of the above estimates, this indicates that the world is in a state of crisis regarding cryptography and quantum computing. In particular, the automotive industry could suffer a devastating blow by its use of broken digital signatures that adversaries can forge with large-scale quantum computers, such as the case of Tesla using Ed25519.

In the worst-case scenario, an adversarial Stuxnet-style nation-state-sponsored attack could be executed within the next fifteen years using a large-scale quantum computer to forge the digital signatures used in OTA software updates. The quantum-enabled attack could allow the installation of malicious AI into the vehicle's path planning and perceptual systems and malware into the firmware of safety-critical ECUs such as the steering, braking, and engine. The attack could mark the beginning of quantum cyber warfare. The number of casualties resulting from this attack could be comparable to that of the September 11th terrorist attacks. Furthermore, automotive companies with a trillion-dollar market capitalization (i.e., Tesla) could face significant financial difficulties and require government bailouts at the taxpayers' expense. Finally, an attack of this magnitude could result in a war.

In the best-case scenario, an attacker would only have five minutes with remote cellular access to a vehicle, and every car would be equipped with a unique public key. Attacking a single car would require retrieving the public key from the vehicle and sending it to a quantum computer to calculate the private key. The quantum computation of the private key would have to be done in less than five minutes, which would require a very efficient and expensive quantum computer. Since every vehicle is equipped with a unique public key, it would cost approximately less than $1 million per vehicle. See Section 11.2 for how the estimate of less than $1 million of the cost for obtaining a private key from a public key using a quantum computer was calculated. A fleet-wide hack of a million cars would cost approximately $1 trillion and require that a million private keys are computed within a short period of time. In the next 30 years, there will not be enough quantum computers on the planet to execute a fleet-wide attack if unique public keys are used in every vehicle. However, attacking a single public key in a car for less than $1 million would be feasible.

The automotive industry should complete the migration to quantum-safe cryptography before the arrival of large-scale quantum computers. According to the National Cybersecurity Center of Excellence (NCCoE) from the National Institute of Standards and Technology (NIST) it may take anywhere from 5 to greater than 15 years to upgrade to quantum-safe cryptographic systems starting from when the standards are published, which is expected in 2024 [16]. Every industry is has different complexity regarding migration, but the point is that the exercise is non-trivial and migration plans should provide sufficient lead time to be completed before large-scale quantum computers arrive. The recommendation is to hope for the best-case scenario but plan for the worst-case scenario. Migration from currently deployed public key cryptography is not risk-free. There is also the risk that a classically equipped adversary could break public key cryptography, both existing and post-quantum. The optimal time for migration and strategy to mitigate the risks faced by digital signatures is discussed in the section on risk mitigation strategies.

There are several risk mitigation strategies to consider, such as moving to post-quantum digital signatures, hybrid digital signatures, hash-based digital signatures, and optimizing the crypto agility of an organization. See Section 11.2.1 for more details regarding risk mitigation strategies.

On October 10th, 2021 an email was sent to Tesla's vulnerability reporting address "VulnerabilityReporting@tesla.com" to disclose the quantum vulnerability discovered in their use of Ed25519 to digitally sign their OTA software updates. As of February 28th, 2022, the author of this thesis has not received a response from Tesla.

Chapter 2

Overview of Quantum Computing

"Nature isn't classical, dammit, and if you want to make a simulation of nature, you'd better make it quantum mechanical, and by golly it's a wonderful problem, because it doesn't look so easy."

Richard P. Feynman

Quantum computers are an emerging technology that can process quantum information. The fundamental unit of quantum information is the qubit, which is short for quantum binary digit. A qubit can exist in a superposition of 0 and 1 simultaneously because of the Laws of Quantum Physics. Furthermore, two qubits can become entangled with each other. As well, qubits can have phases added to them. These phenomena translate into quantum gates that can put a qubit into superposition, entangle two qubits, or add phases to them.

The following are some mathematical properties of a qubit:

A qubit is in a quantum superposition of it's basis states $|0\rangle$ and $|1\rangle$ which is are called "Kets" whose conjugate transpose (adjoint) is represented as $\langle 0|$ and $\langle 1|$ which are called "Bras". $|0\rangle = \begin{pmatrix} 1 \\ 0 \end{pmatrix}$ and $|1\rangle = \begin{pmatrix} 0 \\ 1 \end{pmatrix}$. An arbitrary qubit is represented as $|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$ where $|\psi\rangle \in \mathbb{C}^2$, $|\psi\rangle = \begin{pmatrix} \alpha \\ \beta \end{pmatrix}$. The inner product of the basis states are $\langle i|j\rangle = \delta_{ij}, i, j \in \{0,1\}$, and contains a joke from Paul Dirac who created the "Bra-Ket" notation [153]. The basis states $|0\rangle$ and $|1\rangle$, called the computational basis, form an orthonormal basis. A qubit is a unit vector, where $\||\psi\rangle\| = 1 = \sqrt{\langle\psi|\psi\rangle}$ which implies that $\||\psi\rangle\| = (\bar{\alpha}\ \bar{\beta})\begin{pmatrix}\alpha\\ \beta\end{pmatrix} = |\alpha|^2 + |\beta|^2 = 1$. Note that α and β are called probability amplitudes and $|\alpha|^2$ is the probability of observing $|0\rangle$ and $|\beta|^2$ is the probability of observing $|1\rangle$ when measured in the computational basis. Note that the total probability is $|\alpha|^2 + |\beta|^2 = 1$ since the qubit must be either $|0\rangle$ or $|1\rangle$ when measured.

The following is a set of universal quantum gates [22]:

The $\frac{\pi}{8}$-gate (T): $\begin{pmatrix}1 & 0\\ 0 & e^{i\frac{\pi}{4}}\end{pmatrix}$

Given a single qubit $|\psi\rangle = \alpha|0\rangle + \beta|1\rangle, T|\psi\rangle = \begin{pmatrix}1 & 0\\ 0 & e^{i\frac{\pi}{4}}\end{pmatrix}\begin{pmatrix}\alpha\\ \beta\end{pmatrix} = \alpha|0\rangle + e^{i\frac{\pi}{4}}\beta|1\rangle$. The T gate adds a relative phase to the qubit modifying the probability amplitude of $|1\rangle$ from β to $e^{i\frac{\pi}{4}}\beta$. Note that the probability of observing $|1\rangle$ when measuring in the computational basis is still $|\beta|^2$ since $|e^{i\frac{\pi}{4}}\beta|^2 = e^{-i\frac{\pi}{4}}\bar{\beta}e^{i\frac{\pi}{4}}\beta = \bar{\beta}\beta = |\beta|^2$.

The Hadamard gate (H): $\frac{1}{\sqrt{2}}\begin{pmatrix}1 & 1\\ 1 & -1\end{pmatrix}$

The Hadamard gate when applied to $|0\rangle$ yields an equal superposition of $|0\rangle$ and $|1\rangle$.

$$H|0\rangle = \frac{1}{\sqrt{2}}|0\rangle + \frac{1}{\sqrt{2}}|1\rangle$$

Note that the probability amplitudes of $|0\rangle$ and $|1\rangle$ are $\frac{1}{\sqrt{2}}$ meaning that the probability of measuring a $|0\rangle$ or $|1\rangle$ in the computational basis is each 50% since $|\frac{1}{\sqrt{2}}|^2 = \frac{1}{2}$.

The Controlled-not gate ($CNOT$): $\begin{pmatrix}1 & 0 & 0 & 0\\ 0 & 1 & 0 & 0\\ 0 & 0 & 0 & 1\\ 0 & 0 & 1 & 0\end{pmatrix}$

The following example illustrates the power of the $CNOT$ gate which acts on a two-qubit system.

Start with the two-qubit system $|\psi\rangle = |0\rangle \otimes |0\rangle = |00\rangle$. Apply a Hadamard gate to the first qubit to put it into an equal superposition, and do nothing to the second qubit, i.e., apply the identity gate to the second qubit. Next, apply the $CNOT$ gate to the resulting two-qubit system.

$$(H \otimes I_2)|\psi\rangle = H|0\rangle \otimes I_2|0\rangle = (\frac{1}{\sqrt{2}}|0\rangle + \frac{1}{\sqrt{2}}|1\rangle) \otimes |0\rangle \implies$$

$$|\beta_{00}\rangle = CNOT(H \otimes I_2)|00\rangle = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix} \begin{pmatrix} \frac{1}{\sqrt{2}} \\ 0 \\ \frac{1}{\sqrt{2}} \\ 0 \end{pmatrix} = \begin{pmatrix} \frac{1}{\sqrt{2}} \\ 0 \\ 0 \\ \frac{1}{\sqrt{2}} \end{pmatrix} = \frac{1}{\sqrt{2}}|00\rangle + \frac{1}{\sqrt{2}}|11\rangle$$

By applying the Hadamard gate to the first qubit putting it in an equal superposition of $|0\rangle$ and $|1\rangle$ and then applying a controlled not generates $|\beta_{00}\rangle$ which is known as a Bell state or an Einstein-Poldosky-Rosen (EPR) pair. The EPR pair $|\beta_{00}\rangle$ has the special property that when measured there is a 50% chance that $|00\rangle$ will be observed and a 50% chance that $|11\rangle$ will be observed. This result is true regardless of where the two qubits are physically located in the universe. In the case of $|00\rangle$ this implies that when the first qubit is $|0\rangle$ that the second qubit is always $|0\rangle$. Similarly, in the case of $|11\rangle$ this implies that when the first qubit is $|1\rangle$ that the second qubit is always $|1\rangle$. This implies that 100% of the time that the value of the first qubit and the second qubit are identical. The Bell state $|\beta_{00}\rangle$ is an example of a maximally entangled state. Note that $|\beta_{00}\rangle$ has the special property that it cannot be written as the tensor product of two qubits, i.e.,

$$|\beta_{00}\rangle \neq |\psi_1\rangle \otimes |\psi_2\rangle \text{ where } |\psi_i\rangle = \alpha_i|0\rangle + \beta_i|1\rangle, i \in \{1, 2\}$$

The EPR pair $|\beta_{00}\rangle$ allows for interesting protocols to be implemented on a quantum computer such as quantum teleportation [22]. Note that even though a qubit can, in theory, be teleported across the universe, this cannot happen faster than the speed of light, since two classical bits, whose transmission obeys Einstein's Special Theory of Relativity, must be sent to the receiver of the teleported qubit [22].

For a detailed introduction to quantum computing, please refer to the book "An Introduction to Quantum Computing" by Kaye, Laflamme, and Mosca [22].

A universal set of quantum gates enable new algorithms that cannot be run efficiently on a classical computer. For example, a computation with n qubits would require the storage of 2^n probability amplitudes in a classical computer which exceeds feasible resources in the observable universe as n approaches 300 [92]. In 1994, mathematician Peter Shor discovered quantum algorithms that can solve the factoring and discrete logarithm problems in polynomial-time, which breaks the security of currently deployed public-key cryptography, including Rivest-Shamir-Adelman (RSA), Elliptic-curve cryptography (ECC) and Diffie-Hellman (DH) [23]. Also, Grover's algorithm can attack symmetric-key encryption such as Advanced Encryption Standard (AES) by providing a quadratic speed-up for searching. However, symmetric algorithms can mitigate the risk by doubling the key size (e.g., from AES 128-bit to AES 256-bit keys) [24]. Although quantum computers exist today, these are noisy intermediate-scale quantum computers (NISQ) which means they can only run limited algorithms due to lack of error correction and the limited number of qubits [28]. Running Shor's algorithm to factor an RSA-2048 modulus requires a quantum computer with 4,098 logical qubits [24] comprised of millions of physical qubits with quantum error correction. Given the importance of large-scale quantum computing to public-key cryptography, it is necessary to have a solid estimate of when these machines will arrive for risk management purposes. IBM has announced that it plans to build a 1,121-qubit quantum computer in 2023 [25], but this is still shy of the millions of qubits that will likely be needed to attack an RSA-2048 modulus. Google announced that they will have a useful, error-corrected quantum computer built by the end of 2029 [91]. In Google's announcement they stated, "we're on a journey to build 1,000,000 physical qubits that work in concert inside a room-sized error-corrected quantum computer" [91]. Survey results published by Mosca et al. in 2020 indicate that about half of the respondents believe that there is a 50% chance or higher in 15 years of a quantum computer capable of breaking an RSA-2048 modulus in 24 hours being built [5]. Also, nearly 7% of respondents felt an approximately 50% chance of such a quantum computer existing within the next five years [5]. In risk management, it is common to have a contingency plan for the worst-case scenario, such as the worst 1 in 1,000 chance scenario, as per the Basel II accord for capital risk management [21]. Cyber risk managers should plan for a scenario where large-scale quantum computers arrive within the next five years in the worst-case scenario. Risk mitigating actions need to be taken, such as migrating to quantum-safe cryptography and raising capital to cover losses related to cybercrime, such as data breaches and malware attacks by adversaries using a large-scale quantum computer. It is better to be over-prepared for a non-event than to be under-prepared for a significant event when feasible within the constraints of the business.

Chapter 3

Overview of Post-Quantum Cryptography

To mitigate the risk posed by quantum computers, a new type of cryptography is required that is resistant to quantum cryptanalysis. Today, there is a vast international effort to create new algorithms for public-key encryption, key establishment, and digital signatures that are considered resistant to attacks from large-scale quantum computers. The National Institute of Standards and Technology (NIST) is organizing the standardization of these algorithms and is currently in Round 3.

3.1 Digital Signatures

Post-quantum digital signatures provide authentication, non-repudiation, and data integrity and are believed to be secure against an adversary with a large-scale quantum computer. Below in Section 3.1 are the summaries of the three finalists (Rainbow, CRYSTALS-Dilithium, and FALCON) and three alternatives (GeMMS, SPHINCS+, and Picnic) from the NIST Round 3 post-quantum cryptography standardization process for digital signatures. An in-depth summary of Rainbow will be provided, including the algorithm's details as an example of how post-quantum cryptography works.

Digital Signature	Run Time (ms)	Size (bits)

Dilithium	Key Generation = 0.18
	Sign = 0.82
	Verification = 0.16
		K_{secret} = 22,400
		K_{public} = 9,472
		$\sigma = 16,352$

Falcon	Key Generation = 16.77
	Sign = 5.22
	Verification = 0.05
		K_{secret} = 10,248
		K_{public} = 7,176
		$\sigma = 5,520$

Rainbow	Key Generation = 0.48
	Sign = 0.34
	Verification = 0.83
		K_{secret} = 743,680
		K_{public} = 465,152
		$\sigma = 512$

GeMSS	Key Generation = 13.1
	Sign = 188
	Verification = 0.03
		K_{secret} = 107,502
		K_{public} = 2,817,504
		$\sigma = 258$

Picnic	Key Generation = 0.005
	Sign = 4.09
	Verification = 3.25
		K_{secret} = 128
		K_{public} = 256
		$\sigma = 272,256$

SPHINCS+	Key Generation = 2.95
	Sign = 93.37
	Verification = 3.92
		K_{secret} = 512
		K_{public} = 256
		$\sigma = 135,808$

Table 3.1: Benchmarking Post-Quantum Digital Signatures [19]. Note. Data in Table 3.1 from “Sok: Challenges of post-quantum digital signing in real-world applications.” by T.G. Tan, P. Szalachowski, and J. Zhou, 2019, Cryptology ePrint Archive, Report 2019/1374. p.17 `https://ia.cr/2019/1374`

Rainbow

Rainbow is a multivariate quadratic digital signature based on unbalanced oil and vinegar (UOV). The name Rainbow refers to the many possible layers of UOV that can be present in the signature [105]. Rainbow has one of the smallest signature sizes at 512 bits, has sound signature and verification run times, but suffers from some of the largest public and private key pair sizes as shown in Table 3.1 above. Rainbow is outlined below as defined in [104], [105], [106]. For more details on Rainbow, please refer to its website [118], [119].

Key Generation

Let $0 < v_1 < v_2 < \ldots < v_u < v_{u+1} = n$, where $v_r \in \mathbb{Z}, 1 \leq r \leq u+1$. Set $V_l = \{1, \ldots, v_l\}$ and $O_l = \{v_l + 1, \ldots, v_{l+1}\}$ where O_l and V_l are the indices for the oil and vinegar variables respectively in layer l of the "Rainbow", $1 \leq l \leq u$. Below is the multivariate quadratic equation used in creating the central map F composed of $f^{(k)}$ where $v_1 + 1 \leq k \leq n$.

$$f^{(k)}(x_1, \ldots, x_n) = \sum_{i,j \in V_l} \alpha_{ij}^{(k)} x_i x_j + \sum_{i \in V_l, j \in O_l} \beta_{ij}^{(k)} x_i x_j + \sum_{i \in V_l \cup O_l} \gamma_i^{(k)} x_i + \delta^{(k)}$$

where $k \in O_l$ for some l and $k \in O_{l_1}$ and $k \in O_{l_2}$ implies $l_1 = l_2$ and $\alpha_{ij}^{(k)}, \beta_{ij}^{(k)}, \gamma_{ij}^{(k)}, \delta^{(k)} \in_R \mathbb{F}$, a finite field.

Private Key: Randomly select two invertible linear transformations S and T such that $S : \mathbb{F}^m \to \mathbb{F}^m$ and $T : \mathbb{F}^n \to \mathbb{F}^n$ where $m = n - v_1$. Central map $F : \mathbb{F}^n \to \mathbb{F}^m$ as described above.

Public Key: $P = S \circ F \circ T$ is the public key where $P : \mathbb{F}^n \to \mathbb{F}^m$.

Signature Generation

To sign, first calculate the hash of the document d, $h = H(d)$, where $H : \{0,1\}^* \to \mathbb{F}^m$. Next, compute $P^{-1}(h)$ by solving $T^{-1}(F^{-1}(S^{-1}(h)))$ where S^{-1} and T^{-1} have inverses by definition and F^{-1} refers to any pre-image of the central map F as the solution may not be unique. To do so, solve for $c = S^{-1}(h)$, $c \in \mathbb{F}^m$, then calculate pre-image of c under the central map F.

Algorithm 1 Find a pre-image of the central map F

```
1: Input $c \in \mathbb{F}^m$ and $F$ the central map.
2: Select values $x_i \in_R \mathbb{F}, \forall\, i \in V_1$.
3: for $l = 1$ to $u$ do
4:     if $l = 1$ then $i \in V_1$ where $i$ represents the indices of the vinegar variables.
5:     else $i \in V_l = V_{l-1} \cup O_{l-1}$ where $i$ represents the indices of the vinegar variables.
6:     end if
7:     Solve the system of $v_{l+1} - v_l$ equations $f^{(k)}\ \forall\, k \in O_l$ for the oil variables using Gaussian elimination by substituting in the vinegar variables with indicies i.
8:     If there is no solution then start over and take a different random sample for vinegar variables with indices in $V_1$.
9: end for
10: Output The pre-image of c of the central map is $b = F^{-1}(c)$, $b \in \mathbb{F}^n$ the solution to the above system of equations.
```

Calculate the signature $\sigma = T^{-1}(b)$, $\sigma \in \mathbb{F}^n$.

Signature Verification

To verify the signature, check if $P(\sigma) = H(d)$.

CRYSTALS-Dilithium

Cryptographic Suite for Algebraic Lattices (CRYSTALS)-Dilithium is a lattice-based digital signature based on Module Learning with Errors (MLWE) [113]. Dilithium is a Round 3 NIST finalist. This can be seen by its strong performance in signing and verification time and its small key size relative to the multivariate digital signature schemes, as shown in Table 3.1 above. For more information on CRYSTALS-Dilithium, please refer to its website [115], [118].

FALCON

Fast-Fourier Lattice-based Compact signatures over NTRU (FALCON) are a lattice-based digital signature leveraging Nth Degree Truncated Polynomial Ring Units (NTRU). FALCON is a Round 3 NIST finalist. This can be seen by its strong performance in signing and verification time and its small key size relative to the multivariate digital signature schemes, as shown in Table 3.1 above. FALCON has a somewhat smaller key and signature sizes than its main lattice-based competitor, Dilithium, and one of the fastest verification times amongst all the Round 3 digital signatures; however, it's signature and key generation time is slower compared to Dilithium as shown in Table 3.1 above. Note that NIST expects that at most one lattice-based digital signature will be standardized meaning at most one of FALCON and Dilithium [111]. For more information on FALCON please refer to its website [116], [118].

GeMSS

A Great Multivariate Short Signature (GeMSS), is a multivariate quadratic digital signature based on Hidden Field Equations (HFE). GeMMS has the quickest verification times of all the Round 3 digital signatures and the shortest signature but has the largest public-key size and has the slowest signature time, as shown in Table 3.1 above. For further details on GeMMS, please refer to its website [117], [118].

SPHINCS+

Stateless, Practical, Hash-based, Incredibly Nice, Cryptographic Signatures (SPHINCS+), is a stateless hash-based digital signature scheme. SPHINCS+ has the second slowest signature time and also has a relatively slow verification time. SPHINCS+ has one of the longest signatures of the six Round 3 candidates but has some of the smallest private and public key sizes, as shown in Table 3.1 above. For more information on SPHINCS+, please refer to its website [118], [120].

Picnic

Picnic is a post-quantum digital signature based on Zero-Knowledge proofs. Picnic has the second slowest signature and verification time and a large signature. Still, it has the smallest public and private key size amongst the six Round 3 post-quantum digital signature candidates, as shown in Table 3.1 above. For more information on Picnic, please refer to its website [118], [121].

3.2 Key Establishment

Post-quantum key establishment algorithms provide confidentiality of data and are believed to be secure against an adversary with a large-scale quantum computer. Below in Section 3.2 are the summaries of the four finalists (Classic McEliece, CRYSTALS-KYBER, NTRU, and SABER) and five alternatives (SIKE, FrodoKEM, NTRU Prime, HQC, and BIKE) from the NIST Round 3 post-quantum cryptography standardization process for key establishment.

Classic McEliece

Classic McEliece is a code-based Key Encapsulation Mechanism (KEM) developed by McEliece in 1978 [114]. For more information on Classic McEliece, please refer to its website [114], [118].

CRYSTALS-KYBER

CRYSTALS-KYBER is a lattice-based KEM whose hardness depends upon the difficulty of solving a module learning with errors (MLWE) problem [122]. For more information on CRYSTALS-KYBER, please refer to its website [118], [122].

NTRU

Nth Degree Truncated Polynomial Ring Units (NTRU) is a lattice-based KEM whose hardness is based upon the difficulty of finding the shortest vector in a lattice [112]. For more information on NTRU, please refer to its website [118], [123].

SABER

SABER is a lattice-based KEM whose hardness depends on the difficulty of solving the module learning with rounding (MLWR) problem [124]. For more information on SABER, please refer to its website [118], [124].

SIKE

Supersingular Isogeny Key Encapsulation (SIKE) is a KEM based on isogenies of supersingular elliptic curves. SIKE is not patented and provides perfect forward secrecy [126]. Also, SIKE has the smallest key size, as shown in [113], amongst all the Round 3 candidates for KEM [126]. SIKE is unique because it is the only isogeny-based post-quantum cryptographic algorithm entered into the NIST standardization process and is currently an alternative candidate for KEM in Round 3. For more information on SIKE, please refer to its website [118], [125].

FrodoKEM

FrodoKEM is a lattice-based KEM whose hardness is based on solving the learning with errors (LWE) problem [127]. For more information on FrodoKEM, please refer to its website [118], [127].

NTRU Prime

NTRU Prime is a lattice-based KEM whose hardness is based upon the difficulty of finding the shortest vector in a lattice [112]. For more information on NTRU Prime, please refer to its website [118], [128].

HQC

Hamming Quasi-Cyclic (HQC) is a code-based KEM. For more information on HQC, please refer to its website [118], [129].

BIKE

Bit Flipping Key Encapsulation (BIKE) “is a code-based KEM based on Quasi-Cyclic Moderate Density Parity-Check (QC-MDPC) codes” [130]. For more information on BIKE, please refer to its website [118], [130].

For benchmarking details of the above KEMs, please refer to [112], [113].

Chapter 4

Overview of Autonomous Vehicles

Autonomous vehicles are expected to provide significant safety benefits. According to a study published in 2015 by the U.S. Department of Transportation's National Highway Traffic Safety Administration (NHTSA) 94% of accidents have the driver as the critical reason, where the critical reason is often the last failure leading up to the crash [3]. A further breakdown shows that distracted driving, poor decision-making, poor driving execution, and sleep-related issues are the major types of driver-related critical reasons [3]. Autonomous vehicles do not get distracted, do not have fatigue due to lack of sleep, and should provide a material benefit regarding safety. By 2030 up to 15% of new car sales could be Level 4 autonomous, indicating that the safety benefits should have a material impact by then [4].

Autonomous vehicles have a suite of sensors, models for understanding their environment, and tools for making optimal decisions satisfying the system's constraints to implement real-time automated driving actions.

4.1 Sensors

Sensors are used to measure the environment and provide feedback to other components of the system [34]. Below are the key sensors used in an autonomous vehicle.

Cameras are used for object detection, such as street signs and pedestrians, and measuring angles and depth [1]. In some cases, manufacturers base their sensor technology primarily on cameras instead of more expensive LIDAR systems. Tesla is an example of a company relying mainly on cameras for their sensors [35].

RADAR uses electromagnetic radiation to measure the distance and speed of objects in its environment [144]. This sensor is used in systems such as adaptive cruise control [26].

LIDAR uses laser light to make a 3D map of the environment. The sensor can make 3D maps of pedestrians and small objects which is required for Level 4 autonomous vehicles [1].

4.2 Perceptual System

The sensor data is used by the autonomous vehicle to make perceptions about its environment. Below are the main ways sensor data is used to form perceptions.

Localization is the process of the vehicle identifying its location within its perceived environment. GPS can be used to estimate the vehicle's location, although it does not work well in urban areas where there are issues with line-of-sight to the satellite [2].

Object detection uses sensors such as cameras to identify objects in the environment (e.g., pedestrians) using machine learning classification algorithms.

Object tracking leverages object detection and can track an object as it moves through its environment, such as tracking other cars' estimated velocity and path.

Sensor Fusion uses inputs from various sensors to increase the efficiency and reliability of the sensor data [36]. For example, object detection combines distance, velocity, and colour distribution for reliable and accurate results [36].

The systems above form the main components of the perceptual unit, which feed into the path planning system.

4.3 Path Planning System

The path planning system is the central controller in the autonomous vehicle. Given the information from the perceptual unit, it will calculate the optimal path for the car to take. The path planning system makes frequent decisions solving new path optimization problems every 50 milliseconds in some vehicles [1]. The output of path planning is trajectories sent to various controllers (e.g., steering controller), subsequently relayed to the actuators to execute control actions such as turning.

4.4 State of the Art in Autonomous Vehicles

Numerous companies are using autonomous vehicles on the roads. Below is a summary of some key achievements.

