network

MDPI

Article

Call Me Maybe: Using Dynamic Protocol Switching to Mitigate Denial-of-Service Attacks on VoIP Systems

John Kafke and Thiago Viana *

Cyber and Technical Computing, University of Gloucestershire, Cheltenham GL50 2RH, UK
* Correspondence: tviana1@glos.ac.uk; Tel.: +44-01242-715007

Abstract: Voice over IP is quickly becoming the industry standard voice communication service. While using an IP-based method of communication has many advantages, it also comes with a new set of challenges; voice networks are now accessible to a multitude of internet-based attackers from anywhere in the world. One of the most prevalent threats to a VoIP network are Denial-of-Service attacks, which consume network bandwidth to congest or disable the communication service. This paper looks at the current state of research into the mitigation of these attacks against VoIP networks, to see if the mechanisms in place are enough. A new framework is proposed titled the "Call Me Maybe" framework, combining elements of latency monitoring with dynamic protocol switching to mitigate DoS attacks against VoIP systems. Research conducted around routing VoIP over TCP rather than UDP is integrated into the proposed design, along with a latency monitoring mechanism to detect when the service is under attack. Data gathered from a Cisco Packet Tracer simulation was used to evaluate the effectiveness of the solution. The gathered results have shown that there is a statistically significant improvement in the response times of voice traffic when using the "Call Me Maybe" framework in a network experiencing a DoS attack. The research and findings therefore aim to provide a contribution to the enhancement of the security of VoIP and future IP-based voice communication systems.

Keywords: Voice-over-IP (VoIP); Denial-of-Service (DoS); Transmission Control Protocol (TCP)

Citation: Kafke, J.; Viana, T. Call Me Maybe: Using Dynamic Protocol Switching to Mitigate Denial-of-Service Attacks on VoIP Systems. *Network* **2022**, *2*, 545–567. https://doi.org/10.3390/network2040032

Academic Editor: Rajendra V. Boppana

Received: 29 August 2022
Accepted: 14 October 2022
Published: 18 October 2022

1. Introduction

As the deadlines for the "copper switch-off" approach, with the complete replacement of the traditional Public Switched Telephone Network (PSTN) by the Voice over Internet Protocol (VoIP) comes a different set of challenges unique to this internet-based communication system [1]. This is because the technologies used by VoIP (Session Initiation Protocol (SIP), Media Gateway Control Protocol (MGCP), and H.323) operate in an open environment, as opposed to the closed circuit-switch approach of PSTN. Techniques previously only accomplishable by malicious attackers with physical access to the wires that carried the voice communication are now possible from across the globe. Furthermore, attacks that exploited IP-based services, such as Denial-of-Service, are now able to target the voice communication network.

Despite being one of the easiest and cheapest attacks to execute, DoS attacks pose a significant risk to businesses and users around the world, especially when targeting the VoIP network. Many corporations now use VoIP as their main line of communication between employees via tools such as Skype and Microsoft Teams, and with working from home becoming increasingly common, such an attack would be devastating for small and large organisations alike. When the communication system is disrupted, all areas of a business are affected, and in the worst cases can result in a temporary cessation of operations until the problem is resolved. Therefore, it is essential that security against DoS is kept up to date with the rapid advances in technique of those executing these attacks.

This paper proposes a two-stage 'detecting and rerouting' technique for the mitigation of UDP based attacks. The design uses a latency monitoring system to continuously scan VoIP connections within a network for increases in response times, which could indicate the presence of a DoS attack. Upon detection, the VoIP packets are dynamically rerouted through a twin TCP network set up alongside the original UDP network. The TCP network also contains a UDP restricted stateless firewall, which blocks common UDP based DoS attacks such as UDP flooding. By combining these techniques, a method is set out for the reactive mitigation of UDP based DoS attacks. The paper also presents evidence that VoIP over TCP has a place in highly secure networks despite the disadvantages of using TCP for continuous data transfer.

The latency monitoring system for detecting (D)DoS attacks enables the framework to act as a fool-proof final layer of defense for the network that can be used in conjunction with alternative higher-level techniques. The proposed framework provides as main contributions a scalable, adaptable solution for UDP DoS attacks that incorporates both rules-based and statistical approaches for maximum effectiveness while circumventing the issues commonly associated with these methods.

2. Literature Review

The Voice-over-IP (VoIP) industry is quickly becoming the biggest player in the voice communication market, but as more businesses and users adopt VoIP, the number of attackers looking to disrupt or infiltrate these communication networks is only increasing. One of the greatest threats posed to the VoIP industry are Denial-of-Service (DoS) attacks. Rafique, Akbar, and Farooq [2] explain that this is due to vulnerabilities in the Session Initiation Protocol (SIP). They go on to detail what they describe as the "easiest" method of DoS: flooding. By flooding a SIP server with a large volume of requests, its internal resources (memory, CPU, and bandwidth) are consumed to the point where there are not enough resources left to provide the VoIP service to the regular users. Flooding can be mitigated through stateful SIP servers (as opposed to stateless) and message authentication techniques. While Rafique, Akbar, and Farooq provided an effective overview of flooding as a form of DoS attack, they did not discuss many of the other attack vectors commonly used during DoS attacks.

Sisalem, Kuthan, and Ehlert [3] take a more in-depth approach to their discussion of DoS attacks, going into detail about how different resources can be targeted depending on the type of requests the SIP server is attacked with. For example, to target the CPU, the attack must cause the server to perform a large volume of CPU-intensive tasks such as: verifying identity, executing applications, or communicating with external or non-existent servers. Alternatively, sending a large volume of session initiation requests consumes memory as the server attempts to initialise and run each instance, in what is described as a "brute force" attack. This description is misleading, however, and does not contain traditional brute-force elements, such as trial-and-error, usually associated with the term. A more accurate description of this type of attack would be flooding, which is outlined by Ormazabal et al. [4]. An attack is defined as flooding when it consists of "generating more packets than the recipient can handle, making it too busy processing packets to process legitimate packets". Ormazabal et al. also describe the two categories of flooding: signalling, which involves the use of SIP INVITE and REGISTER requests, or media floods, which involve spamming open ports with "meaningless and/or un-sequenced packets".

It is interesting to note the great vulnerability VoIP systems have to DoS attacks is by and large due to the fact they operate using UDP. UDP minimises packet delay or loss due to the lack of a handshake, yet it is this lack of inherent authentication that makes UDP, and therefore VoIP, more vulnerable to a DoS attack. Kai and Zhe [5] make the additional point that many firewalls close the UDP port for this reason. Ormazabal et al. [4] further backs up this idea, stating that "the fact that SIP runs over UDP, provides opportunities for attacks like spoofing, hijacking, and message tampering." While these papers fail to mention TCP is also susceptible to certain types of DoS, such as the TCP SYN flood, it is

hard to deny the role UDP plays in the vulnerability VoIP has towards DoS. The solution in this paper takes this fact into consideration, avoiding the risks of UDP by rerouting the packets through TCP.

2.1. DoS Countermeasures

Several countermeasures for DoS attacks targeting VoIP systems have been proposed, or even implemented, in recent literature; many of which may be used in conjunction with the design proposed in this paper. The following section evaluates and compares these measures in an aim to conclude the most effective solutions available.

While the majority of this discussion will be in the context of a SIP architecture, it is important to note that VoIP systems are supported on several architectural models. For example, Cauteruccio et al. [6] conduct an investigation into anomalies in Multiple IoT (MIoT) scenarios, putting forward a methodological framework to assist with future research as well as analysing problems associated with the current detection methods for MIoT systems. Despite being focused on MioT systems, elements of the framework proposed by Cauteruccio et al. are especially relevant and transferrable to other systems and architectures such as SIP, specifically the taxonomizing of anomalies into three definitions—presence/success, hard/soft, and contact/content—based on certain characteristics [6]. With this in mind, these categories can be considered when examining the possibility that certain anomalies have arisen as a result of a (D)DoS attack.

A paper by Nazih et al. [7] contains a comprehensive list of approaches to detecting and countering DoS attacks on SIP based networks, gathered from 28 articles published over the period of 2014 to 2020. The countermeasures have been sorted into four categories: finite state machine, rules-based, statistically based, and machine learning. Furthermore, each technique has been analysed and sorted into a table, along with the type of DoS mitigated, as well as the detection time and overall performance. This thorough approach to documenting each prevention method and citing its source article makes this an excellent resource from which research can be conducted. Additionally, the list highlights a significant gap in research, with the document and table therein lacking any mention of mitigation methods involving the rerouting of packets. This research paper aims to fill this gap by proposing a design that will circumvent DoS attacks on SIP based VoIP systems by rerouting the packets through TCP rather than UDP. The advantage of this system is that it can be used in conjunction with any of the previously proposed detection and filtration methods with little drawback.

