MITConnect

A System for Wireless Network Utilization at MIT

Advaith Anand
advaith@mit.edu

Jared Counts
jcounts@mit.edu

Lisandro Jimenez
jimenezl@mit.edu

May 6, 2016

Recitation Leader: Matei Zaharia

Section R01

1. INTRODUCTION

Thousands of students and faculty at MIT rely on its wireless network to connect to the internet. These users expect a consistent and fast experience on the network. MIT's wireless network consists of access points which are used to connect devices to the internet.

Currently, clients connect to access points (APs) by picking the one with greatest signal strength in range. However, this protocol isn't necessarily the best as it doesn't account for congestion on the network, amongst other factors. For example, in the existing protocol some APs can be overloaded while others are underutilized, and there's no mechanism in place to mediate this issue.

In this paper we present a system design to improve upon the existing interface for network communication at MIT, specifically a system to pair up clients with APs. This system will allow a client to connect to in-range APs, recommending nearby ones if none are in range, and support high network utilization at MIT. There are three main use cases for a system of this sort that we have considered. First, consider a client of variable network usage that would like to connect to the network. This client could be within range of multiple underutilized APs and thus connection would be fairly straightforward. However, if none of the APs nearby are free, then this system will recommend an AP out of range but nearby to the client. Second, if many clients try to connect at once within a small location, the APs nearby could be overwhelmed with demand if not managed properly, another use case this system will account for. Third, the network may have areas which have frequent brief connections which it should effectively manage for optimal performance.

Keeping these use cases in mind, a system has been designed with the primary design goals of simplicity, performance, scalability, fault tolerance, and user happiness.

2. DESIGN

2.1 Modules

This system design consists of three modules, as previously mentioned. These include the IS&T Server, APs, and Client Controllers. Their high level roles and connections are shown in figure 1. We will explore the modules in detail in this section.

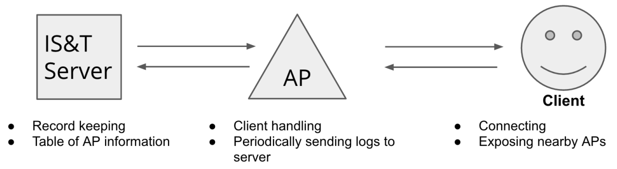

Figure 1: The system modules and their roles at a high level.

2.1.1 IS&T Server

The IS&T server has 10 terabytes of storage, is connected to the internet, and can communicate with APs directly and clients indirectly. The server has two primary responsibilities in this proposed system - (1) communicating with APs to help with the protocols described in section 2.2, and (2) record keeping. In this system implementation there will be a few hash tables stored on the server for fast lookup. These servers are all keyed by the AP MAC addresses and store the AP's locations, IP addresses, and nearby APs (within 500 feet), respectively. The server is also responsible for storing records of client and server history. In particular, for each AP MAC address the server will store a table with the timestamp and number of fulfilled users and number of unhappy users per timestamp, along with the average bandwidth used.

2.1.2 Access Points

Access Points (APs) each have a unique MAC address and have a simple UNIX-like operating system powered by a 1.2 GHz processor, along with 64 MB of flash storage and 32 MB of RAM. In this system design the APs are responsible for connecting clients and load balancing, amongst other tasks. Each AP will store a table of nearby APs, along with their IP addresses. This table will be indexed by MAC addresses. APs will also store a table with AP MAC addresses mapped to free capacities. To constantly monitor itself, each AP will keep a table of users and their average bandwidths, along with last acknowledgement time. This allows it to know utilization information and the number of users connected. APs also partake in the record keeping protocol, and will store information every second to be sent to the server when requested. As mentioned in section 2.1.1, this information will contain details of number of fulfilled and unhappy users, along with the bandwidth used at that timestamp. Finally, APs keep a queue of users trying to connect so they can serve them in sequential order.

2.1.3 Client Controller

The Client Controller is a software module present on every client attempting to connect to the internet. The client controller's primary responsibility is to connect each client to the best possible AP. The controller is given as input the list of (up to 11) APs that the client has found (see section 2.2.2) and is responsible for transmitting this list along with the client's information to a random AP. The controller in addition keeps track of R (maximum number of bits a client will send or receive over a one-second period), G (number of bits the client's applications generated in the past second), and A (the number of bits that the client actually sent in the last second), the data regarding a client's connection.

