electronics

MDPI

Article

PrECast: An Efficient Crypto-Free Solution for Broadcast-Based Attacks in IPv4 Networks

Dalal Hanna, Prakash Veeraraghavan * and Eric Pardede

Department of Computer Science and Information Technology, La Trobe University, Victoria 3086, Australia; D.Hanna@latrobe.edu.au (D.H.); E.Pardede@latrobe.edu.au (E.P.)

* Correspondence: P.Veera@latrobe.edu.au; Tel.: +61-3-9479-1547

Received: 31 March 2018; Accepted: 4 May 2018; Published: 8 May 2018

Abstract: Broadcasting is one of the essential features in the Internet Protocol Ver 4 (IPv4). Attackers often exploit this feature of the IP protocol to launch several attacks against a network or an individual host. Attackers may either be a part of a Local Area Network (LAN) or outside a LAN to launch these attacks. There are numerous papers available in the literature to solve problems resulting from IP broadcasting. However, all these solutions target a specific problem that results from IP broadcasting. Furthermore, these solutions use either a computationally-intensive cryptographic scheme, the a priori relation between the host and the network or a modified protocol stack at every host. In this paper, we provide a seamless and transparent solution to eliminate IP broadcasting and thus eliminate all problems related to IP broadcasting. Our proposed solution is crypto-free and does not need any modification to the protocol stack.

Keywords: IPv4; ARP-poison; IP-broadcasting; DDoS; DHCP-hijacking

1. Introduction

In this paper, we focus our attention on the broadcasting feature in the Internet Protocol (IP) Version 4 (IPv4). Several important IP-based services like the Bootstrap Protocol (BOOTP) and the Dynamic Host Configuration Protocol (DHCP) are based on IP broadcasting. The Routing Information Protocol Ver.1 (RIPv1) uses IP broadcasting to disseminate the routing information [1]. Ethernet broadcasts are used by the Address Resolution Protocol (ARP) and the Neighbour Discovery Protocol (NDP) to translate IPv4 and IPv6 addresses respectively to MAC (Media Access Control) addresses.

Since a source host has no information about the destination host in a broadcast transmission, any malicious host may pretend to be the intended destination host and launch several attacks in a network. An attack may originate from a host inside a Local Area Network (LAN) or from a host outside the network. However, all the attacks are targeted towards a LAN. In Section 2, we create a taxonomy of attacks that result from IPv4 broadcasting. One such important attack is the Address Resolution Protocol (ARP) poisoning problem. We discuss them in detail in Section 3.

There are numerous papers available in the literature to solve security-related problems that result from IP broadcasting. However, all papers concentrate on solving only a particular problem. No holistic approach is made to address this problem in its entirety. Furthermore, every solution either uses cryptographic algorithms to establish a relation between an end-host and the network or uses a non-standard protocol to connect to the network. These assumptions defeat the generality offered by protocol features and the new Bring Your Own Device (BYOD) strategy. This forms the main motivation for the research work presented in this paper.

In this paper, we follow the seven-tier architecture of the Open Systems Interconnection model (OSI) for our reference. The reader may refer to [2] for the activities of each layer in the OSI stack. If the context is clear, we denote IPv4 just as IP.

Unlike other works available in the literature, in this paper, we provide an efficient and seamless solution for all security-related problems that are due to IPv4 broadcasting. Our proposed scheme, called PrECast (Protocol for Eliminating Broadcast) eliminates broadcasting at the network layer without losing the functionality of broadcasting. Our contributions and strengths are as follows:

- The proposed scheme eliminates broadcasting in IPv4 networks, without losing any functionality due to IP broadcasting. Thus, our solution eliminates every security-related issue that is due to broadcasting.
- Unlike other papers available in the literature, the proposed solution is crypto-free and does not require any modification to a protocol or configuration at the client-side. Thus, our scheme does not violate any protocol standards and support BYOD.
- Implementing our scheme requires minimal configuration at the core network, and its operation is autonomous.
- Our solution is future proof and scalable. It can easily be implemented in a Software-Defined Networking (SDN) environment.

The rest of the paper is organized as follows: In Section 2, we create a taxonomy of security-related attacks that are due to IPv4 broadcasting. One such important attack is the ARP-poisoning attack. In Section 3, we review the current solutions to the ARP-poison problem. We also highlight the strengths and weaknesses of various solutions reviewed in this paper. Section 4 deals with the design of the PrECast protocol. In Section 5, we present the performance analysis of the PrECast protocol for convergence and message complexity. In Section 6, we highlight the strength of our protocol against the protocols we reviewed here. Section 7 deals with the conclusions and future directions.

2. Taxonomy of Broadcast-Related Problems in IPv4 Networks

In the literature, all the existing work dealing with IPv4 security issues review only related work specific to their problem. However, this may not help us to provide a holistic solution to attacks against the IP protocol. Thus, we present a taxonomy of broadcast-related problems due to IPv4.

Broadcasting introduces several system and network performance issues and consumes the network bandwidth. Every active host's Network Interface Card (NIC) must interrupt its CPU (Central Processing Unit) in order to process a received broadcast packet. This will affect the CPU performance, especially when a host needs to process several broadcast packets within a short burst of time. The same is applicable to switches, as well. Most often, a host does not benefit from processing the broadcast packet. This is because the host is not the destination being sought; it does not care about the service that is being advertised; or it already knows about the service.

Since broadcasting affects both the individual host's CPU and the network performance, it is exploited often to launch a Denial of Service (DoS) attack. In the absence of a destination-based authentication, any malicious node can pretend to be the destination node and launch a Man-In-The-Middle (MITM) attack.

The attacks that are related to IPv4 broadcasting can be broadly classified into two categories:

1. Attacks that are originated external to a Local Area Network (LAN). The attackers can be anywhere on the Internet and launch an attack towards a network.
2. Attacks that are originated internal to a LAN. Here, an attacker is a host inside a LAN.

2.1. Attacks from An External Network

The main objective for this class of attack is to launch a DoS attack against the target network [3]. The attacker may not be interested in obtaining any valuable information from the target network.

The Internet Control Message Protocol (ICMP) is a supporting protocol in the Internet protocol suite. It is used by network devices, including routers, to send error messages and operational information indicating, for example, that a requested service is not available or that a host or router could not be reached [1]. The Smurf IP DoS attack [4] is a classical example of a DoS attack that exploits the IP broadcast mechanism using ICMP-echo messages. Smurf attack uses a ICMP-echo request with a large payload towards the victim. There are three parties involved in this attack: the attacker, the intermediary and the victim. The intermediary is not a single host, rather all active hosts in a specific IP subnet. One of the criteria for being an intermediary network is that it must have higher upstream bandwidth than the victim's downstream traffic. It also must have a large number of active hosts in the network.

To launch this attack, the attacker first needs to create a forged ICMP-echo-request packet with the victim's IP address as the source address and the IP broadcast address of the intermediary network as the destination address. This echo-request will be received by every active host in the intermediary network. They in turn send an ICMP-reply to the victim's machine. In this case, a single forged ICMP-echo-request from the attacker will result in hundreds of ICMP-echo-replies targeted towards the victim. Thus, the victim is subjected to network congestion that could potentially make the network unusable. If the intermediary network has less upstream bandwidth, this attack will also introduce network congestion in the intermediary network.

A Fraggle attack [4] is a variation of a Smurf attack where an attacker sends a large mount of spoofed random UDP traffic to ports 7 (Echo) and 19 (Chargen) to an IP broadcast address, with the intended victim's spoofed source IP address. It works very similarly to the Smurf attack in that many computers on the network will respond to this traffic by sending traffic back to the spoofed source IP of the victim, flooding it with traffic. Modern routers by default do not forward IP broadcast packets. Thus, most current networks are immune to Smurf and Fraggle attacks.

An attack that is similar to Smurf and Fraggle attacks is the DNS amplification attack [5–7]. Through various techniques, the attacker turns a small DNS query into a much larger payload directed at the target network. The attacker sends a DNS look-up request using the spoofed IP address of the victim to vulnerable DNS servers that support the open recursive relay. The original request is often relayed through botnets for a larger base of attack and further concealment.

These amplifications can increase the size of the requests from around 40 bytes to 4000 bytes, which is well above the maximum Ethernet packet size. The packets that are larger than the path MTU (Maximum Transmission Unit) need to be broken down for transmission and then reassembled at the destination host. This will consume more of the victim's network resources. Like the Smurf attack, botnets [8] may furthermore be used to increase the amplitude of this attack. The attack is hard to protect against using firewalls, as it comes from valid-looking servers with valid-looking traffic.

According to Arbor's report [9], the average size of DNS reflection amplification attacks is growing greatly. The maximum observed data rate of the DNS reflection amplification attack size in first half of 2016 was 480 Gbps. According to them, DDoS remains a commonly-used attack type due to the ready availability of free tools and inexpensive online services that allow anyone with a grievance and an Internet connection to launch an attack. This has led to an increase in the frequency, size and complexity of attacks in recent years.

The attacker who performs the DDoS attack may not benefit directly. They might use the DoS attack as a way of criticizing the company or government organization for exhibiting undesirable political or geopolitical, economic or monetary behaviours. A DDoS attack might also aim to punish the victim for refusing an extortion demand or for causing disruption to the attacker's business model. However, many DDoS victims never learn what motivated the attack [10].

2.2. Attacks from an Internal Network

In this class of attacks, an attacker has a strong motivation of becoming a MITM to sniff for valuable corporate information. He/she may then transmit sensitive internal data using the side-channel to an external network.

