Crossfire Attack Detection in 6G Networks with the Internet of Things(IoT)

Nicholas Perry[0000−0001−6873−2599] and Suman Bhunia[0000−0003−3587−3509]

Department of Computer Science and Software Engineering, Miami University, Oxford, Ohio, USA 45056
{perryna4, bhunias}@miamioh.edu

Abstract. As the internet plays an increasingly vital role in our daily lives, the threat of denial of service (DoS) attacks continues to loom, posing significant challenges to network security. With the proliferation of internet-of-things (IoT) devices, including those in the healthcare sector (IoMT), the need to secure these networks becomes even more critical. The emergence of Mobile Edge Computing (MEC) servers has shifted the focus toward processing data near the network edge to alleviate network congestion. However, a new form of DoS attack, known as the crossfire attack, presents a complex challenge as it is difficult to detect and can have devastating effects on networks. While Software Defined Networks (SDNs) offer promise in mitigating DoS attacks, they also introduce vulnerabilities of their own. This paper explores the current landscape of IoT, IoMT, DoS attacks, and crossfire attacks. It discusses existing defense strategies and proposes a defense mechanism that leverages packet header inspection to differentiate between adversarial and benign packets. The paper concludes with the execution of a crossfire attack in a Mininet environment with the RYU SDN controller, highlighting the need for multiple approaches to protect critical servers in the face of persistent DDoS attacks.

Keywords: IoT · Cross Fire attack · Neural Network · Mininet

1 Introduction

As the internet continues to evolve and become more essential to daily lives than ever, there is a growing population looking to destroy it. Denial of service (DoS) attacks have been a threat to the internet for years and continue to cause issues even in today's networks. The continued improvement and introduction of internet-of-things (IoT) devices have pushed mobile carriers to update their existing infrastructure to support the increased number of devices. With the use of Internet of Medical Things (IoMT) servers, securing the network has become even more critical. If an adversary were to disrupt the operations of IoMT devices, entire hospitals and healthcare networks may be affected, leading to disastrous, if not deadly consequences. Due to the increased load on the network, a focus on Mobile Edge Computing (MEC) has increased. These MEC

servers keep data from going across the center of the network and instead process the data near the edge of the network. A newer type of DoS attack, known as the crossfire attack has plagued the internet. This type of attack is extremely difficult to detect and can be lethal to networks if executed correctly. Software defined networks (SDNs) have been discussed as a promising mitigation technology that could detect DoS attacks. While SDNs are helpful, they are not perfect and open up a different set of vulnerabilities to exploit. The end of DDoS attacks is not in sight, and therefore many different approaches must be taken to protect critical servers.

About 25 billion devices are currently interconnected and by 2025, 60 billion devices are expected to be connected [3]. As the Internet continues to develop, traditional devices are becoming "smart", meaning that they are connected to the Internet. The term "Internet of Things"(IoT) was coined in 1999 by Kevin Ashton which describes a global network of interconnected devices [3]. The motivation for IoT devices is to create large "smart" systems [8]. Technological advancements are the reason for the increased motivation to link devices together [3]. IoT devices take many forms and almost any traditional device can be converted to a smart device. Some examples of IoT devices include smart plugs, smart washing machines, smart lights, smart refrigerators, etc. The Internet of Medical Things (IoMT) is an extension of the IoT with a focus on medical devices. These IoMT devices are medical things that have the capability to transfer information across a network without requiring human-to-human or human-to-computer interaction. These devices enable physicians to diagnose illnesses more easily by connecting various vital parameters using IoMT monitors [16].

The crossfire attack is a type of DoS attack that is more difficult to detect. This attack uses many devices across large geographic regions to send low-intensity requests across the network to various servers on the other side of the network. This is especially problematic with the advent of IoT and IoMT, because even though these devices often have extremely limited processing power, these devices can be compromised and used since the attack only requires low-intensity attacks to be sent from any given device.

Previous works seek to defend against these crossfire attacks using various methodologies. Routing around congestion (RAC) attempts to mitigate the crossfire attack by changing routing decisions based on the congestion of a given link. This solution, though also slows down legitimate traffic as all traffic is routed around that congestion. If an adversary was able to force routing decisions to consistently change, packets may be dropped as the network tries to continually determine the best route but is unable to do so. Another defense strategy, moving target defense (MTD) seeks to make the scanning phase of the attack more difficult by randomly updating routes so that an adversary would not be able to identify a consistently shared link between nodes in the network. This approach also suffers from the fact that oftentimes these routes are non-ideal, meaning legitimate traffic is degraded at the expense of security.

The summary of contributions are:

- The paper proposes a statistical detection model for crossfire attacks using Analysis of Variance (ANOVA) and neural networks.
- The proposed mechanism only uses packet headers and not packet content to determine if a packet is adversarial that achieves an accuracy of 95.3% in detecting these packets
- We evaluated the proposed defense mechanism on a real-world network topology from the ATT North America backbone topology using the Mininet simulation environment

The rest of the article is organized as follows: Section 2 discusses some background information about IoT, IoMT, DoS attacks, and crossfire attacks. Section 3 details the threat model of the attack. Section 4 discusses defense strategies against the crossfire attack. Section 5 Shows the execution of the crossfire attack.