2.2 Processes

2.2.1 Access Point to Access Point Communication

A limitation of the initial system setup is that it has a method for Access Points to communicate with the central IS&T server, but not with each other. This system adds AP to AP communication to improve client connection and redistribution protocols and reduce reliance on the server for vital processes. APs can communicate with each other through their IP addresses. When an access point needs to communicate with another, it will simply send a request for an IP address for the other access point's MAC

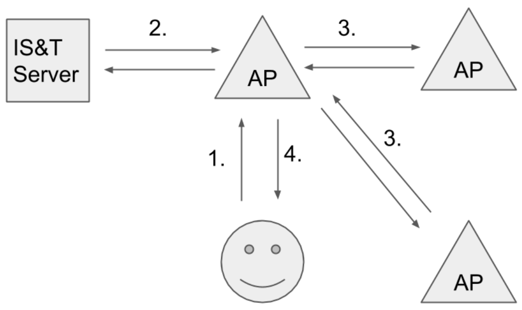

Figure 2: The Delegation phase of Client Connection at a high level. After the client discovers nearby APs, 1. The client sends a delegation request to one of the APs containing a list of the APs it discovered 2. If any of these APs are not already in the delegator's nearby AP table, then it sends a request for the nearby AP IP addresses to IS&T Server. 3. The delegator AP sends a free capacity request to each of the new APs in its nearby AP table. 4. The delegator AP then determines the AP with the most free capacity and informs the client of this AP.

address from the central IS&T server. This will be in the form of a frame encoding the MAC address. As mentioned earlier, the server will store this information in a quick lookup table. Specifically, when the server requests and received congestion information from each AP, the AP will also send its MAC and IP information. Thus the server will then respond with a frame containing the identified AP's IP address, which the communicating AP can now use to contact the other AP.

2.2.2 Client Connection

An access point will initially not know what other access points are around it until the first stage of the first client connection. There are three stages to client connection: Discovery, delegation, and connection. Here we will discuss at a high level how these stages work, and we will evaluate the client connection system in detail in section 3.2.

Discovery

For the discovery stage, when a client is connecting, it first scans all 11 data channels in a random order for nearby access points. As it scans, the client will build a list of the MAC addresses of the access points that it has discovered. The last AP the the client has discovered will be considered the Delegator AP for that client. The Delegator AP will be responsible for assigning the client to the least utilized AP within range of the client. Any AP in the network can serve delegation requests by connecting clients.

Delegation

In the second stage, the client will send a delegation request to its Delegator AP. This stage is described at a high level in figure 2. The delegation request will contain a list of MAC addresses of the other discovered access points. The Delegator AP will add this client and its discovered AP MAC addresses to a delegation queue for processing.

On a thread, each AP will be processing delegation requests in order from their delegation queue. When a client gets processed, the delegator AP will first fill in any missing entries in its table of nearby access points using the list of MAC addresses the client provided. Each access point in this table will have an IP address and free capacity information. When a new entry is being added to this list, an IP address is retrieved by IS&T, followed by the free capacity information which is retrieved from the nearby AP directly using its IP address. Free capacity is defined as a tuple containing number of remaining users an AP can handle as well as the remaining bandwidth. The remaining bandwidth is the total bandwidth capacity minus the average bandwidth usage. The free capacity information in this table will be updated in a round robin on a separate thread, and will be updated at least once per second, as determined in section 3.1.2.

The Delegator AP will then find the AP with the most free capacity from the client's discovered AP list. If the AP with the maximum free capacity, which may include the Delegator AP itself, can handle the client, then the client is assigned to that AP. If no AP is found to have a nonzero remaining user capacity as well as a remaining bandwidth that can match the given user's requested bandwidth, then the rest of the APs in the table are scanned. If one is found from that list, it will be recommended to the user to move in range of that AP.

If none are still found, then the Delegator AP will send a request for more nearby APs from the IS&T server. The server will retrieve a list of APs that are within 500 feet of this AP as well as their IPs and send this list back to the Delegator AP. The Delegator AP can then update its nearby AP list with these APs, retrieve their free capacity, and recommend the one with the maximum free capacity to the user.

Connection

If the Delegator AP responds to the user with its own MAC address, then the client will simply stay on that channel and connect to the network. If the responded AP MAC address is within the client's discovered AP list, then the client will switch to that channel and connect to the network through that AP. Otherwise, the client is instructed to seek out the AP recommended by its Delegator AP.

2.2.3 Load Balancing

There are two main issues our system has to deal with when dealing with load balancing: When to disconnect users from an AP during congestion, and when to stop allowing users to connect.