Included in the survey conducted by Nazih et al. [7], Cadet and Fokum [8] propose their own mitigation system consisting of an Intrusion Protection System (IPS) using Snort [9]. This is a rules-based approach, using filtering based on an analysis of traffic conducted in previous research by Bansal and Pais [10]. For example, genuine users were shown to send on average 1–2 INVITE requests to start a call, so the threshold for dropping the packets is set to 4. The method proposed by Cadet and Fokum [8] is easy to implement and effective against common UDP floods, detecting and stopping an attack in an average time of 500 ms. Additionally, 20% of memory in the SIP server that would have been consumed when dealing with the attacks was saved. However, this approach is not effective when dealing with DoS attacks that are stealthier or utilise multiple devices to distribute the attack. Another flaw is the use of static filters rather than dynamic, reducing the scalability and adaptability of the software. The design proposed in this paper aims to avoid these flaws; firstly, by circumventing the need to detect the attack, and instead opt for a responsive approach, the stealth of the (D)DoS becomes irrelevant. Secondly, the use of dynamic protocol switching aims to create an adaptable solution that can be implemented in countless scenarios.

A rules-based approach, using a combination of a Handler and Bloom filter, is proposed by Ganesan and Msk [11]. This two-tier approach aims to mitigate both high-rate and low-rate attacks and incorporates a dynamic blacklist of malicious traffic in the Handler layer which is passed through a Bloom Filter to keep it continuously updated. This

method analyses several attributes of the packets at the flow level, ensuring that packets coming and going are correctly mapped to the users to mitigate spoofing of packet data, a technique commonly used by attackers. The results showed that while the detection time and false positives/negatives were improved compared to designs utilising either a Handler or Bloom Filter separately, the proposed method still had significant levels of false positives and negatives (1.6% and 4.13%, respectively) when dealing with fake signalling attacks. The "Call Me Maybe" framework aims to reduce the number of false positives and eliminate false negatives by detecting the attacks at a lower level. Instead of analysing packet information, the latencies and response times are used to determine whether the network is suffering an attack. This way, there is no chance of the framework being tricked by a compromised trusted client or advanced obfuscation techniques that could be used to hide packet information.

More recently, Ivy and Priya [12] suggest another trust-based detection and prevention mechanism that improves on the work of Cadet and Fokum [8]. Rather than blocking IP addresses that exceed a request threshold, a dynamic trust level is assigned to each client that changes based on the volume of requests over time. Initially, clients are assigned a baseline trust level which can be increased with continuous non-threatening interaction with the server or decreased if sudden high volumes of requests are detected. Using a dynamic detection system has many advantages. For example, if a high-trust user were to suddenly exhibit high volumes of requests for non-malicious reasons (such as technical issues) their already established trust level would prevent them from being blacklisted from the network. However, trust-based mechanisms proposed by Ganesan and Msk [11] and Ivy and Priya [12] would still leave the network vulnerable to DDoS attacks that utilised compromised hosts with an already established trust level, as it would take much longer for the server to register the threat. Despite this, the idea of a dynamic detection and prevention mechanism is critical to succeeding in defending a network from DoS attack, and for this reason has been heavily incorporated into the design proposed in this paper.

Tas, Unsalver, and Baktir [13] also use a dynamic solution but at a more holistic level, considering traffic patterns through the whole network over periods of time. These previous traffic patterns are compared to the live data being collected from the network to determine whether it is within a normal range. To counter expected differences in intensity (such as weekdays versus the weekend), traffic can be collected on an hourly, daily, weekly, or monthly basis [13]. By using traffic from the network as a whole, fewer resources are required to track individual clients' data, freeing up memory that the server can use elsewhere. Additionally, this method also eliminates the possibility of incorrectly identifying a legitimate client as malicious. Tas, Unsalver, and Baktir found that when using this method against a simulated SIP-based DoS attack, the CPU load was reduced from an average of 71% to 18% [13]. Tsiatsikes et al. [14] also use a statistical approach, based on entropy and log-files, to deal with DoS attacks on SIP servers. While the detection time is significantly longer than that of Cadet and Fokum [8], (1.8–16.8 s), this method implements in-depth offline analysis of audit trail data to identify DoS attacks. As well as this, extra consideration is given to the privacy of the users through the anonymisation and obfuscation of the parts of the log files being analysed.

Finally, Ahmad and Singh [15] propose a combination of a security-enhanced SIP proxy server and an enhanced application layer stateless firewall to help mitigate DoS attacks targeting VoIP systems. Their proposed solution aims to maintain a trusted list of user IP addresses through continuous communication between the SIP server and the firewall. Using this method, the firewall rules can be adjusted automatically based on the information received from the SIP server. Similarly, the 'Call Me Maybe' framework suggests the use of a UDP-restricted stateless firewall that is automatically activated based on the information received from a Control Centre monitoring latency in a network. Additionally, as the design proposed in this paper only incorporates the stateless firewall temporarily, a stateful firewall could also be in place on the network for regular use, reducing the risks associated with using a stateless firewall full-time—namely, lack of traffic and packet inspection.

Due to the success of the above methods, a holistic and dynamic approach to detection was chosen for the method in this paper. By taking the average response times of each device and comparing them to an established 'normal' threshold, a DoS attack can be detected and reactively countered with minimal impact on the VoIP traffic, and therefore minimal disruption of communications.

2.2. Literature Comparison Table

Table 1 displays a comparison between the designs in current literature and the design proposed in this paper.

Table 1. Table comparing designs proposed in recent literature with the 'Call me Maybe' framework.

Solution Description	Advantages	Disadvantages
Rules-based Snort Intrusion Protection System [8]	• Effectively stopped UDP flooding attacks in 500 ms. • Saved 20% of SIP server memory.	• Weak against distributed DoS attacks. • Weak against stealthier DoS attacks, such as when the attacker uses IP masking techniques. • Static filter thresholds.
Handler and Bloom filter [11]	• Uses a dynamic blacklist of malicious hosts. • Aims to mitigate both high-rate and low-rate DoS attacks. • Analyses packets at the flow level for greater detail.	• Significant volume of false positives and negatives. • Despite the dynamic blacklist, the solution is still vulnerable to distributed or stealthy DoS attacks if enough IP addresses are used.
Trust-based detection and prevention system [12]	• Uses a trust level for each host, improving the responsiveness of the attack and reducing false positives/negatives caused by day-to-day latency drops. • The algorithm dynamically updates based on real-time traffic analysis.	• While the use of host trust levels is effective, this method is especially vulnerable to an attack using compromised trusted hosts, as this will take much longer to detect due to the already established trust levels.
Statistical defence mechanism for a Novel DoS attack [13]	• Statistical approach, with the option of using data gathered on an hourly, daily, weekly, or monthly basis. • Shown to reduce CPU load significantly	• As the method is only tested on the novel 'SR-DRDoS' attack, its effectiveness against common forms of DoS attack is unknown.
Statistical entropy-based method [14]	• Deep off-line analysis of log files. • Anonymisation of users. • Easy to deploy and compatible with existing SIP installations.	• Longer detection time compared to other prevention mechanisms • High computation requirements
Enhanced SIP server and stateless firewall [15]	• Targeted towards and built for VoIP networks. • Trusted host list updated dynamically. • Effective against a variety of DoS attacks.	• Not backed up by experimentation. • Raises risks associated with using a stateless firewall full-time due to the lack of detailed traffic and packet inspection.
'Call me Maybe' Framework	• Effective at reducing the impact of UDP flooding attacks. • Dynamically changing protocols provide a quick response and automatic disablement when the DoS attack subsides. • Uses latency to detect DoS attacks, circumventing any stealth techniques that could be used, reducing false positives, and eliminating false negatives. • Can be used in conjunction with all the above solutions.	• As a responsive approach is used, the attack cannot be pre-empted. For this reason, it is recommended that the 'Call me Maybe' framework be used in combination with prevention systems. • Data quality expected to be temporarily reduced during the period the TCP network is activated

2.3. SIP-Based VoIP over TCP

When implementing VoIP communication in a network, the preferred choice of protocol for transmitting the voice data is usually UDP, whereas TCP is reserved for other mechanisms such as registration and initiation [16]. This is due to the performance advantage UDP has over TCP when transmitting a continuous stream of data, due to TCPs retransmission mechanism. There are, however, significant security flaws in UDP that make systems utilising this protocol more vulnerable to attacks, specifically UDP floods. As a result, in certain modern firewalls (known as UDP-restricted stateless firewalls) incoming UDP ports are closed [17]. This is not the case for similar TCP ports; as a result, several mechanisms for VoIP communication over TCP have been proposed in the past. This paper aim to circumvent the issues of UDP while overcoming any challenges that may arise when transmitting a continuous stream of data over TCP.

Yang, Lee and Ko [17] provide a detailed analysis of Voice over TCP (VoTCP) communication quality when used as a technique for traversing URSFs. According to the study, Skype's VoTCP has an average end-to-end delay of 30 ms, and an average jitter of 10 ms. This is in part due to the use of a "proprietary dynamic jitter buffer scheduling algorithm", which balances the quality of the communication with the rate at which the data is transmitted. The algorithm functions largely by monitoring and adjusting both the jitter and the delay of voice traffic. However, this mechanism (Figure 1) in use by Skype is not a "true" VoTCP connection, as TCP is only used between a Skype client (SC) and a sender node (SN) located outside the URSF. Traffic is converted back into UDP to be communicated across the internet. While the method proposed in this paper consists of a total conversion of data from UDP to TCP for the length of the DoS attack, this analysis by Yang, Lee and Ko [17] was a good introduction into this idea.