The Address Resolution Protocol (ARP) [1] is a well-known protocol that uses IP broadcast service within a LAN. ARP is a communication protocol used for resolving a network layer address (e.g., IPv4 address) to a link layer address (e.g., Ethernet address). This has a critical role in the Internet protocol (Version 4) suite. The ARP is a request and response protocol, the messages of which are encapsulated by a link layer protocol. It is communicated within the boundaries of a single local area network and never routed across Layer-3 IP subnets.

In the absence of a destination-based authentication scheme in the ARP protocol, any malicious host may pretend to be a destination host and unicast its own MAC address. Here, the attacker will convince other network users to dynamically associate the victim's IP address with the attacker's MAC address. By doing so, the attacker will become a MITM. He/she can then attract victim's Layer-3 traffic towards him/her. This attack is popularly known as the ARP-poisoning or ARP-spoofing attack. ARP-spoofing may allow an attacker to intercept data frames on a network, modify the traffic or stop all the traffic. Thus, an attacker can breach the confidentiality and integrity of messages. This attack can only be used on networks that use ARP and is thus limited to local area network segments.

The ARP-poison attack is a precursor to a plethora of LAN attacks in a network. In ARP-poisoning attacks, the victims are usually the default gateway for the network, DHCP servers, proxy servers or any other important hosts in the network. Through ARP-poisoning, an attacker can launch the following attacks:

- Gateway and proxy hijacking
- SSL-intercept attack
- DoS attack
- DNS hijacking
- DHCP hijacking

If the ARP-poison victim is the default-gateway or the proxy server of the network, the attacker can attract all IP traffic of the entire network towards him/her. He/she can then look for all valuable information in the network. According to the Calyptix report [11], the SSL-intercept attack constituted 11% of the overall Internet attacks in 2016. If an attacker can divert all the IP traffic towards him/her, he/she can then create an SSL-intercept to obtain all sensitive data from the network users.

Through ARP-poisoning, the attacker can launch a DoS attack through the following ways:

- By dropping all packets.
- The attacker can issue a random TCP-RST (TCP-Reset [1]) command to an already existing TCP connection.
- Forwarding packets to the wrong destination.

DNS hijacking is a growing threat [10]. Domain Name System (DNS) hijacking intercepts legitimate DNS requests from the user and matches them to compromised IP addresses hosted on the attacker's servers. To perform DNS hijacking, an attacker launches an MITM attack, which subverts the users' DNS requests and directs them to their own compromised DNS server. The attackers' compromised DNS server uses a DNS switching Trojan to attach the wrong IP address to the user's DNS request, therefore directing him/her to a spoofed website. This attack is popularly known as pharming and could be employed by scammers in a phishing campaign aimed at stealing personal information. Interestingly, legitimate ISP providers also engage in the ill-fetingactivity to suit their own selfish interests, including placing ads or collecting statistics for big data analysis [10].

By hijacking a DHCP server, an attacker can create a rogue DHCP server. A rogue DHCP server is a DHCP server on a network that is not under the administrative control of the network staff. Through rogue DHCP address leasing, the attacker can offer a compromised gateway address. He/she can also direct all DNS queries to an external compromised DNS server or to one set by the attacker. The main purpose of doing so is to launch an MITM attack and to direct corporate traffic to an external network.

The proxy and gateway hijackings have a similar effect compared with DNS and DHCP hijackings. The attacker can also target individual hosts.

All the above attacks fall into the active category, wherein an attacker actively participate in the network activity. In a passive attack, an attacker will passively listen and collect all valuable information on various network services and important hosts in the network (such as the default gateway, DHCP servers, etc.) by listening to ARP requests. Even though this is not an attack by itself, an attacker can collect sufficient information about the network and launch an attack at a later point in time. He/she may also forward vital information about the network to someone external to the network.

We presented a taxonomy of the above-mentioned attacks in Figure 1.

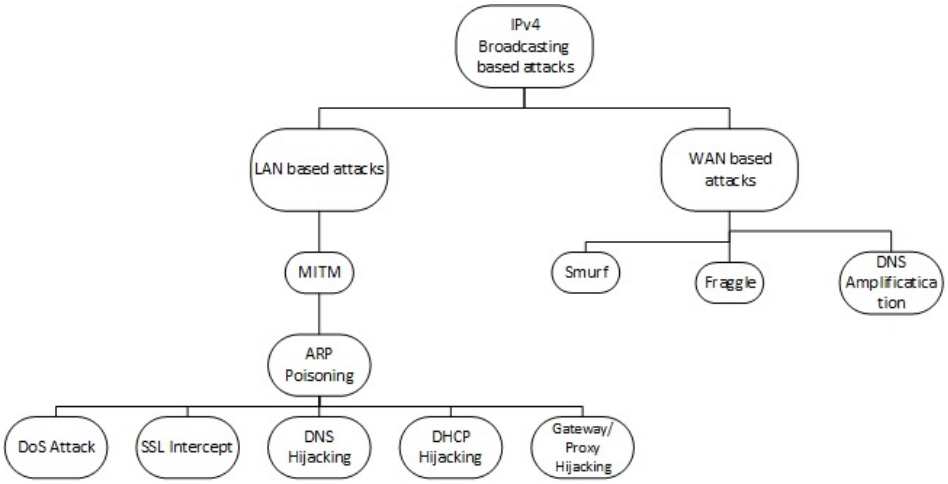

Figure 1. Taxonomy of attacks due to IPv4 broadcasting. MITM, Man-In-The-Middle.

Since the ARP-poisoning attack is a precursor to several LAN-based attacks, in the next section, we review the state-of-the-art solution to solve ARP-poisoning attacks and the strengths and weaknesses of the solutions available in the literature.

3. Mitigating the ARP-Poisoning Attack

Since the ARP-poisoning attack is the precursor to several broadcast-based LAN attacks, in this subsection, we review the current development in this topic. Recall that ARP-poisoning is a type of attack in which a malicious actor sends falsified ARP messages over a local area network. This results in the linking of an attacker's MAC address with the IP address of a legitimate computer or server on the network. Once the attacker's MAC address is connected to an authentic IP address, the attacker will begin receiving any data that are intended for that IP address.

Solutions to ARP-poisoning can broadly fall into two categories:

1. Algorithm-based solution: Here, solutions are mostly implemented within a client-server architecture. An agent running on either a client or the server monitors for ARP-poison attacks and provides solutions.
2. Hardware-based solution: Here, network hardware like switches and routers run specialized software that can detect and mitigate ARP-poisoning.

In this section, we analyse the strengths and weaknesses of each class of solutions.

3.1. Algorithm-Based Solutions

Bruschi, Ornaghi and Rosti [12] proposed a cryptographic scheme to address the ARP-poisoning problem, called a Secure Address Resolution Protocol (S-ARP). They introduced a set of functionalities that enable an integrity and authenticity check on the content of ARP replies using asymmetric key cryptography. In their scheme, every host in the network has a public and a private key pair, certified by a local trusted party on the LAN. The certificate from the trusted party provides a binding between the host identity, its public key and its IP address. Each host sends its signed certificate containing the public key and the IP address to the Authoritative Key Distributor (AKD), which inserts the public key and the IP address in a local database, after validation. Any S-ARP-enabled host is identified by its own IP address and the public/private key pairs. Using this key pair, S-ARP provides message authentication. However, S-ARP does not offer any traffic confidentiality. This feature is left to the higher layers in the OSI stack.

In S-ARP, all ARP-reply messages are digitally signed by the sender with the corresponding private key. At the receiving side, the signature is verified using the host's public key. If the public key of the sender is not available, the receiving host will send a request to the AKD. The AKD in turn sends it to the requesting host in a digitally-signed message. The first step in the S-ARP scheme is to identify the AKD and distribute through a secure channel its public key and MAC address to all the other hosts. Such an operation may be performed manually when a host is installed on the LAN for the first time.

S-ARP has several disadvantages. We list some of the important disadvantages here:

- S-ARP requires a public and private key for every host in the network. The key pair has to be generated either by the host itself or by the network administrator. Leaving it to the network administrator adds an overhead. In the first case, not every host may generate a strong key pair. An attacker may impersonate a node that has a weak key pair. In addition to this, this technique may not scale in a large-scale network.
- The public key and the MAC address of the AKD need to be sent to every host securely. Thus, this scheme can only be implemented in a restricted private network environment.
- Key revocation is one of the important issue that is hard to implement using this scheme.
- An attacker may launch a DDoS attack against the AKD to bring the entire network down.

Lootah, Enck and McDaniel [13] proposed the Ticket-based Address Resolution Protocol (TARP) to solve the ARP-poisoning problem. Their main objective was to provide a cost-effective solution to the ARP-poisoning problem. The major flaw in the design of the ARP protocol is the lack of authentication. Due to this flaw, an attacker can forge an ARP reply to inject a new address association into the victim's cache (called reply spoofing) and forge an ARP reply to replace an address association in the victim's cache (called entry-poisoning). TARP addresses these two vulnerabilities through attestation (called tickets) generated by a centralized server. Their ticketing scheme is based on [14,15]. Each ticket has a validation period.

The TARP process is similar to the ARP mechanism: a host that requires a <MAC, IP >binding will broadcast an ARP request. The host with the requested IP address sends a reply, along with its ticket. The validity of the ticket can be verified using the public key of the Local Ticketing Agent (LTA). Since an adversary has no knowledge of the private key of the LTA, he/she may not be able to forge a ticket. To make their proposal less complex, they propose to use the DHCP server for the role of LTA, as well.