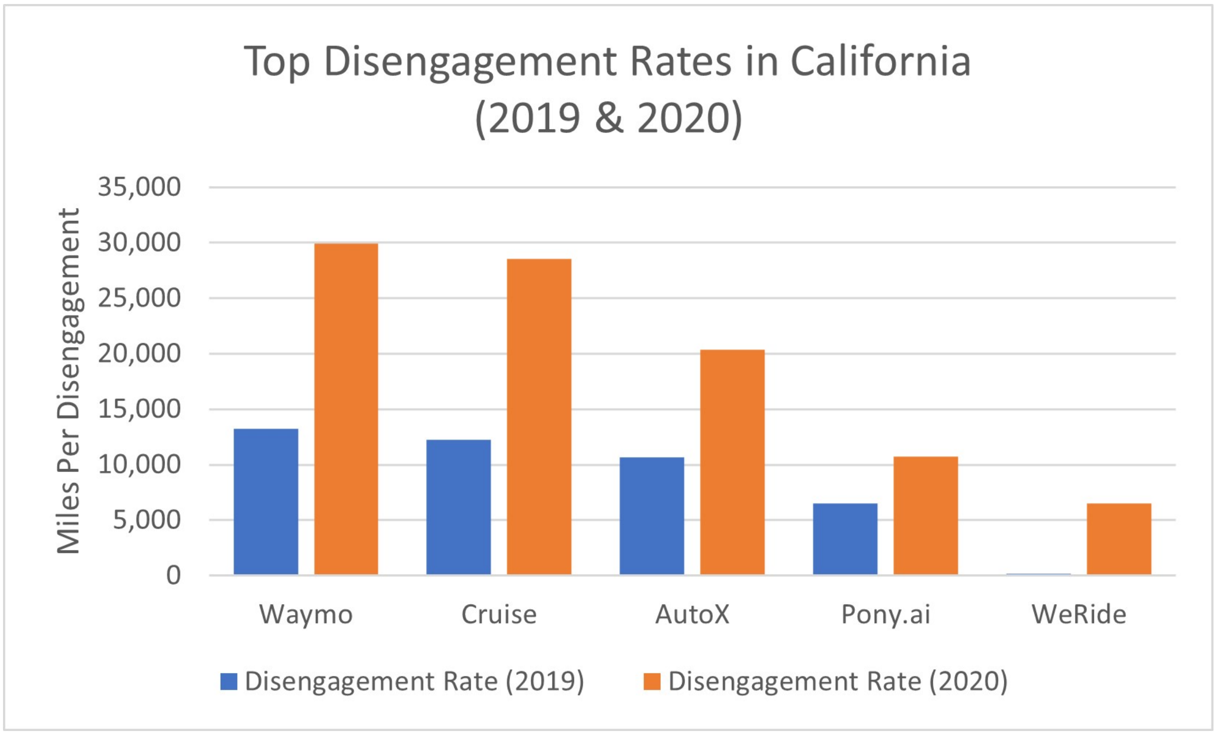


Figure 4.1: Miles per Disengagement Rates of Autonomous Vehicle in California [50]

Figure 4.1 shows the top reported miles per disengagement rate for autonomous vehicle testing in California. The miles per disengagement rate measures the average number of miles driven per request for a human to take over from the autonomous driving system. Waymo and Cruise are leading with high miles per disengagement rates averaging about once every 30,000 miles [50]. Waymo drove over 628,000 miles with 21 disengagements and Cruise over 770,000 miles with 27 disengagements in California during 2020 [50]. Other start-ups are making significant progress, including AutoX, Pony.ai, and WeRide. Note that the trend is for a significant year-over-year increase in all companies' miles per disengagement rates. For example, AutoX reported a rate of 10,685 miles per disengagement in 2019 and 20,367 miles per disengagement in 2020, a 91% year-over-year (YoY) increase

[50]. The five top companies shown in Figure 4.1 are well above the average of 529 miles per disengagement across all 25 companies reporting autonomous vehicle testing results in California for 2020 [50]. The average in 2019 was 321 miles per disengagement and has since increased by 65% YoY in 2020 [50]. The YoY increase in the miles per disengagement rate in 2020 indicates that automation is rapidly increasing across the industry. Note that the five companies in Figure 4.1 are developing Level 4 autonomous vehicles [145], [146], [147], [148], [149]. In 2021, AutoX launched a driverless taxi service which is the first time the public can use an autonomous vehicle without a safety driver in China [52].

Tesla did not report any miles driven in California regarding disengagement rates of autonomous vehicles for 2020 [50]. As of February 2020, Tesla has driven nearly 3 billion miles in autopilot [47],[51]. In Q1 2021, Tesla reported one accident per 4.19 million miles with autopilot engaged. Without autopilot and other safety features, 0.98 million miles per accident compared to NHSTA data showing an average of one accident every 0.48 million miles in the USA [48]. Note that Tesla's autopilot is classified as Level 2 autonomy as of the beginning of 2021 [49].

Chapter 5

Overview of Automotive Security

Automotive security involves protecting cyber systems from malicious attacks [84]. Modern cars are complex containing at least 100 electronic control units (ECUs), five internal networks, 2 miles of cables, and 100 million lines of code [38]. The attack surface consists of long-range, short-range, and local access to the system [39]. The attack surface of an autonomous vehicle includes the following key elements:

5.1 Attack Surface

Local Attack Surface [40]:

1. ECUs (e.g., Steering, Braking, Engine, Transmission, Lighting, Airbag, ADAS) – The software and settings on the ECUs can be modified by an adversary with local access to the vehicle, which could cause performance issues and unsafe control actions to occur.

2. ODB II (Onboard Diagnostic II port) – The OBD II port is used to monitor the car's performance and can communicate with various ECUs. This data connection allows for attacks that can compromise ECUs within the vehicle enabling an adversary to execute unsafe control actions and read information from the network, some of which may be confidential [64].

3. USB – USB connections can be used to load files to the car which gives an attacker the ability to compromise the ECUs by executing malicious code that can compromise

the ECUs connected to the network [66]. USBs could also be used to exfiltrate data from a car.

4. CD – CDs can be used to load files to a car like USB, which can be used to execute malicious code that can compromise the network of the vehicle [41]. Unlike USB, CD players used in cars are read-only and cannot exfiltrate data from the car.

5. Sensors (e.g., LIDAR, RADAR, Cameras) – Sensors can be manipulated by an adversary that modifies the physical environment, sending corrupted feedback about the environment, leading to false perceptions resulting in unsafe control actions [44]. Adversaries can also compromise the sensors via a denial of service (DoS) attack which could result in a controller issuing unsafe control actions.

   Short-range Attack Surface [40]:

6. Bluetooth – Bluetooth is a wireless data connection between computers and the car that can be used to exploit vulnerabilities in the car's network and attack ECUs leading to unsafe control actions [41]. As well, Bluetooth can be used for data exfiltration.

7. TPMS (Tire Pressure Monitoring System) – The TPMS transmits a signal to the TPMS ECU with a unique identifier and tire pressure information [65]. This signal can be sniffed, and the car tracked [65]. In addition, spoofing attacks can send false tire pressure data to the vehicle, and DoS attacks are also possible using signal jamming [65].

8. Passive Keyless Entry (PKE) – PKE sends codes between the key fob and the car via a wireless signal that enables doors to automatically unlock and the ignition to start when the key is within proximity [71]. An adversary can intercept and spoof these codes allowing the contents of the car or even the entire vehicle itself to be stolen [67], [68], [69], [70].

9. Remote Entry Key Fob – The key fob can actively send commands wirelessly to the vehicle to lock and unlock doors. Replay and signal jamming attacks are possible and can be used to circumvent the protection given by rolling codes based on pseudorandom number generators [71], [72].

10. Dedicated Short-Range Communication (DSRC) (e.g., Vehicle-to-Everything (V2X)) – The DSRC can transmit and receive information with enabled cars and infrastructure [152]. An adversary could spoof this information which could lead to unsafe

control actions. As well, a DoS attack could also subsequently lead to unsafe control actions.

11. Wi-Fi – The car can connect wirelessly to Wi-Fi, sending and receiving data between the router and exposing the vehicle to the internet. Spoofing attacks such as a car connecting to an adversarial network can give hackers an entry point to the car's internal network leading to attacks that result in unsafe control actions, including data exfiltration. The adversarial network could be integrated with a drone enabling a drone-based Wi-Fi network attack [74].

    Long-range Attack Surface [40]:

12. Cellular – A cellular connection that can access the car's internal network is excellent for emergencies and diagnostics but gives an attacker a point of entry from anywhere with an internet connection. Cellular is arguably the most dangerous part of a car's attack surface and can lead to remote compromise of safety systems leading to unsafe control actions [41], [66]. Cellular is also vulnerable to a DoS attack.

13. Telematics – The telematics unit is used to monitor vehicles remotely, leveraging a cellular-based internet connection. Like cellular, this is one of the most dangerous parts of a vehicle's attack surface. Depending on the network architecture, an attacker could exploit this connection remotely and launch attacks that can compromise ECUs leading to unsafe control actions [41], [66]. Also, Telematics are vulnerable to DoS attacks.

14. GPS – The GPS is used for localization and navigation, errors of which could lead to unsafe control actions being issued by an autonomous vehicle with incorrect position information. GPS is vulnerable to both DoS and spoofing attacks.

15. Smartphone – The smartphone can be used as a wireless controller of many features of a car over the internet, Wi-Fi, and Bluetooth. As a result, the connectivity allows for exploiting features in the vehicle such as data exfiltration and issuing unauthorized control actions such as an attacker turning the air conditioning on and off repeatedly to damage the battery [75].

16. Radio – digital signals can be transmitted through a radio signal. These signals can be used to supply information to systems in a car, such as navigation [76], [142]. Spoofing these signals can lead to false information being recorded, causing incorrect navigation [76]. Radio is also subject to DoS attacks.

The main types of attacks on the internal network of a car include [39]:

- Sniffing (e.g., Intercepting)
- Denial of Service (DoS)
- Spoofing

Hacking of cars has three significant categories per Miller and Valasek [40]:

- "Remote attack surfaces" [40:5]
- "Cyber-physical features" [40:5]
- "In-vehicle network architectures" [40:5]

Below is a history of key automotive security attacks.

5.2 History of Key Automotive Security Attacks

1. **Hacking traffic information signals**

   In 2007, security researchers from Inverse Path hacked a navigation system that received traffic updates over radio signals (Radio Data System – Traffic Message Channel (RDS-TMC)) [76],[142]. The lack of authentication used by the system allowed the researchers to spoof messages to the navigation system such as "closing roads" and "bad weather" alerts, and security messages such as "Air raid, danger" and "Bomb alert" [76].

2. **An angry former employee bricked over 100 cars just after he was fired [80]**

   An automotive centre in Texas that provides loans to customers with poor credit history used a 3rd party system created by Pay Technologies that acts as a collections tactic for customers that are delinquent on their payments [80]. The system allows a collections agent to remotely immobilize the engine and activate the horns on the car, which is a safer alternative to traditional collection tactics of sending a tow truck to repossess the vehicle [80]. This is because repossession can result in violence including cases when the repossession worker was shot and killed [80]. In February 2010, customers began complaining to the automotive centre that their car would not start and the horn would not turn off [80]. After an initial investigation, it was discovered that these customers were not delinquent on their payments [80]. After five days, over 100 customers complained about the same issue with their cars [80], [142]. An investigation led the police to an IP address of a recently dismissed employee that had used stolen login credentials from a former co-worker to remotely disable the engines and activate the horns of the cars [80]. The former employee was subsequently "charged with felony breach of a computer system" [80].

3. **"Security and privacy vulnerabilities of in-car wireless networks: a tire pressure monitoring system case study" [65]**

   In 2010, researchers from the University of South Carolina and Rutgers University demonstrated attacks on the tire pressure monitoring system (TPMS) [65], [142]. A sniffing attack allows tracking a vehicle from up to 40 metres away using unique identifiers transmitted by the TPMS [65]. The lack of authentication in the TPMS signal enables spoofing attacks to be executed [65]. The spoofing attacks were demonstrated to work between vehicles travelling at 110 kilometres per hour by activating the low tire pressure indicator in the dashboard of the targeted car [65].

4. **The University of San Diego and University of Washington researchers show that critical safety functions can be attacked over a car's internal network using local, short-range, and long-range connections**

   In 2010, a team of researchers from the University of San Diego and the University of Washington showed that virtually all the ECUs in a vehicle could be compromised given access to in-vehicle systems using the OBD-II port [42], [142]. The attack enabled them to compromise many critical safety systems, such as disabling the brakes and stopping the engine [42], [142]. Their attacks were met with criticism since they required physical access to the automobile where an attacker could have alternatively decided to cut the brake lines instead [41].

   In 2011, the teams further showed that they could remotely control the car's internal network via Bluetooth and cellular by taking control of the telematics unit, marking the first time that a car was experimentally demonstrated to be hackable remotely by an adversary with internet connectivity [41], [142]. In addition to sending unsafe control actions to the internal network, data exfiltration is possible by accessing the car's location via GPS and obtaining cabin audio using the hands-free calling microphone [41].

5. **"Connected cars are now a reality, but are they secure?"** [81]

In 2014, Kaspersky Lab and IAB conducted a connected car study using BMW's Connected Drive as a case study [81], [142]. They found that there were several attack vectors such as [81], [142]:

- Stolen credentials, which can be used to open and steal the car using the mobile app.
- If a user enables remote mobile app services, then a stolen phone is like a stolen key to the car.
- Bluetooth drivers for the car can be downloaded from BMW and installed via USB, which opens an attack vector where malicious software can be executed in the vehicle.
- A SIM card controls certain functions within the car, and an attacker could spoof control messages to the vehicle.

6. **Security researchers remotely attack Jeep over a cellular network, disabling critical safety functions such as the engine and brakes and demonstrate their attack on public roads**

   In 2015, Miller and Valasek discovered significant security vulnerabilities in a Jeep Cherokee [43], [66], [142]. The exploit allows for scanning cars connected to the internet and returns their VIN, make, model, IP address, and GPS coordinates [43], [66]. With this information, attackers could choose their victim from the list and send unsafe control actions to the vehicle over the cellular network [43], [66]. Miller and Valasek demonstrated their attack on a public road with journalist Andy Greenberg behind the wheel [43], [66]. They remotely activated the radio, windshield wipers, cabin fan and disabled the engine while on a highway, demonstrating the safety risk to Greenberg and other road participants [43], [66]. Other attacks include activating the turn signals, editing the speedometer, unlocking the doors, disabling the brakes, and controlling the steering [66]. Chrysler issued a recall for 1.4 million vehicles due to this research by Miller and Valasek [43], [66]. The share price of Fiat Chrysler dropped approximately 6% following the announcement of the recall [66].

7. **Hacking the OnStar Remote Link app with "OwnStar"**

   In 2015, Samy Kamkar created a piece of hardware for less than one hundred dollars leveraging a Raspberry Pi named "OwnStar" that can trick a mobile phone into joining his network and launch a man-in-the-middle (MITM) domain name system

(DNS) spoofing attack to obtain login credentials for a user's OnStar Remote Link app [72], [73], [142]. Once the credentials have been obtained, he can locate the car over GPS, unlock the doors, and remote start the vehicle [72], [73], [142].

8. **Keen Lab sends unsafe control actions over Wi-Fi to a Tesla, prompting the release of code signing**

   In 2016, Keen Security Lab from Tencent hacked a Tesla [142] over Wi-Fi and could overwrite the firmware on ECUs, enabling them to send unsafe control actions over the CAN bus through Wi-Fi [13]. The attack prompted Tesla to do an over-the-air (OTA) update requiring code signing on future ECU firmware updates to help mitigate

   In 2017, Keen Security Lab again hacked a Tesla [142] and discovered a way to bypass the code signing control added by Tesla in 2016 and upload custom firmware to the gateway ECU [77], [78]. In addition, the digital signature scheme used by Tesla is the Edwards-curve Digital Signature Algorithm (EdDSA) which is not a quantum-safe digital signature since it is based on the discrete log problem, which can be defeated with quantum cryptanalysis [23], [77], [78], [79].

9. **Disabling a vehicle's security system over Wi-Fi**

   In 2016, Ken Munro, a security researcher from Pen Test Partners, discovered a vulnerability in the Mitsubishi Outlander that allowed him to connect to the Wi-Fi access point embedded in the vehicle enabling him to disable the car's security system [82], [142].

10. **Hacking a Nissan Leaf using its mobile app**

    In 2016, Scott Helme discovered a security vulnerability in the mobile app for the Nissan Leaf, which allowed him to remotely query the current location and turn on the air conditioning of any leaf worldwide by simply knowing the VIN of the vehicle in question [75], [142]. The critical issue was that there was no authentication required to send these control actions to the cars [75]. Helme wrote a script that would turn on the air conditioner when the battery was 100% charged and let the battery drain to 95%. Then, turn off the air conditioning and repeat the process in a loop that can damage the battery in the vehicle [75]. Helme noted that this was not a traditional hack since there was no security system to circumvent, but rather a complete lack of security [75].

11. **"Robust Physical-World Attacks on Deep Learning Visual Classification" [44]**

    Computer vision is one of the vital perceptual technologies used in an autonomous vehicle. The classification engine is powered by deep neural networks (DNNs) [44]. In 2018, Eykholt *et al.* published research demonstrating that slight modifications of the physical world can manipulate the input to the classification algorithms used for computer vision in an autonomous vehicle [44]. Their research applied small perturbations to a stop sign by adding black and white stickers, leading to a 100% misclassification of images in the laboratory setting and 84.8% misclassification of images during a field test [44]. Their attacks led to a stop sign being classified as a speed limit 45 traffic sign which is a significant safety issue [44].

12. **Stealing a Tesla by attacking the key**

    In 2018, a team from Computer Security and Industrial Cryptography group (COSIC) at KU Leuven, including Wouters, created a method for cloning a key of a Tesla Model S in a few seconds, which allowed for unlocking and stealing the car [67],[68],[69].

    In 2020, a team from COSIC at KU Leuven, including Wouters, found vulnerabilities in a Tesla Model X car that allows stealing the car by getting within 5 metres of the key fob [67],[70]. Wouters discovered that he could overwrite the firmware on the key fob over a Bluetooth connection [67],[70]. The process takes about 1.5 minutes, and with the software on the key fob compromised, he was able to pair the key to the vehicle and unlock and start the car [67],[70].

13. **Hacking traction control of a Volkswagen Polo**

    In 2020, a consumer protection agency in the U.K. named "Which?" commissioned researchers from Context Information Security to hack a Volkswagen Polo to identify potential security issues [83], [142]. The researchers discovered that a flaw in the software update process for the infotainment system could be exploited, allowing them to tamper with the traction control system [83], [142]. When combined with another flaw in the key, they say that an attacker could break into a car and implement the hack in 5 minutes, putting a vehicle's safety at risk [83].

14. **Attack of the Drones**

    In 2021, security researchers Weinmann of Kunnamon and Schmotzle of Comsecuris disclosed a hack of a Tesla in which they accessed the infotainment unit of a car over Wi-Fi using a drone with a Wi-Fi dongle attached [74]. This allowed them to open the doors, control air conditioning, change the sound system's volume, and adjust the vehicle's seats [74].

Chapter 6

Overview of STAMP and STPA-Sec

Safety is the process of controlling a system to reduce hazards and losses to an "acceptable level of risk" [55].

The following is the definition of a loss: "*A loss involves something of value to stakeholders. Losses may include a loss of human life or human injury, property damage, environmental pollution, loss of mission, loss of reputation, loss or leak of sensitive information, or any other loss that is unacceptable to the stakeholders.*" [8:16]

The following is the definition of a hazard: "*A hazard is a system state or set of conditions that, together with a particular set of worst-case environmental conditions, will lead to a loss.*" [8:17]

Safety is vital since identified hazards and losses can be mitigated by proactive actions to protect the system's stakeholders through enhanced system control. Safety can often be enhanced by adding controls whose benefits exceed the cost of the controls.

Accidents have traditionally been modeled as component failures based on reliability; however, a systems theory approach is used in the System-Theoretic Accident Model and Processes (STAMP) [7]. Safety issues emerge from both component failures, and interactions of components and losses occur when safety constraints are not enforced on the system [7]. STAMP addresses the limitations of the component failure frameworks by considering both component failures and interactions, which means it analyzes the same causes compared to the traditional frameworks plus additional causes due to interactions [7]. The framework has three components: safety constraints, control structure, and process model [7]. These are leveraged by STPA (System-Theoretic Process Analysis), discussed next.

STPA is a hazard analysis technique whose goal is to identify the causes of hazards and losses. STPA is different from other hazards analysis techniques such as Failure Mode

Effect Analysis (FMEA) and Failure Mode, Effect, and Criticality Analysis (FMECA). The difference is because FMEA and FMECA rely on studying each component separately and coming up with a probability of failure of each component to estimate the system's safety [8]. STPA models safety as not just component failure but emergent from interactions of the components of the system, as does STAMP from which STPA is based [7], [8]. STPA has the following steps [7], [8]:

1. Identify Losses, Hazards, and System-level safety constraints.
2. Create a diagram of the control structure of the system.
3. Derive the set of unsafe control actions (UCAs) from the diagram in Step 2.
4. Identify the causes of losses for each UCA.

In addition to STPA, a modification called STPA-Sec (System-Theoretic Process Analysis for Security) is used to identify security-related causes of losses. The main difference with the analysis is in identifying the causes of losses where the control loop is analyzed for ways that errors could have been caused by an adversary that intentionally did one of the following to a component of the control loop as per Levenson [8:48]:

- Injected
- Spoofed
- Denied Service (i.e., Denial of Service (DoS))
- Tampered
- Intercepted
- Disclosed

Determining the security-related causes of losses for each UCA is of particular interest when an adversary can access a large-scale quantum computer. An adversary with access to a large-scale quantum computer could cause unsafe control actions by breaking public-key cryptographic protocols commonly used for digital signatures and key establishment (e.g., RSA).

Chapter 7

STPA-Sec Analysis on the Path Planning System

STPA-Sec will be applied to an autonomous vehicle's path planning control loop to identify the system's causes of hazards and losses. This will be done by defining the problem, creating a model of the control structure, identifying the UCAs, and finally identifying the causes of the UCAs by an adversary with access to a large-scale quantum computer.

7.1 Define Purpose of Analysis

The hazard analysis aims to identify the causes of unsafe control actions that can lead to losses so actions can be taken to design the system to prevent the hazards and losses from materializing, which requires defining the hazards and losses. Also, system-level constraints must be created to provide requirements to the engineers to prevent hazards in their creations [7].

7.1.1 Define & Frame the Problem

In order to define the purpose of the analysis, the scope, purpose, goals, and key stakeholders of the system that is being analyzed must be defined [9]. The analysis focuses on autonomous vehicles, either taxi-operated or individually owned, including their passengers and other road participants such as pedestrians, and how to get to the destination safely and securely while delivering superb customer satisfaction and shareholder value [9],

[150]. The adversary in this model has access to large-scale quantum computers and can therefore execute quantum cryptanalysis on the cryptographic systems protecting the car. A car has many communication systems, such as vehicle-to-everything (V2X), Wi-Fi, cellular, and Bluetooth. However, this work focuses on the path planning control loop within the car that impacts autonomous decision-making with material impact to the steering, acceleration, and brakes. Passengers, other road participants (e.g., pedestrians), vehicle manufacturers, and vehicle operators are key stakeholders.

7.1.2 Identifying Losses

In the context of an autonomous vehicle's path planning process, the losses used in the analysis are the following, similar to Levenson [8:16]:

- *L1: Loss or harm to life*
- *L2: "Loss or damage to the vehicle"* [8:16]
- *L3: "Loss or damage to objects outside the vehicle"* [8:16]
- *L4: "Loss of customer satisfaction"* [8:16]
- *L5: Loss of confidential information*
- *L6: Loss of shareholder value*

L1 could result from the death or injury of passengers or other road participants. Losses, in this case, can also include death or injury to pets such as a dog.

L2 could result from a dent in the bumper or something more severe like irreparable damage to the engine where the car becomes a write-off.

L3 could result in damage to a streetlamp, or there could be a major accident with another car such that the other vehicle becomes a write-off.

L4 could result from a passenger being unsatisfied with their autonomous taxi experience and providing a poor social media rating. The loss of customer satisfaction could become widespread via a viral social media post.

L5 could result from confidential information in the car, such as sensor data being intercepted and breached.

L6 could result from market reactions to news of hazardous behaviour of an autonomous vehicle governed by behavioural economics.

7.1.3 Identifying System-level Hazards

The following are the system-level hazards where the system is defined as the path planning control loop similar to [8], [9], [10]:

- *H1 Vehicle is taking a safe but incorrect legal path [L4, L6]*
- *H2 Vehicle runs low on energy [L1, L2, L3, L4, L6]*
- *H3 Vehicle uses quantum-vulnerable public-key cryptography [L1, L2, L3, L4, L5, L6]*
- *H4 Vehicle violates traffic laws [L1, L2, L3, L4, L6]*
- *H5 Vehicle is too close to an external object [L1, L2, L3, L4, L6]*

H1 is defined as not taking the correct legal path, which excludes paths that violate traffic laws. Taking a safe but incorrect legal path could lead to L4 and L6 by taking suboptimal paths such as driving around a block for no reason.

H2 is defined as the vehicle running low on energy. H2 would mean that there is a risk that it cannot reach its destination, such as the one requested by a passenger, or to make it to the nearest recharging station. Running out of energy in the middle of a highway or tunnel could result in L1, L2, L3 if there is a collision. If this happens in a safe location with a passenger in the vehicle, L4 would result. If this were a frequent occurrence, news of this issue could result in L6.

H3 is defined as having insecure communication within the path planning control loop using quantum-vulnerable public-key cryptography for authentication and key establishment. Using quantum-safe public-key cryptography is not considered a hazard, assuming it meets the performance requirements. Forged authentication for software and control actions within the path planning control loop could result in unsafe control actions being executing leading to L1, L2, and L3. Breaching data within the path planning control loop could lead to L5. Finally, breach of data and unsafe control actions sent to actuators such as steering would result in negative customer experience and loss of shareholder value in most cases leading to L4 and L6, respectively.

H4 includes driving too fast, including disobeying stops signs and traffic lights. H4 can lead to various losses such as L1, L2, L3. An autonomous vehicle that breaks traffic laws will likely negatively impact customer experience and could also negatively impact shareholder value.

H5 could result from being too close to another car due to an unanticipated erratic move by the other vehicle. Two cars that are too close to each other run the risk of a collision that could result in L1, L2, and L3. Also, an autonomous vehicle that does not maintain a safe distance from other objects would likely result in a negative customer experience and possible loss in shareholder value.

Drive.ai is an example of a company with unsatisfactory disengagement rates in California, once every 53 miles driven in 2019 [50], that filed a notice of closure of the company in 2019 to the State of California [61], [143]. They were subsequently bought by Apple days before their closure occurred [62]. The unsatisfactory disengagement rates could have contributed to the announced closure of the company and the loss of shareholder value. Another example of loss in shareholder value due to unsafe driving is NIO [85]. The driver was in a fatal accident with Navigate on Pilot engaged [85]. The share price of NIO fell from 42.47 USD on August 12th, 2021 to 36.29 USD on August 19th, 2021 on the New York Stock Exchange (NYSE), approximately a 15% drop in share price in one week which is correlated with the accident [63].

7.1.4 Identifying System-level Constraints

The definition of a system-level constraint is the following: "*A system-level constraint specifies system conditions or behaviours that need to be satisfied to prevent hazards (and ultimately prevent losses).*" [8:20]

The following are the system-level constraints designed to prevent system-level hazards:

- *SC1 - Vehicle shall take the correct path [H1, H2, H4, H5]*
- *SC2 – Vehicle shall not get low on energy [H2, H5]*
- *SC3 – Vehicle shall use quantum-safe cryptography where performance requirements allow for it [H3]*
- *SC4 - Vehicle shall obey traffic laws [H2, H4]*
- *SC5 – There shall be sufficient distance between vehicle and external objects [H4, H5]*
- *SC6 - If the vehicle is taking a safe but incorrect legal path, then there shall be a system to identify this error and correct the course.*
- *SC7 - If the vehicle is at risk of being low on energy, then there shall be a system to identify this and recharge.*

- *SC8 - If the vehicle violates traffic laws by speeding, then there shall be a system to identify this violation and reduce the speed to a legal level assuming it is safe to do so.*
- *SC9 - If the vehicle gets too close to an external object, then there shall be a system to identify this and move the car to an appropriate location.*

Note: SC1 is connected to SC6, SC2 is connected to SC7, SC4 is connected to SC8, and SC5 is connected to SC9.