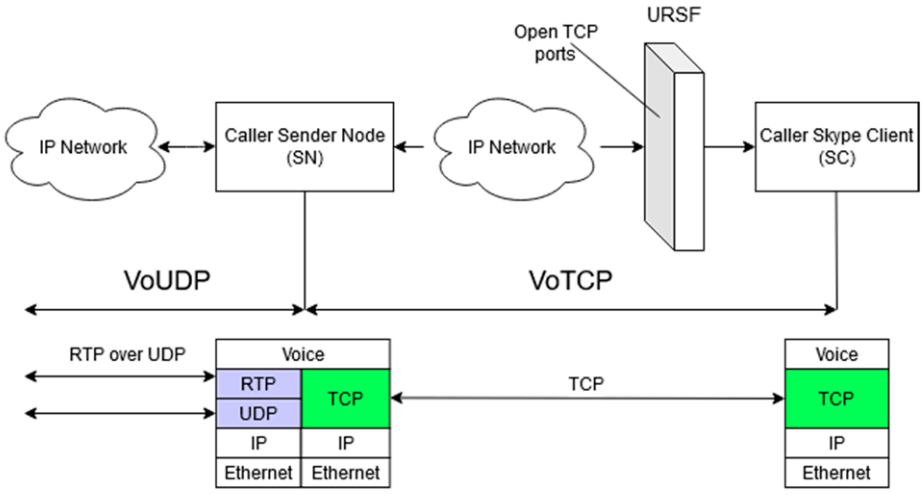

Figure 1. Diagram showing the VoTCP mechanism used by Skype. Recreation from Yang, Lee and Ko (2008).

A "true" VoTCP system is proposed by Satoda, Nihei, and Yoshida [18], using VoIP over multiple TCP connections (VoMTCP) to provide real-time voice communication while mitigating the disadvantages of TCP. As demonstrated in Figure 2b, in a regular TCP connection, if a packet is lost, the following packets on that connection are delayed until the original dropped packet is retransmitted. Satoda, Nihei and Yoshida [18] set out to improve on this inefficient process. In their design, the voice data is divided and distributed over multiple TCP connections, before being reconstructed at the receiving end. This mechanism aims to counteract the delay that would normally be present in real-time TCP communication. Using multiple TCP connections (Figure 3), if a packet is lost on one connection, the others are still able to transmit the remaining voice data packets with no delay. In combination with the use of packet loss concealment (PLC) to cover any packet data loss, the voice data can be reconstructed with minimal delay at the receiver

end. For these reasons, this method is very effective in the case of a high packet loss rate. However, when the connection is more stable, the overall quality decreases due to the natural degradation that occurs during the process of deconstructing and reconstructing packets. Therefore, as the number of connections is static, this design is only reliable if there is a consistent loss of packets. If a more adaptive approach were taken, such as dynamically adjusting the quantity of connections, the applicational potential of this solution would increase dramatically. Despite this, the design demonstrated the possibility of a "true" VoTCP system, able to maintain communication through URSFs with network conditions of "10% packet loss rate and 100 ms round-trip time".

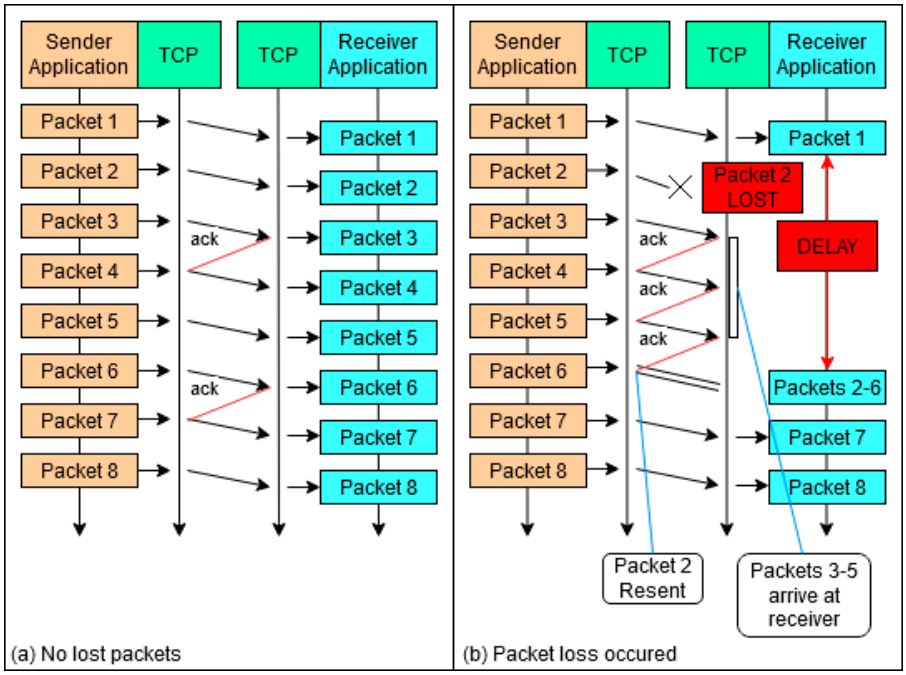

Figure 2. Diagram showing the retransmission mechanism used by TCP when a packet is lost or delayed. Recreation from Satoda, Nidei and Yoshida (2014). (**a**) No lost packets; (**b**) Packet loss occurred.

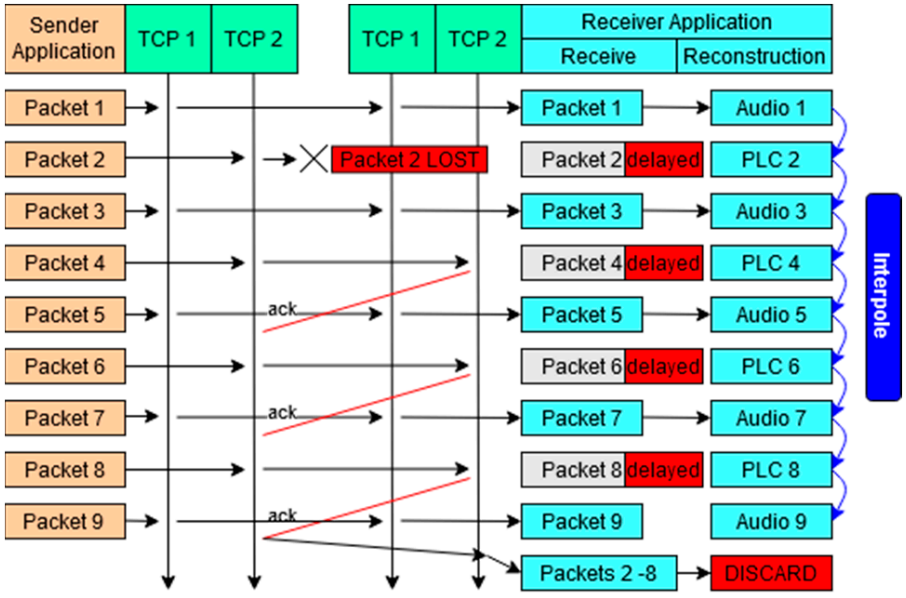

Figure 3. Diagram showing the application of multiple TCP streams to reduce packet loss and delay.

Kai and Ze [5] propose a "disorder TCP transmission strategy" to avoid the delay caused by TCP's retransmission mechanism following a dropped packet. Their strategy aims to counter the same issue as Satoda, Nihei, and Yoshida, pictured in Figure 2, while avoiding the setbacks that arise when using multiple connections. By adding the socket option "SO_UNORDERED", the application layer disregards the consecutiveness of the packets, instead reading data as it arrives. This more intuitive approach means lost packets are skipped over, avoiding the delay of retransmission. To identify the order of the packets, Kai and Ze [5] make use of a strategy that encodes the data of a packet with boundary identifiers. Firstly, the identifier byte "0x00 B" is inserted into both the start and the end of each complete packet before arriving at the application layer of the receiver. Here, the packets must be decoded to restore the complete original voice data. From the evaluation conducted by Kai and Ze, over 90% of the voice data transmitted using disorder-TCP could be heard normally up to a packet loss rate of 2%. This is significantly greater than that of basic TCP, which drops below 90% coherency at just 0.7% packet loss rate, and only slightly below that of UDP. However, the mechanisms involved in disorder-TCP (data coding and boundary identification) require a significantly larger portion of CPU power and processing time than both standard TCP and UDP.

When selecting a VoTCP mechanism for the design in this paper, the above techniques, and many others, were considered. While the design from Kai and Ze does have its advantages, Satoda, Nihei and Yoshida's VoMTCP system would be more appropriate due to the application for the design. As the VoTCP mechanism would only be activated when the network was currently suffering a DoS attack, it is expected that the rate of packet loss would be consistently high. As a result, the flaw of Satoda, Nihei and Yoshida's design (that it becomes less effective at low packet loss rates) becomes redundant.

3. Measuring the Effectiveness of Implementation

To measure the effectiveness of the implementation of the design proposed in this paper, a method for gathering results must be devised. This section evaluates how the effectiveness of the previously discussed designs were measured, as well as comparing the results themselves. This aims to provide the mechanisms and data that can be used to measure the effectiveness of the design proposed in this paper following its implementation.