2 Background

This section discusses the essential knowledge required to successfully execute a crossfire attack, a critical aspect of modern network security. It explores key networking concepts, including the revolutionary 6G technology, the Internet of Things (IoT) and Internet of Medical Things (IoMT), Mobile Edge Computing, and the pervasive security concerns surrounding these advancements. Moreover, it provides a comprehensive examination of the crossfire attack itself, shedding light on its intricacies and implications for network defenses. By thoroughly examining these interconnected topics, this section aims to contribute to the understanding and mitigation of cyber threats in contemporary network environments

2.1 6G

5G/6G Architecture Until recently, mobile communication was handled by fourth-generation (4G) and Long Term Evolution (LTE) systems. Recently with the rise of Internet of Things (IoT) devices and a larger focus on edge computing, a new standard had to be created in order to support the rapidly growing Internet. Fifth-generation (5G) is the next standard of mobile communication that will be able to support such a wide variety of devices simultaneously. 5G services are attempting to meet 3 main constraints as it develops: ubiquitous connectivity, zero latency, and high-speed gigabit connections [11].

Network Architecture As the number of mobile devices exponentially increases, there is a need for an architecture redesign from the previous generation. Differences in the waves used for 5G that permit increased speed require careful consideration due to differences in propagation.

5G cellular network architecture is distinct from previous generations but retains many features of those generations. A renewed focus on interior architecture is necessary. With 4G, an outdoor base station had the ability to allow both

inside and outside users to communicate. Due to the constraints and changes in architecture, the shorter waves of 5G cannot penetrate walls as easily [5]. Therefore, 5G architecture must consider distinct interior architecture to overcome the issue of penetration loss [17]. The use of multiple-input, multiple-output (MIMO) technology can help reduce the burden of penetration loss.

2.2 IoT and IoMT

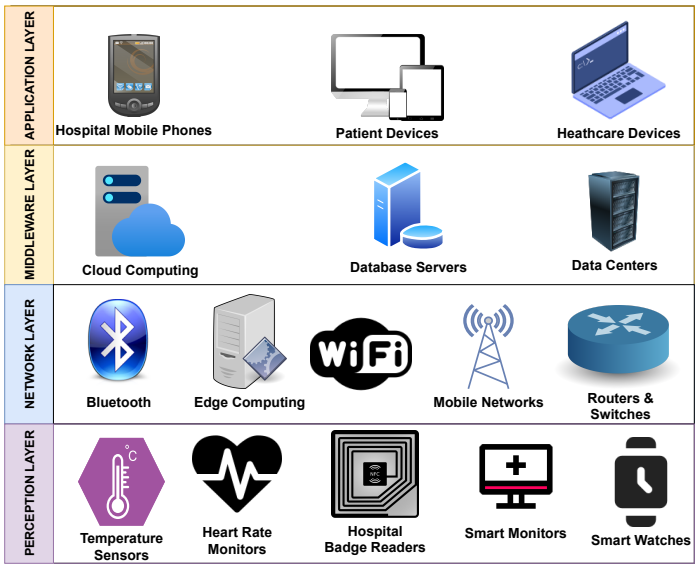


Fig. 1: The 4 Layer IoT Architecture

The Internet of Things has its own distinct architecture that works with the Internet. Typically IoT is categorized into 4 layers. Figure 1 details the types of devices that are present on each layer. The perception layer is the lowest level of the IoT architecture. The perception layer contains the sensor-enabled physical objects which act as endpoints to the IoT ecosystem. The next layer, the network layer, consists of various communication protocols, edge computing, and network connectivity. This layer transfers information securely from IoT end-points to a processing device. The middleware layer receives data from the network layer and stores it in a database. Cloud computing servers and database management systems are typically middleware devices that allow applications and sensors to connect. Finally, the top layer of the four-layer IoT architecture is the application layer. This layer IoT exists as a result of many technologies. These technologies work together to create a holistic system that is able to communicate across the internet. Radio frequency identification (RFID) was a large precursor to IoT as

it allowed machines to record metadata, recognize objects, and control devices through radio waves [8]

2.3 Mobile Edge Computing

Mobile edge computing (MEC) is an architecture where cloud computing services are placed on the edge of the network using mobile base stations [1]. With the ever-increasing need for cloud computing services while using mobile devices, placing computing servers within the radio access network (RAN) and in close proximity to these devices allows mobile traffic to connect to the nearest cloud service edge network. By placing MEC services within the the RAN, bottlenecks associated with traveling through the core of the internet can be reduced [1]. The European Telecommunications Standards Institute characterizes MEC by the following criteria: [12]

1. On-Premises - The edge services should be located at the edge of the network, meaning it should be able to run isolated from the core of the network
2. Proximity - By being close to the source of the data/information, MEC is useful for analytics and data collection
3. Lower Latency - By being closer to the edge devices, latency is considerably reduced. This can be used to reduce latency or improve user experience.
4. Location Awareness - When connected to WiFi or cellular, services can use low-level signaling to determine the location of connected devices.
5. Network Context Information - Real-time network statistics can be used by applications to provide context-specific services that can be monetized and change the experience of mobile communication.

Mobile edge computing can be used in many sectors to offload core services. Augmented reality (AR) systems typically require high computational power. Many users use (AR) on their mobile devices, so computations have to be offloaded to servers. Edge computing would allow these high-demand, low-latency tasks to remain at the edge of the network [1]. Edge computing also will play a key role with respect to web performance and caching HTML content. By deploying content delivery servers at the edge, HTTP requests would travel through these servers that would handle many of these requests, reducing traffic across the core network [1]. MEC services allow 5G to continue to work towards the core goal of "zero-latency" as reducing congestion in the core allows more traffic to be routed. This in turn improves the experience for users of 5G technology.