SC1 means that the vehicle must take the optimal legal path for safety and efficiency. If it does not take that path, then H1 could occur if it takes another legal path. H2 could happen if the path taken depletes the energy. If the optimal legal path is not taken, then an illegal path that disobeys traffics laws may have been taken, in which case H4 would result. Finally, the incorrect path could place the car too close to other objects resulting in H5.

SC2 is violated when the vehicle runs low on energy, which could result in H5 if the vehicle runs out of energy on the road, causing a collision. If SC2 is violated, then it implies H2 by the definition of SC2.

If **SC3** is violated, then it implies H3 by the definition of SC3.

SC4 could be violated if the vehicle is speeding, resulting in H2 due to inefficient energy use. If SC4 is violated, then it implies H4 by the definition of SC4.

If **SC5** is violated, then H4 could occur if another car gets so close that a collision violates traffic laws, such as speeding that leads to a collision. If SC5 is violated, then it implies H5 by the definition of SC5.

IF **SC6** is violated, the system does not course correct, resulting in L4 and L6 in the worst-case scenario.

If **SC7** is violated, the system does not correct the destination, resulting in L1, L2, L3, L4, and L6 in the worst-case scenario.

If **SC8** is violated, the system does not make an appropriate adjustment for speed, resulting in L1, L2, L3, L4, and L6 in the worst-case scenario.

If **SC9** is violated, the system does not move the car to an appropriate distance from an external object, resulting in L1, L2, L3, L4, and L6 in the worst-case scenario.

7.2 Control Structure Diagram

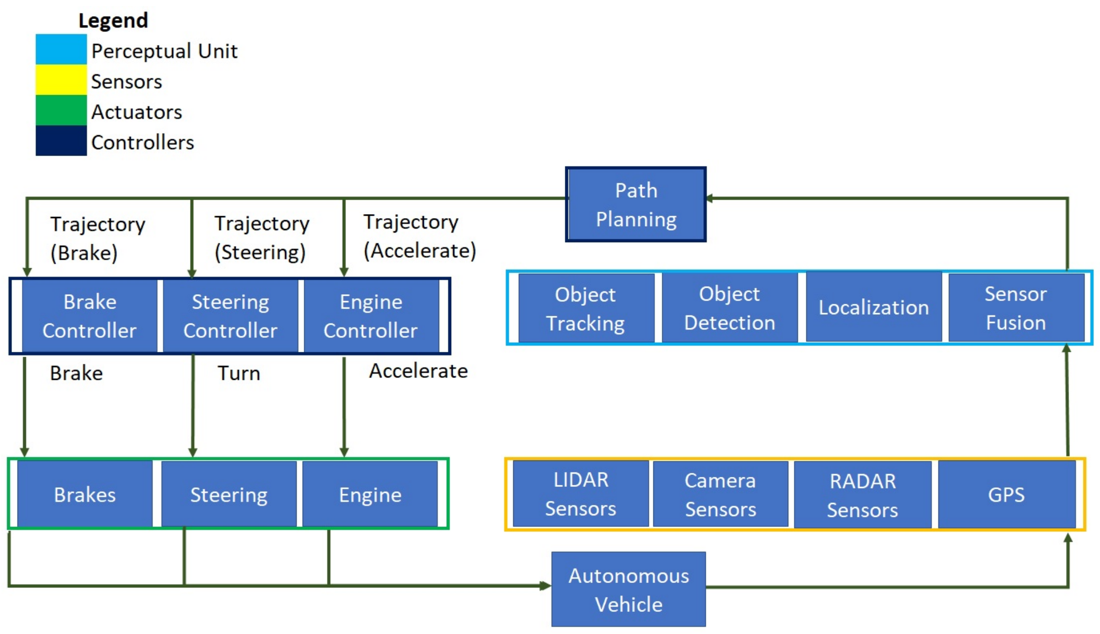


Figure 7.1: Functional Control Structure Diagram.
Note: Figure is modeled after similar diagrams in S. Sharma "Considering saftey and security in av functions", 2019, master of applied science thesis. p.20 [10], Abdulkhaleq *et al.* "A systematic approach based on STPA for developing a dependable architecture for fully automated driving vehicles" 2017, Procedia Engineering, vol. 179, p.47 [11], [30], [57].

Figure 7.1 above illustrates the path planning control loop. The loop contains sensors such as cameras, LIDAR, RADAR, and GPS. The feedback measured by the sensors is sent to the perceptual unit, which builds a model of the environment using tools such as object detection and object tracking. Also, it uses maps and GPS for localization to understand its position within the perceived environment. The perceptual information is sent to the path planning system used as constraints in the optimization problems to calculate the optimal passage through its environment. Path planning executes its strategy by sending instructions to the vehicle controllers, which are relayed as control actions to the brakes, steering, and engine actuators. The actuators subsequently control the motion of the autonomous vehicle, and the control loop repeats itself until the car arrives at its destination.

7.3 Unsafe Control Actions (UCAs)

The following is the definition of an unsafe control action: "*An unsafe control action (UCA) is a control action that, in a particular context and worst-case environment, will lead to a hazard.*" [8:35]

The control structure diagram from Section 7.2 is used to create an inventory of unsafe control actions (UCAs). From the control structure diagram, a control action is analyzed by the following four criteria to see if it could result in a UCA as per Levenson [8:36]:

1. "Not providing the control action leads to a hazard." [8:36]
2. "Providing the control action leads to a hazard." [8:36]
3. "Providing a potentially safe control action too early, too late, or in the wrong order." [8:36]
4. "The control action lasts too long or is stopped too soon (for continuous control actions, not discrete ones)." [8:36]

The controllers that are analyzed for unsafe control actions are:

1. Path Planning System
2. Brake Controller
3. Steering Controller
4. Engine Controller

7.3.1 Path Planning System

Note: "n/a" in Tables 7.1, 7.2, and 7.3 below indicates no applicable UCA since the trajectory control action is not continuous.

Control Loop: Path Planning to Autonomous Vehicle	
Control Action: Trajectory (Brake)	
UCA	Description
Not providing causes hazard	UCA1.1 Path planning system does not send an appropriate trajectory while the vehicle is in motion and needs to come to a stop or to slow down. (H1, H2, H3, H4, H5)
Providing causes hazard	UCA1.2 Path planning system sends an inappropriate trajectory requiring braking while the vehicle needs to accelerate. (H1, H2, H3, H4, H5)
Too early, too late, out of order	UCA1.3 Path planning system sends a trajectory too late when the vehicle needs to slow down or come to a stop. (H1, H2, H3, H4, H5)
Stopped too soon, applied too long	n/a

Table 7.1: Path Planning Trajectory (Brake) UCAs

UCA 1.1 is an example of when not providing a control action leads to a hazard.

There are three UCAs listed in Table 7.1 above for the trajectory sent from the path planning unit to the brake controller. When UCA 1.1 occurs, H1 may occur since a vehicle that does not slow down or stop when required could be taking an incorrect legal path. H2 is also possible as the vehicle could travel until it runs out of energy if it does not apply the brakes when required. H3 could result if the control algorithm issuing the UCA uses quantum-vulnerable public-key cryptography. Similarly, H4 may occur if the brakes need to be applied to obey traffic laws such as speed limits, red lights, or stop signs. Finally, UCA 1.1 might result in H5 because the vehicle is too close to another car.

Control Loop: Path Planning to Autonomous Vehicle	
Control Action: Trajectory (Steer)	
UCA	Description
Not providing causes hazard	UCA2.1 Path planning system does not send an appropriate trajectory when there is an object that it needs to maneuver around while the vehicle is in motion. (H1, H2, H3, H4, H5)
Providing causes hazard	UCA2.2 Path planning system provides inappropriate trajectory resulting in the vehicle changing lanes when it does not need to change lanes. (H1, H2, H3, H4, H5)
Too early, too late, out of order	UCA2.3 Path planning system provides trajectory too late when the vehicle needs to make a lane change. (H1, H2, H3, H4, H5)
Stopped too soon, applied too long	n/a

Table 7.2: Path Planning Trajectory (Steer) UCAs

UCA 2.2 is an example of providing a control action that leads to a hazard.

The table above corresponds to the three UCAs for the steering trajectory issued by the path planning system. H1 could occur if the vehicle changes lanes and travels along a different legal path. H2 could occur if the car changes lanes and exits a highway when it is not supposed to and runs out of energy. H3 could result if the control algorithm issuing the UCA uses quantum-vulnerable public-key cryptography. H4 could happen because of the car driving into oncoming traffic violating traffic laws. H5 could occur if the car changes lanes and hits another vehicle.

Control Loop: Path Planning to Autonomous Vehicle	
Control Action: Trajectory (Accelerate)	
UCA	Description
Not providing causes hazard	UCA3.1 Path planning system does not send an appropriate trajectory when vehicle needs to change lanes. (H1, H2, H3, H4, H5)
Providing causes hazard	UCA3.2 Path planning system provides an inappropriate trajectory when the vehicle needs to slow down or to stop. (H1, H2, H3, H4, H5)
Too early, too late, out of order	UCA3.3 Path planning system provides trajectory too late while the vehicle is in an intersection. (H1, H3, H4, H5)
Stopped too soon, applied too long	n/a

Table 7.3: Path Planning Trajectory (Accelerate) UCAs

UCA 3.3 is an example of a control action issued too late, leading to hazards in the worst-case scenario.

Table 7.3 above contains three UCAs regarding the acceleration trajectory issued by the path planning system. H1 and H4 are possible as the vehicle could take a legal or illegal path through the intersection depending on traffic flow without the guidance of an updated trajectory. H3 could result if the control algorithm issuing the UCA uses quantum-vulnerable public-key cryptography. H5 could occur if the vehicle accelerates too late and gets into a collision in the intersection.

7.3.2 Brake Controller

Control Loop: Brake Controller to Autonomous Vehicle	
Control Action: Brake	
UCA	Description
Not providing causes hazard	UCA4.1 Brake controller does not send a brake control action while the vehicle is in motion and needs to come to a stop or to slow down. (H1, H2, H3, H4, H5)
Providing causes hazard	UCA4.2 Brake controller sends a brake control action while the vehicle is accelerating. (H1, H2, H3, H4, H5)
Too early, too late, out of order	UCA4.3 Brake controller sends a brake control action too early while planning to come to a stop. (H1, H3, H4, H5) — UCA4.4 Brake controller sends a brake control action too late while trying to slow down or come to a stop. (H1, H2, H3, H4, H5)
Stopped too soon, applied too long	UCA4.5 Brake controller stopped the brake control action too soon when the vehicle needed to slow down or to stop. (H1, H3, H4, H5) — UCA4.6 Brake controller applied the brake control action too long when the vehicle needed to slow down. (H1, H3, H4, H5)

Table 7.4: Brake Controller UCAs

UCA 4.3 is an example of providing a control action too early, leading to hazards.

Table 7.4 above contains a list of UCAs for the brake control action issued by the brake controller. H1 could happen if the car stops too soon and remains on a legal path. H3 could result if the control algorithm issuing the UCA uses quantum-vulnerable public-key cryptography. H4 could happen if the brakes are applied too soon and the car is stopped in an intersection, violating traffic laws. H5 could occur if the vehicle applies the brakes too soon and gets too close to another vehicle.

7.3.3 Steering Controller

Control Loop: Steering Controller to Autonomous Vehicle	
Control Action: Turn	
UCA	Description
Not providing causes hazard	UCA5.1 Steering controller does not send a turn control action when there is an object that it needs to maneuver around while the vehicle is in motion. (H1, H3, H4, H5)
Providing causes hazard	UCA5.2 Steering controller provides turn control action when it does not need to change lanes. (H1, H2, H3, H4, H5)
Too early, too late, out of order	UCA5.3 Steering controller provides turn control action too early when making a lane change. (H1, H2, H3, H4, H5) — UCA5.4 Steering controller provides turn control action too late when making a lane change. (H1, H2, H3, H4, H5)
Stopped too soon, applied too long	UCA5.5 Steering controller stops the turn control action too soon when maneuvering around a curve. (H1, H2, H3, H4, H5) — UCA5.6 Steering controller applies the turn control action too long when maneuvering around a curve. (H1, H2, H3, H4, H5)

Table 7.5: Steering Controller UCAs

UCA 5.5 is an example of a control action stopped too soon that leads to hazards.

Above is the table corresponding to the UCAs for the steering control actions issues by the steering controller. H1 could occur if the vehicle stops turning when attempting to make a lane change and stays in the wrong lane. H2 could occur if the vehicle takes the wrong exit off a highway and uses too much energy. H3 could result if the control algorithm issuing the UCA uses quantum-vulnerable public-key cryptography. H4 could occur if the vehicle begins driving on the shoulder without a reason to pull over. H5 could occur if the vehicle gets too close while trying to pass another vehicle due to stopping the steering control action too soon.

7.3.4 Engine Controller

Control Loop: Engine Controller to Autonomous Vehicle	
Control Action: Accelerate	
UCA	Description
Not providing causes hazard	UCA6.1 Engine controller does not send an acceleration control action when the vehicle is changing lanes. (H1, H2, H3, H4, H5)
Providing causes hazard	UCA6.2 Engine controller provides an acceleration control action when the vehicle needs to slow down or to stop. (H1, H2, H3, H4, H5)
Too early, too late, out of order	UCA6.3 Engine controller provides an acceleration control action too early while the vehicle is near an intersection. (H1, H3, H4, H5) — UCA6.4 Engine controller provides an acceleration control action too late while the vehicle is in an intersection. (H1, H3, H4, H5)
Stopped too soon, applied too long	UCA6.5 Engine controller stopped acceleration control action too soon while the vehicle is in an intersection. (H1, H3, H4, H5) — UCA6.6 Engine controller applied acceleration control action too long while the vehicle is changing lanes. (H1, H2, H3, H4, H5)

Table 7.6: Engine Controller UCAs

UCA 6.6 provides an example of applying a control action too long, causing hazards.

Table 7.6 above contains a list of the unsafe control actions associated with the engine controller. H1 could occur if the car changes into the wrong lane. Similarly, H2 could arise if it changes into the wrong lane and exits the highway. H3 could result if the control algorithm issuing the UCA uses quantum-vulnerable public-key cryptography. H4 could occur if the car cross lanes into oncoming traffic. Finally, H5 could arise if the car hits another car due to changing into the wrong lane.

7.4 Controller Constraints

UCAs	Controller Constraints
UCA1.1 Path planning system does not send an appropriate trajectory while the vehicle is in motion and needs to come to a stop or to slow down. (H1, H2, H3, H4, H5)	C1.1 Path planning system must send an appropriate trajectory while vehicle is in motion and needs to come to a stop or to slow down.
UCA2.2 Path planning system provides inappropriate trajectory resulting in the vehicle changing lanes when it does not need to change lanes. (H1, H2, H3, H4, H5)	C2.2 Path planning system must not provide an inappropriate trajectory when it does not need to change lanes.
UCA3.3 Path planning system provides trajectory too late while the vehicle is in an intersection. (H1, H3, H4, H5)	C3.3 Path planning system must not provide trajectory too late while near an intersection.
UCA4.3 Brake controller sends a brake control action too early while planning to come to a stop. (H1, H3, H4, H5)	C4.3 Brake controller must not apply brakes too early while planning to come to a stop.
UCA5.5 Steering controller stops the turn control action too soon when maneuvering around a curve. (H1, H2, H3, H4, H5)	C5.5 Steering controller must not stop the turn control action too soon when maneuvering around a curve.
UCA6.6 Engine controller applied acceleration control action too long while the vehicle is changing lanes. (H1, H2, H3, H4, H5)	C6.6 Engine controller must not apply acceleration control action too long while vehicle is changing lanes.

Table 7.7: Sample of Controller Constraints

The following is the definition of a controller constraint: *"A controller constraint specifies the controller behaviors that need to be satisfied to prevent UCAs."* [8:41]

Above is a sample of safety constraints that must be adhered to at the controller level to help stop UCAs from occurring. A comprehensive list of controller constraints for all of the UCAs can be found in Appendix A.

7.5 Causes of UCAs

The following control loop diagram can be used to diagnose the causes of UCAs.

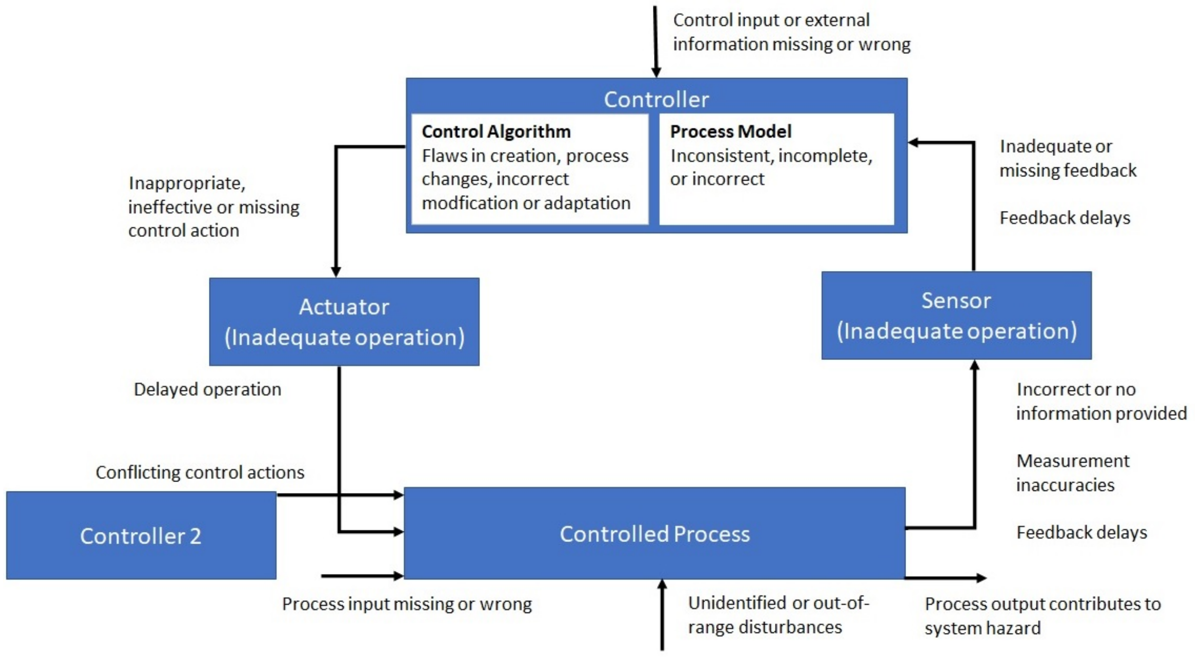

Control input or external information missing or wrong
Controller
Control Algorithm
Flaws in creation, process changes, incorrect modfication or adaptation
Process Model
Inconsistent, incomplete, or incorrect
Inappropriate, ineffective or missing control action
Inadequate or missing feedback
Feedback delays
Actuator (Inadequate operation)
Sensor (Inadequate operation)
Delayed operation
Incorrect or no information provided
Measurement inaccuracies
Conflicting control actions
Controller 2
Controlled Process
Feedback delays
Process input missing or wrong
Unidentified or out-of-range disturbances
Process output contributes to system hazard

Figure 7.2: Diagnostic Control Loop as described in N.G. Levenson, "An STPA Primer, Version 1", 2013, MIT, p. 48 [7] and N.G. Levenson, "Engineering a safer world: Systems thinking applied to safety", 2011, The MIT Press, p.223 [12]

There are a variety of ways that the control loop could fail, including [8]:

1. Input to the controller had an error.
2. Flawed control algorithm.
3. The process model does not accurately reflect reality.
4. The control action was insufficient.

5. The actuator hardware failed.
6. There was a delay in the operation of the actuator.
7. There were feedback delays
8. Feedback measurement error.
9. The sensor hardware failed.
10. Feedback was corrupted that was sent to the controller by the sensor.

A modification is necessary to study security reasons for the control loop failing. For security, consider how the above failures could be caused by an adversary per Levenson by [8:48]:

- Injection
- Spoofing
- Denial of Service (DoS)
- Tampering
- Intercepting
- Disclosing

UCA	Cause	Scenario
UCA1.1 Path planning system does not send an appropriate trajectory while the vehicle is in motion and needs to come to a stop or to slow down. (H1, H2, H3, H4, H5)	The control algorithm has been tampered with	Scenario 1 - The path planning control algorithm intentionally does not send an appropriate trajectory when an obstacle is detected in the vehicle's path. This is due to a malicious software update which used a quantum computer to break the authentication protocol.
UCA1.1 Path planning system does not send an appropriate trajectory while the vehicle is in motion and needs to come to a stop or to slow down. (H1, H2, H3, H4, H5)	Inconsistent process model Incorrect Feedback	Scenario 2 - The path planning system receives incorrect information from the obstacle detection, obstacle tracking, and localization systems that has been spoofed by a malicious software update to the perceptual unit which used a quantum computer to break the authentication protocol. This results in the path planning system not sending an appropriate trajectory when an obstacle is detected in the vehicle's path.

Table 7.8: Loss Scenarios 1 & 2 for UCA 1.1

For example, identify ways an adversary could spoof a control action or tamper with the control algorithm.

The focus of the analysis will be to understand how an adversary with access to a large-scale quantum computer could perform quantum cryptanalysis to compromise the various failure points of the control loop for each unsafe control action.

The table above contains the causes of UCA 1.1 due to attacks by an adversary with a large-scale quantum computer. In Scenario 1, a malicious software update was installed into the path planning system, which caused the car to intentionally not apply the brakes when required. UCA1.1 was enabled by an adversary that broke the quantum-vulnerable digital signature allowing the malware to pass authentication and be installed in the vehicle.

Scenario 2 takes another angle and considers when the information received has been spoofed by the obstacle detection, obstacle tracking, and localization systems. The cause is the same as Scenario 1, a malicious software update by an adversary that broke the quantum-vulnerable digital signature. In Scenario 2, the control algorithm could be operating fine within the path planning unit. However, the spoofed information results in an incorrect process model, which subsequently causes the path planning control algorithm to issue a UCA due to a false understanding of the environment.

UCA	Cause	Scenario
UCA5.2 Steering controller provides turn control action when it does not need to change lanes. (H1, H2, H3, H4, H5)	The control algorithm has been tampered with	Scenario 33 - The steering controller intentionally sends a turn control action when it does not need to change lanes. This is due to a malicious software update which used a quantum computer to break the authentication protocol.

Table 7.9: Loss Scenarios 33 for UCA 5.2

Similarly, malicious software updates can impact the steering, brake, and engine controllers due to an adversary with a large-scale quantum computer, as illustrated in Table 7.9 above in Scenario 33.

The steering, brake, and engine controllers can also be indirectly impacted by a malicious software update to the path planning system even if the software in the controllers has not been modified, as illustrated in Scenario 50 in Table 7.10 below.

UCA	Cause	Scenario
UCA6.4 Engine controller provides an acceleration control action too late while the vehicle is in an intersection. (H1, H3, H4, H5)	Inconsistent process model and Incorrect Input	Scenario 50 -The engine controller receives incorrect information from the path planning system regarding an acceleration control action that is due to an attack which used a quantum computer to break the authentication protocol causing the vehicle to accelerate too late while in an intersection.

Table 7.10: Loss Scenarios 50 for UCA 6.4

The complete inventory of loss scenarios can be found in Appendix B. All the UCAs can be caused by an adversary that has broken the quantum-vulnerable digital signature used for code signing with a quantum computer and installed malware into one or more systems.

Chapter 8

Attack Details

Attacking the software updates in an autonomous vehicle by forging the digital signatures used for code signing is how a quantum computer could attack the path planning control loop. The path planning control loop attack includes attacking the perceptual unit, path planning system, steering, brake, and engine controllers. This attack could result in several UCAs leading to hazards and losses in the worst-case scenario. The following examples will illustrate the feasibility of a software update attack on autonomous vehicles using a quantum computer.

First, an attack on a Tesla by Keen security lab from Tencent allowed the team to control the vehicle over Wi-Fi, resulting in Tesla deploying an update to their cars requiring code signing [6]. The attack works while the vehicle is parked and in motion, and this is the first publicly known attack that enables remote control of a Tesla by exploiting the controller area network (CAN Bus) [13]. The purpose of the code signing update was to prevent the electronic control units (ECUs) from running unauthorized software that could enable them to send UCAs such as braking while in motion when the vehicle does not need to slow down or stop [13].

Another well-known attack is Stuxnet, in which malware was installed in nuclear centrifuges in Iran, causing them to break [27]. The attack used two private keys to forge digital signatures on drivers stolen from JMicron and Realtek [27]. The system infected controllers that changed the rotational speed of the centrifuges to unsafe values until they broke [27]. Estimates are that approximately 1,000 centrifuges were damaged in the Natanz nuclear facility [27]. Due to the malware's level of sophistication, including using four zero-day exploits in the attack, it is suspected that Stuxnet was a nation-state-sponsored attack [27].

The two attacks above were selected because they demonstrate that a software update attack on autonomous vehicles is feasible. In the exploit discovered by the Keen security lab, it was shown that it is possible to update the firmware on ECUs and send unsafe control actions over a wireless connection causing Tesla to deploy code signing to their cars [6]. This defence only works if the digital signatures used in the code signing are resilient to quantum cryptanalysis. Subsequent analysis by Keen security lab shows that Tesla is using EdDSA for its code signing, which an adversary can break with a large-scale quantum computer [23], [77], [78], [79]. In the Stuxnet example, we saw how malware could be used to send unsafe control actions to nuclear centrifuges [96], and one of the main components of the attack was that two stolen private keys were used to sign software. Combining these attacks and adding a nation-state with access to a large-scale quantum computer yields the ingredients needed to implement a Stuxnet-style attack on autonomous vehicles. However, you can quantum-compute the private key instead of breaking into a vehicle manufacturer's office and stealing it.

Below is an excerpt from Section 8.2 regarding cryptographic credentials from the document "Cyber Security Practices for the Safety of Modern Vehicles" published by the NHSTA in 2020, which states that:

"*Any credential obtained from a single vehicle's computing platform should not provide access to multiple vehicles.*" [53:13]

The statement above is significant for a quantum-enabled software update attack. Automotive manufacturers should be vigilant to ensure that each automobile's private-public key pair is unique; otherwise, an attacker could obtain a single public key from any vehicle in the market and quantum-compute the private key needed to forge digital signatures that would authenticate across the entire fleet.

Note that a software update attack on a car using a quantum computer is not a newly discovered attack method. For example, Alexander Truskovsky from ISARA discussed using a quantum computer to attack the software update in 2017 [56].

Chapter 9

Confidential Information

Confidentiality within a vehicle also needs consideration. The following is a list of confidential information that may require public-key cryptography [33]:

1. External camera data
2. Sensor data (e.g., LIDAR, RADAR, Ultrasonic, and Thermal Imaging)
3. GPS data (i.e., current location, previous locations)
4. Vehicle software data
5. Event Data Recorder (EDR) data
6. Internal camera data
7. Internal audio data
8. External audio data
9. Biometric data (e.g., facial recognition, voice recognition)
10. Medical data
11. Payment data
12. Cellphone data
13. Driving style data

Why is the data listed above confidential?