Since TARP adds the missing authentication to the ARP protocol, it solves the ARP-poisoning problem. Compared with the S-ARP protocol, TARP is less complex. There is no need for every host to have a public and private key pair. The same functionality of S-ARP is achieved through the use of the digital certificate from the LTA.

However, we note that their proposed scheme has the following vulnerabilities:

- Like any centralized service, LTA is a single point of failure. Launching a DDoS attack against LTA can bring the entire network down.
- There is no mechanism of distributing the public key of LTA to all clients. In the absence of a secure delivery mechanism, any intruder can pretend to be the LTA and distribute his own public key and sign the <MAC, IP >bindings. In S-ARP, this requirement is explicitly mentioned.
- TARP in its current state cannot detect rogue DHCP servers. In a multilayered network, there is a latency between the legitimate DHCP server and some clients. This may be exploited by a rogue DHCP server that distributes IP addresses and tickets.
- To make the protocol more secure, TARP must provide an efficient ticket revocation mechanism that securely notifies all active clients about which tickets are no longer valid. Any such mechanism will incur severe network overhead in a busy network like a public hotspot or university networks.
- Even though TARP solves the ARP-poisoning problem, still the broadcast mechanism is used for the ARP-request. Thus, TARP suffers performance degradation due to the broadcast mechanism.

In [16], Nam et al. proposed an enhanced version of ARP called MR-ARP to prevent the ARP-poisoning attack in Ethernet-based LAN environments through a voting mechanism. MR-ARP determines the owner of the IP address by giving a higher priority to the previous owner in the case of conflict of the MAC address of the owner. If an MR-ARP node observes a conflict for the owner of an IP address that is not registered in its own ARP cache, the MR-ARP triggers voting on the owner of that IP address among the neighbour nodes to make a decision based on the voting results. Since the decision is made through a voting mechanism, MR-ARP must ensure that fairness in the voting mechanism is maintained. In a wired LAN, all nodes have the same transmission rate and thus are likely to send a similar number of voting packets during a particular interval of time. However, this is not possible in a wireless network. This is due to the fact that the transmission rate in a wireless network may be different for different nodes due to the traffic rate adaption based on the signal-to-noise ratio. To resolve this unfairness in a wireless segment, MR-ARP introduces a computational puzzle in the voting procedure. Thus, MR-ARP has a different fairness mechanism for different nodes.

In a later paper [17], one of the authors of [16] proposed some generalization to MR-ARP and called it the Generalized MR-ARP (GMR-ARP). The fairness in voting was improved in GMR-ARP by dropping reply packets that arrive too early. We noted that both proposals [16,17] have a number of shortfalls:

- In order to implement MR-ARP or GMR-ARP, the protocol stack at the client-side needs to be redesigned. Any end-host must have the new protocol stack installed before connecting to the network. This is an undesirable requirement in a public environment (like a university, airports or restaurants).
- Their voting process consumes a lot of network overhead. This process will also consume resources on every participating host as the broadcasting mechanism is used by the voting process.
- Their proposal requires every host to respond to a voting request as early as possible, and each host is expected to send a certain number of voting replies. However, how soon a node sends its voting reply depends on the number of processes running on the host and the queue length of the buffer at every host. An attacker could use this mechanism to launch a DoS against every host in the network.
- Both MR-ARP and GMR-ARP require every host to maintain a long-term <MAC, IP >binding table. This consumes memory. Furthermore, to keep the ARP cache entries active, hosts have to send an ARP-request to every entry in the long-term table. This incurs unnecessary network overhead.
- Since the neighbourhood density is not known to every host in the network, it is hard to set a threshold value for the number of good votes.

- Even though the authors proposed a scheme to eliminate the overhead due to cryptographic tools, their proposed scheme incurs more traffic due to the voting mechanism in addition to the existing ARP broadcasting overhead.

In general, any algorithm-based solutions to eliminate ARP-poison attacks has the following problems:

- Most of the research proposals introduced some form of message authentication through the use of cryptography. They also used a centralized certificate authority to issue digital certificates. The use of any cryptographic tool at the client-side will introduce CPU latency due to computational requirements. Severe network latency is introduced if the cryptographic computations are done at the server-side. Thus, any scheme that uses cryptography is only suitable for a smaller network.
- DDoS is a potential problem for any scheme that depends on a centralized service.
- Redesigning or implementing a modified TCP/IP stack at the client-side is possible only in a smaller private network. For this type of network, a simpler alternative is to use a static ARP table that not only eliminates the ARP-poisoning problem, but also improves the performance in the absence of any ARP broadcasting. Implementing a modified TCP/IP stack is impossible in a network that uses the concept of BYOD.
- There are few non-cryptographic techniques available in the literature to eliminate the ARP-poisoning attack; however, they use collaborative neighbour-based message authentication. In order for these protocols to work, the network has stringent conditions. One condition is to ensure that the number of well-behaving nodes exceeds the number of attackers. These conditions can easily be met in a private network, but are very difficult to ensure in a public network,. Furthermore, in the absence of any incentive for message authentication (or voting) for other nodes, selfish nodes might restrain from voting in order to save their CPU and other network resources. Thus, a minimum threshold for message authentication may not be maintained.

3.2. Hardware-Based Solution

From the key papers reviewed in Section 3.1, it is apparent that every algorithm-based solution requires either a modification to the protocol stack and/or the use of heavy cryptographic tools, along with previously-established stateful information between a host and the network. Thus, they may not support BYOD concepts.

Cisco introduced a hardware-based solution to address this security concern. Their algorithm runs on their switches and routers and thus does not need any modification at the client device.

Cisco [18] introduced a number of security features in their switches to protect a LAN. DHCP snooping protects the network from rogue DHCP servers. The network administrator configures the port to which a legitimate DHCP server is connected as a trusted port. The remaining ports are configured as untrusted ports. The switch will drop all DHCP-server messages that originate from untrusted ports. The switch also builds and maintains a DHCP binding database that consists of the MAC address, IP address, lease time, binding type, VLAN and interface-ID. In a DHCP-exhaustion attack, an attacker can exhaust the pool of available IP addresses of the DHCP server within seconds. This is done by flooding DHCP-discover packets requesting new IP addresses. The DHCP-snooping technique does not only mitigate rogue DHCP servers, but also defends against DHCP-exhaustion attacks.

Even if switch-port security is in place, tools such as Yersinia or dhcpstarv are used to randomize the field Client Hardware Address (CHADDR) inside the DHCP payload to send multiple DHCP-requests [19]. This process exhausts the DHCP address space within a short time. However, DHCP-snooping is clever enough to read the payload of the DHCP protocol and verify that the source MAC address and CHADDR are the same to defeat such starvation attacks. This extra feature needs to be enabled through an optional command.

Cisco's patented Dynamic ARP Inspection (DAI) [18] is a security feature currently built into their switches that validates ARP packets in a network. DAI intercepts, logs and discards ARP packets with invalid IP-to-MAC address bindings. This capability protects the network from Layer-2 MITM attacks. The DAI feature learns the <MAC, IP >binding from the DHCP-snooping table. Even though the DAI feature solves the ARP-poisoning problem in a LAN, it still has the following problems:

- In a stacked switch environment, DHCP-snooping is managed on the stack master. When a new switch joins the stack, the switch receives the DHCP-snooping configuration from the stack master. Whenever a member leaves the stack, all DHCP-snooping address bindings associated with the switch age out. However, in a non-stacked multi-layered environment, DHCP-snooping is hard to implement.
- As mentioned in [20], many network components require changing MAC addresses. Therefore, care should be taken when defining protections against ARP-poisoning. This includes the Hot Standby Router Protocol (HSRP), hosts configured with load balancing solutions such as Microsoft NLB (Network Load Balancing), clustering solutions, and so on. However, disabling the DAI feature on these ports could weaken the security. An intelligent attacker may exploit this weakness and launch an attack.
- As we mentioned before, DAI learns the legitimate <MAC, IP >bindings from the DHCP-snooping database. Thus, all static <MAC, IP >bindings must be manually entered into the DHCP-snooping database. For DAI, an Access Control List (ACL) must be created to include all static <MAC, IP >bindings. This process creates more overheads for the system administrator. Any change in the binding must be manually entered into the ACL. It is more common in an organization to assign static IP addresses to servers and printers. As soon as they are decommissioned, their bindings need to be removed from the ACL, as well as from the DHCP-snooping database.
- The DHCP-snooping configuration requires a highly skilled network administrator when DHCP Option 82 is enabled. For more detail on DHCP-Option 82, please refer to [1].
- DAI will not protect wireless hosts against the ARP-poisoning attack.

To our knowledge, none of the research work in the past addressed the shortfalls mentioned above. This is a main motivating factor for our solution architecture in the following section.

4. The PrECast Solution Architecture

In this section, we provide a detailed specification of our proposed PrECast architecture to eliminate IPv4 broadcasting and attacks resulting from IPv4 broadcasting. Similar to the DAI architecture, our solution is hardware-based and thus requires no configuration at the client-side. We focus on important network and system performance metrics while designing our solution. They are efficiency, scalability, flexibility and zero-configuration at the client-side. Broadcasting consumes considerable network bandwidth, CPU and buffer resources at every single network device such as switches and hosts on the network. However, several network protocols like ARP, DHCP and the Routing Information Protocol (RIPv1) depend heavily on broadcasting services. In our university network, we estimated that almost 70% of the broadcast traffic is due to the ARP protocol. Before we describe the PrECast architecture, we explain how LAN switches are modified to run our proposed solution.