2.4 Security Concerns

IoT and IoMT devices may provide useful services, however, they currently present a large security problem in the world of networking. The first concern that arises with the introduction of IoT is that creating additional devices that are addressable can allow attackers to intrude [8]. Security measures are only as good as the weakest link, and the introduction of new devices opens the door to

additional vulnerabilities that could be exploited by an adversary. Due to the low cost of IoT devices, corners may be cut in terms of manufacturing. Oftentimes, IoT devices may use default or anonymous logins which an adversary can use to intrude on a network [13]. These concerns are magnified in healthcare settings. If sensors are compromised, they may report false data, or no data at all leading to misdiagnoses, or in the worst case, a patient unable to call for medical staff in a time of emergency. Therefore, it is necessary that additional security and safety measures are in place to prevent these critical devices from failing or becoming compromised.

2.5 Crossfire Attack

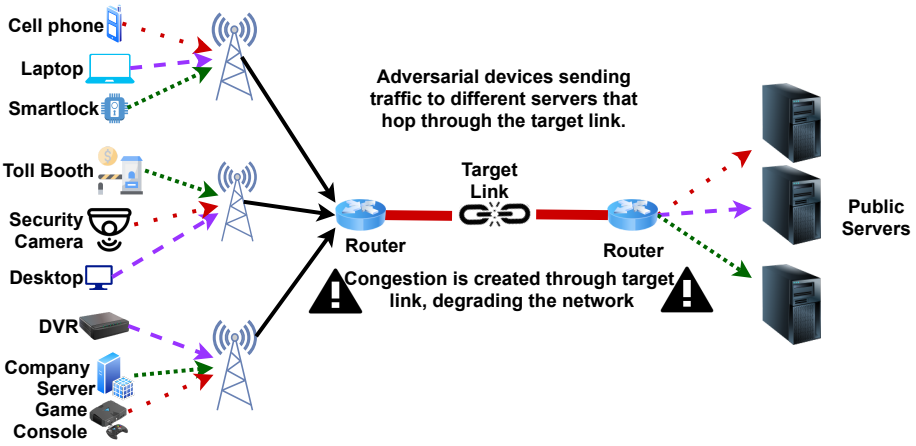


Fig. 2: Execution of the crossfire attack

The crossfire attack is a type of Link Flooding attack that attempts to degrade or disable connections to a specific geographical region of the internet. This attack is perpetuated by directing low-intensity requests to various public servers that share a common link to flood that shared link. The attack uses multiple attacking nodes, each sending low-intensity traffic to different destination nodes. This type of attack does not affect one specific destination, instead, it targets a large geographical region, served by the target link node. This type of attack is devastating to a specific geographical region, as both upstream and downstream traffic is affected [7]. The crossfire attack is more difficult to detect, as crossfire is an *indirect* attack, contrary to most other attacks. Since the attack spreads low-intensity traffic to various destinations, this allows the attack traffic to blend in with legitimate traffic and is virtually undetectable in standard DoS detection and mitigation protocols, at least until after substantial damage has been done [7].

To execute a crossfire attack, first, a potential adversary would select a list of public servers within the targeted areas and a set of decoy servers away from the target area. Since these servers are publicly accessible, they can be easily found. Next, the adversary would have to perform recognisance and generates a link map. This link map would be a map of layer 3 links that connect their decoy servers to public servers. The map enables the adversary to select a set of "target servers" that when flooded, would be able to cut off the target area from the open internet. The adversary then coordinates the decoy servers to flood the target link, effectively blocking most flows to the target area. Each individual server sends low-intensity traffic so as to not arouse suspicion of anomaly-based detection system. The individual low-intensity flows are indistinguishable from legitimate traffic. Finally, the adversary begins to flood each target link one at a time to not trigger automatic packet route mutation [7].

2.6 Related Works

MTD One proposed crossfire defense solution is the moving target defense (MTD). Traditional networks are static in nature, allowing attacks to spend as much time as needed to gather information and find vulnerabilities [4]. MTD has historically been a warfare strategy but recently has been adopted into the IT world. A moving target defense can first delay network mapping and reconnaissance. ICMP and UDP scans can be disrupted by imposing fake hosts and services on random ports that do not exist. The fake listeners can increase the time and workload an attacker would need in order to launch an attack [6]. Additionally, a mutable network can be created that changes IP addresses and ports of network hosts dynamically. This means that machines will not be able to be located at the same address at any given time [4]. Oftentimes, MTD takes the form of random route and address mutations. Randomization has been a common strategy in moving target defenses. This strategy is based on the idea that if addresses of targets within the network constantly change or access policies between the attacker and target change, then the attack success rate will drastically reduce. An attacker's ability to effectively do reconnaissance is sharply diminished as well due to the ever-changing network landscape [6]. Obfuscation of links via SDN has also been proposed to confuse and thwart attackers [2]. By obfuscating links between the attacker and target, an adversary would not be able to identify common links, a key step in performing a crossfire attack.

Rerouting-Based Defenses Moving target defense can also be used to create virtual "routable IPs". While the real IPs of hosts remain unchanged and static, the virtual IPs assigned by the network consistently changed frequently and synchronously. However, the higher the rate of mutation, the more overhead is required to run the network [4]. This type of approach is often used by load-balancers to send traffic to different destinations depending on network load.