Ormazabal et al. [4] proposes a two-layer firewall filter that covers both media and signalling traffic to mitigate DoS attacks by blocking malicious traffic. The implementation used a server with a Deep Packet Processing Module (DPPM) through which the filters are applied. The attacker machines were composed of 17 machines running Ubuntu Server OS, one of which would act as the "controller", capable of monitoring the data generated from the experiment, such as the Calls Per Second (CPS) and CPU load. The choice to use physical machines, rather than a simulation software meant the attack would work just as it would in a real-world scenario. However, this means measuring information from the network was more challenging, as the "controller" would have to be a device within the network itself. Additionally, this meant the scale of the attack was limited by the capacity of the hardware, and as a result, the full capabilities of the system could not be examined. For these reasons, as well as resource constraints, the design proposed in this paper instead opts for the simulation tool Cisco Packet Tracer combined with the Webex Teams API.

The system was tested using four types of traffic to perform the DoS attack: spoofed messages, floods of requests, responses, and out-of-state messages. By disabling digest authentication, a service required for the filters to work, the filtering system could be temporarily disabled, allowing a calibration of the baseline CPU load from legitimate traffic. Following this, the four types of attack were performed against the server individually at first, then all together in a composite attack. The results gained by Ormazabal et al. [4] demonstrated enabling the filtering mechanisms decreased CPU load by around 40% during each attack, even at high loads of traffic. Additionally, the filters were described to have had "zero false negatives, and negligible false positives (1%)" across all four types of attacks. Because of this, CPU load was a better indicator of the effectiveness of

the filter. Overall, however, the results table is quite shallow, lacking key information such as the packet loss rates during the attacks or the coherency of the transmitted voice data. This data would have been useful when comparing performance with other DoS mitigation strategies.

A more comprehensive set of results is gathered by Yu [19] in a study of over 8 million intrusion attempts into a company VoIP network, operating using an IP-PBX. Additionally, included in the paper is an evaluation of three methods (disconnection of IP-PBX from the internet, firewall protection, and application protection) and their effectiveness when implemented in the IP-PBX. It should be noted that malicious intrusion attempts were classed as any communication request that failed to authenticate, it is therefore possible that packets with legitimate reasons for failing the authentication (such as an incorrect password) are included in the dataset. The initial results consist of an in-depth analysis of data gathered from the syslog and tcpdump files. Yu [19] evaluates the individual operations that occur on the server as each malicious INVITE request is received. This level of detail would also be useful to investigate how the mitigation techniques operate, providing a more qualitative point of comparison. Yu [19] also performs quantitative analysis of the results, providing figures for the performance of the server under varying rates of packet flooding. The packet rate and CPU usage, as well as the average call setup time are recorded. Additionally, the system is tested under varying scales of DoS attack, until maximum CPU capacity is reached and the calls begin to timeout. This demonstration of performance is much more informative than the static rate of attack used by Ormazabal et al. [4]. However, Yu [19] only provides data on two types of attack compared to the four demonstrated by Ormazabal et al. Improving on these metrics, the method in this paper will be tested using variations in the scale and strength of the DoS attack. This will provide a comprehensive analysis of the effectiveness and limitations of the DoS mitigation strategy, from which comparisons can be drawn to other proposed strategies.

4. The "Call Me Maybe" Framework

The 'Call Me Maybe' framework is a design framework for VoIP networks that will provide the network with additional protection against DoS attacks. At its core, the framework is built around the addition of a 'shadow' TCP network that can be used as a fallback when the regular network is suffering a (D)DoS attack. Supporting this core idea, 4 principles make up the framework to ensure maximum effectiveness and reproducibility. The 'Call Me Maybe' framework principles are as follows:

I. The network should provide the capability for VoIP phones to operate over both TCP and UDP.
II. As part of the TCP network, a UDP-restricted firewall should be in place to aid in the mitigation of a UDP flooding attack.
III. The dynamic switch between TCP and UDP should be automatic and in response to latency rises caused by a denial-of-service attack.
IV. To enable dynamic switching between TCP and UDP, the network should have in place a form of latency monitoring to capture changes in response times.

Using the above principles, the 'Call Me Maybe' framework aims to provide an additional layer of security to VoIP networks that can be used in conjunction with any underlying protection mechanisms to mitigate distributed and non-distributed DoS attacks. While TCP and UDP VoIP networks were selected for this research, it is acknowledged that the design is not limited to alternating protocols, and can be used in other scenarios, for example, dynamically switching between VoIP and PSTN. Figure 4 displays a diagram of the processes involved in the framework.

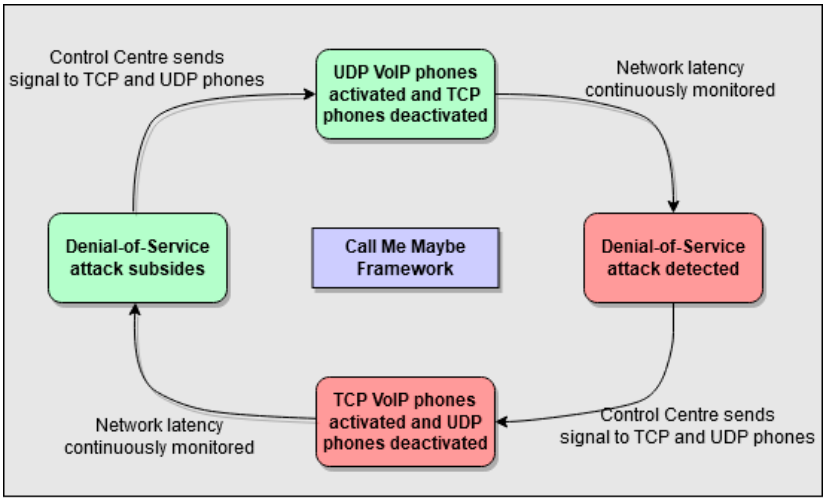

Figure 4. Diagram of the processes involved in the Call Me Maybe framework.

4.1. Principle I: TCP and UDP Capability in a Network

The core principle of the 'Call Me Maybe' framework is the use of a 'shadow' TCP network that exists with the UDP network, providing an alternative protocol and port through which communication can be carried out if the network is under attack. This TCP network can exist through a combination of software changes—i.e., updating the configuration of the VoIP phones and router to use the new protocol—and hardware changes, where entirely new network devices are used. The protocol change has several benefits, for example, by using TCP, network security features incompatible with UDP such as a UDP-restricted stateless firewall (URSF) can be enabled [17]. The URSF will mitigate the DoS attack, while the TCP VoIP network will allow voice traffic to continue for the duration the network is under attack. Ideally, the VoIP phones will be set up with the capability of transferring quickly to the TCP network when required. The speed of this is essential as any delay can result in lost voice packets.

4.2. Principle II: Use of a UDP-Restricted Firewall

The use of VoIP over TCP enables the unique opportunity for VoIP networks to temporarily function in environments where UDP ports are closed, allowing them to take advantage of the strength of TCP against DoS attacks that it derives from the 3-way handshake used for authentication [5]. Once the network has transferred over to TCP, the firewall may then be used to block all incoming UDP ports to act as an additional layer of security against the DoS attack. With the UDP ports blocked, the impact of attacks such as UDP flooding on a network are dramatically reduced. Once again, this can be performed through ad hoc configuration changes, hardware changes, or a combination of the two.

4.3. Principle III: Dynamic and Automatic Protocol Switching

For the proposed framework to be truly effective in reducing the impact of a (D)DoS attack, it is essential that it is able to respond to attacks on-demand and as quickly as possible. For this reason, it is a key principle of the design that the protocol switching should occur automatically and as soon as the attack is confirmed. This can be accomplished in a variety of ways, but for the design adopted in this paper a Control Centre (CC) was implemented that would send out a 'switch' signal to the VoIP phones, instructing them to alternate between UDP and TCP depending on the detection of a DoS attack. For this dynamic switching to function efficiently, it is essential that reliable metrics are used to calculate when to switch the protocol over to the TCP network.

4.4. Principle IV: Latency Monitoring

In order to provide real-time protection to a network and facilitate the dynamic protocol switching described in Principle III, a form of detecting DoS attacks as they occur must also be implemented. In the case of the "Call Me Maybe" framework, this function is built into the CC described above, allowing it to monitor the latency of voice traffic within a network, while also dynamically switching the VoIP phones between the two protocols. By detecting the attacks through latency monitoring, a responsive, rather than preventative, approach is adopted. By using a responsive approach, not only are any chances of false negatives eliminated, but the framework is also put in a position to function as an underlying form of defence for any VoIP network and is able to be used in conjunction with any established preventative solutions. This system therefore works to create an adaptable and scalable solution, that will reduce the impact of DoS attacks by reflexively switching the network being used by the VoIP phones to an alternative protocol, in this case TCP.

5. Experimentation

This chapter discusses how a solution in line with the "Call Me Maybe" framework was implemented and evaluated using Cisco Packet Tracer and Webex. The discussion includes the overall network topology, IP addressing and routing, as well as explanation of the Python code used on the devices. Additionally, included is demonstration of the working system using Webex Teams as a command line interface to initiate communication between selected VoIP phones. Each section contains screenshots with additional information.