1. **External camera data**

   External camera data captures pedestrians' face and location information. As well as information on other road participants such as vehicles. This information is stored for perceptual analysis and building new machine learning models. This information would be classified as confidential for individual consumer vehicles and commercially operated fleets such as autonomous taxis. Failure of an autonomous taxis fleet to protect this confidential information could result in class action lawsuits.

2. **Sensor data (e.g., LIDAR, RADAR, Ultrasonic, and Thermal Imaging)**

   Although less evident than external camera data, forms of electromagnetic radiation other than visible light such as the kinds used in LIDAR (laser light), RADAR (microwave radiation), and thermal imaging (infra-red light), and other waves such as ultrasonic, could become classified as confidential information [33]. This is because observation of such information could reveal some personal characteristics of pedestrians and other road participants [33]. An autonomous taxi company recording millions of LIDAR, RADAR, thermal imaging, and ultrasonic scans of pedestrians and other road participants could face a class action lawsuit if this data is not treated as confidential.

3. **GPS data**

   Current and previous locations are confidential information that may be stored in a vehicle. Adversaries can use the current location to launch real-time physical attacks such as abducting passengers, and law enforcement can use previous locations for criminal investigations [33]. GPS data has also been used when couples divorce as evidence that their spouse was unfaithful for decisions such as awarding alimony [60].

4. **Vehicle software data**

   Vehicle software data, especially those containing proprietary artificial intelligence algorithms and security functions, could be valuable to adversaries, including the company's competitors, and would be classified as confidential.

5. **Event data recorder (EDR) data**

The event data recorder contains information describing the state of the vehicle before an accident. [151] Law enforcement or insurance companies could use this information for safety, legal, and criminal investigations and should be held confidential. [33], [151]

6. **Internal camera data**

   The internal camera data can be used for safety monitoring the driver, payments using facial recognition, vehicle security, and entertainment such as virtual and augmented reality, to name a few. This data will significantly benefit passengers regarding safety, security, and entertainment but must be held confidential.

7. **Internal audio data**

   Hands-free calling, passenger conversations, and commands issued to the car's computer could contain confidential information that needs protection.

8. **External audio data**

   Audio recordings from other road participants, such as pedestrians, would be considered confidential. Also, audio leading up to the moments before an accident could be used by insurance or in criminal investigation and would be confidential.

9. **Biometric data (e.g., facial recognition, voice recognition)**

   Biometric authentication will be used for security to prevent vehicle theft, safety to avoid hazards such as intoxicated driving, voice control of vehicle systems, and payments. The biometric data contains personally identifiable information (PII), which is confidential.

10. **Medical data**

    Medical diagnostic data captured in an ambulance or autonomous vehicle using sensors such as cameras or other tools would contain personal health information (PHI) that is confidential. DNA information could be used for diagnostic purposes and must be handled securely, especially for long-term confidentiality.

11. **Payment data**

Embedded payment systems apps within cars could conveniently allow users to pay for fuel, restaurants, and parking metres [59]. Still, they could carry confidential information about a credit card that could be subject to payment card industry data security standards (PCI-DSS) [59].

12. **Cellphone data**

    Contacts, call history, and text messages could be obtained and require security to protect confidential information.

13. **Driving style data**

    Data regarding driving style such as steering, braking, and acceleration may be stored in a vehicle and possibly shared with insurance companies [33]. This data can impact insurance premiums and should be treated as confidential [33].

To the best of the thesis author's knowledge, the in-vehicle network connecting the electronic control units (ECUs) does not currently use public-key cryptography to secure confidential information in any consumer vehicle. However, there are several papers on in-vehicle communication with protocols leveraging public-key cryptography. In [31] by Zelle *et al.* from Fraunhofer SIT and Audi, TLS 1.2 is evaluated on in-vehicle networks. In terms of confidentiality, it is noted that encryption may add additional overhead to the networks, such as latency and bandwidth [31]. RSA and ECDSA were selected for digital signatures to benchmark cryptographic protocols, and Elliptic Curve Diffie-Hellman (ECDHE) was selected for key exchange [31]. In a paper by Groza and Murvay from Politechnica University of Timisoara on confidentiality within an in-vehicle network, they propose using the Diffie-Hellman (DH) key exchange [32]. The common factor in the papers published above is that they are not using quantum-safe cryptography to secure their network. This means that an adversary with a large-scale quantum computer could breach confidential information, especially information that requires long-term confidentiality [15].

See Section 3 above for an outline of the quantum-safe key establishment algorithms currently in Round 3 of the NIST standardization process used to protect confidential information, which can mitigate the risk of breach of in-vehicle confidential information listed above.

Chapter 10

Suggested Improvements to STPA-Sec

This section contains ideas on how to improve STPA-Sec. One improvement would be to add a risk assessment component to the analysis. It currently only identifies causes of UCAs, resulting in hazards and losses in the worst-case scenario. STPA-Sec does not contain probability estimates for hazards because there could be components whose probabilities cannot be estimated in a complex system [7]. There is a risk that significant hazards could be ignored if the calculated probability is perceived as immaterial [7]. However, a risk-based approach for prioritizing which causes of UCAs to solve for first is recommended so that the most material risks are addressed first. Even if the probability of the hazard occurring is difficult to estimate, the severity given the hazards should be more manageable to estimate. A technique commonly used for risk modelling in cyber security is Monte Carlo simulation [37].

Another area to improve upon would be having a more explicit framework for cyber security. The NIST Cyber Security framework can be leveraged to provide more structure to the STPA-Sec analysis. The following are examples of sections of the NIST framework that would provide more clarity when doing the STPA-Sec analysis:

IDENTIFY (ID) – Asset Management (AM) [58]

"ID.AM-2: Software platforms and applications within the organization are inventoried" [58:24]

This is a useful exercise to go through during an STPA-Sec analysis, especially when considering a quantum-enabled adversary since knowledge of system-wide use of public-key cryptography is critical for risk mitigation. This is like Phase 1 in the Quantum Risk Assessment (QRA) framework by Mosca and Mulholland [15].

PROTECT (PR) – Data Security (DS) [58]

"PR.DS-2: Data-in-transit is protected" [58:32]

Knowledge of cryptographic protection of data-in-transit is important in the STPA-Sec analysis since it is vulnerable to quantum cryptanalysis if quantum-vulnerable public-key cryptography is used. In some cases, no public-key cryptography is used to protect data-in-transit but should be considered during the analysis as a possible risk to confidentiality with quantum-safe cryptography as the mitigant.

"PR.DS-6: Integrity checking mechanisms are used to verify software, firmware, and information integrity" [58:33]

Knowledge of which cryptographic protection is used for authentication and integrity in the STPA-Sec analysis is critical since a quantum-enabled adversary can forge quantum-vulnerable digital signatures. In some cases, no public-key cryptography is used to protect the authenticity and integrity of software, firmware, and information integrity, but should be considered during the analysis as a possible risk with quantum-safe cryptography as the mitigant.

DETECT (DE) – Anomalies and Events (AE) [58]

"DE.AE-3: Event data are collected and correlated from multiple sources and sensors" [58:38]

When doing the STPA-Sec analysis, it is important to understand which sensors are sending feedback to the controller and whether multi-sensor data fusion is used. An adversary spoofing a single sensor's feedback can cause UCAs to occur, whether the sensor's feedback is being used by itself or in a sensor fusion setting.

Cyber security can also be enhanced through improvements to the diagnostic control loop, specifically regarding cyber-security-related causes of losses. For example, the framework can be enhanced to include cryptographic causes of losses in a situation where the adversary has access to a large-scale quantum computer. Another enhancement would be to use existing quantum risk frameworks such as a QRA in addition to the existing STPA-Sec approach to evaluate the risk of cryptographically vulnerable UCAs. The analysis would include an assessment of when it is expected that an adversary would have the capabilities of executing a quantum-enabled attack [15].

Enhancement to the causes of UCAs: Quantum Cyber Security

Below are the six techniques that an adversary can use to attack the various components of a control loop as per Levenson [8:48]:

- Injecting
- Spoofing
- Denial of Service (DoS)
- Tampering
- Intercepting
- Disclosing

The quantum-enabled versions of the attacks to the control loop are described below:

Note: this is not a fundamentally new result, but it explicitly states how to perform the above cyber security attacks on the control loop with a quantum computer.

Quantum-enabled Injecting:

A quantum adversary could inject messages via a man-in-the-middle (MITM) attack. They could break the authentication protocol between the communicating parties by forging digital signatures.

Quantum-enabled Spoofing: Spoofing can also be conducted by breaking the authentication by forging digital signatures using a quantum computer. Spoofing can make messages appear as if they originate from a legitimate source, such as in a MITM attack.

Quantum-enabled Denial of Service: By leveraging quantum-enabled injecting and spoofing, an adversary could flood the network with messages attacking the availability of critical functions by breaking the authentication protocol with a quantum computer resulting in a quantum-enabled DoS attack.

Quantum-enabled Tampering: A file could be tampered with, and the adversary can disguise it as legitimate software by breaking the authentication and forging digital signatures using a quantum computer.

Quantum-enabled Intercepting: A file could be intercepted by eavesdropping and copying it. Suppose the file is encrypted and was communicated over a public channel using a quantum-vulnerable communication protocol such as TLS. In that case, it can be stored, and quantum cryptanalysis applied to the key exchange in the future.

Quantum-enabled Disclosing: Similarly, once the file has been copied, an adversary would only need to wait until there is a sufficiently powerful quantum computer to decrypt the file and disclose it.

Attacking the Controller with a Quantum Computer: The control algorithm can be tampered with and spoofed by an adversary that used a quantum computer to forge the digital signatures for software signing. Also, the process model can be tampered with by manipulating the feedback to the controller using a quantum computer such as via a MITM attack.

Attacking the Control Actions with a Quantum Computer: Control actions can be injected with a MITM attack using a quantum computer. Control actions can also be spoofed by breaking the authentication protocol with a quantum computer. Similarly, a file could be tampered with during a MITM attack that broke the quantum-vulnerable authentication. Control action can also be intercepted, decrypted, and disclosed by an adversary that broke the quantum-vulnerable communication protocol.

Attacking the Feedback with a Quantum Computer: Feedback can be injected with a MITM attack using a quantum computer. Feedback can also be spoofed by breaking the authentication protocol with a quantum computer. Similarly, a file could be tampered with during a MITM attack that broke the quantum-vulnerable authentication. Feedback can also be intercepted, decrypted, and disclosed by an adversary that broke the quantum-vulnerable communication protocol.

Chapter 11

Quantum Risk Management

Large-scale quantum computers are a material threat to the safety and security of our technological world. Knowledge of when these machines will arrive, and their impact on industries such as financial services, automotive, and defence will allow risk managers to formulate strategies to mitigate the risk. Below is a framework for estimating the likelihood and severity of losses due to the arrival of large-scale quantum computers and is based off of Hubbard's approach in "How to Measure Anything in Cybersecurity Risk" [37] and is based on Monte Carlo simulation using survey data from experts.

11.1 Quantum Cyber Risk Analytics

The analysis focuses on providing estimates of the probability of a large-scale quantum computer breaking RSA-2048 in 24 hours within 5, 10, 15, 20, and 30 years. We are interested in the mean probability and high percentile events such as the 75th, 90th, 95th, 99th, 99.6th, 99.8th, and 99.9th percentiles. In other words, in the worst 1 in 4, 1 in 10, 1 in 20, 1 in 100, 1 in 250, 1 in 500, and 1 in 1,000 chance scenario what is the predicted probability that RSA-2048 will be broken in 24 hours within a given time frame. Specific industries have regulations regarding how extreme the worst-case scenario should be. For example, financial institutions that use the Advanced Internal Rating-Based (AIRB) approach for capital risk management are required by the Basel II accord to hold sufficient capital at the 99.9th percentile of possible loss [21]. The following is taken from Section 5.1 regarding confidence level concerning the calibration of capital models from the document "An Explanatory Note on the Basel II IRB Risk Weight Functions" published by the Basel Committee on Banking Supervision (BCBS):

"The confidence level is fixed at 99.9%, i.e. an institution is expected to suffer losses that exceed its level of tier 1 and tier 2 capital on average once in a thousand years". [21:11]

Tables 11.1-11.5 below show the estimated probability of RSA-2048 being broken and distinguish between if the survey results contained all participants or just those closer to experiments. A minimum or maximum column shows the estimates using the minimum of the interval versus the maximum of the interval, with more details provided in the figures below. The following are instructions on how to read the table given by an example. If the reader believes the experimentalists are more knowledgeable regarding when RSA-2048 will be broken than the total set of respondents to the survey, then the recommendation is to use their estimates. The estimate of the probability that RSA-2048 will be broken in 24 hours within the next fifteen years based on the experimentalists is at a minimum 99% and a maximum of 99.999% at the 99.9th percentile which can be defined as the worst-case scenario, for example, using Basel II [21]. To be conservative, use the minimum value of 99%. This would imply an approximate 99 in 100 chance that RSA-2048 will be broken in 24 hours within the next fifteen years in the worst 1 in 1,000 chance scenario. To the best of the thesis author's knowledge, this is the first time the worst 1 in 1,000 chance scenario for the probability that RSA-2048 will be broken in 24 hours within fifteen years has been calculated and is, therefore, the first known quantum stress test.

The quantum stress test results in Table 11.3 are alarming. From an experimentalist point of view, the mean probability that RSA-2048 will be broken in 15 years, which is the most conservative, is 28.71%. However, at the 99.9th percentile, the worst-case scenario probability is 99.065% which is approximately 3.5X the mean of 28.71%. This means that the quantum risk to the automotive, financial services, defence, and other industries is about 3.5 times higher in the worst 1 in 1,000 chance scenario compared to the average scenario. The world should be preparing for RSA-2048 being broken in 24 hours within 15 years as something that is almost certain to happen in the worst 1 in 1,000 chance scenario based on estimates derived from the opinions of quantum computing experts.

The tables below show the analysis of trying to estimate various percentiles for the probability that RSA-2048 will be broken within 5, 10, 15, 20, and 30 years. The p-values are low for the Kolmogorov-Smirnov Goodness-of-Fit test for 5 and 10 year time horizons since they are less than 5%. When a p-value is below 5%, the null hypothesis is rejected. In the case of the Kolmogorov-Smirnov Goodness-of-Fit test, the null hypothesis is that the data fits the Beta distribution. Rejecting the null hypothesis means that the estimates in Tables 11.1 and 11.2 should not be used since the Beta distribution does not fit the data well.

Stress testing aims to ensure that your business has sufficient capital to survive the worst-case scenario, for example, at the 99.9th percentile of loss. By only considering the mean probability of RSA-2048 being broken in 24 hours, it seems reasonable to only be concerned with the quantum risk materializing within ten or more years. This thought process contradicts the fundamentals of risk management. The goal is for the business to survive in the long term. Although it cannot be concluded what the worst 1 in 1,000 chance scenario risk of a quantum computer attack is based on the analysis below within the next 5-years it could be significantly greater than 2%. On average there is essentially low risk in the next 5 years, but in the worst 1 in 1,000 chance scenario there could be material risk that the world is not prepared for over the next 5-years.

11.1.1 Risk Estimates

Model of Probability of RSA-2048 Broken in 24 Hours within 5-years				
Estimate	All Respondents Minimum	All Respondents Maximum	Experimentalist Minimum	Experimentalist Maximum
Mean Probability	2.70%	8.68%	1.61%	5.04%
75th Percentile Probability	0.95%	12.51%	0.07%	8.69%
90th Percentile Probability	7.70%	26.06%	2.79%	17.90%
95th Percentile Probability	16.49%	35.95%	9.28%	24.92%
99th Percentile Probability	39.70%	55.34%	32.49%	39.92%
99.6th Percentile Probability	51.53%	63.98%	45.72%	47.38%
99.8th Percentile Probability	59.29%	69.48%	54.60%	52.48%
99.9th Percentile Probability	66.00%	74.19%	62.35%	57.14%
Kolmogorov-Smirnov (KS) Statistic	0.61	0.30682	0.66667	0.33333
p-value	$8.11 * 10^{-15}$	0.000505	$1.09 * 10^{-9}$	0.009656
alpha	0.09369035	0.3907659	0.05065984	0.4294992
beta	3.46008137	0.039582018	3.2353836	6.3463942

Table 11.1: Model of Probability that RSA-2048 is broken in 24 hours within 5-years

Model of Probability of RSA-2048 Broken in 24 Hours within 10-years				
Estimate	All Respondents Minimum	All Respondents Maximum	Experimentalist Minimum	Experimentalist Maximum
Mean Probability	13.50%	31.74%	9.90%	23.00%
75th Percentile Probability	17.72%	36.50%	8.35%	26.85%
90th Percentile Probability	48.97%	62.94%	37.31%	48.94%
95th Percentile Probability	68.08%	76.32%	59.68%	62.15%
99th Percentile Probability	90.76%	92.09%	88.53%	81.90%
99.6th Percentile Probability	95.61%	95.83%	94.71%	88.27%
99.8th Percentile Probability	97.52%	97.44%	97.09%	91.58%
99.9th Percentile Probability	98.60%	98.43%	98.40%	93.97%
Kolmogorov-Smirnov (KS) Statistic	0.22727	0.17822	0.29167	0.19517
p-value	0.02123	0.1222	0.03370	0.32000
alpha	0.1913312	0.4125433	0.1252435	0.4351915
beta	1.2010385	1.4108337	1.1430453	2.0495458

Table 11.2: Model of Probability that RSA-2048 is broken in 24 hours within 10-years

Model of Probability of RSA-2048 Broken in 24 Hours within 15-years				
Estimate	All Respondents Minimum	All Respondents Maximum	Experimentalist Minimum	Experimentalist Maximum
Mean Probability	32.30%	54.30%	28.71%	49.71%
75th Percentile Probability	53.56%	85.54%	36.78%	81.89%
90th Percentile Probability	82.88%	97.05%	65.38%	96.13%
95th Percentile Probability	92.50%	99.13%	79.21%	98.83%
99th Percentile Probability	98.96%	99.95%	94.07%	99.93%
99.6th Percentile Probability	99.67%	99.99%	97.15%	99.99%
99.8th Percentile Probability	99.860%	99.997%	98.366%	99.996%
99.9th Percentile Probability	99.941%	99.999%	99.065%	99.999%
Kolmogorov-Smirnov (KS) Statistic	0.15909	0.15909	0.16804	0.16667
p-value	0.2154	0.2154	0.5069	0.5176
alpha	0.3483992	0.639881	0.360429	0.5450435
beta	0.8028486	0.5666328	1.236318	0.577151

Table 11.3: Model of Probability that RSA-2048 is broken in 24 hours within 15-years

Model of Probability of RSA-2048 Broken in 24 Hours within 20-years				
Estimate	All Respondents Minimum	All Respondents Maximum	Experimentalist Minimum	Experimentalist Maximum
Mean Probability	59.23%	81.08%	55.58%	77.95%
75th Percentile Probability	89.61%	99.39%	87.45%	98.99%
90th Percentile Probability	98.03%	99.98%	97.50%	99.94%
95th Percentile Probability	99.44%	100%	99.27%	99.99%
99th Percentile Probability	99.97%	100%	99.96%	100%
99.6th Percentile Probability	99.99%	100%	99.99%	100%
99.8th Percentile Probability	99.998%	100%	99.998%	100%
99.9th Percentile Probability	99.9996%	100%	99.9993%	100%
Kolmogorov-Smirnov (KS) Statistic	0.18182	0.18182	0.16667	0.16667
p-value	0.1090	0.1090	0.5176	0.5176
alpha	0.7956180	1.1852389	0.7101527	1.2302788
beta	0.5480761	0.2841521	0.5613523	0.3204002

Table 11.4: Model of Probability that RSA-2048 is broken in 24 hours within 20-years

Model of Probability of RSA-2048 Broken in 24 Hours within 30-years				
Estimate	All Respondents Minimum	All Respondents Maximum	Experimentalist Minimum	Experimentalist Maximum
Mean Probability	76.32%	91.88%	76.83%	91.56%
75th Percentile Probability	97.72%	99.94%	97.90%	99.97%
90th Percentile Probability	99.75%	100%	99.76%	100%
95th Percentile Probability	99.95%	100%	99.95%	100%
99th Percentile Probability	100%	100%	99.999%	100%
99.6th Percentile Probability	100%	100%	100%	100%
99.8th Percentile Probability	100%	100%	100%	100%
99.9th Percentile Probability	100%	100%	100%	100%
Kolmogorov-Smirnov (KS) Statistic	0.17783	0.27273	0.18345	0.25
p-value	0.1237	0.002873	0.3945	0.09956
alpha	1.4318305	2.555214	1.5980069	1.975151
beta	0.4147751	0.2249397	0.4202609	0.1952276

Table 11.5: Model of Probability that RSA-2048 is broken in 24 hours within 30-years

Tables 11.1-11.5 above show the results of fitting Beta distributions to the survey results from the "Quantum Threat Timeline Report 2020" by Mosca and Piani [5]. Tables 11.6 and 11.7 are taken from that report and show the survey results. 44 experts from the quantum computer industry were surveyed and asked what they thought the probability of RSA-2048 being broken in 24 hours within a 5, 10, 15, 20, and 30 year time period is. [5] 24 of the respondents were experimentalists, and their responses are shown separately and combined in the total 44. [5]

(All Respondents) Survey Results Regarding RSA-2048 Being Broken in 24 Hours					
How likely	5 years	10 years	15 years	20 years	30 years
Extremely unlikely (< 1% chance)	27 (61%)	10 (23%)	1 (2%)	0 (0%)	0 (0%)
Very unlikely (< 5% chance)	11 (25%)	13 (30%)	6 (14%)	1 (2%)	0 (0%)
Unlikely (< 30% chance)	3 (7%)	10 (23%)	14 (32%)	5 (11%)	1 (2%)
Neither likely nor unlikely (about 50% chance)	3 (7%)	6 (14%)	11 (25%)	10 (23%)	7 (16%)
Likely (> 70% chance)	0 (0%)	5 (11%)	5 (11%)	16 (36%)	13 (30%)
Very likely (> 95% chance)	0 (0%)	0 (0%)	7 (16%)	7 (16%)	11 (25%)
Extremely likely (> 99% chance)	0 (0%)	0 (0%)	0 (0%)	5 (11%)	12 (27%)
Total number of respondents (percentage)	44 (100%)	44 (100%)	44 (100%)	44 (100%)	44 (100%)

Table 11.6: (All Respondents) Probability of RSA-2048 Being Broken in 24 Hours [5]
Note. From the "Quantum Threat Timeline Report 2020", by M. Mosca and M. Piani, 2021, evolutionQ. Published by the Global Risk Institute in Financial Services (GRI). p.25 https://globalriskinstitute.org/publications/quantum-threat-timeline-report-2020/

(Experimentalist) Survey Results Regarding RSA-2048 Being Broken in 24 Hours					
How likely	5 years	10 years	15 years	20 years	30 years
Extremely unlikely ($< 1\%$ chance)	16 (67%)	7 (29%)	1 (4%)	0 (0%)	0 (0%)
Very unlikely ($< 5\%$ chance)	7 (29%)	7 (29%)	4 (17%)	1 (4%)	0 (0%)
Unlikely ($< 30\%$ chance)	0 (0%)	6 (25%)	8 (33%)	3 (13%)	1 (4%)
Neither likely nor unlikely (about 50% chance)	1 (4%)	2 (8%)	5 (21%)	6 (25%)	3 (13%)
Likely ($> 70\%$ chance)	0 (0%)	2 (8%)	3 (13%)	8 (33%)	7 (29%)
Very likely($> 95\%$ chance)	0 (0%)	0 (0%)	3 (13%)	4 (17%)	7 (29%)
Extremely likely ($> 99\%$ chance)	0 (0%)	0 (0%)	0 (0%)	2 (8%)	6 (25%)
Total number of respondents (percentage)	24 (100%)	24 (100%)	24 (100%)	24 (100%)	24 (100%)

Table 11.7: (Experimentalists) Probability of RSA-2048 Being Broken in 24 Hours [5]. Note. From the “Quantum Threat Timeline Report 2020”, by M. Mosca and M. Piani, 2021, evolutionQ. Published by the Global Risk Institute in Financial Services (GRI). p.26 https://globalriskinstitute.org/publications/quantum-threat-timeline-report-2020/

Appendix D below contains the respondent-level datasets used as inputs into a function for fitting Beta distributions. The Beta distributions were fit using “R Studio” [134], based on the open-source statistical analysis software “R” [132] leveraging the “fitdistplus” package [133]. See Appendix C for a sample of the code used. The Kolmogorov-Smirnov Goodness-of-Fit statistic was optimized when fitting the Beta distribution parameters.

The Beta distribution is modelling the distribution of possible probabilities of RSA-2048 being broken in 24 hours by a large-scale quantum computer in the next 5, 10, 15, 20, and 30 years. If the dataset is based on the minimum value, the minimum value of the interval was used. If the dataset is based on the maximum value, the maximum value of the interval was used, similar to the analysis in [5].

11.1.2 Monte Carlo Simulation

Now that Beta distributions have been fit to the survey data, as shown above, the next step is to run a Monte Carlo simulation. To do this, generate N uniformly distributed random numbers from the interval (0,1) [37]. Typically N is some large number such as 10,000 or 1,000,000 but can vary depending on the simulation [37]. Next, leverage the fitted Beta distributions from above to calculate the inverse of the cumulative Beta distribution (Inverse_Beta) [37], [135] at the randomly generated uniformly distributed values previously generated as specified above.

For example, suppose the uniformly distributed random number is $r = 0.99$. Next, suppose we are doing a simulation using the Beta distribution that was fit for all respondents on a 15 year time horizon, taking the minimum value of their responses. The Inverse_Beta$(\alpha = 0.3483992, \beta = 0.8028486, r = 0.99) = 98.96\%$ as per Table 11.3 above. Repeat this process for all of the randomly generated uniformly distributed numbers as per above. This will create N estimates for the likelihood that RSA-2048 will be broken within a fifteen-year time horizon based on the minimum of all respondents.

The next step is to simulate the severity of losses, given that quantum cryptanalysis occurs within the next fifteen years. This would require conducting a survey with experts in the automotive industry to provide a 90% confidence interval as per the methodology specified in “How to Measure Anything in Cybersecurity Risk” [37]. Using a similar procedure as above, generate N estimates of severity using the distribution fitted to the survey results as per “How to Measure Anything in Cybersecurity Risk” [37]. Note that the dataset does not currently exist to conduct the simulation for loss severity and is part of future work in autonomous vehicle research.

Finally, multiply the values from each of the N simulations of likelihood and severity together to get N simulations of the total loss resulting from a quantum computer attack on autonomous vehicles. Again, since this is a simulation, there will be a total loss distribution, and percentiles can be calculated such as the 99.9th.

11.1.3 Model Limitations and Risks

The following are limitations and risks on the above framework:

1. The input dataset is generated by human experts giving their opinion on the probability of an outcome.

2. There are only 44 total respondents and 24 experimentalists so there could be sample bias impacting the results.

3. The Kolmogorov-Smirnov (KS) statistic is used to optimize the model parameters and test the Goodness-of-Fit so the optimization of the parameters could optimize the p-values.

4. Distributions other than Beta could possibly fit the data more closely.

5. The model should not be used to make predictions about 5 and 10 year horizons as the p-values are low.

6. The data is rounded to the minimum and maximum of the interval and may not represent the exact value a respondent would have provided. The rounding done to generate the minimum and maximum values of an interval could be a source of error in the calculated Kolmogorov-Smirnov statistic [131].

7. The function used to fit the Beta distribution by optimizing the Kolmogorov-Smirnov statistic is designed for continuous data [133]. The rounding to the minimum and maximum of the interval could be a source of model error.