Another proposed way to mitigate large-scale DDoS attacks is by using a routing around congestion (RAC) defense. The RAC defense works by routing

traffic between a service deployer and a critical autonomous system around degraded links. RAC defense asserts that attack traffic is irrelevant and does not need to be filtered when using this defense [14]. The RAC defense offers path isolation by dynamically creating detour routes for critical flow [15].

Infeasibility of Current Mitigation Techniques While the proposed defense solutions may work in theory, they are infeasible in nature. Rerouting-based defenses like RAC are not feasible in production servers. RAC defense uses border gateway protocol poisoning to avoid specific autonomous systems. The current border gateway protocol is incompatible and may not be able to be updated in such a way as to make this defense feasible. Even so, if this defense were made possible, it could be misused with malicious intent to attack autonomous systems [15]. Rerouting-based defense may also be able to be hijacked to force rerouting constantly. This in practice may cause packets to get dropped or delayed. Additionally, the aforementioned overhead required to implement a moving target defense may not be practical on large-scale networks.

Current Detection Efforts Current defense mechanisms treat both legitimate and attack traffic the same, degrading the performance of legitimate users. Current attack traffic detection methods point to detecting DoS and DDoS attacks, not link-flooding attacks. These detection efforts often rely on traffic being directed to a singular end-point. Therefore, models that detect standard DoS and DDoS attacks may not be able to accurately detect crossfire attack traffic.

One study, Narayanadoss et al. [10] proposes a deep-learning model to detect crossfire attacks in intelligent transport systems. This study provides a machine-learning-based model to detect vehicles in a network that are involved in the attack. The created models reflected a detection rate of 80% in a network of 35 nodes [10]. This model includes data irrelevant to traditional networks (vehicle speed) that may impact the model's accuracy in networks that are not intelligent transport systems. Additionally, as the network grew, detection accuracy decreased. Only a maximum of 35 nodes were implemented. As noted by the author, as the number of nodes increased, "[m]any legitimate flows could be detected as part of attacking traffic as they may have a temporal correlation with other attacking flows" [10]. This model may prove infeasible in larger networks, such as networks in large cities. Beyond this singular model, significant work has not been done to detect crossfire attacks and classify traffic based on characteristics.

3 Threat Model

To successfully execute a crossfire attack, adversaries must be able to mask malicious traffic behind legitimate traffic to avoid detection by the system. The aforementioned mitigation techniques focus on adversaries using the same attack nodes repeatedly. If an adversary were to have a botnet sufficiently large, they

would be able to slowly introduce attack nodes into the attack, and remove attack nodes that have become ineffective. By staying within the confines of thresholds, an attack could be executed for longer without being detected.

During the reconnaissance phase of the attack, an adversary would use multiple attack nodes to execute *trace route* commands. These commands would be done at low intensity, and low rate to avoid route mutation from the SDN. By spreading out these *trace route* commands, common links can be discovered and mapped with minimal error due to route mutation. This would allow the adversary to create a route map, and understand where secondary nodes are used when the primary link is being flooded. For maximum efficiency, the adversary would choose a time during routine peak demand, as SDNs would anticipate this stress on the system, and additional strain may be attributed to regular demand fluctuations.

Once launching the attack, the nodes would monitor the route of the low-intensity traffic to destination servers, to ensure that the traffic is being routed through the link node. If a node determines that it has been rerouted, it shall continue directing traffic to the target node, and wait before disconnecting and changing targets. This can prevent the "common denominator" defense. The moving target defense can be leveraged in itself to disrupt legitimate traffic. By forcing the SDN to continually change routes, legitimate traffic can be slowed down beyond usability.

The adversary would not launch all attack nodes at once, as this may cause a spike in demand, which the system would detect. Instead, the adversary would gradually increase attack nodes in order to mask the increase in demand as organic demand, thereby potentially circumventing anomaly-based detection [7].

As the attack propagates on the network, constant monitoring of network routing would be required. As the system responds to the attack, we would monitor the change in performance during route mutation, and when the attack is taking place undetected. This would allow for the practicality of leveraging route mutation-based mitigation to be measured.

4 Defense Mechanism

Since crossfire attacks are so lethal, it is important to detect when they are occurring as soon as possible. Therefore, using a software defined network (SDN) is ideal so that a holistic view of the network can be obtained. The use of SDN is proposed as an ideal approach to obtain a holistic view of the network. In an SDN, the control plane is decoupled from the data plane, allowing centralized control and management of the network. The SDN architecture consists of OpenFlow switches that forward network traffic based on instructions received from the SDN controller. Each OpenFlow switch in the network reports the packet headers of incoming packets to the SDN controller. Packet headers contain important information such as source and destination IP addresses, transport protocol, port numbers, etc. By inspecting these headers, the SDN controller can gain visibility into the network and analyze the characteristics of