Modification to Switching Devices

IPv4 broadcasting is restricted to an IP subnet and a single Ethernet broadcast domain. The Ethernet broadcast domain consists of Ethernet switches, repeaters and hubs. Since repeaters and hubs are obsolete, today's network uses only switches to create a local-area segment. In this subsection, we present what essential software modification needs to be done at switches that will support the PrEcast protocol. This modification must be done by the switch manufacturer.

Since Ethernet is the most popular and the de-facto Layer-2 protocol, we demonstrate our solution through Ethernet segments.

By design, Ethernet switches do not care about the content of any higher layer protocols encapsulated inside an Ethernet frame. As a first step, a manufacturer modifies this feature, so that switches open an Ethernet frame and record the source and the destination IP address present in an IP packet. Today's corporate network uses Layer-3 switches, mainly for QoS support, manageability, access control, and so on. Thus, these switches need no modification and support Layer-3 features natively.

Switches use a special type of memory called Content Addressable Memory (CAM) to store port numbers and MAC address relations in order to switch a frame effectively [21]. PrECast adds one more field to include the IP addresses of the hosts associated with the MAC addresses. We call this table a Modified CAM (MCAM) table. The structure of the traditional CAM table and the MCAM table is presented in Figure 2. The first column represents the port ID or the switch ID. If a host is directly connected to the switch, this field represents the port number to which the host is connected. If the MAC address corresponds to a remote host, then this field represents the ID of the switch to which the host is connected. The MCAM table can either be kept at the DRAM of a switch or the CAM architecture can be modified to include a 32-bit IP address along with other standard information.

If a switch cannot be modified, it can still be a part of the PrECast infrastructure. However, the IPv4 broadcast-related issues still exist and are confined to hosts that are connected with a switch that does not support the PrECast protocol.

Port ID	MAC Address
1	AA:BB:CC:DD:EE:GG
4	XX:YY:ZZ:LL:MM:NN

Port ID/Switch ID	MAC Address	IP Address
1	AA:BB:CC:DD:EE:GG	192.168.1.23
475234	XX:YY:ZZ:LL:MM:NN	192.168.1.72

Figure 2. Traditional Content Addressable Memory (CAM) table and a modified CAM table.

5. Setting up the PrECast Infrastructure

Since switches are the building blocks of a local area network, as the first line of action in securing a network, the network hardware must be physically secured. Furthermore, the running configuration on these switches needs to be protected through the use of strong passwords. Any attempt to change the access privilege level or the password must trigger an alarm.

The network administrator must install the cryptographic key for each switch. Only the legitimate switches that are configured by the system administrator will take part in the secure multicast process outlined in our proposal. Any other switches that try to join the multicast group will be rejected. This may include switches that are plugged into the network by end-users to expand their connectivity, which may not support the PrECast infrastructure. The administrator must also set appropriate time-out values mentioned in Section 5.1

These are the only activities that require the administrator's involvement. All other processes mentioned in the PrECast protocol are automatic and do not need any involvement from the administrator. Since the inter-switch communication needs to be protected in a corporate environment from eavesdropping, the administrator needs to install certificates and private keys irrespective of the PrECast architecture. Thus, the processes that involve the administrator in the PrECast protocol do not requires an extra overhead.

5.1. The PrECast Algorithm

The first step in the PrECast algorithm is to build the MCAM table dynamically. A host can either obtains its IP address dynamically from the DHCP server or its IP address is assigned manually by the system administrator. In these two cases, we describe how MCAM tables are built.

In any corporate network, there will be few hosts that use static IP address; for example, DHCP servers, primary and secondary DNS servers, default gateway, and so on. Since our paper deals with issues that arise due to IPv4 broadcasting, we assume that hosts that are connected to our network have Layer-3 connectivity through IPv4. To gain Layer-3 connectivity to the network, every host on the network must have a unique IP address assigned to it. The IP addresses can be either assigned statically by the system administrator (in this case, the IP address will never change) or a host may obtain its IP address dynamically from a DHCP server.

5.1.1. IP Address Is Configured to a Host Manually

We assume that the IP address is assigned to Host A manually. The conventional CAM table build-up process is as follows: During the boot-up process (or during the IP address binding process), Host A will issue a G-ARP (Gratuitous ARP [1]) broadcast packet to everyone to ensure that the assigned IP address is unique in the network. We assume that Host A is connected to Switch S. The switch port to which Host A is connected is the first network device to receive this encapsulated Ethernet frame. In the conventional process, the switch will record the port from which this Ethernet frame is received and the source MAC address present in the frame to its CAM table. Once this is done, the switch will forward this frame to every other active port in the switch, except the port from which this frame was received. Other switches receiving this broadcast frame will follow the same process as that of Switch S. In the event of any IP address conflict, a host may send a unicast reply to this G-ARP request. The G-ARP reply will be switched back to Host A using the entries available in the CAM table of every switch in the reverse path.

We modify this conventional process as follows. Note that we have not made any changes to the protocol standards and operation, thus even if a switch does not support the PrECast process, it can still coexist with the PrECast infrastructure:

Since our PrECast protocol works transparently, Host A broadcasts a G-ARP packet through switch port S_A of the switch S. This packet is then be verified by the switch for any possible IP address conflict as follows:

- Switch S will first check with its MCAM table for a possible IP address conflict by referring to its MCAM table. In case of any IP conflict, Switch S will craft a G-ARP reply and inform Host A.
- If there is no entry available for the same IP address in its MCAM table, S will multicast the G-ARP request to all other switches that are securely known to S. Switches that receive this G-ARP frame will check with their MCAM table for a possible address conflict. The switch that detects an address conflict will send a G-ARP reply to S; S will in turn craft a reply back to Host A.
- If the IP address is not mapped onto any other MAC address, S will not receive any reply from other switches until the time-out set out by the ARP-standard. After this time-out, switch S will add the switch-port S_A, the MAC address of Host A and its IP address to the MCAM table.
- In the absence of any G-ARP reply, Host A will bind the IP address like the conventional process.

5.1.2. A Host Obtained Its IP Address through a DHCP Server

In this case, after booting the system, Host A will broadcast a DHCP-discover packet. This packet in general is sent to every host in the network. The DHCP server available in the host's IP network will unicast its reply to the host with the IP address and other relevant network information.

We modify this traditional scheme to eliminate broadcasting due to the DHCP-discover process in the network as follows. Like the previous subsection, the switch port S_A receiving this DHCP-discover packet will note down Host A's MAC address and unicast to the DHCP server available in the network.

The DHCP server information is known to the switch through the manual configuration process outlined in "the boot strap process". The switch will wait for the DHCP-offer message from the server. Whenever the DHCP server offers the lease of an IP address to the requested host, the switch will complete the MCAM table that will now contain the port ID (which is S_A), the MAC address and the IP address of Host A.

Since there are only two different ways of obtaining the IP address for a host, in both schemes, the switch will transparently build its MCAM table.

The state transition diagram of the MCAM table build-up process of a switch is presented in Figure 3.

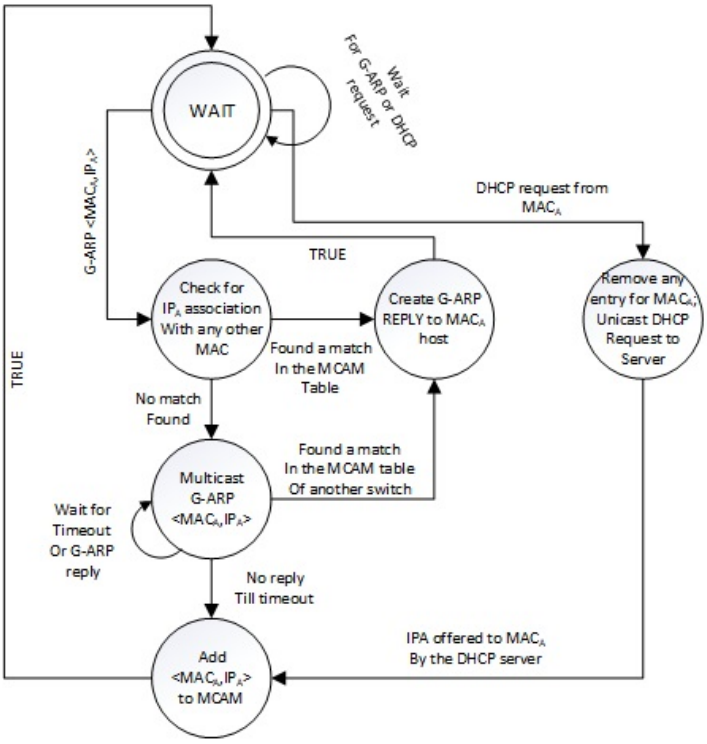

Figure 3. MCAM table build-up process. G-ARP, Gratuitous ARP; MCAM, Modified CAM.

5.1.3. The PrECast ARP Request Process

Our approach is similar to the conventional ARP request process. Host A needing to know Host B's MAC address will broadcast an ARP request packet. This packet will be received by Switch S through the switch port S_A. Instead of forwarding to every host in the network, PrECast will modify the ARP process as follows:

- Switch S will check its MCAM table for a possible <MAC, IP> binding for Host B. If the required ARP-mapping information is available, S will send a unicast ARP-reply to Host A.
- If there is no binding information available at Switch S, S will securely multicast the ARP-request to all other switches in the network. The switch to which Host B is connected (say Switch T) will send a multicast reply containing the <MAC, IP> binding of Host B to every switch in the network. Switches receiving this multicast reply will add Switch T's ID, <MAC, IP> in their MCAM table for possible use in the sequel. Eventually, Switch S will send an ARP-reply back to Host A based on the information obtained from Switch T.
- If no switch knew about Host B, like the conventional ARP process, the ARP request will go for time-out. This will be known to all switches in the network, through the absence of any multicast reply.