8. The model is very simplistic in its approach to estimating total loss. It assumes that either there will not be quantum cryptanalysis, in which case there will be no loss, or there will be quantum cryptanalysis, in which case there will be an immense loss. The model does not have different tiers of loss built into it for different levels of quantum cryptanalysis risk, such as the price of the quantum computer.

9. The Monte Carlo simulation framework is only meant to provide an incremental quantitative improvement compared to the purely qualitative risk matrix approach. [37]

11.1.4 Model Assumptions

The following are the assumptions on the above framework:

1. $Loss_{(p1(i),p2(i))} = P(Q)_{(p1(i))} * [Loss|Q]_{(p2(i))} + P(NoQ)_{(p1(i))} * [Loss|NoQ]_{(p2(i))}$
   $= P(Q)_{(p1(i))} * [Loss|Q]_{(p2(i))} \because [Loss|NoQ]_{(p2(i))} = 0$

   where:

   $Loss_{(p1(i),p2(i))}$ = the ith loss estimate generated by the Monte Carlo simulation.

   $P(Q)_{(p1(i))}$ = the probability of quantum cryptanalysis happening at percentile $p1(i)$.

   $[Loss|Q]_{(p2(i))}$ = the loss estimate given quantum cryptanalysis happening at percentile $p2(i)$.

   $P(NoQ)_{(p1(i))}$ = the probability of no quantum cryptanalysis at percetile $p1(i)$.

   $[Loss|NoQ]_{(p2(i))}$ = the loss estimate given no quantum cryptanalysis at percentile $p2(i)$ which is 0.

   $p1(i), p2(i) \in_R (0,1)$ are the uniformly randomly generated percentiles used in the Monte Carlo simulation to estimate probability and severity of losses. For example, $p1(i) = 0.99$ represents the 99th percentile.

2. $L_N = \{Loss_{(p1(i),p2(i))} | 1 \leq i \leq N, i \in \mathbb{Z}\}$

   where N is the number of Monte Carlo simulations.

3. $L_{N(p)}$ is the estimated loss at percentile p after N Monte Carlo simulations from the set of all loss estimates L_N.

4. $P(Q)_{(p1(i))}$ is estimated using the results of the fitted Beta distribution which is based on survey results estimating the risk of quantum cryptanalysis happening in 24 hours and is therefore a lower bound of quantum cryptanalysis happening ever.

11.1.5 Future Enhancements to the Model

The following are some future enhancements to the above framework:

1. Increase the sample size of the respondents in the survey if possible.

2. Explore distributions other than Beta to see if they fit the data more closely.
3. Explore new modelling techniques that can model the 5-year and 10-year time horizons. For example, how to build a model to calculate the 99.9th percentile where 27 out of 44 of your observations are 0.
4. Explore different ways to collect the survey data to enhance the model and reduce the potential model error introduced by rounding to the minimum and maximum of the interval.
5. Explore different techniques to build a model when the data is not exactly defined as a point, but the inputs are an interval.
6. Add more granular time dimensions when estimating the loss severity and probability of quantum cryptanalysis being possible. For example, estimating the loss severity over the next fifteen years is a lot different than estimating if it happens in 0-5 years, 5-10 years, and 10-15 years since you effectively have to average over these three time buckets.
7. When calculating risk, add a cost dimension into the calculation as cheaper quantum computers would pose a more considerable risk due to their prevalence.
8. Instead of having a single estimate of loss severity, as mentioned in the Monte Carlo section above, do not treat this as a constant but as a random variable that itself has a distribution that needs to be simulated. This will require conducting a survey as mentioned in the Monte Carlo section above to generate an estimated loss severity distribution.

11.2 Quantum Risk Assessment

Quantum-vulnerable public-key cryptography is threatened by the potential existence of large-scale quantum computers in the future. In order to mitigate the quantum risk alternatives need to be evaluated such as migrating to quantum-safe cryptography. In the context of a path planning control loop, quantum-vulnerable digital signatures are at risk. One crucial question is when an attacker such as a nation-state would have access to a large-scale quantum computer capable of forging quantum-vulnerable digital signatures. The answer is uncertain today as the technology is still maturing; however, a large-scale quantum computer could exist in the next five years. The estimated chance of a quantum computer capable of breaking RSA-2048 in 24 hours being built in the next five years as of 2020 is low, around 2% using a conservative estimate based upon survey results from experts in the field [5]. However, in the worst 1 in 1,000 chance scenario this risk could be significantly greater than 2% in the next 5-years. Another crucial question is what is the risk of newly developed quantum-safe cryptography being broken and is this risk less than the risk of quantum cryptanalysis. Finally, when is the optimal time to migrate to post-quantum digital signatures.

According to the National Cybersecurity Center of Excellence (NCCoE) from the National Institute of Standards and Technology (NIST) it may take anywhere from 5 to greater than 15 years to upgrade to quantum-safe cryptographic systems starting from when the standards are published, which is expected in 2024 [16]. This implies that the implementation would be complete for materially impacted systems between 2029 and 2039, possibly going into the 2040's. 86% of respondents to a survey published in the 'Quantum Threat Timeline Report 2020' believe that there is at least a 50% chance that a large-scale quantum computer capable of breaking an RSA-2048 modulus in 24 hours will exist within 20 years [5]. There is approximately a 99 in 100 chance that RSA-2048 will be broken in 24 hours within 15 and 20 years in worst 1 in 1,000 and 1 in 20 chance scenario respectively based on the analysis in Section 11.1. Therefore, the arrival of large-scale quantum computers could likely overlap the migration to quantum-safe cryptography based on the estimates above and possibly happen well before the migration is complete.

Another point to consider is the cost of obtaining a private key. An estimate by Mariantoni in 2014 placed the cost of building a superconducting quantum computer at $1 Billion, which can factor a 2000-bit number in 24 hours being available in 15 years [18]. If the quantum computer solved one 2000-bit number every day for three years, this would result in an average cost of fewer than 1 Million dollars for a private key. At this price range, adversarial nation-states would have no trouble affording a private key, and it would be a bargain compared to the risk of physically stealing a private key from a company. If large-scale quantum computers arrive before quantum-safe digital signatures are deployed in autonomous vehicles, then a Stuxnet-style attack using a quantum computer would be feasible. A quantum-computer-enabled fleet-wide Stuxnet style attack would only be possible if the same public-private key pair were used in automobiles, as discussed in Section 8. Note that attacking individual vehicles at the cost of approximately less than one million dollars per vehicle would be possible even if unique public-private key pairs were used, so there is a material risk either way.

A Stuxnet-style attack would result in some or all the software in the path planning control loop, including the path planning system, perceptual unit, steering, engine, and brake controllers, being modified, which could cause all the 27 UCAs identified in Section

7.3 to occur. In the worst-case, the UCAs would result in H1, H2, H3, H4, and H5 and L1, L2, L3, L4, and L6. These losses include life, vehicle, property, customer satisfaction, and shareholder value. Infecting millions of cars with a malicious over-the-air (OTA) software update at the same time is possible, per an NHTSA report [29]. Suppose one million vehicles became infected with malware that used a quantum computer to forge digital signatures and that issued a series of UCAs at a specified point in time, then the malware could put 1,000,000 cars at material risk. The loss of life could be very high. Also, even if everyone were safe, trust in the brand and the autonomous vehicle industry could be significantly impacted. A large fraction of the potential 15% share in new car sales that Level 4 autonomous vehicles could have in 2030 would likely evaporate [4]. The severity can be measured by the loss of life and the negative economic impact on the automotive industry. Based on the severity alone, it is recommended that automotive companies evaluate possible alternatives to their existing digital signatures, including migrating to quantum-safe cryptography, hybrid cryptography, hash-based digital signatures, and their current level of crypto agility. See Section 11.2.1 below regarding risk mitigation strategies for more details. In addition to severity, the possibility of large-scale quantum computers arriving in 15 years being at an approximate 99 in 100 chance in the worst 1 in 1,000 chance scenario and the expected migration time of at least five years indicates that automotive companies need to begin migration planning right away.

11.2.1 Risk Mitigation Strategies

Adversaries forging digital signatures can lead to severe consequences for the automotive industry resulting in immeasurable suffering and loss of life. One clear attack method uses a quantum computer to perform cryptanalysis on existing digital signature schemes such as RSA or EdDSA. NIST is currently standardizing cryptography with an expected completion date of 2024. To mitigate the quantum risk, migrating to one of the new post-quantum cryptographic algorithms is an option. However, this option is not risk-free. Post-quantum cryptography is also vulnerable to attacks that can enable an adversary to forge a digital signature without using a quantum computer. For example, Ward Beullens published an attack on Rainbow that returns the secret key of a Round 2 instance using about 53 hours of computational runtime on average running on a laptop [155]. An adversary recovering the secret key is a risk that needs to be considered, even with the use of post-quantum cryptography as illustrated by Beullens mentioned above. A critical question to answer is when is the optimal time to transition to post-quantum digital signatures. If automotive manufacturers were to migrate to post-quantum digital signatures overnight, the risk of an attack on post-quantum cryptography using classical methods would likely be higher than the risk of a quantum computer existing overnight capable of breaking existing cryptography such as RSA and EdDSA or the risk of RSA and EdDSA being broken by classical methods. On the other hand, the risk of a quantum computer existing within 10 years capable of quantum cryptanalysis is likely higher than the risk of post-quantum cryptography being broken by classical methods. For example, based on the estimates in Table 11.2, a conservative estimate for the mean probability that RSA-2048 will be broken in 24 hours within 10 years by the experimentalists is approximately 10%. An approach for quantifying the risk of post-quantum cryptography being broken would be to conduct a survey similar to the one by Mosca and Piani [5]. For example, at least 50 thought leaders in the field of post-quantum cryptanalysis could be surveyed and asked what is the probability that a specific post-quantum digital signature (e.g., CRYSTALS-Dilithium) will be broken within the next 5, 10, 15, 20, and 30 years by quantum and classically equipped adversaries. The framework used in Section 11.1 can be used to estimate the risk at various percentiles. Comparing the results of the analysis between the Mosca and Piani quantum survey and the post-quantum cryptography survey would help guide the recommendation as to when is the optimal time to transition to post-quantum digital signatures. Note that although Basel II recommends evaluating the risk of the worst 1 in 1,000 chance scenario [21], it may be beneficial to also look at the mean and 1 in 10 risks. This is because as N gets large the worst 1 in N chance scenario risk converges to 100% so it may be easier to discriminate which risk is more acceptable at lower values of N. Note that the survey has not been conducted and is part of future work in quantum risk management research.

Quantum and classically equipped adversaries threaten digital signatures used for authentication of OTA software updates in cars. The following are possible risk-mitigating actions that can be taken.

Hybrid digital signatures can be used so that the risk of using a new post-quantum signature can be offset by using a traditional digital signature such as RSA or EdDSA [156]. Another option is to use two post-quantum digital signatures to sign a software update. For example, when updating the software in a car, sign the software with SPHINCS+ and CRYSTALS-Dilithium. In this situation, an adversary would have to break two post-quantum digital signatures to bypass the authentication. The double post-quantum digital signature would provide resilience to quantum cryptanalysis and redundancy so that you can use the newer post-quantum digital signatures with a backup plan in case one of them is broken. The risk reduction could allow for deployment sooner as the risk of quantum cryptanalysis could be higher than the risk of two post-quantum digital signatures being broken. According to the experimentalists, a conservative estimate of the mean risk of a quantum computer existing within the next 5 years based on estimates from Table 11.1 is approximately 2%. It is possible that the risk of two post-quantum digital signatures being broken within 5-years is much less than 2%, but further analysis and surveys would need to be conducted to confirm this and is part of future work in quantum risk management. The use of double post-quantum digital signatures could be the path forward in the short-term, but the risk of additional complexity introduced by hybrid signatures also needs to be considered.

Another consideration is to use post-quantum hash-based digital signatures. Their security is more understood compared to other post-quantum digital signatures, and they only require a "secure cryptographic hash function" [157]. Stateful hash-based digital signatures such as XMSS or stateless ones such as SPHINCS+ may be a safer alternative in the short-term regarding the risk of being broken compared to other post-quantum digital signatures. As mentioned above, SPHINCS+ could be combined with another digital signature to provide additional protection at the cost of increasing complexity.

Another risk mitigant is crypto agility. This is the ability to quickly adapt cryptographic primitives used without significant manual intervention or system changes [16]. When designing a car, crypto agility should be considered as there could be a need to quickly switch the digital signatures used for OTA software updates. If this cannot be done remotely, costly recalls could be required to update the cryptographic keys at automotive service centres.

Another option is to continue using existing public-key cryptography such as RSA and EdDSA in the short term. This might be the best option until NIST standardizes post-quantum cryptography in 2024. As of today, the probability of breaking existing public-key cryptography using classical methods may be less than the probability of breaking post-quantum cryptography. In addition, the risk of breaking existing public-key cryptography using classical and quantum methods combined may be less than the risk of breaking post-quantum cryptography in the short term. In order to confirm this, a survey should be conducted with cryptography experts regarding the probability of breaking existing public-key cryptography using classical methods similar to the surveys outlined above and is part of future work in quantum risk management research. Eventually, the quantum cryptanalysis risk will likely become sufficiently high that migration to an alternative mentioned above will be necessary to mitigate the risk of currently deployed public-key cryptography being broken by a quantum computer. The optimal migration time frame to one of the above alternatives will be informed by the comparison of the survey results mentioned above.

Chapter 12

Summary and Recommendations for Quantum-Safe Autonomous Vehicles

An autonomous vehicle's path planning control loop is not immune to quantum computer attacks. Several controllers in the vehicle use code signing to guarantee that the software originates from the correct vendor and that the software integrity is maintained. The code signing is based on digital signatures such as RSA and EdDSA, whose security depends on the difficulty of factoring and finding discrete logarithms [79]. Quantum computer algorithms discovered by Peter Shor can be used to factor and find discrete logarithms in polynomial time, which means that these problems can be efficiently solved once a large-scale quantum computer exists [23]. Estimating when a quantum computer will arrive is a complex problem. There is a slight possibility that there will be one in 5 years from now. However, looking further down the road, approximately 9 out of 10 experts believe there is at least a 50% chance that a large-scale quantum computer capable of breaking an RSA-2048 modulus in 24 hours will exist within 20 years as of 2020 [5]. In the worst 1 in 1,000 chance scenario, there is an approximate 99 in 100 chance that RSA-2048 will be broken within 15 years based on the analysis from Section 11.1.

Secure code signing is vital to the safety and security of the autonomous vehicle industry. The Keen Security Lab hack from 2016 showed that it is possible to modify the firmware of ECUs and wirelessly control a vehicle and send UCAs to it [13]. Tesla responded to this attack by implementing code signing to mitigate the risk, albeit quantum-vulnerable code-signing using Ed25519 [6], [77], [78]. Stuxnet is another example where two private keys were stolen and used to sign software, which caused UCAs in Iranian nuclear centrifuges, resulting in losses [27], [96]. The ability to retrieve the private key of an automotive manufacturer using quantum or classical computational methods could enable an adversary to execute a Stuxnet-style attack on autonomous vehicles. Suppose manufacturers used the same public-private key pair as discussed in Section 8. In that case, all that is needed is a single private key, which would cost in the neighbourhood of less than \$1 million to quantum-compute with the ability to infect a million vehicles with a malicious software update, or less than \$1 per vehicle. A malicious software update could enable all 27 UCAs from the path planning control loop to be sent to the brakes, steering, and engine. The 27 UCAs could potentially result in numerous casualties and significantly negatively impact the autonomous vehicle industry's consumer confidence and market capitalization, which Level 4 is estimated to be up to 15% of new car sales for the overall automotive industry in 2030 [4]. Fortunately, it appears that there are mathematical problems that quantum computers cannot efficiently solve, or no known algorithm exists. These problems form the basis for post-quantum cryptography, as outlined in Section 3. Considering that only a single private key may be needed to launch a Stuxnet-style attack on a collection of autonomous vehicles and that the loss severity is extreme, the recommendation is to review the risk mitigation strategies as discussed in Section 11.2.1. One possible path forward would be to use two post-quantum digital signatures such as a hash-based signature combined with a lattice-based signature to mitigate the risk of quantum cryptanalysis and the risk of using newly developed post-quantum signatures at the cost of additional complexity. With this option, automotive manufacturers could potentially migrate to post-quantum digital signatures once they become standardized in 2024 and reduce the overall risk profile. However, more analysis is needed to confirm this is the case, as discussed in Section 11.2.1.

Quantum computers may answer essential questions about quantum properties of materials such as high temperature and room temperature superconductors. Room temperature superconductors could revolutionize power generation via efficient nuclear fusion power plants by optimizing the energy consumption of the superconducting magnets. Quantum computers could help us understand black holes [28], quantum gravity via the holographic principle, and string theory [98]. Perhaps we could even discover new laws of physics by understanding our current theories more deeply, such as gleaning insights into quantum geometry and m-theory. Quantum computers could even help build better autonomous vehicles by improving their batteries [97] and making them hover (imagine an autonomous taxi in a major city hovering on a surface enabled by room temperature superconducting magnets). Quantum sensors such as quantum magnetometers could enhance the safety of the vehicles through improved navigation when GPS is not available such as when the line of sight is blocked by skyscrapers in major cities [2], [141]. There is also the dark side of quantum computing to consider. As outlined in this thesis, quantum computers can be used for cyber warfare. They can perform quantum cryptanalysis to forge digital signatures on the software updates of autonomous vehicles, leading to critical safety issues.

Hiroshima was the first to suffer the devastating impact of a nuclear fission bomb on August 6th, 1945 [99]. Nagasaki was subsequently attacked with a nuclear fission bomb on August 9th, 1945 [100]. The world's first nuclear fission power plant was turned on December 20th, 1951 [101]. This is over 6 years after the fission bombs were used. This illustrates that one of the first major use cases of fission technology was as a weapon of mass destruction. Only after the dark side of fission technology was implemented was a use case that is arguably mostly positive for humanity, nuclear fission power generation implemented. Of course, even the benefits of nuclear fission energy, such as climate change mitigation, come with safety risks, as observed in the cases of Chernobyl [102] and Fukushima [103].

Quantum information technology is in some regards similar to nuclear fission in the sense that they are both powerful technologies predicted by physics and some of the very first major use cases can be viewed as destructive (weapon of mass destruction in the case of fission technology and quantum cyber warfare on autonomous vehicles for quantum computer technology). Similarly, both fission energy and quantum information technology can significantly benefit humanity but require ethical use and carry safety and security risks. Through the use of quantum-safe cryptography, humanity can enjoy the possible benefits of quantum information technology such as nuclear fusion power plants enabled by room temperature superconductors without having to experience quantum computers used as weapons of mass destruction.

Chapter 13

Future Work in Quantum-Safe Autonomous Vehicles

Future work would include creating an inventory of all cryptographic protocols in an autonomous vehicle and identifying those vulnerable to quantum cryptanalysis [15]. Another area to work on is applying the STPA-Sec framework to other areas of an autonomous vehicle such as vehicle-to-vehicle, vehicle-to-infrastructure, Wi-Fi, and Cellular data. Particular attention should be paid to see if any confidential information communicated using quantum-vulnerable protocols that could be intercepted, decrypted, and disclosed by an adversary with a large-scale quantum computer such as sensor data being uploaded to a vehicle manufacturer. Also, a MITM attack could be launched by forging digital signatures, enabling an adversary to send corrupt data such as an incorrect destination to a vehicle. Any system that uses digital signatures should be reviewed for safety and security risks. A possible enhancement would be to investigate if quantum random number generators could be used to improve the cryptographic protocols used in the vehicle. Testing the quantum-safe cryptographic protocols with autonomous vehicles is another next step. Finally, additional quantum cyber risk analytics should be completed as outlined in Section 11 on quantum risk management. This would include creating a survey from experts in the automotive industry to be used as a dataset to power a Monte Carlo simulation so the distribution of total losses as a result of a quantum computer attack on autonomous vehicles can be estimated as per the methodology in "How to Measure Anything in Cybersecurity Risk" [37].

Beyond Quantum-Safe Autonomous Vehicles

Tesla announced in 2021 that it is planning to develop humanoid robots in the future [88]. These robots are a natural extension of autonomous vehicles and should be made quantum-safe. Dubai used a humanoid robot to patrol the city for the first time in 2017 and plans to have 25% of the police workforce be robots by 2030 [89]. Another company that Elon Musk founded, Neuralink, is developing a machine-brain interface device that may allow humans to connect their minds to artificial intelligence. It should also be quantum-safe to maintain confidentiality, integrity, and authenticity. Also, a network of the human mind or the internet of the brain (IoB) should be made quantum-safe. Futurist Michio Kaku discussed the possibility of "Brain Net" in 2012 [154]. Progress in this direction is being made. For example, a brain network model called "BrainNet" was published in 2019 by Jiang *et al.* [90].

Quantum-safe autonomous drones, boats, planes, spacecraft, and weaponry (e.g., tanks) must also be considered.

References

[1] T. Denton, *Automated driving and driver assistance systems.* Abington, England: Routledge, 2019.

[2] E. Javanmardi, Y. Gu, M. Javanmardi, and S. Kamijo, “Autonomous vehicle self-localization based on abstract map and multi-channel lidar in urban area,” *IATSS Research*, vol. 43, pp. 1–13, Apr. 2019 [Online]. Available: https://doi.org/10.1016/j.iatssr.2018.05.001.

[3] NHTSA, “Critical reasons for crashes investigated in the national motor vehicle crash causation survey,” Tech. Rep. DOT HS 812 115, U.S. DoT., USA, 2015. Accessed: Mar. 23, 2021 [Online]. Available: https://crashstats.nhtsa.dot.gov/Api/Public/Publication/812115.

[4] P. Gao, H.-W. Kaas, D. Mohr, and D. Wee, “Automotive revolution – perspective towards 2030: How the convergence of disruptive technology-driven trends could transform the auto industry.” McKinsey & Co., New York, New York, USA, 2015. Accessed: Mar. 23, 2021 [Online]. Available: https://www.mckinsey.com/industries/automotive-and-assembly/our-insights/disruptive-trends-that-will-transform-the-auto-industry/de-DE.

[5] M. Mosca and M. Piani, “Quantum threat timeline report 2020.” evolutionQ, Kitchener, Ontario, Canada, Global Risk Institute in Financial Services (GRI), Toronto, Ontario, Canada, 2021. Accessed: Apr. 1, 2021 [Online]. Available: https://globalriskinstitute.org/publications/quantum-threat-timeline-report-2020/.

[6] S. Halder, A. Ghosal, and M. Conti, “Secure ota software updates in connected vehicles: A survey.” arXiv:1904.00685 [cs.CR], 2019.

[7] N. G. Levenson, *An STPA Primer, Version 1*, pp. 1–55. Cambridge, MA, USA: MIT, 2013.

[8] N. G. Levenson and J. P. Thomas, *STPA Handbook*. Cambridge, MA, USA: MIT, 2018.

[9] D. P. Pereira, C. M. Hirata, R. M. Pagliares, and F. L. de Lemos, "Stpa-sec: System-theoretic process analysis for security - flight management system." STAMP Workshop 2017, MIT, 2017. Accessed: May. 27, 2020 [Online]. Available: http://psas.scripts.mit.edu/home/wp-content/uploads/2017/03/Felipe-Oliveira_STPASec-For-Flight-Management-System.pdf.

[10] S. Sharma, "Considering saftey and security in av functions," master of applied science thesis, Electrical and Computer Engineering, University of Waterloo, Waterloo, Ontario, Canada, 2019 [Online]. Available: https://uwspace.uwaterloo.ca/bitstream/handle/10012/15005/Sharma_Shefali.pdf?isAllowed=y&sequence=3.

[11] A. Abdulkhaleq, D. Lammering, S. Wagner, J. Röder, N. Balbierer, L. Ramsauer, T. Raste, and H. Boehmert, "A systematic approach based on stpa for developing a dependable architecture for fully automated driving vehicles," *Procedia Engineering*, vol. 179, pp. 41–51, Jan. 2017 [Online]. Available: https://doi.org/10.1016/j.proeng.2017.03.094.

[12] N. G. Levenson, *Engineering a safer world: Systems thinking applied to safety*. Cambridge, MA, USA: The MIT Press, 2011.

[13] S. Nie, L. Liu, and Y. Du, "Free-fall: Hacking tesla from wireless to can bus." Keen Security Lab of Tencent, Shenzhen, China, 2017. Accessed: Mar. 25, 2021 [Online]. Available: https://www.blackhat.com/docs/us-17/thursday/us-17-Nie-Free-Fall-Hacking-Tesla-From-Wireless-To-CAN-Bus-wp.pdf.

[14] A. Matrosov, E. Rodionov, D. Harley, and J. Malcho, "Stuxnet under the microscope." eset, Accessed: Apr. 9, 2021 [Online]. Available: https://www.esetnod32.ru/company/viruslab/analytics/doc/Stuxnet_Under_the_Microscope.pdf.

[15] M. Mosca and J. Mulholland, "A methodology for quantum risk assessment." evolutionQ, Kitchener, Ontario, Canada, Global Risk Institute in Financial Services (GRI), Toronto, Ontario, Canada, 2017. Accessed: Apr. 11, 2021 [Online]. Available: https://globalriskinstitute.org/publications/3423-2/.

[16] W. Barker, W. Polk, and M. Souppaya, "Getting ready for post-quantum cryptography: Exploring challenges associated with adopting and using post-quantum cryptographic algorithms." NIST, USA, Apr. 2021. Accessed: Jan. 16, 2022 [Online]. Available: https://nvlpubs.nist.gov/nistpubs/CSWP/NIST.CSWP.04282021.pdf.

[17] Security Magazine, "Business losses to cybercrime data breaches to exceed $5 trillion by 2024." SecurityMagazine.com, Aug. 2019. Accessed: Jan. 23, 2022 [Online]. Available: https://www.securitymagazine.com/articles/90806-business-losses-to-cybercrime-data-breaches-to-exceed-5-trillion-by-2024.

[18] M. Mariantoni, "Matteo mariantoni - invited talk - building a superconducting quantum computer." Institute for Quantum Computing (IQC), University of Waterloo, Waterloo, Ontario, Canada, (Oct. 23, 2014). Accessed: Jan. 16, 2022 [Online Video]. Available: https://www.youtube.com/watch?v=wWHAs--HA1c.

[19] T. G. Tan, P. Szalachowski, and J. Zhou, "Sok: Challenges of post-quantum digital signing in real-world applications." Cryptology ePrint Archive, Report 2019/1374, 2019. Available: https://ia.cr/2019/1374.

[20] NIST, "Submission requirements and evaluation criteria for the post-quantum cryptography standardization process." USA, 2016. Accessed: Apr. 13, 2021 [Online]. Available: https://csrc.nist.gov/CSRC/media/Projects/Post-Quantum-Cryptography/documents/call-for-proposals-final-dec-2016.pdf.

[21] Basel Committee on Banking Supervision, "An explanatory note on the basel ii irb risk weight functions." Bank for International Settlements, Basel, Switzerland, 2005. Accessed: Apr. 14, 2021 [Online]. Available: https://www.bis.org/bcbs/irbriskweight.htm.

[22] P. Kaye, R. Laflamme, and M. Mosca, *An Introduction to Quantum Computing*. Oxford, England: Oxford University Press, 2007.

[23] P. W. Shor, "Algorithms for quantum computation: discrete logarithms and factoring," in *Proceedings 35th annual symposium on foundations of computer science*, pp. 124–134, 1994. Available: https://doi.org/10.1109/SFCS.1994.365700.