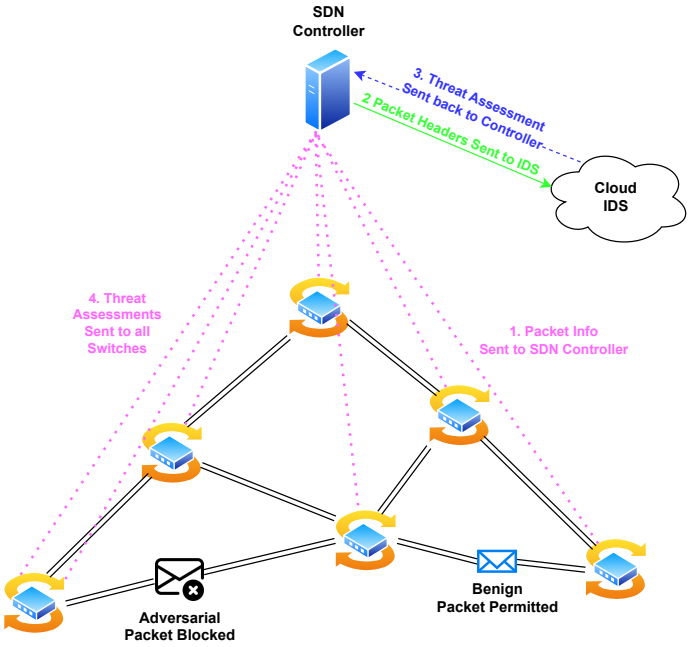


Fig. 3: Defense Mechanism

the packets flowing through it. To defend against crossfire attacks, a traffic classification model is proposed to determine whether a packet is adversarial (part of the attack) or benign. This model is implemented on the SDN controller. It leverages machine learning or rule-based techniques to analyze the packet headers and make an informed decision about the nature of the packet. The SDN controller sends the packet headers to a cloud-based IDS for further analysis. The IDS hosts the proposed traffic classification model, which evaluates the received packet headers and determines if they correspond to an adversarial or benign packet. The IDS is equipped with computational resources and advanced analysis techniques to perform this task effectively. Once the IDS determines that a packet is adversarial, the SDN controller instructs the respective OpenFlow switch to drop the offending packet(s) from the processing pipeline. By discarding the malicious packets, congestion on the network can be reduced, preventing the crossfire attack from spreading further. In summary, this defense mechanism combines the capabilities of SDN, packet header inspection, a traffic classification model, and a cloud-based IDS to detect and mitigate crossfire attacks. By inspecting packet headers, identifying adversarial packets, and dropping them in real time, the mechanism helps protect the network from the detrimental effects of crossfire attacks, minimizing potential damage and maintaining network performance. Figure 3 details the mechanism.

5 Experiment Setup

The execution of a crossfire attack, in theory, appears straightforward. By flooding a specific link, the attacker aims to overwhelm it with traffic. This process involves identifying potential routes that utilize the targeted link and generating low-intensity requests across those routes to flood the link effectively. However, in practice, executing a crossfire attack can be challenging due to the limited availability of information regarding the specific routes taken by packets. The lack of public access to this crucial routing data presents a significant hurdle for attackers attempting to orchestrate such attacks. In this section, we delve into the intricacies of executing a crossfire attack, exploring the methodologies used to overcome these obstacles and the implications of this type of attack on network performance and security.

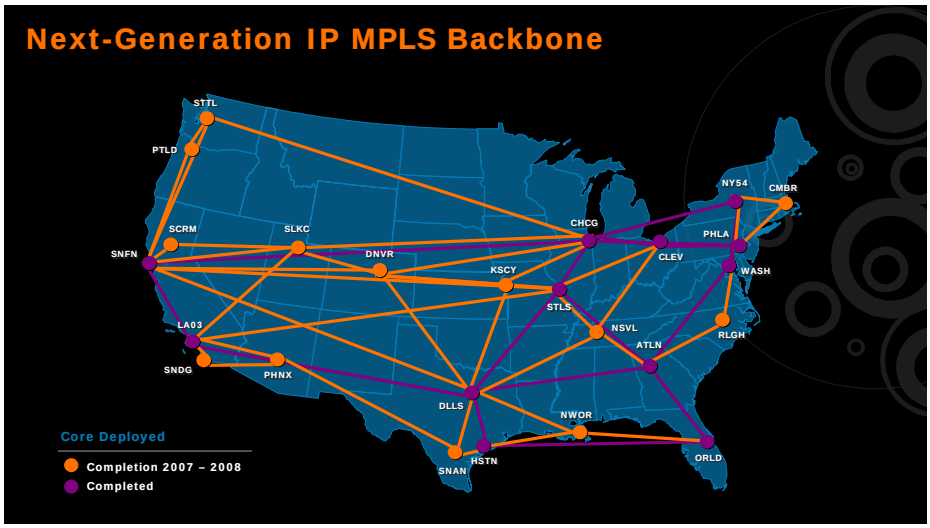


Fig. 4: Test Network Diagram [9]

Our crossfire attack was executed on a network simulated by the MiniNet network simulator and the RYU SDN controller. These tools allowed for the creation and management of a virtual network environment for experimentation and analysis. To set up the network, a python script was employed, which utilized the ATT North America Backbone network from The Internet Topology Zoo [9] as the basis for the network configuration. The network topology, as depicted in Figure 4, provides a visual representation of the structure and interconnections of the various network components. It showcases the arrangement of links within the simulated network. To introduce additional functionality and explore specific scenarios, a Mobile Edge Computing (MEC) server was strategically placed between the ORLD and CLEV nodes. The MEC server served as a

centralized computing platform that brought computing resources closer to the network edge, enabling efficient processing and analysis of data generated within the network. In this particular setup, the MEC server had a direct connection between the ORLD and CLEV nodes, facilitating seamless communication and data exchange between them. The choice of the ATT North America Backbone network from The Internet Topology Zoo [9] was driven by its complexity and size, which allowed for a more realistic simulation of network traffic. By utilizing a network with a sufficient number of components and diverse connections, researchers and analysts could better understand and evaluate the performance, scalability, and security aspects of network systems under various conditions

5.1 Executing the Attack

Once the network setup is complete, the testing scenario involves a sequence of events. First, a ping request is sent from the NY54 node, which represents an external connection, to the MEC (Multi-Access Edge Computing) server. This initial interaction confirms the connectivity between these nodes.