As can be seen from the above steps, the PrECast ARP algorithm is transparent to end-hosts. Hosts neither need a modified protocol stack, nor to be aware of the existence of the PrECast system running in the network.

The state-transition diagram of an ARP-lookup process for the proposed PrECast algorithm is presented in Figure 4.

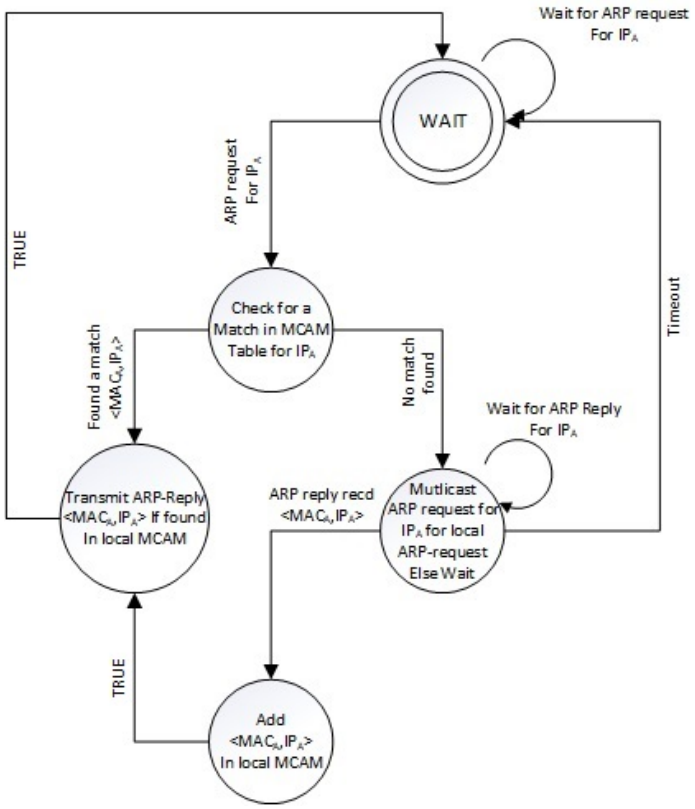

Figure 4. Protocol for Eliminating Broadcast (PrECast) ARP-lookup process.

5.1.4. MCAM Table Maintenance

We now present the MCAM table maintenance process due to the ageing process. The traditional ARP-cache table stored at every host has a time-out value. The time-out value depends on the particular OS running on the host. Having a time-out value has two advantages: Since we are removing any stale ARP cache entries from the memory, having a time-out value can conserve the memory. Time-out also has another important purpose. Within the duration of the time-out, the binding relation between the MAC and IP address of a host might have changed. For example, in a DHCP-based environment, the IP address lease may have expired, and the same IP address is allocated to another host. In a static IP environment, the administrator might have replaced a faulty network adapter that resulted in a change in the binding (this event may be quite rare in a network).

Since the MCAM table kept at a switch is similar to the ARP-cache table, it is essential to perform the above maintenance. This is to ensure that the entries kept in the MCAM table are current and valid.

In our proposed scheme, we consider two types of time-out values. The first one is the forced-time-out value. The forced-time-out option is used whenever there is a change in the MAC and IP binding relation. The second one is the traditional time-out value that deals with stale entries. We explain them below in detail.

The Forced-Time-Out:

Let t be the time-out value. Either between [0, t], or while an <MAC, IP> entry is flagged as dormant, a host may leave the network. Whenever a host leaves the network, we remove its <MAC, IP> binding from every switch in the network. In a dynamic environment, the same IP address is issued to another host with a different MAC address. We give below an example of cases where a forced-time-out is used to remove entries from every switch in the network.

Let us assume that Host A is connected to Switch S. There are a number of cases that can trigger a forced-time-out. They are:

- Host A issue a DHCP-release message after closing its connection.
- The DHCP server has issued a lease expiry message to Host A, and Host A has not responded to it.
- No L1 or L2 activity from Host A for a period of time (this period of time is a parameter set by the system administrator).
- Host A has changed its network card.

If one or more of the above cases happen, Switch S will trigger an unsolicited ARP multicast reply with $< ID_S, MAC_A, ptyset >$ with a zero time-out value. Here, ID_S is the switch ID for S; MAC_A is the MAC address of Host A; and the IP field is empty. All switches that receive this multicast reply will remove the entry for Host A from their MCAM table.

The Normal-Time-Out:

Every entry present in the MCAM table has a time-out value. If the switch detects a new frame with a particular MAC and IP combination, then the timer for the corresponding entry in its MCAM table is reset to the current system time. This means that, whenever a <MAC, IP> binding combination is in use, the corresponding entry remains fresh in the MCAM table. If a <MAC, IP> binding is not in use for the entire duration set in the time-out value, then the corresponding entry in the MCAM table is flagged as dormant. This process is different from the ARP-cache time-out process, wherein any stale entry will be removed from the memory. The main reason for us to propose this is to ensure that, whenever the same <MAC, IP> binding is needed, it is readily available without incurring any load on the network.

If a host connected to a switch issues an ARP request to another host for which its binding is flagged as dormant, the switch will change this entry from dormant to active, and a new timer value is set. Rather than removing an unused entry from the MCAM table, flagging them as dormant saves a pair of multicast request and a reply, if the entry is needed at a later point of time.

Based on the above two time-out values, PrECast can assure that the active entries kept in the MCAM table of any switch are always fresh.

5.1.5. Robustness of the PrECast Protocol

In this subsection, we analyse the robustness of the PrECast protocol. As we mentioned before, a switch that does not implement the PrECast protocol can still coexist with the PrECast infrastructure. However, the broadcast-based IPv4 attacks still exist and are confined to hosts that are connected to the unsupported switches. The PrECast switch-port to which a non-supportive switch is connected will record the <MAC, IP> mapping of all the hosts connected with the non-supportive switch in its MCAM table.

We now discuss the case when a PrECast supportive switch, say S_1, goes down. Assume that the switch S_1 is connected with port P of a switch S_2. Whenever, S_1 is down, S_2 will not detect any Layer-1 activity on port P. If there are no activities for the time set out by the administrator, S_2 will initiate the forced-time-out procedure set in Section 5.1.4. S_2 will multicast an unsolicited ARP-reply for all the hosts connected with switch S_1. All active switches that receive this unsolicited ARP-reply will

remove the host entries from their MCAM table. A similar procedure is followed if a host connected to a switch port is down.

If the binding between a host and the switch is broken and still there is a Layer-1 activity between the switch and the host, a forced-time-out procedure will not be invoked; rather, the respective MCAM entry will be marked as dormant. Whenever the broken link is restored, the MCAM entry will be changed from dormant to active. Thus, this process will not incur an extra overhead.

6. Performance Analysis of PrECast

In this section, we compare the performance of PrECast against the conventional ARP protocol. First, we present the asymptotic analysis dealing with the message complexity and the convergence time. These two parameters are inter-related and important for the overall network performance. In the second part of this section, we analyse the performance of PrECast using simulation.

6.1. Communication Message Complexity

Communication message complexity in our context determines the amount of network traffic due to the traditional ARP scheme and the proposed PrECast.

As we mentioned before, ARP broadcast is applicable only to a single IP broadcast domain. We assume the underlying network topology in which PrECast is deployed follows Cisco's recommended three-tier architecture. We assume that there are n active hosts in the network. These include PCs, mobile hosts, printers, fax machines, etc. Our LAN has m access layer switches and a distribution layer switch. The hosts are connected to the access layer switches, and the access layer switches are connected to the distribution layer switch.

Even though access layer switches and the distribution layer switch can be connected in any fashion to offer redundancy and fault tolerance, they eventually form a tree architecture to avoid any loop in the network. Thus, the entire topology can be viewed as a rooted-tree with $(n + m + 1)$ vertices and $(n + m)$ edges. The root of the tree is the distribution layer switch. The m access layer switches are the children of the distribution layer switch, and the hosts form leaf nodes.

Whenever a host broadcast a new ARP request, the request needs to be forwarded through every active port of the access and the distribution layer switches. Thus, a new ARP request traverses through every edge of the tree. Since there are n active hosts in the network, the communication message complexity for the traditional ARP is $O(n \times (n + m)) = O(n^2 + nm)$.

In PrECast, let a host issue a new ARP-request for a destination. If the destination's <MAC, IP> binding is not available in the MCAM table of the host's switch, the switch will multicast a request to the remaining m switches. Thus, in this case, the multicast request traverses only through m-edges connecting $(m + 1)$ switches. Thus, the worst case message complexity is $O(nm)$.

Even though, the worst case complexity of PrECast is $O(nm)$, in reality, PrECast issues a smaller number of multicast requests. This is because, at the startup, a switch has more <MAC, IP> bindings stored in its MCAM table than an end-host, which knows nothing in a traditional ARP scheme.

As we can observe, PrECast has a savings of $O(n^2)$ messages, even in the worst case scenario. The savings is significant in terms of bandwidth, host CPU and other performance resources in a medium to a large-scale network.