[24] T. Haner, M. Roetteler, and K. M. Svore, "Factoring using 2n+2 qubits with toffoli based modular multiplication." arXiv:1611.07995v2 [quant-ph], 2017.

[25] J. Gambetta, "Ibm's roadmap for scaling quantum technology." IBM Research Blog, 2020. Accessed: Jan. 17, 2022 [Online]. Available: https://research.ibm.com/blog/ibm-quantum-roadmap.

[26] Wikipedia contributors, "Adaptive cruise control." Wikipedia, The Free Encyclopedia, Accessed: Jan. 17, 2022 [Online]. Available: https://en.wikipedia.org/w/index.php?title=Adaptive_cruise_control&oldid=1062638874.

[27] Wikipedia contributors, "Stuxnet." Wikipedia, The Free Encyclopedia, Accessed: Jan. 17, 2022 [Online]. Available: https://en.wikipedia.org/w/index.php?title=Stuxnet&oldid=1065286737.

[28] J. Preskill, "Quantum computing in the nisq era and beyond." arXiv:1801.00862v3 [quant-ph], 2018.

[29] NHTSA, "Cybersecurity of firmware updates," Tech. Rep. DOT HS 812 807, U.S. DoT., USA, Oct. 2020. Accessed: Mar. 18, 2021 [Online]. Available: https://www.nhtsa.gov/sites/nhtsa.gov/files/documents/cybersecurity_of_firmware_updates_oct2020.pdf.

[30] MATLAB, "Understanding sensor fusion and tracking, part 1: What is sensor fusion?," (Oct. 21, 2019). Accessed: Jan. 17, 2022 [Online Video]. Available: https://www.youtube.com/watch?v=6qV3YjFppuc.

[31] D. Zelle, C. Krauß, H. Strauß, and K. Schmidt, "On using tls to secure in-vehicle networks," in *Proceedings of the 12th International Conference on Availability, Reliability and Security*, pp. 1–10, 2017. Available: https://doi.org/10.1145/3098954.3105824.

[32] B. Groza and P.-S. Murvay, "Identity-based key exchange on in-vehicle networks: Can-fd & flexray," *Sensors*, vol. 19, no. 22, p. 4919, Nov. 2019 [Online]. Available: https://doi.org/10.3390/s19224919.

[33] Norton Rose Fulbright, "The privacy implications of autonomous vehicles." Norton Rose Fulbright Data Protection Report, Accessed: Sep. 21, 2021 [Online]. Available: https://www.dataprotectionreport.com/2017/07/the-privacy-implications-of-autonomous-vehicles/.

[34] Wikipedia contributors, "Sensor." Wikipedia, The Free Encyclopedia, Accessed: Jan. 17, 2022 [Online]. Available: https://en.wikipedia.org/w/index.php?title=Sensor&oldid=1059883466.

[35] M. Humphries, "Tesla is developing a self-driving system that only uses cameras." PCMag, Accessed: Jan. 17, 2022 [Online]. Available: https://www.pcmag.com/news/tesla-is-developing-a-self-driving-system-that-only-uses-cameras.

[36] D. J. Yeong, G. Velasco-Hernandez, J. Barry, and J. Walsh, "Sensor and sensor fusion technology in autonomous vehicles: A review," *Sensors*, vol. 21, no. 6, p. 2140, Mar. 2021 [Online]. Available: https://doi.org/10.3390/s21062140.

[37] D. W. Hubbard and R. Seiersen, *How to Measure Anything in Cybersecurity Risk*. Hoboken, NJ, USA: John Wiley & Sons, Inc., 2016.

[38] D. P. Möller and R. E. Haas, *Guide to Automotive Connectivity and Cybersecurity: Trends, Technologies, Innovations and Applications*, ch. 6, pp. 265–377. Cham, Switzerland: Springer Nature Switzerland AG, 2019.

[39] Wikipedia contributors, "Automotive security." Wikipedia, The Free Encyclopedia, Accessed: Jan. 17, 2022 [Online]. Available: https://en.wikipedia.org/w/index.php?title=Automotive_security&oldid=1053152927.

[40] D. Clare, S. Fry, H. Handschuh, H. Patil, C. Poulin, A. Wasicek, R. Wood, D. A. Brown, G. Cooper, I. Gilvarry, D. Grawrock, A. Rajan, A. Tatourian, R. Venugopalan, C. Vishik, D. Wheeler, and M. Zhao, "Automotive security best practices: Recommendations for security and privacy in the era of the next-generation car." Santa Clara, CA, USA, McAfee, 2017. Accessed: Jul. 20, 2021 [Online]. Available: https://www.mcafee.com/enterprise/en-us/assets/white-papers/wp-automotive-security.pdf.

[41] S. Checkoway, D. McCoy, B. Kantor, D. Anderson, H. Shacham, S. Savage, K. Koscher, A. Czeskis, F. Roesner, and T. Kohno, "Comprehensive experimental analyses of automotive attack surfaces," in *20th USENIX Security Symposium (USENIX Security 11)*, (San Francisco, CA, USA), USENIX Association, Aug. 2011. Available: https://www.usenix.org/conference/usenix-security-11/comprehensive-experimental-analyses-automotive-attack-surfaces.

[42] K. Koscher, A. Czeskis, F. Roesner, S. Patel, T. Kohno, S. Checkoway, D. McCoy, B. Kantor, D. Anderson, H. Shacham, and S. Savage, "Experimental security analysis of a modern automobile," in *2010 IEEE symposium on security and privacy*, pp. 447–462, 2010. Available: https://doi.org/10.1109/SP.2010.34.

[43] A. Greenberg, "Hackers remotely kill a jeep on the highway—with me in it." WIRED, Accessed: Jul. 27, 2021 [Online]. Available: https://www.wired.com/2015/07/hackers-remotely-kill-jeep-highway/.

[44] K. Eykholt, I. Evtimov, E. Fernandes, B. Li, A. Rahmati, C. Xiao, A. Prakash, T. Kohno, and D. Song, "Robust physical-world attacks on deep learning models." arXiv:1707.08945v5 [cs.CR], 2018.

[45] A. Greenberg, “This bluetooth attack can steal a tesla model x in minutes.” WIRED, Accessed: Jul. 27, 2021 [Online]. Available: https://www.wired.com/story/tesla-model-x-hack-bluetooth/.

[46] A. Greenberg, “Hackers can steal a tesla model s in seconds by cloning its key fob.” WIRED, Accessed: Jul. 27, 2021 [Online]. Available: https://www.wired.com/story/hackers-steal-tesla-model-s-seconds-key-fob/.

[47] A. Karpathy, “Andrej karpathy - ai for full-self driving at tesla.” Matriod, (Apr. 20, 2020). Accessed: Jul. 27, 2021 [Online Video]. Available: https://www.youtube.com/watch?v=hx7BXih7zx8.

[48] Tesla, “Tesla vehicle safety report.” Tesla.com, Accessed: Jul. 28, 2021 [Online]. Available: https://www.tesla.com/en_CA/VehicleSafetyReport.

[49] Wikipedia contributors, “Tesla autopilot.” Wikipedia, The Free Encyclopedia, Accessed: Jul. 28, 2021 [Online]. Available: https://en.wikipedia.org/w/index.php?title=Tesla_Autopilot&oldid=1035245942.

[50] State of California Department of Motor Vehicles, “Disengagement reports,” Accessed: Jul. 28, 2021 [Online]. Available: https://www.dmv.ca.gov/portal/vehicle-industry-services/autonomous-vehicles/disengagement-reports/.

[51] L. Fridman, “Tesla vehicle deliveries and autopilot mileage statistics.” LexFridman.com, Accessed: Jul. 28, 2021 [Online]. Available: https://lexfridman.com/tesla-autopilot-miles-and-vehicles/.

[52] AutoX, Shenzhen, China, “Autox opens its fully driverless robotaxi service to the public in china (english),” (Jan. 27, 2021). Accessed: Jul. 29, 2021 [Online Video]. Available: https://www.youtube.com/watch?v=O69YEWpSacU.

[53] NHTSA, “Cybersecurity best practices for the safety of modern vehicles,” Tech. Rep. Draft 2020 Update, U.S. DoT., USA, 2020. Accessed: Jul. 19, 2021 [Online]. Available: https://www.nhtsa.gov/sites/nhtsa.gov/files/documents/vehicle_cybersecurity_best_practices_01072021.pdf.

[54] S. Hollister, “Hackers obtain ps3 private cryptography key due to epic programming fail? (update).” engadget, Accessed: Jan. 24, 2022 [Online]. Available: https://www.engadget.com/2010-12-29-hackers-obtain-ps3-private-cryptography-key-due-to-epic-programm.html?guccounter=1&guce_referrer=aHR0cHM6Ly93d3cuZ29vZ2xlLmNhLw&guce_referrer_sig=AQAAALWkQXXrRQKMkfpNNMtAMFidwg0

RE7udblKslBoqUDjsq2lbNV_FqJ0lrs4OlXW7bxj-WNvSPV77NRSr5BEkMKUbPHaK8Gm2XiN5VvI23ntKUzrkyP_0YxIQmZ3Qzi6amDcJTePJyqSON7ZOL50Z5t4s1SEwp9BsBaaykajmtYMY.

[55] Wikipedia contributors, "Safety." Wikipedia, The Free Encyclopedia, Accessed: Aug. 3, 2021 [Online]. Available: https://en.wikipedia.org/w/index.php?title=Safety&direction=prev&oldid=1052372398.

[56] A. Truskovsky, "Do you know about quantum's threat to ota software updates?." ISARA, Accessed: Aug. 3, 2021 [Online]. Available: https://www.isara.com/blog-posts/quantums-threat-ota-software-updates.html.

[57] QNX, "Ultimate guide to autonomous systems." BlackBerry, Accessed: Aug. 3, 2021 [Online]. Available: https://blackberry.qnx.com/en/ultimate-guides/autonomous-systems.

[58] NIST, "Framework for improving critical infrastructure cybersecurity," Tech. Rep. Version 1.1, USA, 2018. Accessed: Aug. 17, 2021 [Online]. Available: https://doi.org/10.6028/NIST.CSWP.04162018.

[59] Visa, "Road ahead: Connected cars coming to a lot near you." Visa.com, Accessed: Jan. 24, 2022 [Online]. Available: https://usa.visa.com/visa-everywhere/innovation/connected-car.html.

[60] L. Guillen, "Using gps tracking devices in divorce." DivorceNet, NOLO, Accessed: Jan. 24, 2022 [Online]. Available: https://www.divorcenet.com/resources/divorce/divorce-basics/gps-tracking-cheating-spouses-vehicle.

[61] State of California Employment Development Department, "Warn report," Accessed: Feb. 20, 2022 [Online]. Available: https://www.edd.ca.gov/Jobs_and_Training/warn/WARN_Report_for_7-1-2018_to_06-25-2019.pdf.

[62] Wikipedia contributors, "Drive.ai." Wikipedia, The Free Encyclopedia, Accessed: Aug. 29, 2021 [Online]. Available: https://en.wikipedia.org/w/index.php?title=Drive.ai&direction=prev&oldid=1061157803.

[63] NYSE, "Quote data," Accessed: Aug. 29, 2021 [Online]. Available: https://www.nyse.com/quote/XNYS:NIO.

[64] Wikipedia contributors, "On-board diagnostics." Wikipedia, The Free Encyclopedia, Accessed: Sep. 20, 2021 [Online]. Available: https://en.wikipedia.org/w/index.php?title=On-board_diagnostics&direction=prev&oldid=1046097709.

[65] I. Rouf, R. Miller, H. Mustafa, T. Taylor, S. Oh, W. Xu, M. Gruteser, W. Trappe, and I. Seskar, "Security and privacy vulnerabilities of In-Car wireless networks: A tire pressure monitoring system case study," in *19th USENIX Security Symposium (USENIX Security 10)*, (Washington, DC, USA), USENIX Association, Aug. 2010. Available: https://www.usenix.org/conference/usenixsecurity10/security-and-privacy-vulnerabilities-car-wireless-networks-tire-pressure.

[66] C. Miller and C. Valasek, "Remote exploitation of an unaltered passenger vehicle." Black Hat, (Dec. 29, 2015). Accessed: Sep. 20, 2021 [Online Video]. Available: https://www.youtube.com/watch?v=MAcHkASmXEc.

[67] D. Brouckmans, "Belgian security researchers from cosic and imec steal a tesla model x in minutes." Research Group COSIC, KU Leuven, Accessed: Sep. 20, 2021 [Online]. Available: https://www.esat.kuleuven.be/cosic/news/belgian-security-researchers-from-cosic-and-imec-steal-a-tesla-model-x-in-minutes/.

[68] L. Wouters, "Fast, furious and insecure: Passive keyless entry and start in modern supercars." Research Group COSIC, KU Leuven, Accessed: Sep. 20, 2021 [Online]. Available: https://www.esat.kuleuven.be/cosic/news/fast-furious-and-insecure-passive-keyless-entry-and-start-in-modern-supercars/.

[69] L. Wouters, E. Marin, T. Ashur, B. Gierlichs, and B. Preneel, "Fast, furious and insecure: Passive keyless entry and start systems in modern supercars," *IACR Transactions on Cryptographic Hardware and Embedded Systems*, vol. 2019, no. 3, pp. 66–85, 2019. https://doi.org/10.13154/tches.v2019.i3.66-85.

[70] J. Liu, "Belgian security researchers from ku leuven and imec demonstrate serious flaws in tesla model x keyless entry system." imec, Accessed: Sep. 20, 2021 [Online]. Available: https://www.imec-int.com/en/press/belgian-security-researchers-ku-leuven-and-imec-demonstrate-serious-flaws-tesla-model-x.

[71] Wikipedia contributors, "Remote keyless system." Wikipedia, The Free Encyclopedia, Accessed: Sep. 20, 2021 [Online]. Available: https://en.wikipedia.org/w/index.php?title=Remote_keyless_system&direction=prev&oldid=1045720575.

[72] S. Kamkar, "Def con 23 - samy kamkar - drive it like you hacked it: New attacks and tools to wireles." Def Con 23, (Dec. 2, 2015). Accessed: Sep. 20, 2021 [Online Video]. Available: https://www.youtube.com/watch?v=UNgvShN4USUc.

[73] S. Kamkar, "Radio hacking: Cars, hardware, and more! - samy kamkar - appsec california 2016." OWASP AppSec, (Mar. 21, 2016). Accessed: Sep. 20, 2021 [Online Video]. Available: https://www.youtube.com/watch?v=1RipwqJG50c.

[74] R.-P. Weinmann and B. Schmotzle, "Cansecwest 2021: Tbone drone vs tesla - ralf-philippe weinmann benedikt schmotzle, comsecuris." CANSECWEST 2021, (Apr. 28, 2021). Accessed: Sep. 20, 2021 [Online Video]. Available: https://www.youtube.com/watch?v=krSj81thN0w.

[75] S. Helme, "Hacking the nissan leaf - scott helme." SteelCon 2016, (Jul. 19, 2016). Accessed: Sep. 20, 2021 [Online Video]. Available: https://www.youtube.com/watch?v=egws2_WSUUE.

[76] A. Barisani and D. Bianco, "Unusual car navigation tricks: Injecting rds-tmc traffic information signals." CanSec West, Inverse Path Ltd., 2007. Accessed: Jan. 24, 2022 [Online]. Available: https://github.com/abarisani/abarisani.github.io/blob/master/research/rds/cansecwest_2007.pdf.

[77] Keen Security Lab, Tencent, "Over-the-air: How we remotely compromised the gateway, bcm, and autopilot ecus of tesla cars." Black Hat USA 2018, (Oct. 5, 2019). Accessed: Sep. 20, 2021 [Online Video]. Available: https://www.youtube.com/watch?v=N2uD1PoHaUE.

[78] S. Nie, L. Liu, Y. Du, and W. Zhang, "Over-the-air: How we remotely compromised the gateway, bcm, and autopilot ecus of tesla cars." Keen Security Lab, Tencent, Black Hat USA, 2018. Accessed: Sep. 20, 2021 [Online]. Available: https://i.blackhat.com/us-18/Thu-August-9/us-18-Liu-Over-The-Air-How-We-Remotely-Compromised-The-Gateway-Bcm-And-Autopilot-Ecus-Of-Tesla-Cars-wp.pdf.

[79] Wikipedia contributors, "Eddsa." Wikipedia, The Free Encyclopedia, Accessed: Sep. 20, 2021 [Online]. Available: https://en.wikipedia.org/w/index.php?title=EdDSA&direction=prev&oldid=1054647482.

[80] M. Goodman, *Future crimes: Everything is connected, everyone is vulnerable and what we can do about it*, ch. 12, pp. 352–354. Canada: Doubleday Canada, 2015.

[81] Kaspersky Lab and IAB, "Connected cars are now a reality, but are they secure?." Kaspersky.com, Accessed: Sep. 20, 2021 [Online]. Available: https://www.kaspersky.com/about/press-releases/2014_connected-cars-are-now-a-reality-but-are-they-secure.

[82] K. Munro, "Def con 24 car hacking village - ken munro - the mitsubishi hack explained." DEF CON 24, (Nov. 4, 2016). Accessed: Sep. 20, 2021 [Online Video]. Available: https://www.youtube.com/watch?v=aeWlfvJ5pEo.

[83] A. Laughlin, "We hacked a ford focus and volkswagen polo." Which?, Accessed: Sep. 20, 2021 [Online]. Available: https://www.which.co.uk/news/2020/04/we-hacked-a-ford-focus-and-a-volkswagen-polo/.

[84] Wikipedia contributors, "Automotive security." Wikipedia, The Free Encyclopedia, Accessed: Sep. 20, 2021 [Online]. Available: https://en.wikipedia.org/w/index.php?title=Automotive_security&direction=prev&oldid=1053152927.

[85] J. Rosevear, "Why nio stock is dropping today." NASDAQ.com, Accessed: Sep. 21, 2021 [Online]. Available: https://www.nasdaq.com/articles/why-nio-stock-is-dropping-today-2021-08-16.

[86] M. Fujiwara, A. Waseda, R. Nojima, S. Moriai, W. Ogata, and M. Sasaki, "Unbreakable distributed storage with quantum key distribution network and password-authenticated secret sharing," *Scientific reports*, vol. 6, 28998, 2016. Available: https://doi.org/10.1038/srep28988.

[87] M. Sasaki, "Masahide sasaki - tokyo qkd network for long-lived systems." SFB 1119 Crossing, (Dec. 5, 2017). Accessed: Sep. 21, 2021 [Online Video]. Available: https://www.youtube.com/watch?v=bGOFO8IBEOg&list=PLPGoYUDW8EJBafBvVBTCggiaJL3oKIzQM&index=10.

[88] E. Musk *et al.*, "Tesla ai day." Tesla, (Aug. 19, 2021). Accessed: Sep. 22, 2021 [Online Video]. Available: https://www.youtube.com/watch?v=j0z4FweCy4M&t=7514s.

[89] Wikipedia contributors, "Dubai police force." Wikipedia, The Free Encyclopedia, Accessed: Sep. 22, 2021 [Online]. Available: https://en.wikipedia.org/w/index.php?title=Dubai_Police_Force&direction=prev&oldid=1048895321.

[90] L. Jiang, A. Stocco, D. M. Losey, J. A. Abernethy, C. S. Prat, and R. P. Rao, "Brainnet: a multi-person brain-to-brain interface for direct collaboration between brains," *Scientific reports*, vol. 9, 6115, 2019. Available: https://doi.org/10.1038/s41598-019-41895-7.

[91] E. Lucero, "Unveiling our new quantum ai campus." Google, Accessed: Sep. 29, 2021 [Online]. Available: https://blog.google/technology/ai/unveiling-our-new-quantum-ai-campus/.

[92] N. Turok, *The universe within: from quatum to cosmos.* Toronto, ON, Canada: House of Anansi Press Inc., 2021.

[93] Wikipedia contributors, "Maximum likelihood estimation." Wikipedia, The Free Encyclopedia, Accessed: Nov. 2, 2021 [Online]. Available: https://en.wikipedia.org/w/index.php?title=Maximum_likelihood_estimation&direction=prev&oldid=1058836461.

[94] Wikipedia contributors, "White hat (computer security)." Wikipedia, The Free Encyclopedia, Accessed: Nov. 6, 2021 [Online]. Available: https://en.wikipedia.org/w/index.php?title=White_hat_(computer_security)&direction=prev&oldid=1054035023.

[95] E. Musk *et al.*, "Elon musk 'what if all teslas get hacked?' (fleet-wide hack 1.0)." Elon Musk Viral Videos, (Dec. 10, 2018). Accessed: Nov. 7, 2021 [Online Video]. Available: https://www.youtube.com/watch?v=5rMA-eVAgzg.

[96] N. G. Leveson, "Srecon19 europe/middle east/africa - a systems approach to safety and cybersecurity." USENIX, (Oct. 31, 2019). Accessed: Nov. 7, 2021 [Online Video]. Available: https://www.youtube.com/watch?v=LKI6oRZ49T0.

[97] J. Garcia, "Ibm and daimler use quantum computer to develop next-gen batteries." IBM, Accessed: Nov. 9, 2021 [Online]. Available: https://www.ibm.com/blogs/research/2020/01/next-gen-lithium-sulfur-batteries/.

[98] C. Wilson and J. Donohue, "Quantum today: Simulating quantum particles on a lattice." Institute for Quantum Computing (IQC), (Nov. 8, 2021). Accessed: Nov. 9, 2021 [Online Video]. Available: https://www.youtube.com/watch?v=VrRx58L2tn8.

[99] Wikipedia contributors, "Hiroshima." Wikipedia, The Free Encyclopedia, Accessed: Nov. 9, 2021 [Online]. Available: https://en.wikipedia.org/w/index.php?title=Hiroshima&direction=prev&oldid=1054733501.

[100] Wikipedia contributors, "Nagasaki." Wikipedia, The Free Encyclopedia, Accessed: Nov. 9, 2021 [Online]. Available: https://en.wikipedia.org/w/index.php?title=Nagasaki&direction=prev&oldid=1058200418.

[101] Office of Nuclear Energy, "9 notable facts about the world's first nuclear power plant - ebr-i." Energy.gov, Accessed: Nov. 9, 2021 [Online]. Available: https://www.energy.gov/ne/articles/9-notable-facts-about-worlds-first-nuclear-power-plant-ebr-i.

[102] Wikipedia contributors, "Chernobyl disaster." Wikipedia, The Free Encyclopedia, Accessed: Nov. 9, 2021 [Online]. Available: https://en.wikipedia.org/w/index.php?title=Chernobyl_disaster&direction=prev&oldid=1054514173.

[103] Wikipedia contributors, "Fukushima nuclear disaster." Wikipedia, The Free Encyclopedia, Accessed: Nov. 9, 2021 [Online]. Available: https://en.wikipedia.org/w/index.php?title=Fukushima_nuclear_disaster&direction=prev&oldid=1054646798.

[104] J. Ding and D. Schmidt, "Rainbow, a new multivariable polynomial signature scheme," in *Applied Cryptography and Network Security: Third International Conference, ACNS 2005*, (New York, NY, USA), pp. 164–175, Springer Berlin Heidelberg, Jun. 2005. Available: https://books.scholarsportal.info/uri/ebooks/ebooks2/springer/2011-04-28/3/9783540315421.

[105] J. Ding, M.-S. Chen, M. Kannwischer, J. Patarin, A. Petzoldt, D. Schmidt, and B.-Y. Yang, "Rainbow - algorithm specification and documentation: The 3rd round proposal." NIST Post-Quantum Cryptography Round 3 Submissions, Accessed: Apr. 17, 2021 [Online]. Available: https://csrc.nist.gov/projects/post-quantum-cryptography/round-3-submissions.

[106] J. Ding, M.-S. Chen, M. Kannwischer, J. Patarin, A. Petzoldt, D. Schmidt, and B.-Y. Yang, "Rainbow." First NIST PostQuantum Standardization Workshop, 2018. Accessed: Apr. 8, 2021 [Online]. Available: https://csrc.nist.gov/CSRC/media/Presentations/Rainbow/images-media/Rainbow-April2018.pdf.

[107] L. Lamport, "Constructing digital signatures from one way function," Tech. Rep. CSL 98, SRI International, Oct. 1979. Accessed: May. 27, 2021 [Online]. Available: https://www.microsoft.com/en-us/research/uploads/prod/2016/12/Constructing-Digital-Signatures-from-a-One-Way-Function.pdf.

[108] D. J. Bernstein, J. Buchmann, and E. Dahmen, *Post-Quantum Cryptography*, pp. 35–93. Germany: Springer-Verlag Berlin Heidelberg, 2009.

[109] R. C. Merkel, "A certified digital signature." Submitted to CACM, 1979. Accessed: May. 28, 2021 [Online]. Available: http://www.merkle.com/papers/Certified1979.pdf.

[110] A. Casanova, J.-C. Faugère, G. Macario-Rat, J. Patarin, L. Perret, and J. Ryckeghem, "Gemss: A great multivariate short signature." NIST Post-Quantum Cryptography Round 3 Submissions, Accessed: Apr. 4, 2021 [Online]. Available: https:

//csrc.nist.gov/projects/post-quantum-cryptography/round-3-submissions.

[111] D. Moody, "Nist status update on the 3rd round." NIST, USA, Accessed: Oct. 1, 2021 [Online]. Available: https://csrc.nist.gov/CSRC/media/Presentations/status-update-on-the-3rd-round/images-media/session-1-moody-nist-round-3-update.pdf.

[112] V. B. Dang, F. Farahmand, M. Andrzejczak, K. Mohajerani, D. T. Nguyen, and K. Gaj, "Implementation and benchmarking of round 2 candidates in the nist post-quantum cryptography standardization process using hardware and software/hardware co-design approaches." Cryptology ePrint Archive, Report 2020/795, 2020 Accessed: Oct. 1, 2021 [Online]. Available: https://eprint.iacr.org/2020/795.pdf.

[113] K. Gaj, "Implementation and benchmarking of round 2 candidates in the nist post-quantum cryptography standardization process using fpgas." NIST, USA, Accessed: Oct. 1, 2021 [Online]. Available: https://csrc.nist.gov/CSRC/media/Projects/post-quantum-cryptography/documents/round-3/seminars/oct-2020-gaj-kris-presentation.pdf.

[114] M. R. Albrecht, D. J. Bernstein, T. Chou, C. Cid, J. Gilcher, T. Lange, V. Maram, I. von Maurich, R. Misoczki, R. Niederhagen, K. G. Paterson, E. Persichetti, C. Peters, P. Schwabe, N. Sendrier, J. Szefer, C. J. Tjhai, M. Tomlinson, and W. Wang, "Classic mceliece," Accessed: Oct. 1, 2021 [Online]. Available: https://classic.mceliece.org/.

[115] S. Bai, L. Ducas, E. Kiltz, T. Lepoint, V. Lyubashevsky, P. Schwabe, G. Seiler, and D. Stehlé, "Crystals-dilithium," Accessed: Oct. 4, 2021 [Online]. Available: https://pq-crystals.org/dilithium/.

[116] P.-A. Fouque, J. Hoffstein, P. Kirchner, V. Lyubashevsky, T. Pornin, T. Prest, T. Ricosset, G. Seiler, W. Whyte, and Z. Zhang, "Falcon fast-fourier lattice-based compact signatures over ntru," Accessed: Oct. 4, 2021 [Online]. Available: https://falcon-sign.info/.

[117] A. Casanova, J.-C. Faugère, G. Macario-Rat, J. Patarin, L. Perret, and J. Ryckeghem, "Gemss: A great multivariate short signature," Accessed: Oct. 4, 2021 [Online]. Available: https://www-polsys.lip6.fr/Links/NIST/GeMSS.html.