Following the establishment of the network, servers within the network begin initiating HTTP connections randomly across the infrastructure. Approximately 80% of the servers are engaged in requesting HTTP resources at any given time. This random traffic generation simulates unpredictable and legitimate network activity, replicating real-world usage patterns.

After a period of 30 seconds dedicated to legitimate traffic flows, the attack commences. Multiple zombie servers, compromised devices controlled by the attacker, start streaming video traffic over TCP (Transmission Control Protocol) connections to each other. TCP is deliberately chosen for this attack to obscure the nature of the traffic being transmitted. To further obfuscate the content, the videos are streamed over HTTPS (Hypertext Transfer Protocol Secure), making it difficult to distinguish the packets as video traffic based on their packet types.

Each zombie server strategically selects its destinations, ensuring that the attacking video packets pass through, but do not end at, either the ORLD or CLEV nodes. This strategic routing aims to block external connections to the MEC server. Consequently, the switches connecting these networks experience congestion due to the significant volume of data flowing through each link.

The attack is executed in three phases, with a third of the zombie servers initiating the attack during each phase. This staged approach helps distribute the attack traffic and potentially evade detection or mitigation measures.

Throughout the entire process, packet headers are captured using Wireshark, a widely used network protocol analyzer. Despite the use of HTTPS, which encrypts the content of the packets, the packet headers remain visible. Therefore, in this scenario, the network would only have access to information contained in the packet headers to analyze and identify the attack.

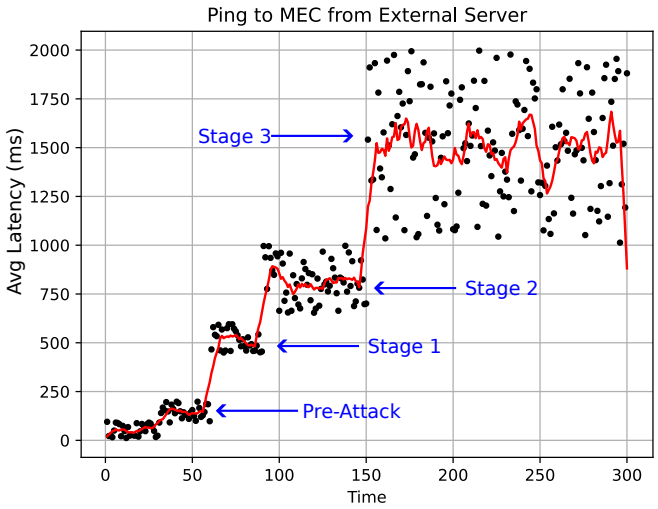


Fig. 5: Average Latency Over Time

5.2 Attack Impact

After running the network for 5 minutes, the ping round trip times were collected and plotted in Figure 5. The moving average was plotted as a line. As the attack continues, the ping increases on average. During the first stage, the average ping remained around 50-100ms. Once the zombie servers began attacking, the average increased to about 500ms. After the second phase, the ping increases to about 800ms. Finally, during the third phase the ping increases and hovers around 1500ms.

6 Crossfire Detection

Detecting a crossfire attack directly can be difficult. Since HTTPS encrypts the packets, the contents of the packets cannot be inspected. Only the headers of each packet are able to be inspected. Therefore, each model created only analyzes the headers of the packets and makes decisions based on those headers.

After running the experiment described in Section 5, packet headers were collected for every packet sent on the network. About 30,000 packets were collected. After the packets were collected, the data was aggregated. First, a standard Analysis of Variance (ANOVA) was conducted. After conducting the initial ANOVA,

6.1 Analysis of Variance (ANOVA)

An analysis of variance was first performed with all the data. The initial ANOVA gave the results depicted in Table 1 and Table 2. As shown, the largest predictor of adversarial packets were the window size, the ack_rtt, and the time_relative.

Table 1: Initial ANOVA Full Model Test

Model	-LogLikelihood	DF	ChiSquare	Prob>ChiSq
Difference	176.18978	12	352.3796	<.0001
Full	408.10276			
Reduced	584.29254			

Table 2: Initial ANOVA Parameter Estimates

Term		Estimate	Std Error	ChiSquare	Prob>ChiSq
Intercept	Unstable	-6.18762050	89752.103	0.00	0.9999
tcp.window_size		0.00057399	8.7147e-5	43.38	<.0001
tcp.len		-0.00051130	0.0003489	2.150	0.1428
tcp.stream		-0.08826740	0.0343179	6.620	0.0101
tcp.flags[0x00000010]	Unstable	-15.9381260	89752.102	0.000	0.9999
tcp.flags[0x00000011]	Unstable	9.78867015	116551.53	0.000	0.9999
tcp.flags[0x00000012]	Unstable	12.0348821	117261.45	0.000	0.9999
tcp.flags[0x00000018]	Unstable	-15.0194920	89752.102	0.000	0.9999
tcp.analysis.ack_rtt		-63.1695000	7.2849454	75.19	<.0001
frame.time_relative		0.11828939	0.0393738	9.030	0.0027
frame.time_delta		1.88407238	1.1200274	2.830	0.0925
tcp.time_relative		25.6221427	3.1284271	67.08	<.0001
tcp.time_delta		3.24388117	4.7044249	0.480	0.4905

After running the original ANOVA, we removed any non-significant factors to achieve the following ANOVA shown in Table 3 and Table 4. The test as a whole is able to determine whether or not a packet is adversarial based on the window size, ack_rtt, frame time, and tcp time. This model may not be practical given a node that consistently has a significant delay. If a node has significant delay, all packets may be marked as adversarial. Additionally, during an attack, the delay of packets through the congested links may present a problem where the model detects all packets as adversarial and blocks essentially all connections, worsening the effects of the attack.