6.2. Convergence Analysis

One of the optimization schemes available to reduce the number of ARP broadcast requests is to cache the recent <MAC, IP> bindings. If the MAC address of the recently-received ARP-reply is needed at a later point in time, it can be retrieved from the cache. According to the standard, all active hosts on the network hearing an ARP broadcast from Host A should update their own ARP cache entry for Host A. Even though this feature potentially reduces the number of ARP broadcasts in the network, it poses a serious security problem. Thus, several operating systems do not support this feature. Furthermore, any unsolicited ARP reply will be silently dropped.

In the traditional ARP process, the ARP cache table is empty for every host during the startup. Every time a host issues an ARP resolution, it adds an entry to the ARP cache table. There are no more ARP broadcasts in the network under the following two conditions:

1. Every host knows about the MAC address of all other hosts in the network.
2. All hosts know about the MAC address of other frequently-communicating hosts in the network.

Condition (1) is a stronger condition and includes Condition (2). There is no need for every host to know about all other hosts in the network. Thus, the ARP convergence mostly depends on Condition (2). For the completeness of the paper, we analyse both of these conditions.

Let us assume that there are r frequently-used hosts in the network (like default gateway, DNS server, and so on). Let an arbitrary host A need the <MAC, IP> binding for Host B. Host A will issue an ARP broadcast, only when the MAC address is not available in A's ARP cache table. Thus, the probability of Host A issuing an ARP broadcast for Host B is $(n-1-t)/(n-1)$, where t is the number of ARP entries present in A's ARP-cache table at the time of a request. Initially, when A's ARP-cache table is empty, this probability is one. When there is only one host in the network not known to A, this probability is $1/(n-1)$. When the remaining $n-1$ host's <MAC, IP> bindings are available with a host, then this probability is zero. After this stage, Host A will not issue any ARP broadcast.

The convergence time is the time it takes for this probability to change from one to either $1/(n-1)$ or $(n-r)/(n-1)$, depending on whether we are interested in Condition (1) or (2). The quickest convergence happens when every ARP resolution results in a broadcast message and a new entry is added to the ARP cache table of a host.

In order to keep the ARP-cache fresh, it has a time-out value. The time-out process is discussed in the later part of this paper. In the absence of any time-out:

1. The quickest time in which Host A knows about the remaining $(n-1)$ hosts in the network is $(n-1)/\lambda$, where λ is the arrival rate of the ARP-requests with the probability $\prod_{i=1}^{n-1}(\frac{n-i}{n-1})$.
2. The quickest time in which Host A knows about all the frequently-used r hosts in the network is r/λ with the probability $\prod_{i=1}^{r}(\frac{n-i}{n-1})$.

We now analyse the convergence of PrECast. We assume that there are $n_1, n_2, \cdots, n_m$ number of hosts connected with each access-layer switch, respectively. We analyse the quickest convergence of PrECast through edge covering for graphs. Before we establish a bijective relation between PrECast and edge covering, we define some of the standard graph-theoretic terminologies used in this paper.

A graph G consists of a finite non-empty set $V = V(G)$ of vertices together with a set E = E(G) of unordered pairs of distinct vertices of V. The pair $e = \{u, v\}$ of vertices in E is called an edge of G. We also write an edge $e = \{u, v\}$ as $e = uv$.

If $e = uv \in E$, then u and v are called adjacent vertices and e is incident with each of its two vertices u and v.

If every pair of vertices of a graph is adjacent, it is then called a complete graph. A complete graph on n vertices is denoted by K_n.

A graph $G = (V, E)$ is said to be an r-partite graph (where r is a positive integer greater than one) if its vertex set can be partitioned as $V = V_1 \cup V_2 \cup \cdots \cup V_r$, such that if uv is an edge of G, then u is in some V_i and $v \in V_j$ for some $i \neq j$. If $uv \in E$ for every $u \in V_i$ and $v \in V_j$ for $i \neq j$, then G is called the complete r-partite graph. It is denoted by $K_{n_1,n_2,\cdots,n_r}$, where $\mid V_i \mid = n_i$.

A subset M of the edge set E of a graph $G = (V, E)$ is an independent edge set or matching in G, if no two edges of M have a common vertex. A matching M is maximal if there is no other matching M' such that $\mid M' \mid > \mid M \mid$. A maximal matching M is called a maximum matching if M has the maximum number of edges than any other matching.

Let $G = (V,E)$ be a graph. A set $S \subseteq E$ is called an edge cover of G if every vertex of G is incident with at least one edge of S. S is called a minimum edge cover of G if $\mid S \mid$ is of minimum size among all edge covering of G.

Standard graph-theoretic terms not defined here can be found in Bondy and Murty [22].

As we mentioned before, if two hosts A and B are connected to the same switch that implements the PrECast protocol, then the ARP resolution issued by A for B will be handled by the switch itself. However, if B is connected to another switch and the <MAC, IP> binding for B is not available with the switch to which A is connected, the ARP request will be multicasted. Let T be the topology that consists of a distribution layer switch connected with m access layer switches $S_1, S_2, \cdots, S_m$. Let $n_1, n_2, \cdots, n_m$ be the number of hosts connected with these switches, respectively. Without loss of generality, we assume that $n_1 \leq n_2 \cdots \leq n_m$.

For the given topology T running PrECast, we construct a graph $G = (V, E)$ as follows: The set of vertices of G comprises the hosts in the network. If $uv \in G$ are connected in G if and only if the ARP resolution from u to v may result in a multicast, the necessary and sufficient condition for this to happen is that u and v must be connected to a different switch. Since this relation is symmetric, the resultant graph is an undirected graph. It is easy to see that the resultant graph is the complete m-partite graph $K_{n_1,n_2,\cdots,n_m}$. For the rest of this section, T and G denote the above topologies.

The following theorem establishes a relation between an edge covering of G and how all switches in T learn about the MAC address of all hosts in the network.

Theorem 1. *Let $S \subseteq E(G)$. S is an edge cover of G if and only if all the switches in the network learn the <MAC, IP> bindings of every hosts in the network through ARP resolution either from host u to v or from host v to u for all $uv \in S$.*

Proof. Let a host u be connected to a switch S. For other switches in the network to know the MAC address of u, either there must be an ARP request from u or to u that resulted in a multicast among switches. If there are r-pairs of hosts u_i, v_i through which all switches learn the MAC address of every host in the network, then $S = \{u_i v_i \mid 1 \leq i \leq r\}$ is an edge cover of G.

Conversely, let $S \subseteq E(G)$ be an edge cover of G. Let u be any arbitrary vertex of G. We can then find a $v \in V(G)$ such that $uv \in S$. By the definition of G, the hosts representing u and v are not connected with the same switch. Thus, an ARP resolution either from u or from v towards the other host will result in a multicast. Based on this multicast, all switches in the network learn about the MAC address of both u and v. □

The following corollary is immediate from the above theorem.

Corollary 1. *Let $S \subseteq E(G)$ be a minimum edge cover of G and $k = \mid S \mid$. Then, exactly k ARP resolutions are needed in T for all switches in the network to learn the <MAC, IP> binding of every host in the network.*

We now focus our attention towards finding the minimum edge cover number for a complete m-partite graph.

The following theorem is true for any graph without any isolated vertex.

Theorem 2. *Let a graph G have no isolated vertices. Then, the cardinality of the minimum edge cover C of G is equal to the cardinality of the maximum matching M of G increased by $\mid V(G) \mid - 2\times \mid M \mid$.*

Proof. Let M be a maximum matching of G. Then, M saturates $2\times \mid M \mid$ vertices. Thus, there are $\mid V(G) \mid - 2\times \mid M \mid$ unsaturated vertices of G. Since M is a maximum matching, the set of M-unsaturated vertices of G forms an independent set. The maximum matching M can be extended to a minimum edge cover C of G by choosing an edge from $E(G) - M$ for every M-unsaturated vertex. Thus, $\mid C \mid = \mid M \mid + \mid V(G) \mid - 2\times \mid M \mid$. □

The following two theorems are due to Sitton [23], who calculates the size of the maximum matching for a complete m-partite graph.

Theorem 3. *Let $K_{m_1,m_2,\cdots,m_n}$ be a complete multipartite graph with m_i vertices in the i-th part labelled so that $m_1 \leq m_2 \leq \cdots \leq m_n$. If $m_n \geq m_1 + m_2 + \cdots + m_{n-1}$, then:*

1. *The number of edges in any maximum matching M is $m_1 + m_2 + \cdots + m_{n-1}$.*
2. *The maximum matching is obtained by connecting all vertices in the parts of $m_1, m_2, \cdots, m_{n-1}$ to m_n.*

Theorem 4. *Given any complete multipartite graph $K_{m_1,m_2,\cdots,m_n}$ with $m_1 \leq m_2 \leq \cdots \leq m_n$ and $m_n < m_1 + m_2 + \cdots + m_{n-1}$, then any maximum matching will have $\lfloor n/2 \rfloor$ edges, where $n = m_1 + m_2 + \cdots + m_{n-1} + m_n$.*

Based on all the above theorems, we have the following results:

Theorem 5. *Let $G = K_{n_1,n_2,\cdots,n_m}$ be a complete multipartite graph with n_i vertices in the i-th part labelled so that $n_1 \leq n_2 \leq \cdots \leq n_m$. If $n_m \geq n_1 + n_2 + \cdots + n_{m-1}$, then the number of edges in the minimum edge cover S is n_m.*

Proof. Let M be a maximum matching. Then, from Theorem 4, $| M | = (n_1 + n_2 + \cdots + n_{m-1})$. Let S be a minimum edge covering for G. From Theorem 3, $| S | = | M | + | V |$ - $2\times | M | = | V | - | M |$. That is, $| S | = (n_1 + n_2 + \cdots + n_m)$ - $(n_1 + n_2 + \cdots + n_{m-1}) = n_m$. □

Theorem 6. *Let G be the complete multipartite graph defined above. If $n_m < n_1 + n_2 + \cdots + n_{m-1}$, then the number of edges in the minimum edge cover of G is $\lceil n/2 \rceil$, where $n = n_1 + n_2 + \cdots + n_m$.*

Proof. Let M be a maximum matching and S be a minimum edge covering for G. From Theorem 5, $| M | = \lfloor n/2 \rfloor$. We prove this theorem based on whether the G has n odd or even number of vertices.