For the results of the experiment to be as legitimate as possible, the network (Figure 5) was designed in a way that is realistic while remaining relatively simplistic. The reason for this is that during testing excessive packet traffic often caused CPT to crash or add additional latency to the network that would not have occurred in a real environment. The design consists of two areas: Site A and Site B, both containing six traditional UDP VoIP phones and six VoIP phones modified to use TCP, implemented using the 'Thing' component in CPT and programmed using Python.

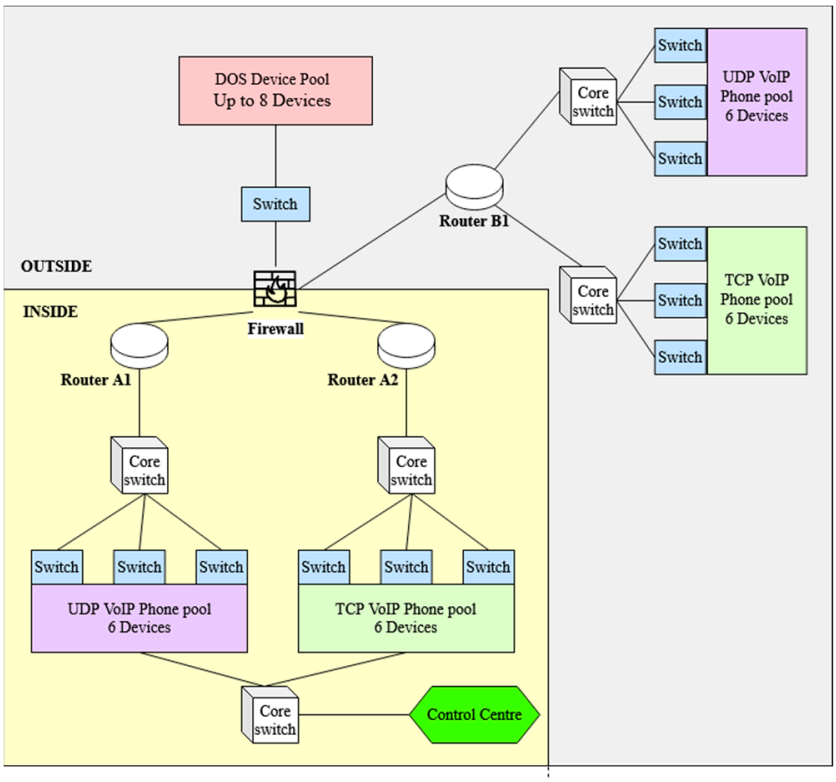

Figure 5. Topology of the Cisco Packet Tracer network.

For the purpose of the experiment, the phones in Site A act as the transmitters and operate through one of two gateway routers depending on the protocol in use (UDP or TCP) to communicate a message to the receiver phone on Site B. The UDP phones use Router-A1 as a gateway, while the TCP phones use Router-A2. This is done to allow the firewall to be set up as a UDP-restricted stateless firewall (URSF) on Router-A2, while also acting as a standard firewall for Router-A1. As a result, the firewall allows both UDP and TCP traffic into the network through Router-A1, while blocking incoming UDP traffic heading to Router-A2. This demonstrates the capabilities using VoIP-over-TCP provides to the network, as using a URSF on Router A1 would prevent the UDP voice traffic from travelling back from Site B to Site A.

The DoS attack being carried out on the network is a distributed UDP flooding attack using eight 'Things' programmed to cycle through a range of ports, continuously spamming both routers with messages. The attack causes latency on the network, as with every new port on which a message is received the routers are required to check the port to see if it is open and return an ICMP message refusing the connection.

Finally, the design also contains a 'Control Centre', which is used to detect whether a DoS attack is currently occurring on the network. If an attack is detected, the Control Centre notifies both sets of VoIP phones on Site A in order to switch the transmissions over to the TCP phones. The Control Centre is also responsible for gathering the response times and packet loss rates of the VoIP communications to be used in the data analysis. Additionally, recorded are the times at which the DoS attacks were detected, which are then compared to the actual initiation times of the attacks to judge the effectiveness of the detection system.

The IP addressing follows a basic scheme, outlined in Table 2. Site B is included in the 'outside' VLAN to evaluate the effectiveness of how VoIP communications between the 'inside' and 'outside' networks are handled alongside malicious DoS traffic also originating from the 'outside' network.

Table 2. IP addressing scheme used in the network design.

Host	IP Address(es)	Mask	Gateway
VOIP-A11 to A16	192.168.1.11–192.168.1.16	/24	192.168.1.1
VOIP-A111 to A116	192.168.1.111–192.168.1.116	/24	192.168.1.2
VOIP-B21 to B26	192.168.2.21–192.168.2.26	/24	192.168.2.1
VOIP-B221 to B226	192.168.2.221–192.168.2.226	/24	192.168.2.1
RTR-A1	192.168.1.1 (in) 10.0.0.2 (out)	/24 /16	-
RTR-A2	192.168.1.2 (in) 10.0.0.3 (out)	/24 /16	-
RTR-B1	192.168.2.2 (in) 209.165.200.226 (out)	/24 /24	-
FIREWALL	10.0.0.1 (in) 209.165.200.225 (out)	/16 /24	-
DOS-1 to 8	209.165.200.240 to 209.165.200.247	/24	209.165.200.225
Control Centre	1.0.0.1	/24	-

Cisco Webex is a meeting/messaging application with an easily accessible API that enables the chatrooms to be used for communication between external programs. Using Cisco Webex version 41.4, several 'rooms' were created to act as a command interface for the VoIP phones, which used HTTP GET and POST requests to read and write content to the chatrooms. Five rooms were created for the purpose of this solution: input and output rooms for Site A and Site B, as well as an additional room for gathering data. The input rooms (shown in Figure 6) were used to 'dial' a number and initiate a connection from one site to the other using an ID unique to each phone (e.g., VOIP-B221 would be dialled using the ID '221'). Following this connection, the voice traffic would begin to travel to and from each site, appearing in the 'output' chatroom (Figure 7) that corresponds with the phone receiving the message. Finally, a data collection room was also present (Figure 8), containing records of all communications between VoIP phones on the network. The timestamps of the messages in the data collection room were used as the main source of data to measure the effectiveness of the solution, which is discussed in Section 5.

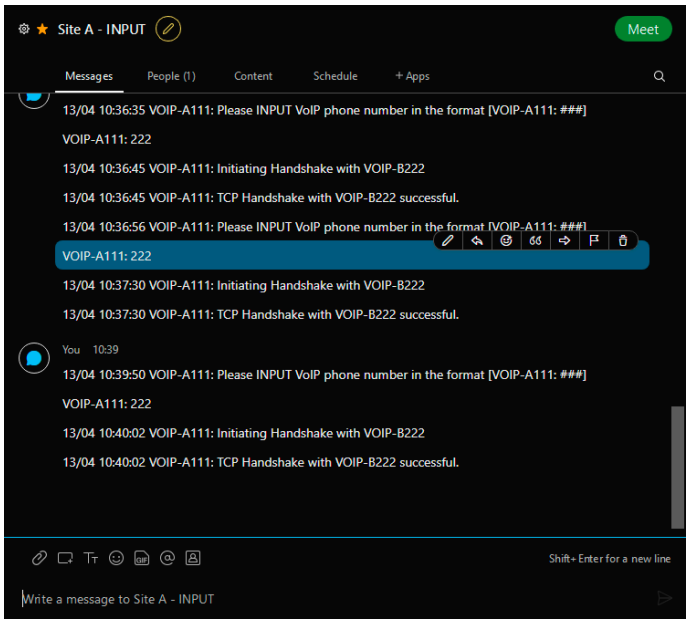

Figure 6. Cisco Webex input room.

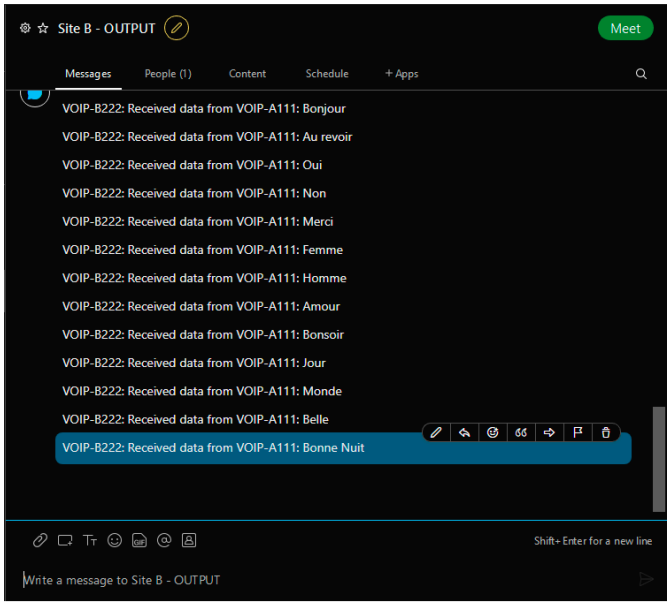

Figure 7. Cisco Webex output room.