[118] NIST, "Post quantum cryptography round 3 submissions," Accessed: Oct. 4, 2021 [Online]. Available: https://csrc.nist.gov/Projects/post-quantum-cryptography/round-3-submissions.

[119] J. Ding, M.-S. Chen, M. Kannwischer, J. Patarin, A. Petzoldt, D. Schmidt, and B.-Y. Yang, "Rainbow signature," Accessed: Oct. 4, 2021 [Online]. Available: https://www.pqcrainbow.org/.

[120] J.-P. Aumasson, D. J. Bernstein, W. Beullens, C. Dobraunig, M. Eichlseder, S. Fluhrer, S.-L. Gazdag, A. Hülsing, P. Kampanakis, S. Kölbl, T. Lange, M. M. Lauridsen, F. Mendel, R. Niederhagen, C. Rechberger, J. Rijneveld, P. Schwabe, and B. Westerbaan, "Sphincs+," Accessed: Oct. 4, 2021 [Online]. Available: https://sphincs.org/.

[121] G. Zaverucha, M. Chase, D. Derler, S. Goldfeder, D. Kales, J. Katz, V. Kolesnikov, C. Orlandi, S. Ramacher, C. Rechberger, D. Slamanig, and X. Wang, "Picnic," Accessed: Oct. 4, 2021 [Online]. Available: https://microsoft.github.io/Picnic/.

[122] R. Avanzi, J. Bos, L. Ducas, E. Kiltz, T. Lepoint, V. Lyubashevsky, J. M. Schanck, P. Schwabe, G. Seiler, and D. Stehlé, "Crystals-kyber," Accessed: Oct. 4, 2021 [Online]. Available: https://pq-crystals.org/kyber/.

[123] C. Chen, O. Danba, J. Hoffstein, A. Hülsing, J. Rijneveld, T. Saito, J. M. Schanck, P. Schwabe, W. Whyte, K. Xagawa, T. Yamakawa, and Z. Zhang, "Ntru," Accessed: Oct. 4, 2021 [Online]. Available: https://ntru.org/.

[124] J.-P. D'Anvers, A. Karmakar, S. S. Roy, F. Vercauteren, J. M. B. Mera, M. V. Beirendonck, and A. Basso, "Saber," Accessed: Oct. 4, 2021 [Online]. Available: https://www.esat.kuleuven.be/cosic/pqcrypto/saber/.

[125] D. Jao, R. Azarderakhsh, M. Campagna, C. Costello, L. D. Feo, B. Hess, A. Hutchinson, A. Jalali, K. Karabina, B. Koziel, B. LaMacchia, P. Longa, M. Naehrig, G. Pereira, J. Renes, V. Soukharev, and D. Urbanik, "Sike," Accessed: Oct. 4, 2021 [Online]. Available: https://sike.org/.

[126] Wikipedia contributors, "Supersingular isogeny key encapsulation." Wikipedia, The Free Encyclopedia, Accessed: Oct. 4, 2021 [Online]. Available: https://en.wikipedia.org/w/index.php?title=Supersingular_isogeny_key_exchange&direction=prev&oldid=1051796863.

[127] E. Alkim, J. W. Bos, L. Ducas, P. Longa, I. Mironov, M. Naehrig, V. Nikolaenko, C. Peikert, A. Raghunathan, D. Stebila, K. Easterbrook, and B. LaMacchia, "Frodokem," Accessed: Oct. 4, 2021 [Online]. Available: https://frodokem.org/.

[128] D. J. Bernstein, B. B. Brumley, M.-S. Chen, C. Chuengsatiansup, T. Lange, A. Marotzke, B.-Y. Peng, N. Tuveri, C. van Vredendaal, and B.-Y. Yang, "Ntru prime," Accessed: Oct. 4, 2021 [Online]. Available: https://ntruprime.cr.yp.to/.

[129] C. A. Melchor, N. Aragon, S. Bettaieb, L. Bidoux, O. Blazy, J. Bos, J.-C. Deneuville, A. Dion, P. Gaborit, J. Lacan, E. Persichetti, J.-M. Robert, P. Véron, and G. Zémor, "Hqc," Accessed: Oct. 4, 2021 [Online]. Available: http://pqc-hqc.org/.

[130] N. Aragon, P. S. L. M. Barreto, S. Bettaieb, L. Bidoux, O. Blazy, J.-C. Deneuville, P. Gaborit, S. Ghosh, S. Gueron, T. Güneysu, C. A. Melchor, R. Misoczki, E. Persichetti, J. Richter-Brockmann, N. Sendrier, J.-P. Tillich, V. Vasseur, and G. Zémor, "Bike - bit flipping key encapsulation," Accessed: Oct. 4, 2021 [Online]. Available: https://bikesuite.org/.

[131] rdrr.io, "ks.test: Kolmogorov-smirnov tests," Accessed: Jan. 10, 2022 [Online]. Available: https://rdrr.io/r/stats/ks.test.html.

[132] R Core Team, *R: A Language and Environment for Statistical Computing.* R Foundation for Statistical Computing, Vienna, Austria, 2021. Available: https://www.R-project.org/.

[133] M.-L. Delignette-Muller and C. Dutang, "fitdistrplus: An R package for fitting distributions," *Journal of Statistical Software*, vol. 64, no. 4, pp. 1–34, 2015. Available: https://www.jstatsoft.org/article/view/v064i04.

[134] RStudio Team, *RStudio: Integrated Development Environment for R*. RStudio, PBC, Boston, MA, 2021. Available: http://www.rstudio.com/.

[135] Microsoft (Excel), "Beta.inv function," Accessed: Feb. 6, 2022 [Online]. Available: https://support.microsoft.com/en-us/office/beta-inv-function-e84cb8aa-8df0-4cf6-9892-83a341d252eb.

[136] J. Tsitsiklis, "L06.5 total expectation theorem." MIT OpenCourseWare, (Apr. 24, 2018). Accessed: Feb. 6, 2022 [Online Video]. Available: https://www.youtube.com/watch?v=GnEyIawrWBg.

[137] Wikipedia contributors, “Law of total expectation.” Wikipedia, The Free Encyclopedia, Accessed: Feb. 6, 2022 [Online]. Available: https://en.wikipedia.org/w/index.php?title=Law_of_total_expectation&oldid=1061972846.

[138] M. Mosca, “Cybersecurity in a quantum world: will we be ready?.” NIST, 2015. Accessed: Feb. 6, 2022 [Online]. Available: https://csrc.nist.gov/csrc/media/events/workshop-on-cybersecurity-in-a-post-quantum-world/documents/presentations/session8-mosca-michele.pdf.

[139] tester, “Find quantiles of a specified beta distribution in R.” stackoverflow, 2020. Accessed: Oct. 15, 2021 [Online]. Available: https://stackoverflow.com/questions/64803765/find-quantiles-of-a-specified-beta-distribution-in-r.

[140] P. Tatti, “Is there an R function to find the probability of certain data being created from a beta distribution?.” stackoverflow, 2020. Accessed: Jan. 10, 2022 [Online]. Available: https://stackoverflow.com/questions/60479538/is-there-an-r-function-to-find-the-probability-of-certain-data-being-created-fro.

[141] SBQuantum, “Navigation for autonomous vehicles in gps denied environments.” sbquantum.com, Accessed: Feb. 7, 2022 [Online]. Available: https://sbquantum.com/.

[142] S. Tengler, “Top 25 auto cybersecurity hacks: Too many glass houses to be throwing stones.” Forbes, Accessed: Jul. 27, 2021 [Online]. Available: https://www.forbes.com/sites/stevetengler/2020/06/30/top-25-auto-cybersecurity-hacks-too-many-glass-houses-to-be-throwing-stones/?sh=2df5a70f7f65.

[143] K. Korosec, “Apple acquires self-driving startup drive.ai on the brink of closure.” Tech Crunch, Accessed: Feb. 20, 2022 [Online]. Available: https://techcrunch.com/2019/06/25/self-driving-startup-drive-ai-is-closing-down/.

[144] Wikipedia contributors, “Radar.” Wikipedia, The Free Encyclopedia, Accessed: Feb. 21, 2022 [Online]. Available: https://en.wikipedia.org/w/index.php?title=Radar&oldid=1072329221.

[145] Wikipedia contributors, “Waymo.” Wikipedia, The Free Encyclopedia, Accessed: Feb. 21, 2022 [Online]. Available: https://en.wikipedia.org/w/index.php?title=Waymo&oldid=1072780691.

[146] Wikipedia contributors, "Cruise (autonomous vehicle)." Wikipedia, The Free Encyclopedia, Accessed: Feb. 21, 2022 [Online]. Available: https://en.wikipedia.org/w/index.php?title=Cruise_(autonomous_vehicle)&oldid=1073198657.

[147] AutoX, "Autox builds the first level 4 robotaxi production line in china." AutoX.ai, (Dec. 21, 2021). Accessed: Feb. 21, 2022 [Online Video]. Available: https://www.youtube.com/watch?v=BEI-u-x9lTU.

[148] E. Walz, "Toyota-backed startup pony.ai builds autonomous driving computing unit built on the nvidia drive orin processor." FutureCar, Accessed: Feb. 21, 2022 [Online]. Available: https://www.futurecar.com/5145/Toyota-backed-Startup-Pony-ai-Builds-Autonomous-Driving-Computing-Unit-Built-on-the-NVIDIA-DRIVE-Orin-Processor/.

[149] WeRide, "Who are we." WeRide.ai, Accessed: Feb. 21, 2022 [Online]. Available: https://www.weride.ai/en/.

[150] W. Young Jr., "Understanding stpa-sec through a simple roller coaster example." 2016 STAMP Conference, Boston, MA, 2016. Accessed: May. 27, 2020 [Online]. Available: http://psas.scripts.mit.edu/home/wp-content/uploads/2016/01/24-BillYoung-W2016.pdf.

[151] Wikipedia contributors, "Event data recorder." Wikipedia, The Free Encyclopedia, Accessed: Feb. 27, 2022 [Online]. Available: https://en.wikipedia.org/w/index.php?title=Event_data_recorder&oldid=1073650157.

[152] Wikipedia contributors, "Dedicated short-range communications." Wikipedia, The Free Encyclopedia, Accessed: Feb. 27, 2022 [Online]. Available: https://en.wikipedia.org/w/index.php?title=Dedicated_short-range_communications&oldid=1044367760.

[153] Wikipedia contributors, "Bra–ket notation." Wikipedia, The Free Encyclopedia, Accessed: Feb. 27, 2022 [Online]. Available: https://en.wikipedia.org/w/index.php?title=Bra-ket_notation&oldid=1068398107.

[154] M. Kaku, "Can we have brain-to-brain communication? — michio kaku —big think." Big Think, (Sep. 26, 2012). Accessed: Feb. 28, 2022 [Online Video]. Available: https://www.youtube.com/watch?v=KXin82A6maM.

[155] W. Beullens, "Breaking rainbow takes a weekend on a laptop." Cryptology ePrint Archive, Report 2022/214, 2022 Accessed: Apr. 10, 2022 [Online]. Available: https://ia.cr/2022/214.

[156] E. Crockett, C. Paquin, and D. Stebila, "Prototyping post-quantum and hybrid key exchange and authentication in tls and ssh." Cryptology ePrint Archive, Report 2019/858, 2019 Accessed: Apr. 10, 2022 [Online]. Available: https://ia.cr/2019/858.

[157] A. Hülsing, S.-L. Gazdag, D. Butin, and J. Buchmann, "Hash-based signatures: An outline for a new standard," in *NIST Workshop on Cybersecurity in a Post-Quantum World*, pp. 1–12, 2015.

Appendices

Appendix A

Controller Constraints

UCAs	Controller Constraints
UCA1.1 Path planning system does not send an appropriate trajectory while the vehicle is in motion and needs to come to a stop or to slow down. (H1, H2, H3, H4, H5)	C1.1 Path planning system must send an appropriate trajectory while vehicle is in motion and needs to come to a stop or to slow down.
UCA1.2 Path planning system sends an inappropriate trajectory requiring braking while the vehicle needs to accelerate. (H1, H2, H3, H4, H5)	C1.2 Path planning system must not send an inappropriate trajectory requiring braking while vehicle is accelerating.
UCA1.3 Path planning system sends a trajectory too late when the vehicle needs to slow down or come to a stop. (H1, H2, H3, H4, H5)	C1.3 Path planning system must not send a trajectory too late while the vehicle needs to slow down or come to a stop.

Table A.1: Controller Constraints - Path Planning Trajectory (Brake)

UCAs	Controller Constraints
UCA2.1 Path planning system does not send an appropriate trajectory when there is an object that it needs to maneuver around while the vehicle is in motion. (H1, H2, H3, H4, H5)	C2.1 Path planning system must send an appropriate trajectory when there is an object that it needs to maneuver around while the vehicle is in motion.
UCA2.2 Path planning system provides inappropriate trajectory resulting in the vehicle changing lanes when it does not need to change lanes. (H1, H2, H3, H4, H5)	C2.2 Path planning system must not provide an inappropriate trajectory when it does not need to change lanes.
UCA2.3 Path planning system provides trajectory too late when the vehicle needs to make a lane change. (H1, H2, H3, H4, H5)	C2.3 Path planning system must not provide trajectory too late when making a lane change.

Table A.2: Controller Constraints - Path Planning Trajectory (Steering)

UCAs	Controller Constraints
UCA3.1 Path planning system does not send an appropriate trajectory when vehicle needs to change lanes. (H1, H2, H3, H4, H5)	C3.1 Path planning system must send an appropriate trajectory when the vehicle needs to change lanes.
UCA3.2 Path planning system provides an inappropriate trajectory when the vehicle needs to slow down or to stop. (H1, H2, H3, H4, H5)	C3.2 Path planning system must not provide an inappropriate trajectory when the vehicle needs to slow down or to stop.
UCA3.3 Path planning system provides trajectory too late while the vehicle is in an intersection. (H1, H3, H4, H5)	C3.3 Path planning system must not provide trajectory too late while near an intersection.

Table A.3: Controller Constraints - Path Planning Trajectory (Acceleration)

UCAs	Controller Constraints
UCA4.1 Brake controller does not send a brake control action while the vehicle is in motion and needs to come to a stop or to slow down. (H1, H2, H3, H4, H5)	C4.1 Brake controller must apply brakes while vehicle is in motion and needs to come to a stop or to slow down.
UCA4.2 Brake controller sends a brake control action while the vehicle is accelerating.(H1, H2, H3, H4, H5)	C4.2 Brake controller must not apply brakes while vehicle is accelerating.
UCA4.3 Brake controller sends a brake control action too early while planning to come to a stop. (H1, H3, H4, H5)	C4.3 Brake controller must not apply brakes too early while planning to come to a stop.
UCA4.4 Brake controller sends a brake control action too late while trying to slow down or come to a stop. (H1, H2, H3, H4, H5)	C4.4 Brake controller must not apply brakes too late while trying to slow down or come to a stop.
UCA4.5 Brake controller stopped the brake control action too soon when the vehicle needed to slow down or to stop. (H1, H3, H4, H5)	C4.5 Brake controller must not stop applying the brakes too soon when the vehicle needs to slow down or to stop.
UCA4.6 Brake controller applied the brake control action too long when the vehicle needed to slow down. (H1, H3, H4, H5)	C4.6 Brake controller must not apply the brakes too long when the vehicle needs to slow down.

Table A.4: Controller Constraints - Brake Controller

UCAs	Controller Constraints
UCA5.1 Steering controller does not send a turn control action when there is an object that it needs to maneuver around while the vehicle is in motion. (H1, H3, H4, H5)	C5.1 Steering controller must turn when there is an object that it needs to maneuver around while the vehicle is in motion.
UCA5.2 Steering controller provides turn control action when it does not need to change lanes. (H1, H2, H3, H4, H5)	C5.2 Steering controller must not provide a turn control action when it does not need to change lanes.
UCA5.3 Steering controller provides turn control action too early when making a lane change. (H1, H2, H3, H4, H5)	C5.3 Steering controller must not provide a turn control action too early when making a lane change.
UCA5.4 Steering controller provides turn control action too late when making a lane change. (H1, H2, H3, H4, H5)	C5.4 Steering controller must not provide turn control action too late when making a lane change.
UCA5.5 Steering controller stops the turn control action too soon when maneuvering around a curve. (H1, H2, H3, H4, H5)	C5.5 Steering controller must not stop the turn control action too soon when maneuvering around a curve.
UCA5.6 Steering controller applies the turn control action too long when maneuvering around a curve. (H1, H2, H3, H4, H5)	C5.6 Steering controller must not apply the turn control action too long when maneuvering around a curve.

Table A.5: Controller Constraints - Steering Controller

UCAs	Controller Constraints
UCA6.1 Engine controller does not send an acceleration control action when the vehicle is changing lanes. (H1, H2, H3, H4, H5)	C6.1 Engine controller must send an appropriate acceleration control action when the vehicle is changing lanes.
UCA6.2 Engine controller provides an acceleration control action when the vehicle needs to slow down or to stop. (H1, H2, H3, H4, H5)	C6.2 Engine controller must not provide an acceleration control action when the vehicle needs to slow down or to stop.
UCA6.3 Engine controller provides an acceleration control action too early while the vehicle is near an intersection. (H1, H3, H4, H5)	C6.3 Engine controller must not provide an acceleration control action too early while near an intersection.
UCA6.4 Engine controller provides an acceleration control action too late while the vehicle is in an intersection. (H1, H3, H4, H5)	C6.4 Engine controller must not provide acceleration control action too late while in an intersection.
UCA6.5 Engine controller stopped acceleration control action too soon while the vehicle is in an intersection. (H1, H3, H4, H5)	C6.5 Engine controller must not stop the acceleration control action too soon while vehicle is in an intersection.
UCA6.6 Engine controller applied acceleration control action too long while the vehicle is changing lanes. (H1, H2, H3, H4, H5)	C6.6 Engine controller must not apply acceleration control action too long while vehicle is changing lanes.

Table A.6: Controller Constraints - Engine Controller

Appendix B

Loss Scenarios

UCA	Cause	Scenario
UCA1.1 Path planning system does not send an appropriate trajectory while the vehicle is in motion and needs to come to a stop or to slow down. (H1, H2, H3, H4, H5)	The control algorithm has been tampered with	Scenario 1 - The path planning control algorithm intentionally does not send an appropriate trajectory when an obstacle is detected in the vehicle's path. This is due to a malicious software update which used a quantum computer to break the authentication protocol.
UCA1.1 Path planning system does not send an appropriate trajectory while the vehicle is in motion and needs to come to a stop or to slow down. (H1, H2, H3, H4, H5)	Inconsistent process model — Incorrect Feedback	Scenario 2 - The path planning system receives incorrect information from the obstacle detection, obstacle tracking, and localization systems that has been spoofed by a malicious software update to the perceptual unit which used a quantum computer to break the authentication protocol. This results in the path planning system not sending an appropriate trajectory when an obstacle is detected in the vehicle's path
UCA1.2 Path planning system sends an inappropriate trajectory requiring braking while the vehicle needs to accelerate. (H1, H2, H3, H4, H5)	The control algorithm has been tampered with	Scenario 3 - The path planning control algorithm intentionally sends an inappropriate trajectory while the vehicle needs to accelerate. This is due to a malicious software update which used a quantum computer to break the authentication protocol.

Table B.1: Loss Scenarios - Path Planning Trajectory (Brake) 1 of 2

UCA	Cause	Scenario
UCA1.2 Path planning system sends an inappropriate trajectory requiring braking while the vehicle needs to accelerate.(H1, H2, H3, H4, H5)	Inconsistent process model — Incorrect Feedback	Scenario 4 - The path planning system receives incorrect information from the obstacle detection, obstacle tracking, and localization systems that has been spoofed by a malicious software update to the perceptual unit which used a quantum computer to break the authentication protocol. This results in the path planning system sending an inappropriate trajectory requiring braking while the vehicle needs to accelerate.
UCA1.3 Path planning system sends a trajectory too late when the vehicle needs to slow down or come to a stop. (H1, H2, H3, H4, H5)	The control algorithm has been tampered with	Scenario 5 - The path planning control algorithm intentionally sends a trajectory too late when when the vehicle needs to slow down or come to a stop. This is due to a malicious software update which used a quantum computer to break the authentication protocol.
UCA1.3 Path planning system sends a trajectory too late when the vehicle needs to slow down or come to a stop. (H1, H2, H3, H4, H5)	Inconsistent process model — Incorrect Feedback	Scenario 6 - The path planning system receives incorrect information regarding the location of the vehicle that has been spoofed by a malicious software update to the perceptual unit which used a quantum computer to break the authentication protocol. This results in the path planning system sending a trajectory too late when the vehicle needs to slow down or come to a stop.

Table B.2: Loss Scenarios - Path Planning Trajectory (Brake) 2 of 2

UCA	Cause	Scenario
UCA2.1 Path planning system does not send an appropriate trajectory when there is an object that it needs to maneuver around while the vehicle is in motion. (H1, H2, H3, H4, H5)	The control algorithm has been tampered with	Scenario 7 - The path planning system intentionally does not send appropriate trajectory when there is an object in its way. This is due to a malicious software update which used a quantum computer to break the authentication protocol.
UCA2.1 Path planning system does not send an appropriate trajectory when there is an object that it needs to maneuver around while the vehicle is in motion. (H1, H2, H3, H4, H5)	Inconsistent process model — Incorrect Feedback	Scenario 8 - The path planning system receives incorrect information regarding objects that has been spoofed by a malicious software update to the perceptual unit which used a quantum computer to break the authentication protocol. This results in the path planning system not sending an appropriate trajectory when there is an object the vehicle needs to maneuver around.
UCA2.2 Path planning system provides inappropriate trajectory resulting in the vehicle changing lanes when it does not need to change lanes. (H1, H2, H3, H4, H5)	The control algorithm has been tampered with	Scenario 9 - The path planning system intentionally sends an inappropriate trajectory when it doesn't need to changes lanes. This is due to a malicious software update which used a quantum computer to break the authentication protocol.

Table B.3: Loss Scenarios - Path Planning Trajectory (Steering) 1 of 2

UCA	Cause	Scenario
UCA2.2 Path planning system provides inappropriate trajectory resulting in the vehicle changing lanes when it does not need to change lanes. (H1, H2, H3, H4, H5)	Inconsistent process model — Incorrect Feedback	Scenario 10 - The path planning system receives incorrect information from the obstacle detection, obstacle tracking, and localization systems that has been spoofed by a malicious software update to the perceptual unit which used a quantum computer to break the authentication protocol. This results in the path planning system sending an inappropriate trajectory when it does not need to change lanes.
UCA2.3 Path planning system provides trajectory too late when the vehicle needs to make a lane change. (H1, H2, H3, H4, H5)	The control algorithm has been tampered with	Scenario 11 - The path planning system intentionally sends trajectory too late when planning to make a lane change. This is due to a malicious software update which used a quantum computer to break the authentication protocol.
UCA2.3 Path planning system provides trajectory too late when the vehicle needs to make a lane change. (H1, H2, H3, H4, H5)	Inconsistent process model — Incorrect Feedback	Scenario 12 - The path planning system receives incorrect information from the obstacle detection, obstacle tracking, and localization systems that has been spoofed by a malicious software update to the perceptual unit which used a quantum computer to break the authentication protocol. This results in the path planning system sending a trajectory too late when planning to make a lane change.

Table B.4: Loss Scenarios - Path Planning Trajectory (Steering) 2 of 2

UCA	Cause	Scenario
UCA3.1 Path planning system does not send an appropriate trajectory when vehicle needs to change lanes. (H1, H2, H3, H4, H5)	The control algorithm has been tampered with	Scenario 13 - The path planning system intentionally does not provide an appropriate trajectory when the vehicle is changing lanes. This is due to a malicious software update which used a quantum computer to break the authentication protocol.
UCA3.1 Path planning system does not send an appropriate trajectory when vehicle needs to change lanes. (H1, H2, H3, H4, H5)	Inconsistent process model — Incorrect Feedback	Scenario 14 - The path planning system receives incorrect information regarding the road or location of the vehicle that has been spoofed by a malicious software update to the perceptual unit which used a quantum computer to break the authentication protocol. This results in the path planning system not providing an appropriate trajectory when the vehicle needs to change lanes.
UCA3.2 Path planning system provides an inappropriate trajectory when the vehicle needs to slow down or to stop. (H1, H2, H3, H4, H5)	The control algorithm has been tampered with	Scenario 15 - The path planning system intentionally provides an inappropriate trajectory when the vehicle needs to slow down or to stop. This is due to a malicious software update which used a quantum computer to break the authentication protocol.

Table B.5: Loss Scenarios - Path Planning Trajectory (Acceleration) 1 of 2

UCA	Cause	Scenario
UCA3.2 Path planning system provides an inappropriate trajectory when the vehicle needs to slow down or to stop. (H1, H2, H3, H4, H5)	Inconsistent process model — Incorrect Feedback	Scenario 16 - The path planning system receives incorrect information regarding the road, location of the vehicle, or objects in its path that has been spoofed by a malicious software update to the perceptual unit which used a quantum computer to break the authentication protocol causing the path planning system to provide an inappropriate trajectory when it needs to slow down or to stop.
UCA3.3 Path planning system provides trajectory too late while the vehicle is in an intersection. (H1, H3, H4, H5)	The control algorithm has been tampered with	Scenario 17 - The path planning system intentionally provides trajectory too late when the vehicle is near an intersection. This is due to a malicious software update which used a quantum computer to break the authentication protocol.
UCA3.3 Path planning system provides trajectory too late while the vehicle is in an intersection. (H1, H3, H4, H5)	Inconsistent process model — Incorrect Feedback	Scenario 18 - The path planning system receives incorrect information regarding the road, location of the vehicle, or objects in the vehicle's path that has been spoofed by a malicious software update to the perceptual unit which used a quantum computer to break the authentication protocol causing the path planning system to provide a trajectory too late while near an intersection.

Table B.6: Loss Scenarios - Path Planning Trajectory (Acceleration) 2 of 2

UCA	Cause	Scenario
UCA4.1 Brake controller does not send a brake control action while the vehicle is in motion and needs to come to a stop or to slow down. (H1, H2, H3, H4, H5)	The control algorithm has been tampered with	Scenario 19 - The brake controller intentionally does not send a brake control action when the vehicle is in motion and needs to come to a stop or to slow down. This is due to a malicious software update which used a quantum computer to break the authentication protocol.
UCA4.1 Brake controller does not send a brake control action while the vehicle is in motion and needs to come to a stop or to slow down. (H1, H2, H3, H4, H5)	Inconsistent process model — Incorrect Input	Scenario 20 - The brake controller receives incorrect information from the path planning system regarding a brake control action that is due to an attack which used a quantum computer to break the authentication protocol. This results in the brake controller not sending a brake control action when an obstacle is detected in the vehicle's path.
UCA4.2 Brake controller sends a brake control action while the vehicle is accelerating. (H1, H2, H3, H4, H5)	The control algorithm has been tampered with	Scenario 21 - The brake controller intentionally sends a brake control action while the vehicle is accelerating. This is due to a malicious software update which used a quantum computer to break the authentication protocol.