6.2 Neural Network

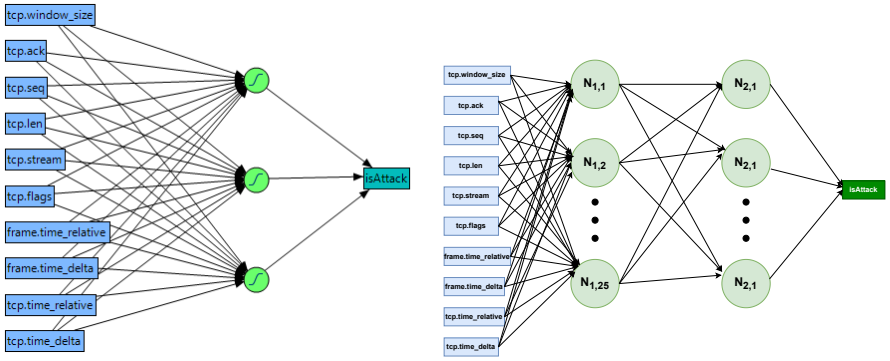
A neural network was also created based on all the criteria. The diagram for the neural network is drawn in Figure 6. The confusion matrices for training and

Table 3: Revised ANOVA Full Model Test

Model	-LogLikelihood	DF	ChiSquare	Prob>ChiSq
Difference	105.98066	4	211.9613	<.0001
Full	478.31188			
Reduced	584.29254			

Table 4: Revised ANOVA Parameter Estimates

Term	Estimate	Std Error	ChiSquare	Prob>ChiSq
Intercept	-13.9094560	3.0268445	21.120	<.0001
tcp.window_size	0.000396320	6.8154e-5	33.810	<.0001
tcp.analysis.ack_rtt	-38.0040650	3.7611744	102.10	<.0001
frame.time_relative	0.01735100	0.0023833	53.000	<.0001
tcp.time_relative	15.8395049	2.3819797	44.220	<.0001



(a) 2×3 neural network (b) 2×25 neural network

Fig. 6: Neural Network Diagrams Using Hyperbolic Tangent Nodes

Table 5: Confusion Matrix for Training Data for 1×3 neural network

		Predicted	
		Adversarial	Benign
Actual	Adversarial	19206	33
	Benign	242	4928

Table 6: Confusion Matrix for Validation Data for 1×3 neural network

		Predicted	
		Adversarial	Benign
Actual	Adversarial	6373	12
	Benign	72	1679

Table 7: Confusion Matrix for Training Data with 2×25 neural network

		Predicted	
		Adversarial	Benign
Actual	Adversarial	19308	28
	Benign	140	4933

Table 8: Confusion Matrix for Validation Data with 2×25 neural network

		Predicted	
		Adversarial	Benign
Actual	Adversarial	6375	10
	Benign	31	1720

validation data are pictured in Table 5 and Table 6. The neural network correctly predicted 99.82% of legitimate packets and 95.3% of attack packets correctly in the training data. In the validation data, the model correctly identified 99.81% of legitimate packets and 95.88% of attack packets in the validation data. An additional neural network was created with 25 nodes by 2 layers, which yielded negligibly better results. The second model only yields an extremely limited increase. Tables 7 and 8 Since these devices are IoT devices and have limited processing power, keeping the neural network model as minimal as possible is ideal. Therefore, using a smaller model for IoMT devices is the better approach.

7 Conclusion

As the internet continues to evolve and become increasingly essential to daily lives, the threat of cyber attacks, particularly Denial of Service (DoS) attacks, looms large. The rise of Internet of Things (IoT) devices, including Internet of Medical Things (IoMT) devices, has further exacerbated the need for robust security measures to protect critical networks, such as those in hospitals and healthcare systems. The advent of Mobile Edge Computing (MEC) servers has addressed some of the challenges posed by the growing number of IoT devices by processing data near the edge of the network, reducing the strain on the central network infrastructure. However, this progress has also introduced new vulnerabilities that can be exploited by attackers. One of the emerging threats is the crossfire attack, a sophisticated and difficult-to-detect type of DoS attack. The crossfire attack targets a specific geographical region by flooding low-intensity traffic from multiple devices, causing congestion and disrupting network operations. Traditional DoS detection and mitigation protocols struggle to identify and counter this type of attack effectively.

While Software Defined Networks (SDNs) have been proposed as a promising mitigation technology for DoS attacks, they are not without their vulnerabilities and limitations. Therefore, it is crucial to explore multiple approaches and strategies to protect critical servers from these evolving threats. The security concerns surrounding IoT and IoMT devices must be addressed to prevent potential intrusions and compromises. The low cost and default login credentials of many IoT devices make them attractive targets for attackers. Robust security measures and safety protocols should be implemented to ensure the integrity and reliability of these critical devices. Moving forward, the adoption of moving target defense (MTD) strategies, such as route and address mutation, and the use of obfuscation techniques can enhance network security and make it more challenging for attackers to carry out crossfire attacks. Rerouting-based defenses and the deployment of intrusion detection systems that inspect packet headers can also contribute to the detection and prevention of adversarial packets. As the number of interconnected devices continues to grow, with an estimated 60 billion devices expected to be connected by 2025, the importance of securing critical servers and networks cannot be overstated. Ongoing research and collaboration among cybersecurity experts, network administrators, and device manufacturers

are essential to developing effective defense mechanisms and ensuring the uninterrupted operation of vital services, particularly in healthcare settings. Overall, protecting critical servers from DoS attacks, including the evolving crossfire attack, requires a multi-faceted approach that combines advanced technologies, robust security protocols, and proactive defense strategies. By addressing these challenges and investing in cybersecurity measures, we can safeguard the integrity and reliability of the Internet and its essential services for the benefit of all.