Case 1. Let n be an even number, and $n = 2k$.

Then, $| M | = k$. Thus, $| S | = k + 2k - 2k = k$. In this case, the maximum matching itself is a minimum edge covering for G.

Case 2. Let n be an odd number, and $n = 2k + 1$.

Then, $| M | = k$. Thus, $| S | = k + (2k + 1) - 2k = k + 1 = \lceil n/2 \rceil$. In this case, there is only one vertex of G that is not saturated by the maximum matching M. Thus, S contain edges of M along with one edge that is incident with the M-unsaturated vertex of G. □

Based on Theorems 5 and 6, we can precisely estimate how soon PrECast converges depending on how many hosts are connected to each access layer switch. Let λ be the arrival rate of ARP requests in the network. Then PrECast converges in n_m/λ or $\lceil n/2 \rceil/\lambda$ time depending on whether $n_m \geq n_1 + n_2 + \cdots + n_{m-1}$ or not.

6.3. Simulation

In this subsection, we present the message complexity and convergence analysis through simulation. We wrote a discrete event simulator using C. We implemented the traditional ARP protocol and PrECast for comparison. We set the ARP arrival rate to be two packets per minute for all hosts in the network. We considered a topology of having five access layer switches connected with a distribution layer switch. There are 20 hosts connected with every access layer switch. Our simulation topology is given in Figure 5. We ran the experiment for 60 min and collected the statistics on message complexity. We repeated the experiment 100 times and took the average to remove any simulation artefacts.

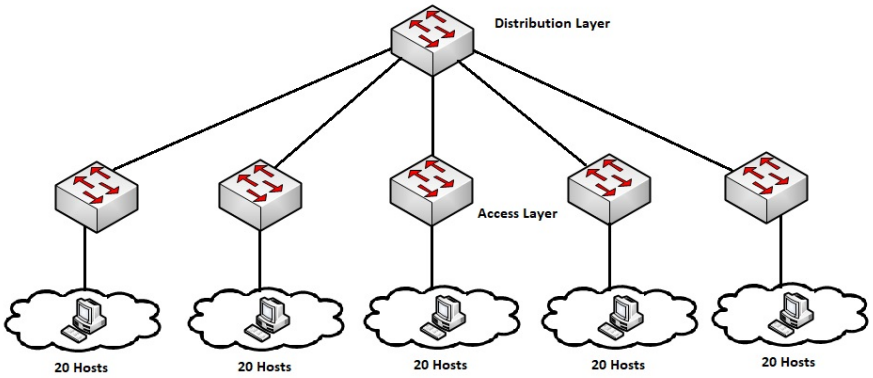

Figure 5. Simulation topology.

In our simulation, we considered the ARP protocol with two different types of ARP-cache clearance processes. In the first type, all entries in the ARP cache table are cleared after the time-out. In this case, if the host needs the MAC address of a recently-used host, it needs to initiate the ARP resolution again. The second ARP-cache clearance scheme only clears the ARP-table entries that were not used until the time-out period. All recently-used entries are kept intact. For more detail on ARP-cache algorithms, please refer to [1]. The performance graphs are presented in Figures 6–9. It can clearly be seen that the PrECast protocol outperforms the conventional ARP protocol.

In this section, we have analysed the theoretical performance of the PrECast architecture in terms of the message complexity and time to converge. We also simulated the PrECast protocol and compared its performance against the traditional ARP protocol. However, certain real-time performance parameters like CPU load on the switches and ARP-table lookup latency due to the PrECast protocol cannot be evaluated using simulation or theoretical results. Our future work involves implementing the PrECast protocol in real hardware and studying the performance parameters such as message complexity, time to converge, CPU load and ARP table lookup latency. We now summarize the implementation tasks and the main guidelines of the experimental test.

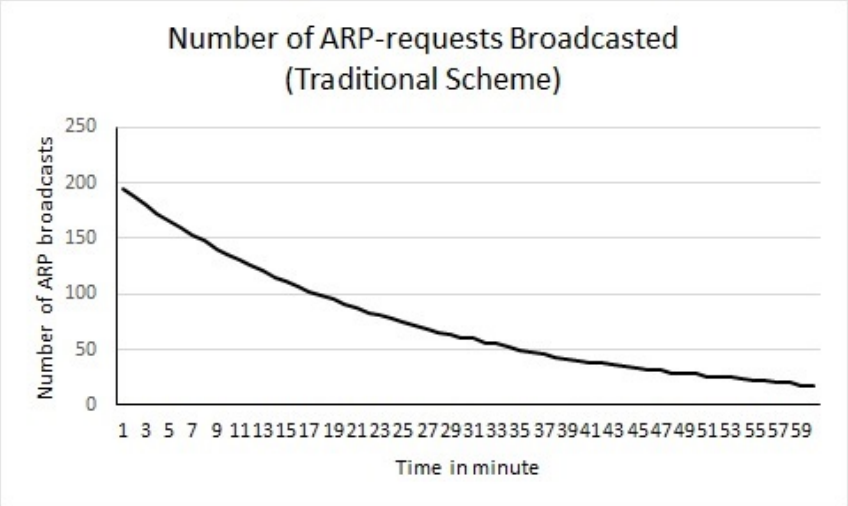

Figure 6. Traditional ARP process with no time-out.

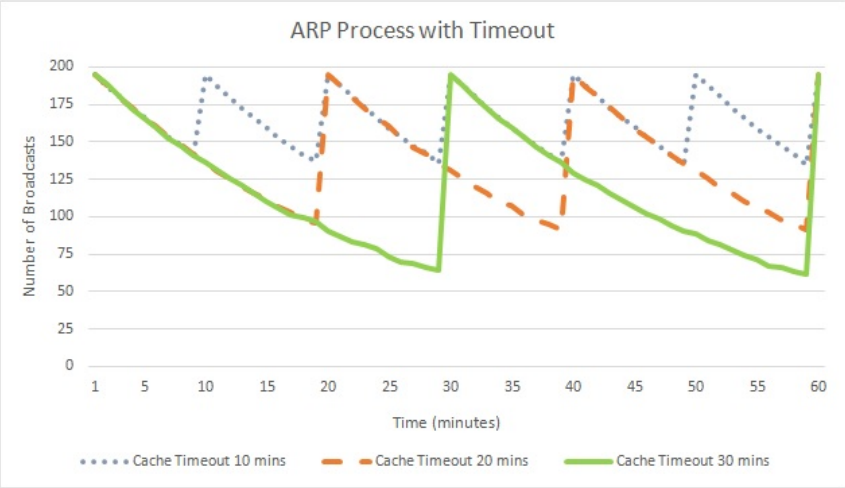

Figure 7. ARP process with time-out = 10, 20 and 30 min (Type 1).

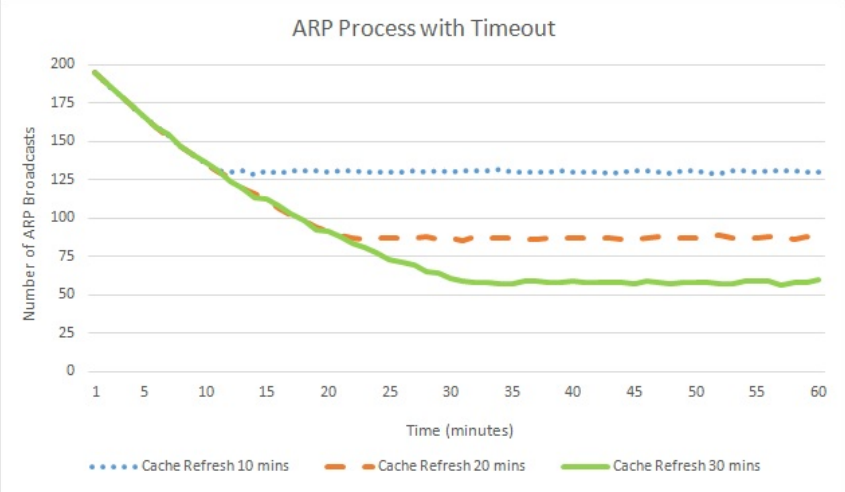

Figure 8. ARP process with time-out = 10, 20 and 30 min (Type 2).