Table B.7: Loss Scenarios - Brake Controller 1 of 4

UCA	Cause	Scenario
UCA4.2 Brake controller sends a brake control action while the vehicle is accelerating. (H1, H2, H3, H4, H5)	Inconsistent process model — Incorrect Input	Scenario 22 - The brake controller receives incorrect information from the path planning system regarding a brake control action that is due to an attack which used a quantum computer to break the authentication protocol. This results in the brake controller sending a brake control action when the vehicle is accelerating.
UCA4.3 Brake controller sends a brake control action too early while planning to come to a stop. (H1, H3, H4, H5)	The control algorithm has been tampered with	Scenario 23 - The brake controller intentionally sends a brake control action too early when coming to a stop. This is due to a malicious software update which used a quantum computer to break the authentication protocol.
UCA4.3 Brake controller sends a brake control action too early while planning to come to a stop. (H1, H3, H4, H5)	Inconsistent process model — Incorrect Input	Scenario 24 - The brake controller receives incorrect information from the path planning system regarding a brake control action that is due to an attack which used a quantum computer to break the authentication protocol. This results in the path planning system sending a brake control action too early when planning to come to a stop.

Table B.8: Loss Scenarios - Brake Controller 2 of 4

UCA	Cause	Scenario
UCA4.4 Brake controller sends a brake control action too late while trying to slow down or come to a stop. (H1, H2, H3, H4, H5)	The control algorithm has been tampered with	Scenario 25 - The brake controller intentionally sends a brake control action too late when coming to a stop. This is due to a malicious software update which used a quantum computer to break the authentication protocol.
UCA4.4 Brake controller sends a brake control action too late while trying to slow down or come to a stop. (H1, H2, H3, H4, H5)	Inconsistent process model — Incorrect Input	Scenario 26 - The brake controller receives incorrect information from the path planning system regarding a brake control action that is due to an attack which used a quantum computer to break the authentication protocol. This results in the path planning system sending a brake control action too late when trying to slow down or come to a stop.
UCA4.5 Brake controller stopped the brake control action too soon when the vehicle needed to slow down or to stop. (H1, H3, H4, H5)	The control algorithm has been tampered with	Scenario 27 - The brake controller intentionally stopped the brake control action too soon when the vehicle needed to slow down or to stop. This is due to a malicious software update which used a quantum computer to break the authentication protocol.

Table B.9: Loss Scenarios - Brake Controller 3 of 4

UCA	Cause	Scenario
UCA4.5 Brake controller stopped the brake control action too soon when the vehicle needed to slow down or to stop. (H1, H3, H4, H5)	Inconsistent process model — Incorrect Input	Scenario 28 - The brake controller receives incorrect information from the path planning system regarding a brake control action that is due to an attack which used a quantum computer to break the authentication protocol. This results in the path planning system stopping the brake control action too soon when the vehicle needs to slow down or to stop.
UCA4.6 Brake controller applied the brake control action too long when the vehicle needed to slow down. (H1, H3, H4, H5)	The control algorithm has been tampered with	Scenario 29 - The brake controller intentionally applies the brake control action too long when the vehicle needs to slow down or to stop. This is due to a malicious software update which used a quantum computer to break the authentication protocol.
UCA4.6 Brake controller applied the brake control action too long when the vehicle needed to slow down. (H1, H3, H4, H5)	Inconsistent process model — Incorrect Input	Scenario 30 - The brake controller receives incorrect information from the path planning system regarding a brake control action that is due to an attack which used a quantum computer to break the authentication protocol. This results in the path planning system applying the brake control action too long when the vehicle needs to slow down.

Table B.10: Loss Scenarios - Brake Controller 4 of 4

UCA	Cause	Scenario
UCA5.1 Steering controller does not send a turn control action when there is an object that it needs to maneuver around while the vehicle is in motion. (H1, H3, H4, H5)	The control algorithm has been tampered with	Scenario 31 - The steering controller intentionally does not send a turn control action when there is an object in its way. This is due to a malicious software update which used a quantum computer to break the authentication protocol.
UCA5.1 Steering controller does not send a turn control action when there is an object that it needs to maneuver around while the vehicle is in motion. (H1, H3, H4, H5)	Inconsistent process model — Incorrect Input	Scenario 32 - The steering controller receives incorrect information from the path planning system regarding a turn control action that is due to an attack which used a quantum computer to break the authentication protocol. This results in the steering controller not sending a turn control action there is an object the vehicle needs to maneuver around.
UCA5.2 Steering controller provides turn control action when it does not need to change lanes. (H1, H2, H3, H4, H5)	The control algorithm has been tampered with	Scenario 33 - The steering controller intentionally sends a turn control action when it does not need to change lanes. This is due to a malicious software update which used a quantum computer to break the authentication protocol.

Table B.11: Loss Scenarios - Steering Controller 1 of 4

UCA	Cause	Scenario
UCA5.2 Steering controller provides turn control action when it does not need to change lanes. (H1, H2, H3, H4, H5)	Inconsistent process model — Incorrect Input	Scenario 34 -The steering controller receives incorrect information from the path planning system regarding a turn control action that is due to an attack which used a quantum computer to break the authentication protocol. This results in the steering controller sending a turn control action when it does not need to change lanes.
UCA5.3 Steering controller provides turn control action too early when making a lane change. (H1, H2, H3, H4, H5)	The control algorithm has been tampered with	Scenario 35 - The steering controller intentionally sends a turn control action too early when planning to make a lane change. This is due to a malicious software update which used a quantum computer to break the authentication protocol.
UCA5.3 Steering controller provides turn control action too early when making a lane change. (H1, H2, H3, H4, H5)	Inconsistent process model — Incorrect Input	Scenario 36 - The steering controller receives incorrect information from the path planning system regarding a turn control action that is due to an attack which used a quantum computer to break the authentication protocol. This results in the path planning system sending a turn control action too early when planning to make a lane change.

Table B.12: Loss Scenarios - Steering Controller 2 of 4

UCA	Cause	Scenario
UCA5.4 Steering controller provides turn control action too late when making a lane change. (H1, H2, H3, H4, H5)	The control algorithm has been tampered with	Scenario 37 - The steering controller intentionally sends a turn control action too late when planning to make a lane change. This is due too a malicious software update which used a quantum computer to break the authentication protocol.
UCA5.4 Steering controller provides turn control action too late when making a lane change. (H1, H2, H3, H4, H5)	Inconsistent process model — Incorrect Input	Scenario 38 - The steering controller receives incorrect information from the path planning system regarding a turn control action that is due to an attack which used a quantum computer to break the authentication protocol. This results in the path planning system sending a turn control action too late when planning to make a lane change.
UCA5.5 Steering controller stops the turn control action too soon when maneuvering around a curve. (H1, H2, H3, H4, H5)	The control algorithm has been tampered with	Scenario 39 - The steering controller intentionally stops the turn control action too soon when maneuvering around a curve. This is due to a malicious software update which used a quantum computer to break the authentication protocol.

Table B.13: Loss Scenarios - Steering Controller 3 of 4

UCA	Cause	Scenario
UCA5.5 Steering controller stops the turn control action too soon when maneuvering around a curve. (H1, H2, H3, H4, H5)	Inconsistent process model — Incorrect Input	Scenario 40 - The steering controller receives incorrect information from the path planning system regarding a turn control action that is due to an attack which used a quantum computer to break the authentication protocol. This results in the path planning system stopping the turn control action too soon when maneuvering around a curve.
UCA5.6 Steering controller applies the turn control action too long when maneuvering around a curve. (H1, H2, H3, H4, H5)	The control algorithm has been tampered with	Scenario 41 - The steering controller intentionally applies the turn control action too long when maneuvering around a curve. This is due to a malicious software update which used a quantum computer to break the authentication protocol.
UCA5.6 Steering controller applies the turn control action too long when maneuvering around a curve. (H1, H2, H3, H4, H5)	Inconsistent process model — Incorrect Input	Scenario 42 - The steering controller receives incorrect information from the path planning system regarding a turn control action that is due to an attack which used a quantum computer to break the authentication protocol causing it to apply the turn control action too long.

Table B.14: Loss Scenarios - Steering Controller 4 of 4

UCA	Cause	Scenario
UCA6.1 Engine controller does not send an acceleration control action when the vehicle is changing lanes. (H1, H2, H3, H4, H5)	The control algorithm has been tampered with	Scenario 43 - The engine controller intentionally does not provide an acceleration control action when the vehicle is changing lanes. This is due to a malicious software update which used a quantum computer to break the authentication protocol.
UCA6.1 Engine controller does not send an acceleration control action when the vehicle is changing lanes. (H1, H2, H3, H4, H5)	Inconsistent process model — Incorrect Input	Scenario 44 - The engine controller receives incorrect information from the path planning system regarding an acceleration control action that is due to an attack which used a quantum computer to break the authentication protocol. This results in the path planning system not providing an acceleration control action when the vehicle is changing lanes.
UCA6.2 Engine controller provides an acceleration control action when the vehicle needs to slow down or to stop. (H1, H2, H3, H4, H5)	The control algorithm has been tampered with	Scenario 45 - The engine controller intentionally provides an acceleration control action when the vehicle needs to slow down or to stop. This is due to a malicious software update which used a quantum computer to break the authentication protocol.

Table B.15: Loss Scenarios - Engine Controller 1 of 4

UCA	Cause	Scenario
UCA6.2 Engine controller provides an acceleration control action when the vehicle needs to slow down or to stop. (H1, H2, H3, H4, H5)	Inconsistent process model — Incorrect Input	Scenario 46 - The engine controller receives incorrect information from the path planning system regarding an acceleration control action that is due to an attack which used a quantum computer to break the authentication protocol causing the vehicle to accelerate when it needs to slow down or to stop.
UCA6.3 Engine controller provides an acceleration control action too early while the vehicle is near an intersection. (H1, H3, H4, H5)	The control algorithm has been tampered with	Scenario 47 - The engine controller intentionally provides an acceleration control action too early when the vehicle is near an intersection. This is due to a malicious software update which used a quantum computer to break the authentication protocol.
UCA6.3 Engine controller provides an acceleration control action too early while the vehicle is near an intersection. (H1, H3, H4, H5)	Inconsistent process model — Incorrect Input	Scenario 48 - The engine controller receives incorrect information from the path planning system regarding an acceleration control action that is due to an attack which used a quantum computer to break the authentication protocol causing the vehicle to accelerate too early while near an intersection.

Table B.16: Loss Scenarios - Engine Controller 2 of 4

UCA	Cause	Scenario
UCA6.4 Engine controller provides an acceleration control action too late while the vehicle is in an intersection. (H1, H3, H4, H5)	The control algorithm has been tampered with	Scenario 49 - The engine controller intentionally provides an acceleration control action too late when the vehicle is in an intersection. This is due to a malicious software update which used a quantum computer to break the authentication protocol.
UCA6.4 Engine controller provides an acceleration control action too late while the vehicle is in an intersection. (H1, H3, H4, H5)	Inconsistent process model — Incorrect Input	Scenario 50 -The engine controller receives incorrect information from the path planning system regarding an acceleration control action that is due to an attack which used a quantum computer to break the authentication protocol causing the vehicle to accelerate too late while in an intersection.
UCA6.5 Engine controller stopped acceleration control action too soon while the vehicle is in an intersection. (H1, H3, H4, H5)	The control algorithm has been tampered with	Scenario 51 - The engine controller intentionally stopped the acceleration control action too soon while the vehicle is in an intersection. This is due to a malicious software update which used a quantum computer to break the authentication protocol.

Table B.17: Loss Scenarios - Engine Controller 3 of 4

UCA	Cause	Scenario
UCA6.5 Engine controller stopped acceleration control action too soon while the vehicle is in an intersection. (H1, H3, H4, H5)	Inconsistent process model — Incorrect Input	Scenario 52 - The engine controller receives incorrect information from the path planning system regarding an acceleration control action that is due to an attack which used a quantum computer to break the authentication protocol causing the engine controller to stop issuing an acceleration control action too soon while the vehicle is in an intersection .
UCA6.6 Engine controller applied acceleration control action too long while the vehicle is changing lanes. (H1, H2, H3, H4, H5)	The control algorithm has been tampered with	Scenario 53 - The engine controller intentionally applied the acceleration control action too long while the vehicle is changing lanes. This is due to a malicious software update which used a quantum computer to break the authentication protocol.
UCA6.6 Engine controller applied acceleration control action too long while the vehicle is changing lanes. (H1, H2, H3, H4, H5)	Inconsistent process model — Incorrect Input	Scenario 54 - The engine controller receives incorrect information from the path planning system regarding an acceleration control action that is due to an attack which used a quantum computer to break the authentication protocol resulting in the acceleration control action being applied too long while the vehicle is changing lanes.

Table B.18: Loss Scenarios - Engine Controller 4 of 4

Appendix C

Sample R Code for All Respondents - 15 years (Minimum)

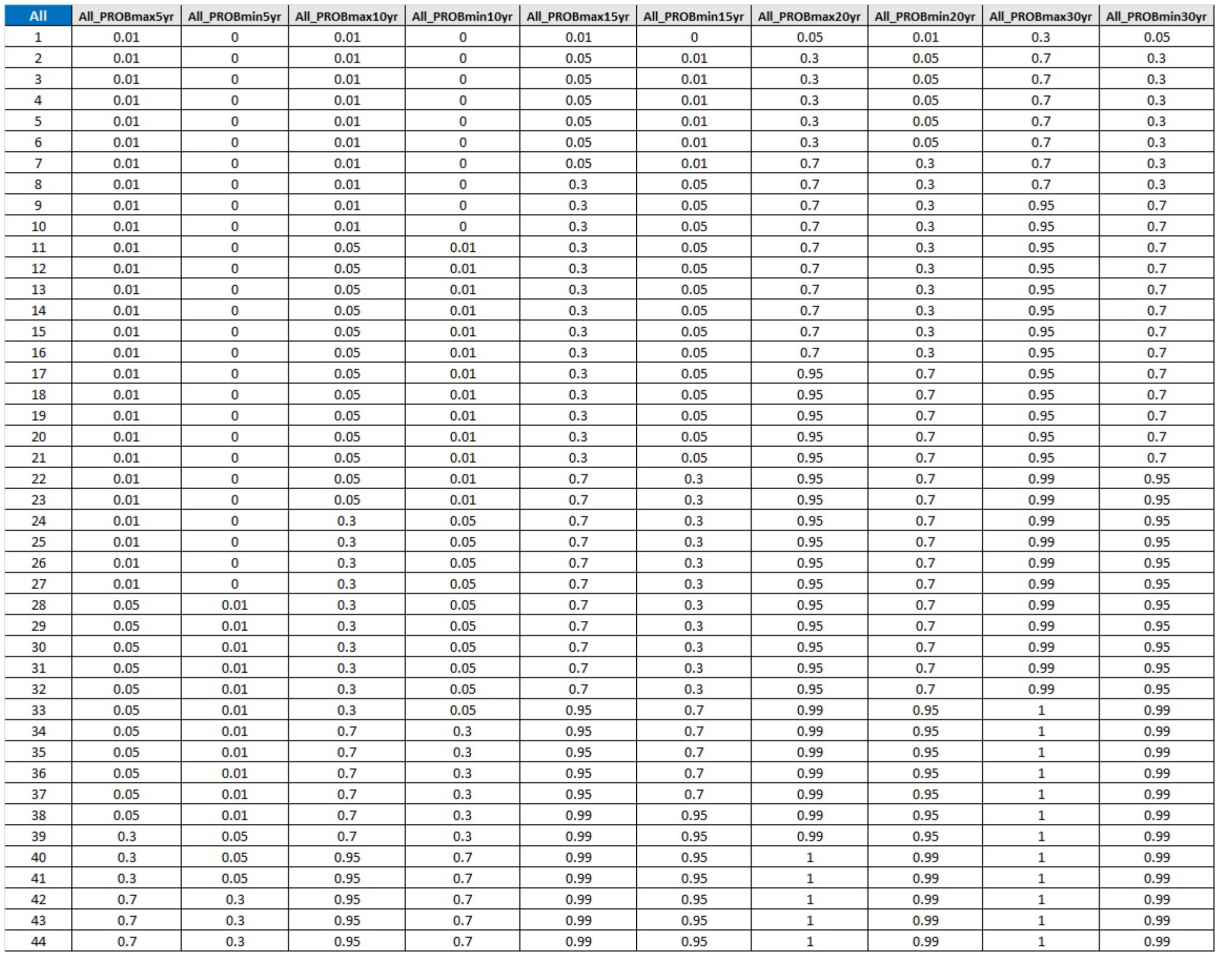
```
> library(fitdistrplus) # [133]
Loading required package: MASS
Loading required package: survival
> percentiles <- c(0.75,0.9,0.95,0.99,0.996,0.998,0.999) #percentiles [139]
> All_PROBmin15yr <- read.table("All_PROBmin15yr.txt") #input data
> All_PROBmin15yr
     V1
1  0.00
2  0.01
3  0.01
4  0.01
5  0.01
6  0.01
7  0.01
8  0.05
9  0.05
10 0.05
11 0.05
12 0.05
13 0.05
14 0.05
15 0.05
16 0.05
17 0.05
18 0.05
19 0.05
20 0.05
21 0.05
22 0.30
23 0.30
24 0.30
25 0.30
26 0.30
27 0.30
28 0.30
29 0.30
30 0.30
31 0.30
```

```
32 0.30
33 0.70
34 0.70
35 0.70
36 0.70
37 0.70
38 0.95
39 0.95
40 0.95
41 0.95
42 0.95
43 0.95
44 0.95

> #fit model [133]
> fit_beta_All_PROBmin15yr <- fitdist(All_PROBmin15yr$V1, "beta",
method = "mge", gof = "KS")
> fit_beta_All_PROBmin15yr
Fitting of the distribution ' beta ' by maximum goodness-of-fit
Parameters:
        estimate
shape1 0.3483992
shape2 0.8028486
> #Perform a Kolmogorov-Smirnov Test [140]
> ks.test(All_PROBmin15yr$V1, "pbeta", shape1 = 0.3483992, shape2 = 0.8028486)

One-sample Kolmogorov-Smirnov test

data:  All_PROBmin15yr$V1
D = 0.15909, p-value = 0.2154
alternative hypothesis: two-sided

Warning message:
In ks.test(All_PROBmin15yr$V1, "pbeta", shape1 = 0.3483992, shape2 = 0.8028486) :
  ties should not be present for the Kolmogorov-Smirnov test

> #Calculate the percentiles of the fitted Beta distribution [139]
> qbeta(p=percentiles, shape1 = 0.3483992, shape2 = 0.8028486)
```

```
[1] 0.5356206 0.8288229 0.9249758 0.9896464 0.9966846 0.9986008 0.9994097

> mean(All_PROBmin15yr$V1)
[1] 0.3229545
```

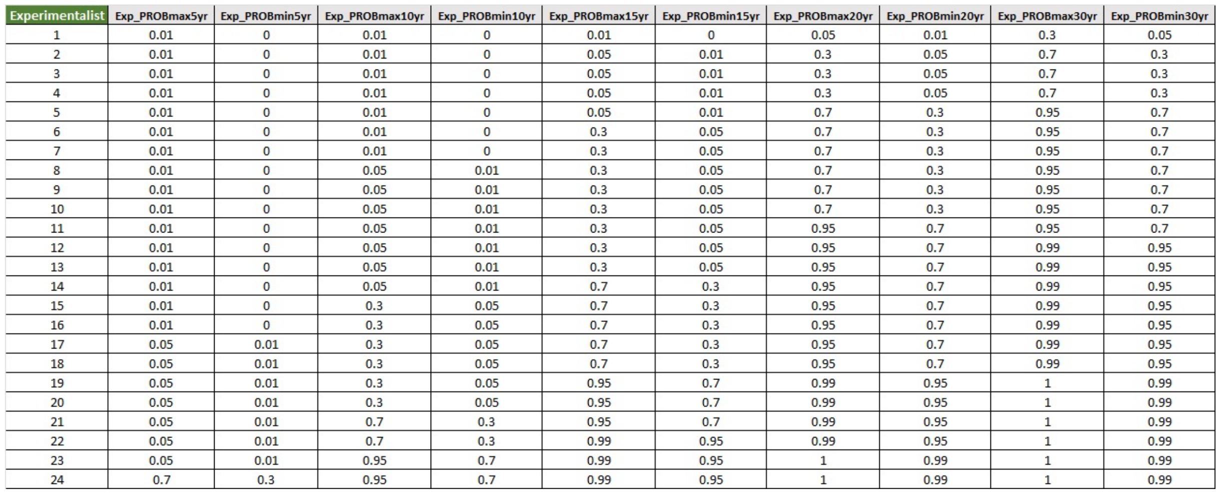
Appendix D

Modelling Datasets



All	All_PROBmax5yr	All_PROBmin5yr	All_PROBmax10yr	All_PROBmin10yr	All_PROBmax15yr	All_PROBmin15yr	All_PROBmax20yr	All_PROBmin20yr	All_PROBmax30yr	All_PROBmin30yr
1	0.01	0	0.01	0	0.01	0	0.05	0.01	0.3	0.05
2	0.01	0	0.01	0	0.05	0.01	0.3	0.05	0.7	0.3
3	0.01	0	0.01	0	0.05	0.01	0.3	0.05	0.7	0.3
4	0.01	0	0.01	0	0.05	0.01	0.3	0.05	0.7	0.3
5	0.01	0	0.01	0	0.05	0.01	0.3	0.05	0.7	0.3
6	0.01	0	0.01	0	0.05	0.01	0.3	0.05	0.7	0.3
7	0.01	0	0.01	0	0.05	0.01	0.7	0.3	0.7	0.3
8	0.01	0	0.01	0	0.3	0.05	0.7	0.3	0.7	0.3
9	0.01	0	0.01	0	0.3	0.05	0.7	0.3	0.95	0.7
10	0.01	0	0.01	0	0.3	0.05	0.7	0.3	0.95	0.7
11	0.01	0	0.05	0.01	0.3	0.05	0.7	0.3	0.95	0.7
12	0.01	0	0.05	0.01	0.3	0.05	0.7	0.3	0.95	0.7
13	0.01	0	0.05	0.01	0.3	0.05	0.7	0.3	0.95	0.7
14	0.01	0	0.05	0.01	0.3	0.05	0.7	0.3	0.95	0.7
15	0.01	0	0.05	0.01	0.3	0.05	0.7	0.3	0.95	0.7
16	0.01	0	0.05	0.01	0.3	0.05	0.7	0.3	0.95	0.7
17	0.01	0	0.05	0.01	0.3	0.05	0.95	0.7	0.95	0.7
18	0.01	0	0.05	0.01	0.3	0.05	0.95	0.7	0.95	0.7
19	0.01	0	0.05	0.01	0.3	0.05	0.95	0.7	0.95	0.7
20	0.01	0	0.05	0.01	0.3	0.05	0.95	0.7	0.95	0.7
21	0.01	0	0.05	0.01	0.3	0.05	0.95	0.7	0.95	0.7
22	0.01	0	0.05	0.01	0.7	0.3	0.95	0.7	0.99	0.95
23	0.01	0	0.05	0.01	0.7	0.3	0.95	0.7	0.99	0.95
24	0.01	0	0.3	0.05	0.7	0.3	0.95	0.7	0.99	0.95
25	0.01	0	0.3	0.05	0.7	0.3	0.95	0.7	0.99	0.95
26	0.01	0	0.3	0.05	0.7	0.3	0.95	0.7	0.99	0.95
27	0.01	0	0.3	0.05	0.7	0.3	0.95	0.7	0.99	0.95
28	0.05	0.01	0.3	0.05	0.7	0.3	0.95	0.7	0.99	0.95
29	0.05	0.01	0.3	0.05	0.7	0.3	0.95	0.7	0.99	0.95
30	0.05	0.01	0.3	0.05	0.7	0.3	0.95	0.7	0.99	0.95
31	0.05	0.01	0.3	0.05	0.7	0.3	0.95	0.7	0.99	0.95
32	0.05	0.01	0.3	0.05	0.7	0.3	0.95	0.7	0.99	0.95
33	0.05	0.01	0.3	0.05	0.95	0.7	0.99	0.95	1	0.99
34	0.05	0.01	0.7	0.3	0.95	0.7	0.99	0.95	1	0.99
35	0.05	0.01	0.7	0.3	0.95	0.7	0.99	0.95	1	0.99
36	0.05	0.01	0.7	0.3	0.95	0.7	0.99	0.95	1	0.99
37	0.05	0.01	0.7	0.3	0.95	0.7	0.99	0.95	1	0.99
38	0.05	0.01	0.7	0.3	0.99	0.95	0.99	0.95	1	0.99
39	0.3	0.05	0.7	0.3	0.99	0.95	0.99	0.95	1	0.99
40	0.3	0.05	0.95	0.7	0.99	0.95	1	0.99	1	0.99
41	0.3	0.05	0.95	0.7	0.99	0.95	1	0.99	1	0.99
42	0.7	0.3	0.95	0.7	0.99	0.95	1	0.99	1	0.99
43	0.7	0.3	0.95	0.7	0.99	0.95	1	0.99	1	0.99
44	0.7	0.3	0.95	0.7	0.99	0.95	1	0.99	1	0.99

Figure D.1: (All Respondents) Datasets Based on Survey Results based off of Table 11.6 [5]



Experimentalist	Exp_PROBmax5yr	Exp_PROBmin5yr	Exp_PROBmax10yr	Exp_PROBmin10yr	Exp_PROBmax15yr	Exp_PROBmin15yr	Exp_PROBmax20yr	Exp_PROBmin20yr	Exp_PROBmax30yr	Exp_PROBmin30yr
1	0.01	0	0.01	0	0.01	0	0.05	0.01	0.3	0.05
2	0.01	0	0.01	0	0.05	0.01	0.3	0.05	0.7	0.3
3	0.01	0	0.01	0	0.05	0.01	0.3	0.05	0.7	0.3
4	0.01	0	0.01	0	0.05	0.01	0.3	0.05	0.7	0.3
5	0.01	0	0.01	0	0.05	0.01	0.7	0.3	0.95	0.7
6	0.01	0	0.01	0	0.3	0.05	0.7	0.3	0.95	0.7
7	0.01	0	0.01	0	0.3	0.05	0.7	0.3	0.95	0.7
8	0.01	0	0.05	0.01	0.3	0.05	0.7	0.3	0.95	0.7
9	0.01	0	0.05	0.01	0.3	0.05	0.7	0.3	0.95	0.7
10	0.01	0	0.05	0.01	0.3	0.05	0.7	0.3	0.95	0.7
11	0.01	0	0.05	0.01	0.3	0.05	0.95	0.7	0.95	0.7
12	0.01	0	0.05	0.01	0.3	0.05	0.95	0.7	0.99	0.95
13	0.01	0	0.05	0.01	0.3	0.05	0.95	0.7	0.99	0.95
14	0.01	0	0.05	0.01	0.7	0.3	0.95	0.7	0.99	0.95
15	0.01	0	0.3	0.05	0.7	0.3	0.95	0.7	0.99	0.95
16	0.01	0	0.3	0.05	0.7	0.3	0.95	0.7	0.99	0.95
17	0.05	0.01	0.3	0.05	0.7	0.3	0.95	0.7	0.99	0.95
18	0.05	0.01	0.3	0.05	0.7	0.3	0.95	0.7	0.99	0.95
19	0.05	0.01	0.3	0.05	0.95	0.7	0.99	0.95	1	0.99
20	0.05	0.01	0.3	0.05	0.95	0.7	0.99	0.95	1	0.99
21	0.05	0.01	0.7	0.3	0.95	0.7	0.99	0.95	1	0.99
22	0.05	0.01	0.7	0.3	0.99	0.95	0.99	0.95	1	0.99
23	0.05	0.01	0.95	0.7	0.99	0.95	1	0.99	1	0.99
24	0.7	0.3	0.95	0.7	0.99	0.95	1	0.99	1	0.99

Figure D.2: (Experimentalists) Datasets Based on Survey Results based off of Table 11.7 [5]