References

1. Abbas, N., Zhang, Y., Taherkordi, A., Skeie, T.: Mobile Edge Computing: A Survey. IEEE Internet of Things Journal **5**(1), 450–465 (2017)
2. Aydeger, A., Saputro, N., Akkaya, K., Rahman, M.: Mitigating crossfire attacks using sdn-based moving target defense. In: 2016 IEEE 41st Conference on Local Computer Networks (LCN). pp. 627–630 (2016). https://doi.org/10.1109/LCN.2016.108
3. Balaji, S., Nathani, K., Santhakumar, R.: IoT Technology, Applications and Challenges: A Contemporary Survey. Wireless Personal Communications **108**(1), 363–388 (Sep 2019). https://doi.org/10.1007/s11277-019-06407-w
4. Gudla, C., Sung, A.H.: Moving Target Defense Application and Analysis in Software-Defined Networking. In: 2020 11th IEEE Annual Information Technology, Electronics and Mobile Communication Conference (IEMCON). pp. 0641–0646 (2020). https://doi.org/10.1109/IEMCON51383.2020.9284847
5. Gupta, A., Jha, R.K.: A Survey of 5G Network: Architecture and Emerging Technologies. IEEE Access **3**, 1206–1232 (2015). https://doi.org/10.1109/ACCESS.2015.2461602
6. Kampanakis, P., Perros, H., Beyene, T.: SDN-based solutions for Moving Target Defense network protection. In: Proceeding of IEEE International Symposium on a World of Wireless, Mobile and Multimedia Networks 2014. pp. 1–6 (2014). https://doi.org/10.1109/WoWMoM.2014.6918979
7. Kang, M.S., Lee, S.B., Gligor, V.D.: The Crossfire Attack. In: 2013 IEEE Symposium on Security and Privacy. pp. 127–141 (2013). https://doi.org/10.1109/SP.2013.19
8. Kavre, M., Gadekar, A., Gadhade, Y.: Internet of Things (IoT): A Survey. In: 2019 IEEE Pune Section International Conference (PuneCon). pp. 1–6 (2019). https://doi.org/10.1109/PuneCon46936.2019.9105831
9. Knight, S., Nguyen, H., Falkner, N., Bowden, R., Roughan, M.: The Internet Topology Zoo. Selected Areas in Communications, IEEE Journal on **29**(9), 1765 –1775 (October 2011). https://doi.org/10.1109/JSAC.2011.111002
10. Narayanadoss, A.R., Truong-Huu, T., Mohan, P.M., Gurusamy, M.: Crossfire attack detection using deep learning in software defined its networks. In: 2019 IEEE 89th Vehicular Technology Conference (VTC2019-Spring). pp. 1–6 (2019). https://doi.org/10.1109/VTCSpring.2019.8746594
11. Panwar, N., Sharma, S., Singh, A.K.: A Survey on 5G: The Next Generation of Mobile Communication. Physical Communication **18**, 64–84 (2016)
12. Patel, M., Hu, Y., Hédé, P., Joubert, J., Thornton, C., Naughton, B., Ramos, J.R., Chan, C., Young, V., Tan, S.J., et al.: Mobile-Edge Computing – Introductory Technical White Paper (Sep 2019), https://www.scirp.org/reference/ReferencesPapers.aspx?ReferenceID=2580032

13. Patton, M., Gross, E., Chinn, R., Forbis, S., Walker, L., Chen, H.: Uninvited Connections: A Study of Vulnerable Devices on the Internet of Things (IoT). In: 2014 IEEE joint intelligence and security informatics conference. pp. 232–235. IEEE (2014)
14. Smith, J.M., Schuchard, M.: Routing Around Congestion: Defeating DDoS Attacks and Adverse Network Conditions via Reactive BGP Routing. In: 2018 IEEE Symposium on Security and Privacy (SP). pp. 599–617 (2018). https://doi.org/10.1109/SP.2018.00032
15. Tran, M., Kang, M.S., Hsiao, H.C., Chiang, W.H., Tung, S.P., Wang, Y.S.: On the Feasibility of Rerouting-Based DDoS Defenses. In: 2019 IEEE Symposium on Security and Privacy (SP). pp. 1169–1184 (2019). https://doi.org/10.1109/SP.2019.00055
16. Vishnu, S., Ramson, S.J., Jegan, R.: Internet of Medical Things (IoMT)-An Overview. In: 2020 5th International Conference on Devices, Circuits and Systems (ICDCS). pp. 101–104. IEEE (2020)
17. Wang, C.X., Haider, F., Gao, X., You, X.H., Yang, Y., Yuan, D., Aggoune, H.M., Haas, H., Fletcher, S., Hepsaydir, E.: Cellular architecture and key technologies for 5G wireless communication networks. IEEE Communications Magazine **52**(2), 122–130 (2014). https://doi.org/10.1109/MCOM.2014.6736752