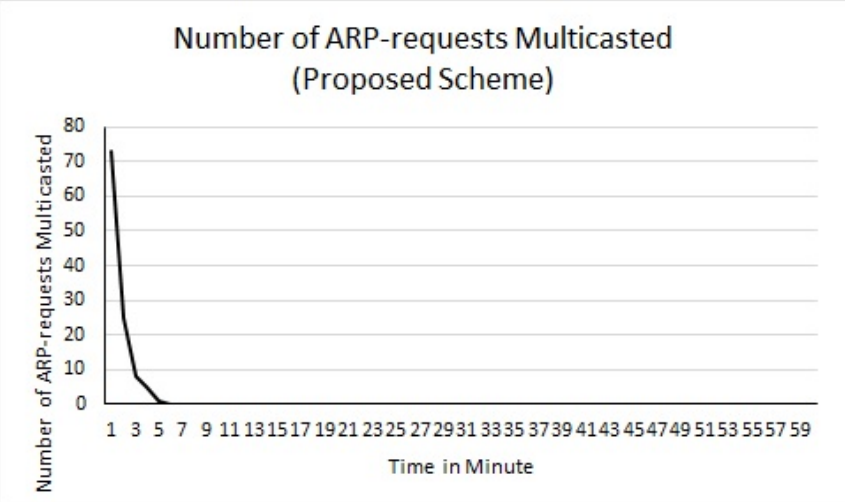

Figure 9. Number of ARP multicasts in PrECast.

As the first task, we modify the kernel of a switch operating system, so that the switching process opens a Layer-2 frame and records the source IP address in order to create the MCAM table. The MCAM table is kept in the RAM of the switches. We then start an inter-switch communication process to create a multicast group. All PrECast-enabled switches join this multicast group. The pre-configured keys are installed by the network administrator in these switches for secure inter-switch communication. As per the state-transition diagram presented in Figures 3 and 4, PrECast switches handle ARP and DHCP queries. Then, we collect short-term and long-term statistics and compared with the results presented in this paper. We strongly believe that the results obtained from the experimental network, when available, will complete and corroborate the results presented in this paper. However, at the current stage, we leave the implementation as our future work.

7. Analysis of PrECast

In this section, we analyse the strength of our proposed architecture by comparing with the existing solution available in the literature.

- From Section 2, it is apparent that broadcasting in IPv4 is responsible for several security problems. However, in the literature, every proposal only addresses a particular type of security pitfall like ARP-poison, DHCP-snooping, etc. One type of solution cannot be modified to provide a solution to another type of attack. However, our holistic approach provides a solution to all security pitfalls that are due to IPv4 broadcasting.
- Several proposals that are available in the literature use cryptographic schemes, the key distribution centre and a pre-defined relation between a host and the network, etc. Firstly, the use of cryptographic schemes may not be efficient for a mobile host due to its CPU and power requirements.
- For a solution that uses a key distribution centre (KDC), the KDC will be a single-point attack. Thus, any DoS attack towards KDC can bring the entire network operation down.
- A pre-defined relation between a host and a network (like only the registered MAC addresses' area is allowed to connect to the network) can be imposed on a small private network. However, this relation cannot be extended in a public network like an airport or university network, which encourages the concept of BYOD.

Our proposed solution does not use cryptographic schemes, thus being power- and CPU-friendly on mobile devices. Since there is no KDC involved in our solution, it is not a single-point failure. Further, pre, since there is no need to establish a predefined relation between a host and a network, our solution not only supports BYOD, but also can scale from a smaller to a larger network.

- Message overhead: Several proposed solutions in the literature require the use of control messages, thus incurring more overhead. However, our proposed protocol did not incur any extra overhead.
- Transparent implementation: If any implemented solution is known to an end-user/attacker, potentially, the solution may be targeted for attacks. However, our proposed solution may be implemented transparently without end-user's knowledge.
- Our proposed solution requires fewer configurations at the core network, compared with other crypto-based schemes that require and maintain the installation of a KDC.
- Our proposed solution requires zero-configuration at the host side. A host may not be aware of the existence of such a solution. Thus, any hacking attempt exploiting IP broadcasting is recorded by our system, and the network administrator can take immediate action against a host with malicious intention.
- In our paper, we used message complexity and convergence time as our performance parameters. Since, we have not changed the state-diagram of the protocol, the CPU load at the switches, end-devices and the RTT (round-trip time) is the same as without our protocol.

- Our solution is scalable; since our proposed solution runs as a service inside a switch, it can be implemented on any switch that runs through a network operating system. It can also be implemented using the SDN (software-defined network) architecture.

8. Conclusions and Future Direction

PrECast eliminates a number of LAN-based attacks that are due to IPv4 broadcasting.

If DDoS attacks such as Smurf, Fraggle or DNS amplification originate from an external network towards a host inside a LAN, then PrECast will not offer any solution. The solution to these problems will form our future work.

Our proposal may not offer a solution to the ARP-poisoning attack in a domain where the switches do not support PrECast. PrECast can isolate the ARP-poisoning problem to an unsupported switch domain in a mixed environment. Our future work involves running PrECast as a proxy-like service to solve the ARP-poisoning attack in a mixed environment. It is still possible to launch an attack if any malicious code modifies the MCAM table kept at a switch. Protecting the MCAM table from any malicious code is also our future work. Our future work also involves evaluating the performance of the PrECast protocol implemented through real hardware.

Author Contributions: The research work was carried out by D.H. under the supervision of P.V. and E.P. towards her PhD.

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

References

1. Fall, K.R.; Richard Stevens, W. *TCP/IP Illustrated, Volume 1: The Protocols*; Addison-Wesley Professional Computing Series; Pearson Education: Indianapolis, IN, USA, 2011.
2. Tanenbaum, A.S.; Wetherall, D.J. *Computer Networks*; Pearson International; Pearson Education: Indianapolis, IN, USA, 2010.
3. Wang, A.; Chang, W.; Mohaisen, A.; Chen, S. How Distributed Are Today's DDoS Attacks? In Proceedings of the ACM SIGSAC Conference on Computer and Communications Security, Scottsdale, AZ, USA, 3–7 November 2014; pp. 1511–1513.
4. Kasperskey Lab. What is a Smurf Attack? Available online: https://usa.kaspersky.com/resource-center/definitions/smurf-attack (accessed on 31 March 2018).
5. Deshpande, T.; Katsaros, P.; Smolka, S.A.; Stoller, S.D. Stochastic Game-Based Analysis of the DNS Bandwidth Amplification Attack Using Probabilistic Model Checking. In Proceedings of the Tenth IEEE European Dependable Computing Conference, Newcastle, UK, 13–16 May 2014.
6. Fachkha, C.; Bou-Harb, E.; Debbabi, M. Fingerprinting Internet DNS Amplification DDoS Activities. In Proceedings of the 6th IEEE International Conference on New Technologies, Mobility and Security (NTMS), Dubai, UAE, 30 March–2 April 2014.
7. US-CERT. DNS Amplification Attacks. Available online: https://www.us-cert.gov/ncas/alerts/TA13-088A (accessed on 31 March 2018).
8. Rodriguez-Gomez, R.A.; Macia-Fernandez, G.; Garcia-Teodoro, P. Survey and taxonomy of botnet research through life-cycle. *ACM Comput. Surv.* **2013**, *45*. [CrossRef]
9. Arbornetworks. Global DDoS Attack Data for thE First Half-2016. 2016. Available online: https://www.arbornetworks.com/arbor-networks-releases-global-ddosattack-data-for-1h-2016 (accessed on 31 March 2018).
10. Info Sec Institute. How to Stop DNS Hijacking. 2014. Available online: http://resources.infosecinstitute.com/stop-dns-hijacking/ (accessed on 31 March 2018).
11. Calyptix. Top 7 Network Attack Types in 2016. 2016. Available online: https://www.calyptix.com/top-threats/top-7-network-attack-types-2016/ (accessed on 31 March 2018).
12. Bruschi, D.; Ornaghi, A.; Rostin, E. S-ARP: A secure address resolution protocol. In Proceedings of the IEEE 19th Annual IEEE conference on Computer Security Applications, Las Vegas, NV, USA, 8–12 December 2003.
13. Lootah, W.; Enck, W.; McDaniel, P. TARP: Ticket-based address resolution protocol. *Comput. Netw.* **2007**, *51*, 4322–4337. [CrossRef]

14. Seo, K.; Lynn, C.; Kent, S. Public-key infrastructure for the secure border gateway protocol (S-BGP). In Proceedings of the IEEE Conference on DARPA Information Survivability, Washington, DC, USA, 12–14 June 2001.
15. McDaniel, P.; Aiello, W.; Butler, K.; Loannidis, J. Origin authentication in inter-domain routing. *Comput. Netw.* **2006**, *50*, 2953–2980. [CrossRef]
16. Nam, S.Y.; Kim, D.; Kim, J. Enhanced ARP: Preventing ARP poisoning based man-in-the-middle attacks. *IEEE Commun. Lett.* **2010**, *2*, 187–189. [CrossRef]
17. Nam, S.Y.; Djuraev, S.; Park, M. Colloborative approach to mitigating arp poisoning-based man-in-the-middle attacks. *Comput. Netw.* **2013**, *57*, 3866–3884. [CrossRef]
18. Bhaiji, Y. *Network Security Technologies and Solutions*; (CCIE Professional Development Series); Cisco Press: Indianapolis, IN, USA, 2016.
19. Yersinia. Available online: http://www.yersinia.net/index.htm (accessed on 31 March 2018).
20. Sans Infosec. Layer 2 Network Protections against Man in the Middle Attacks. Available online: https://isc.sans.edu/forums/diary/layer+2+network+protections+against+man+in+the+middle+attacks/7567/ (accessed on 31 March 2018).
21. Wikipedia. Content-Addressable Memory. Available online: https://en.wikipedia.org/wiki/Content-addressable_memory (accessed on 31 March 2018).
22. Bondy, J.A.; Murty, U.S.R. *Graph Theory With Applications*; Springer: New York, NY, USA, 2011.
23. Sitton, D. Maximum matchings in a complete multipartite graph. *Electron. J. Undergrad. Math.* **1996**, *2*, 6–16.