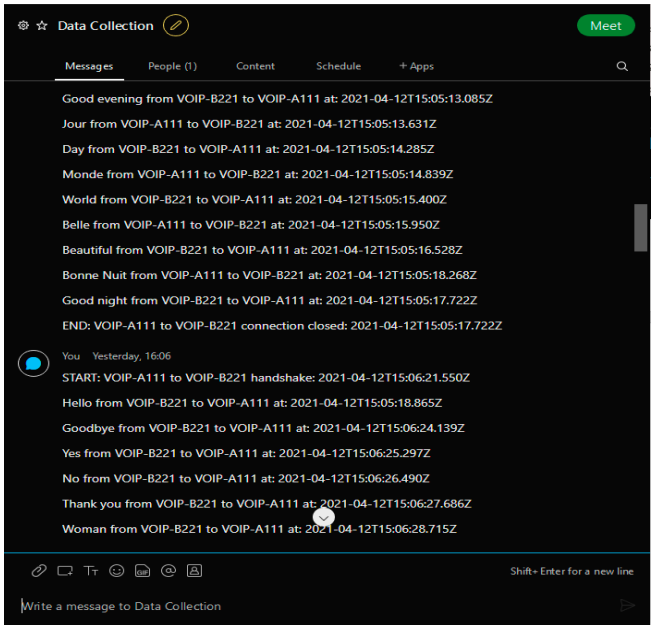

Figure 8. Cisco Webex data collection room.

The VoIP phones are composed of a Python code file, executed within CPT using the 'Thing' component. The main function of the phones is to initiate a connection from a Site A phone to a Site B phone to transmit voice data. For the purpose of this experiment, the voice data being used is a list of thirteen French words, which, upon being received by a Site B phone, are translated into English and transmitted back to Site A. The phones are controlled through Cisco Webex using a command in the following formats: 'VOIP-AXX: YY' for UDP phones, and 'VOIP-AXXX: YYY' for TCP phones. Additionally, during each stage of communication, the VoIP phones record the timestamps of communication in the Webex 'Data Collection' room.

When a connection is first initiated using a Webex command, the first step is to perform a TCP handshake on port 5060, simulating a SIP handshake [20]. This handshake is carried out to ensure the destination phone is receptive and trigger the transmission of voice data by changing the 'state' of the phone to '1'. Following a handshake, the date and time of the corresponding Webex messages are recorded to the Webex 'Data Collection' room as part of the timeline of events.

Once the handshake is complete, the simulated voice traffic is sent from the broadcasting phone to the receiving phone. The traditional VoIP phones use UDP port 2000 to transmit the data, using ROUTER-A1 as the gateway, while the modified phones use TCP port 2001 and ROUTER-A2 as the gateway. Using a separate port and router for the set of TCP phones allows them to be dynamically switched to if the network is suffering a DoS attack, with minimal interference from the side of the network containing the UDP phones. Both the UDP and TCP phones are programmed to transmit 'voice' data in similar ways. The phones on Site A start with a list of thirteen French words to transmit to the Site B phones via their respective protocol. The phones on Site B contain dictionaries with both the English and French words and are used to translate and return the word to the Site A phones. Both sets of phones use an in-built delay of 0.5 s between receiving and sending a message to give a more realistic simulation of voice data while also giving the Webex chatrooms time to catch up between messages. Due to constraints in CPT, actual audio data was unable to be used, however, based on the timestamps and quality of the simulated data, a recreation of what each call may have sounded like was able to be constructed.

Once all thirteen words have been transmitted between both phones, the VoIP connection closes, and the date and time are recorded in the 'Data Collection' room in Webex. Recording the start and end time of each call enables efficient and easy gathering of data

using the Control Centre, which can then be used to measure the effectiveness of the solution proposed in this paper. As well as the set of UDP and TCP VoIP phones, Site A also contains a 'Control Centre' responsible for activating the TCP phones when a DoS attack is detected. The Control Centre (CC) is also used to measure the latency and packet loss of the traffic communicated between the phones on both sites, using the data gathered in the 'Data Collection' Webex room.

The CC continuously calculates the latency using the timestamps of the Webex messages output when a phone receives a packet of data, as these are accurate to the millisecond. For every set of data with response times above a set threshold (in this case 0.7), the CC increases the alert level by one. If the alert level reaches a certain value (in this case 5), the network is considered to be under attack, and the UDP VoIP phones are temporarily disabled while the TCP phones are activated. The CC communicates with the TCP and UDP VoIP phones over UDP port 1000 (Note that the protocol and port used here do not affect, nor are affected by, the rest of the experiment as the instructions go directly to the VoIP phones, rather than through one of the gateway routers). The CC continues to monitor the latency of the TCP communication, and once the alert level drops below the set threshold, the option to return to the UDP phones becomes available. If the DoS attack is believed to have ended, the UDP phones can be safely returned to. The value '0.7 s' was selected as the threshold response time to allow for some leeway from times when the network is naturally congested on top of the in-built 0.5 s delay the VoIP phones have between receiving and sending a message.

The CC is also used as the main data collection mechanism for the experiment. The CC stores the response times of each call in a list, of which the values are then transferred to a CSV file. The data collection was incorporated into the CC to give more realistic results in the experiment, as a real-life network would gather information in a similar way. The devices used to carry out the UDP flood DoS attack are also CPT 'Things' programmed in Python. Webex was unable to be integrated into these devices as the high rates of packet traffic in conjunction with reading and writing to chatrooms caused CPT to crash, compromising the validity of any results that could have been gathered. Multiple DoS 'Things' were included in the design to test the methodology against a range of scales of attack, from one attacker up to eight. Each device works by selecting a UDP port in a set range, sending a 'spam' message to both ROUTER-A1 and ROUTER-A2, incrementing the port number by one, and repeating the message. Each device uses a different range of port numbers in order to put as much stress on the two routers as possible; the first device uses a port range of 1–500, the second uses 501–1000, the third uses 1001–1500, etc.

6. Results, Analysis, and Discussion

The data gathered was generated by running the simulation a total of 1080 times, using variations in the VoIP phones used, methodology, and strength of the DoS attack. The scale of the DoS attack was controlled using variations in the number of machines used to attack the network, from zero devices (control) up to a maximum of eight. The independent *t*-Test was selected to compare the two independent samples (UDP phones and TCP phones) at each strength of DoS attack. Initially the *t*-Test is carried out on the control experiment to ensure there is no inherent variation in the values produced when using the two methodologies. This section also contains discussion and analysis of the results, which are used to evaluate the effectiveness of the implemented solution.

Table 3 shows the results from a *t*-Test when comparing the response times gathered in the control experiment. The *p* values for the control experiment are both greater than 0.05, and as a result it can be concluded that there is no statistically significant difference between the mean response times for the UDP VoIP phones and the TCP VoIP phones when the network is not experiencing a DoS attack. Therefore, the statistical tests can be reliably carried out on the remaining results. It should also be noted that the quantities of packets lost were measured in the control experiment, however no voice packets were found to have failed their transmission between the two phones.

Table 3. Results from an independent sample *t*-Test.

	UDP Response Time (s)	TCP Response Time (s)
Mean	0.588236668	0.586134868
Variance	0.006270646	0.002491745
Observations	60	60
Hypothesized Mean Difference	0	-
df	99	-
t Stat	0.173922477	-
P (T <= t) one-tail	0.431140767	-
t Critical one-tail	1.660391156	-

When examining the results, initially using the proposed methodology does not appear to provide a significant advantage to the network. For the experiment using one DoS device (Table 4), the means for the UDP phones and the TCP phones are 0.7171 s and 0.7078 s, respectively. The p value is 0.32, and as this is greater than 0.05, the response times for the TCP phones are not significantly different to those for the UDP phones. Therefore, the null hypothesis must be accepted, and the alternative hypothesis rejected. Both the mean response time for the TCP and UDP phones is increased compared to the control experiment, indicating that the DoS attack is having an effect even with the methodology in place. This is possibly due to the structure of the network; one firewall is responsible for both networks, and as a result may be becoming congested when handling larger volumes of traffic. While there is a minor improvement in response time (−1.3%) when using the TCP phones, it is not statistically significant at this scale of attack.

Table 4. Results, Experiment 1.

	UDP Response Time (s)	TCP Response Time (s)
Mean	0.717095047	0.707801941
Variance	0.018879195	0.005781404
Observations	60	60
Hypothesized Mean Difference	0	-
df	92	-
t Stat	0.458389553	-
P (T <= t) one-tail	0.323876787	-
t Critical one-tail	1.661585397	-

However, as can be seen in Tables 5 and 6, from Experiment 2 onwards the p value is less than 0.05, suggesting the results are statistically significant. Additionally, this trend suggests that the proposed solution is far more effective when a network is facing a greater strength of attack. Based on this data it is therefore suggested that the threshold for activating the TCP phones should be increased to beyond the value of 0.7, used in the experiments, to around 0.9, where the benefit is greatly increased. Figure 9 displays a graph of the mean response times of the two types of phones during each strength of DoS attack.

Table 5. Results, Experiment 2.

	UDP Response Time (s)	TCP Response Time (s)
Mean	0.872771008	0.833576175
Variance	0.015352186	0.012659159
Observations	60	60
Hypothesized Mean Difference	0	-
df	117	-
t Stat	1.814000748	-
P (T <= t) one-tail	0.036120153	-
t Critical one-tail	1.657981659	-

Table 6. Results, Experiment 3.

	UDP Response Time (s)	TCP Response Time (s)
Mean	0.940874563	0.870644687
Variance	0.019226699	0.004839068
Observations	60	60
Hypothesized Mean Difference	0	-
df	87	-
t Stat	3.506692348	-
P (T <= t) one-tail	0.000360028	-
t Critical one-tail	1.662557349	-

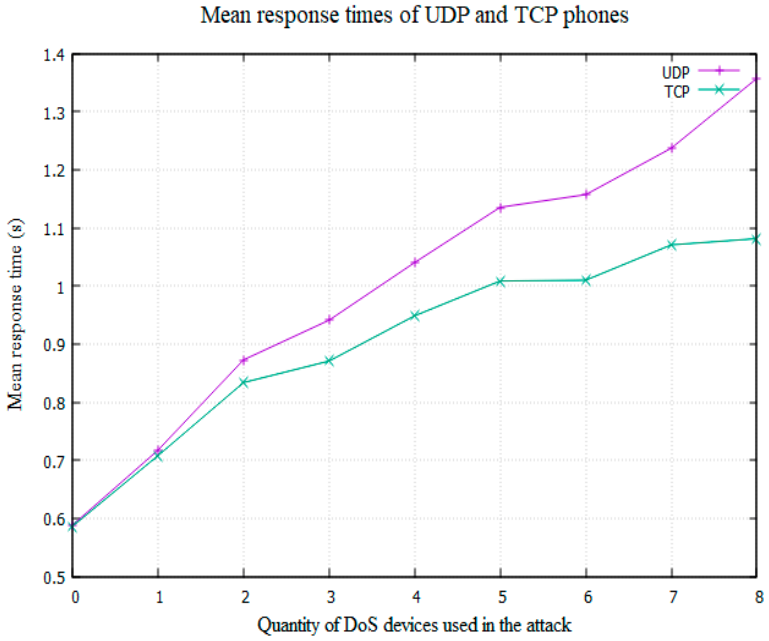

Figure 9. Graph displaying the mean response times of the UDP and TCP phones.

Tables 7 and 8 display the outcome when the independent *t*-Test was carried out on the response times when a network was experiencing a DDoS attack from four and five devices, respectively.

Table 7. Results, Experiment 4.

	UDP Response Time (s)	TCP Response Time (s)
Mean	1.040081206	0.948375321
Variance	0.037276289	0.011119926
Observations	60	60
Hypothesized Mean Difference	0	-
df	91	-
t Stat	3.228993223	-
P (T <= t) one-tail	0.000864284	-
t Critical one-tail	1.661771155	-

Table 8. Results, Experiment 5.

	UDP Response Time (s)	TCP Response Time (s)
Mean	1.135312606	1.008461956
Variance	0.017175358	0.024002806
Observations	60	60
Hypothesized Mean Difference	0	-
df	115	-
t Stat	4.842112008	-
P (T <= t) one-tail	2.01876×10^{-6}	-
t Critical one-tail	1.65821183	-

From Experiment 6 to Experiment 8 (Tables 9–11), the mean response time for the TCP phones appears to begin to level off at around 1.01, while the response time for the UDP phones continues to increase. This results in an increasing disparity between the two sets of data and is again evidence that the "Call Me Maybe" framework is most effective when a network is suffering a strong DDoS attack. The experiment must be conducted against an even greater strength of DoS attack to see if this trend continues.

Table 9. Results, Experiment 6.

	UDP Response Time (s)	TCP Response Time (s)
Mean	1.156956044	1.009764763
Variance	0.073380824	0.018630804
Observations	60	60
Hypothesized Mean Difference	0	-
df	87	-
t Stat	3.758688662	-
P (T <= t) one-tail	0.000154308	-
t Critical one-tail	1.662557349	-

Table 10. Result, Experiment 7.

	UDP Response Time (s)	**TCP Response Time (s)**
Mean	1.236800923	1.070474602
Variance	0.068140906	0.029979311
Observations	60	60
Hypothesized Mean Difference	0	-
df	102	-
t Stat	4.112987187	-
P (T <= t) one-tail	3.96196×10^{-5}	-
t Critical one-tail	1.659929976	-

Table 11. Result, Experiment 8.

	UDP Response Time (s)	**TCP Response Time (s)**
Mean	1.356986179	1.081433178
Variance	0.033293907	0.034034441
Observations	60	60
Hypothesized Mean Difference	0	-
df	118	-
t Stat	8.225869094	-
P (T <= t) one-tail	1.45582×10^{-13}	-
t Critical one-tail	1.657869522	-

The quantities of packets lost were also measured during the experiments, however, in most cases zero packets were found to have been dropped during the communication. The packet loss rate was only found to increase when the network was experiencing a very high load (see Table 12).

Table 12. Table displaying the total quantity of packets dropped across all the experiments run at each strength of DoS attack.

No. of DoS Devices	**Packets Dropped (UDP)**	**Packets Dropped (TCP)**
0 (Control)	0	0
1	0	0
2	0	0
3	1	0
4	1	0
5	2	0
6	1	2
7	3	10
8	5	19

The speed at which the Control Centre detects a DoS attack and activates the TCP phones was also recorded. This is of course influenced by the alert and latency thresholds, which were set to '5' and '0.7', respectively, meaning a DoS attack would be detected after five calls during which the average response time was greater than 0.7. Figure 10 displays a graph showing the average time taken to detect a DoS attack at each strength of attack. The average detection time increases in correlation with the strength of the attack. This is likely due to the mechanism of action, which detects an attack by measuring the response times of a set number of completed calls. Therefore, the longer a call takes to complete, the longer it will take to detect the attack. To combat this, improvements should be made to the detection mechanism that will increase the rate of detection when the response times are high. For example, when the average response time of a phone call is above a certain threshold (e.g., '1 second') the alert level could be increased by two rather than one, reducing the number of completed calls required to trigger the activation of the TCP phones.

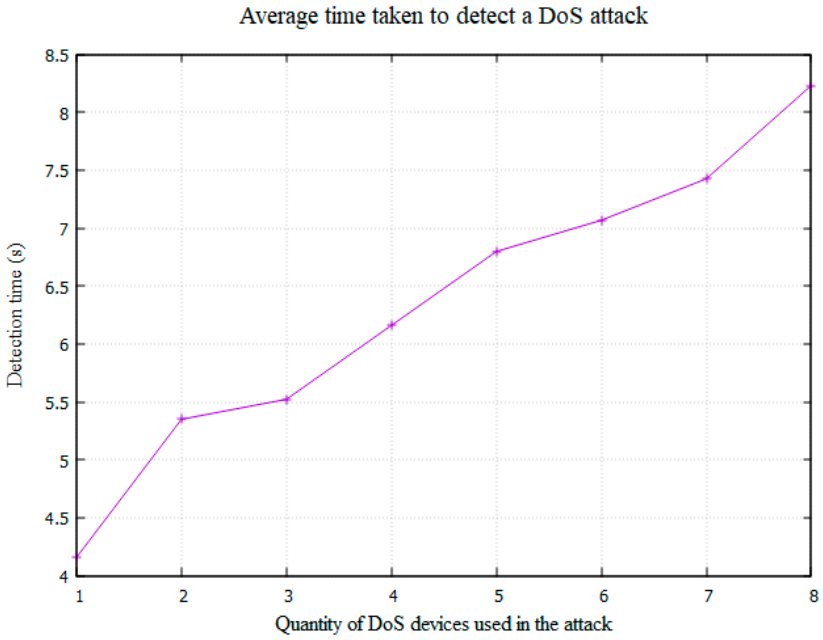

Figure 10. Average time taken to detect the DoS attack at each strength of attack.

7. Conclusions

As defined by Nazih et al. [7], there are four approaches to mitigating UDP DoS attacks: finite state machine (FSM), rules-based, statistically based, and machine learning based. The "Call Me Maybe" framework falls under the rules-based category, as it switches on/off depending on a set threshold, but also uses statistical elements as the threshold value is dynamically updated based on previously gathered network data. This addresses issues common in other proposed rules-based approaches, which can often be rigid and easy to circumvent if the rules are not updated according to changes in the network. The combination of rules-based and statistics-based approaches ensures the "Call Me Maybe" framework is scalable to a wide range of attacks and can be adapted across a variety of network designs. This improves upon alternative approaches, such as machine learning, which often have significant overhead costs and consume large amounts of processing power to build the prediction models. Alternatively, finite state machine approaches, while effective, are often complex and require rigorous designs and therefore would not be suitable for the same wide-scale applications as the proposed framework.

For a rules-based detection system, the 'Call me Maybe' framework demonstrates a competitive detection time even under the strongest attack, where a detection time of 8.23 s was recorded. This is an 8.6% reduction when compared to the work of Ganesan and Msk [11], who use a rules-based 'Handler and Bloom filter' based on packet information to detect DoS flooding attacks in an average time of 9 s when using their most effective

design (when 6 packet attributes are incorporated into the algorithm). Additionally, due to the more general method of gathering the response times across all voice communications the 'Call me Maybe' framework can be applied to defend against all variations of DoS and DDoS, compared to other works which are often restricted to a specific type of flooding attack, such as with Cadet and Fokum [8], whose paper proposes a method specific to INVITE, REGISTER, and BYE floods.

The "Call Me Maybe" framework is intended as a final barrier of defence against UDP (D)DoS attacks due to its latency-monitoring approach. This means that alternative, higher-level, mitigation techniques may be implemented in combination with the framework without any risk of interference. For example, a first line of defence such as Ivy and Priya's [12] trust-based model blocks suspicious clients and allows trusted users. However, this can be bypassed if a trusted user suddenly starts emitting malicious traffic. Without the "Call Me Maybe" framework in place at an underlying level to react to the detected drop in latency, the network would be left vulnerable to the impact of (D)DoS attacks. Additionally, due to the use of captured latency times as the measure by which the defence mechanism (TCP network) is triggered, any chance of the framework not activating when the network is suffering an attack is eliminated; when high latency is detected for a set period of time, the "Call Me Maybe" framework is guaranteed to come into effect.

7.1. Limitations

It should be noted that while implementing the proposed methodology did appear to result in a statistically significant reduction in the response time of voice traffic, the latency remained significantly higher than that in the control experiment. As a result, it must be acknowledged that the solution is partially effective, not providing the full protection originally desired. This is possibly due to the design of the network used in the experiment, which involved a single firewall acting as the gateway between the 'inside' network and the 'outside' network, containing the DoS devices and external site. As both the malicious packets and legitimate traffic were required to pass through this firewall, it is possible that this caused additional delay in the network. To improve the design, two separate firewalls could be used to ensure there is no cross-over between the UDP and TCP VoIP networks.

Finally, as a variation in packets lost was only able to be sufficiently observed in experiments 7 and 8, a reliable conclusion is unable to be made from the data. However, the fact that the packet loss is significantly higher in the calls made using the TCP phones is something that should be examined further. It is possible that this is due to the retransmission mechanism TCP uses, discussed in Section 2.3, in which one failed transmission causes a chain of packets to be delayed or lost. To improve the design, multiple TCP connections could be used to retransmit lost packets in a way similar to that suggested by Satoda, Nihei and Yoshida [18]. However, to perform a more conclusive evaluation, more results are required, and a similar experiment using greater scales of DoS attack should be carried out.

7.2. Contributions

Overall, this research paper has proposed and evaluated the "Call Me Maybe" security framework for VoIP networks. The design of the framework was based on advancements of ideas presented in recent literature, with the goal of providing an adaptable protection mechanism against DoS attacks targeting VoIP networks. The design consisted of three main features: integration of an alternative network in which the VoIP devices operate using TCP, a method of detecting the presence of a DoS attack on the network, and a method of dynamically alternating between the two protocols. To test the efficacy of the proposed framework, Cisco Packet Tracer was used to implement a network that utilised the "Call Me Maybe" security framework. The Cisco Packet Tracer design used in this work is available in the Supplementary Materials. The response time and packet loss rates were tested when using the standard VoIP protocol (UDP), in comparison to the proposed design using VoIP over TCP. Examining the results using the independent t-Test, it was concluded that the proposed design (VoIP over TCP) improves the response times to a statistically

significant degree when a network is experiencing a (D)DoS attack from between two and eight devices. The results also showed the response time for the TCP phones began to level off when around six DoS devices were conducting an attack. As the solution was only tested to a maximum of eight devices, further research must be done to evaluate the efficacy of the design at greater scales of DoS attack and to see if this trend continues. The detection times were also analysed, showing that as the strength of DoS attack increased, so did the time it took to detect the attack.

Supplementary Materials: The following supporting information can be downloaded at: https://www.mdpi.com/article/10.3390/network2040032/s1, Packet Tracer file used in the experiments.

Author Contributions: Conceptualization, J.K. and T.V.; methodology, J.K. and T.V.; software, J.K.; validation, J.K.; formal analysis, J.K.; investigation, J.K.; resources, J.K. and T.V.; data curation, J.K.; writing—original draft preparation, J.K.; writing—review and editing, J.K. and T.V.; supervision, T.V.; project administration, T.V. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Data Availability Statement: Data available on request.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Godlovitch, I.; Kroon, P. *Copper Switch-off: European Experience and Practical Considerations (No. WIK-Consult White Paper)*; WIK-Consult GmbH: Bad Honnef, Germany, 2020.
2. Rafique, M.Z.; Akbar, M.A.; Farooq, M. Evaluating DoS Attacks against Sip-Based VoIP Systems. In Proceedings of the GLOBECOM 2009—2009 IEEE Global Telecommunications Conference, Honolulu, HI, USA, 30 November–4 December 2009; pp. 1–6.
3. Sisalem, D.; Kuthan, J.; Ehler, S. Denial of service attacks targeting a SIP VoIP infrastructure: Attack scenarios and prevention mechanisms. *IEEE Netw.* **2006**, *20*, 26–31. [CrossRef]
4. Ormazabal, G.; Sarvesh, N.; Eilon, Y.; Henning, S. Secure sip: A scalable prevention mechanism for dos attacks on sip based voip systems. In Proceedings of the International Conference on Principles, Systems and Applications of IP Telecommunications, Berlin/Heidelberg, Germany, 1–2 July 2008; pp. 107–132.
5. Kai, S.; Guan, Z.; Meng, C. VoIP transmission mechanism based on TCP. *J. China Univ. Posts Telecommun.* **2016**, *23*, 90–96. [CrossRef]
6. Cauteruccio, F.; Cinelli, L.; Corradini, E.; Terracina, G.; Ursino, D.; Virgili, L.; Savaglio, C.; Liotta, A.; Fortino, G. A framework for anomaly detection and classification in Multiple IoT scenarios. *Future Gener. Comput. Syst.* **2021**, *114*, 322–335. [CrossRef]
7. Nazih, W.; Elkilani, W.; Dhahri, H.; Abdelkader, T. Survey of Countering DoS/DDoS Attacks on SIP Based VoIP. *Networks. Electron.* **2020**, *9*, 1827. [CrossRef]
8. Cadet, F.; Fokum, T. Coping with denial-of-service attacks on the IP telephony system. In Proceedings of the SoutheastCon 2016, Norfolk, VA, USA, 30 March–3 April 2016; pp. 1–7.
9. Roesch, M. Snort: Lightweight Intrusion Detection for Networks. In Proceedings of the 13th USENIX Conference on System Administration, Seattle, DC, USA, 7–12 November 1999; pp. 229–238.
10. Bansal, A.; Pais, A. Mitigation of Flooding Based Denial of Service Attack against Session Initiation Protocol Based VoIP System. In Proceedings of the 2015 IEEE International Conference on Computational Intelligence Communication Technology, Ghaziabad, India, 13–14 February 2015; pp. 391–396.
11. Ganesan, V.; Msk, M. A scalable detection and prevention scheme for voice over internet protocol (VoIP) signaling attacks using handler with Bloom filter. *Int. J. Netw. Manag.* **2018**, *28*, 1995. [CrossRef]
12. Ivy, B.P.U.; Priya, M.A. Detection and Prevention of Distributed Denial of Service Attacks in VoIP. *Taga J. Graphic Technol.* **2018**, *14*, 1985–2000.
13. Tas, I.M.; Unsalver, B.G.; Baktir, S. A Novel SIP Based Distributed Reflection Denial-of-Service Attack and an Effective Defense Mechanism. *IEEE Access* **2020**, *8*, 112574–112584. [CrossRef]
14. Tsiatsikas, Z.; Dimitris, G.; Georgios, K.; Angelos, D. An efficient and easily deployable method for dealing with DoS in SIP services. *Comput. Commun.* **2015**, *57*, 50–63. [CrossRef]
15. Goode, B. Voice over Internet protocol (VoIP). *Proc. IEEE* **2002**, *90*, 1495–1517. [CrossRef]
16. Ahmad, W.; Singh, D. VoIP security: A model proposed to mitigate DDoS attacks on SIP based VoIP network. In *A Multi-Disciplinary Research Book*; Research for Resurgence: Nagpur, India, 2018; Volume 1, pp. 37–48.
17. Hae-Yong, Y.; Kyung-Hoon, L.; Sung-Jea, K. Communication quality of voice over TCP used for firewall traversal. In Proceedings of the 2008 IEEE International Conference on Multimedia and Expo, Hannover, Germany, 23 June–26 April 2008; pp. 29–32.

18. Satoda, K.; Nihei, K.; Yoshida, H. Quality evaluation of voice over multiple TCP connections. In Proceedings of the 2014 International Conference on Computing, Networking and Communications (ICNC), Honolulu, HI, USA, 3–6 February 2014; pp. 141–146.
19. Yu, J. An Empirical Study of Denial of Service (DoS) against VoIP. In Proceedings of the International Conference on Ubiquitous Computing and Communications and 2016 International Symposium on Cyberspace and Security, Granada, Spain, 14–16 December 2016; pp. 54–60.
20. Tam, K.; Goh, H. Session Initiation Protocol. In Proceedings of the 2002 IEEE International Conference on Industrial Technology, Bangkok, Thailand, 11–14 December 2002; pp. 1310–1314.