Disconnecting Users

An issue that must be addressed by our system is what happens when users all surge in usage, thus congesting the AP beyond its limits. The AP must make a decision in order to keep users happy. When an AP's bandwidth usage reaches its maximum capacity, the AP will begin to drop

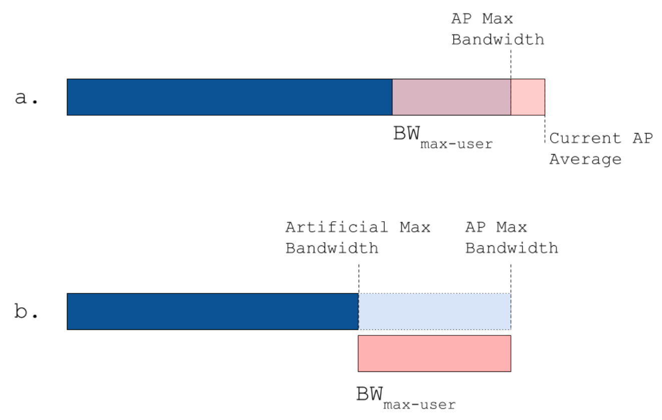

Figure 3: In a., we see that the AP is over capacity. After 15 seconds of this, the AP will kick off the largest user, whose bandwidth is denoted by $BW_{max-user}$. In b., we see that the AP has artificially lowered the bandwidth limit by $BW_{max-user}$. The AP will now appear to be fully congested to other APs.

packets, and users will begin to be unhappy. When this occurs, the AP will wait for 15 seconds. This is the time period between users reporting that they are unhappy. This gives the network time to settle below the AP limit if it is only a temporary surge. However, if the bandwidth usage does not drop down to an acceptable level, then the AP will begin to kick off the largest user of the network. The AP is able to calculate this quickly since each AP contains a table of connected users and their usage. After disconnecting the largest user, the AP again waits to see if the total bandwidth usage drops down to an acceptable level. In addition, the AP will reduce its max bandwidth capacity artificially by the disconnected user's last known bandwidth usage. This is so that the client will not simply reconnect, causing further issues. However, we may be sitting on unused capacity when this happens, so the AP will perform additive increase on the artificial bandwidth limit until it reaches its physical bandwidth limit. Our system prioritizes satisfying more users, so removing a large user in order to accommodate more users is an acceptable tradeoff. Figure 3 describes an example scenario.

Refusing Users
One potential issue that could arise with our system is that a user attempting to connect to an AP may not have enough capacity to serve that user. This might arise when the delegator AP has stale information as to the state of congestion within the other APs. Our system mitigates this by refusing users that are trying to connect to congested APs. During the connection phase, a user will send their maximum bandwidth that they will use during their connection session, called R_{user}. This user will never require more bandwidth than their supplied R_{user}. If the current AP bandwidth usage, averaged over the past 5 seconds, plus R_{user} exceeds the AP's max bandwidth limit, the AP will refuse the user's connection request. The user then performs the client connection protocol until an acceptable AP is found. If a user's R_{user} changes in the middle of a connection session, simply check the difference as above, and if the change results in the AP limit being exceeded, disconnect the user from the AP. Figure 4 describes an example scenario.

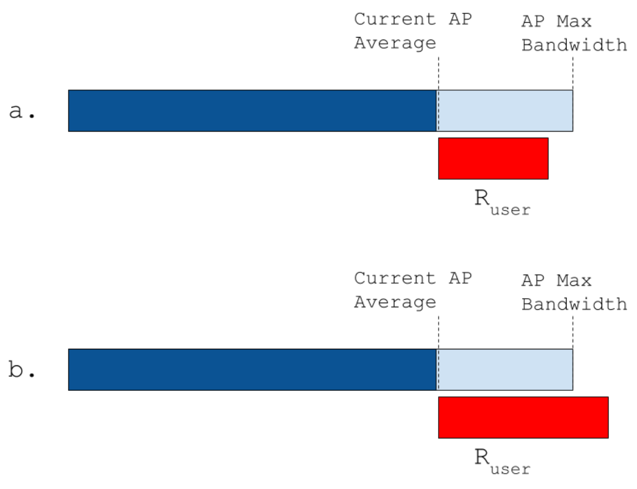

Figure 4: Two example connection scenarios. In a., the user's R_{user} will not make the sum go over the AP's max bandwidth limit, so they will be allowed to connect. In b., R_{user} will make the sum go over the max bandwidth limit, so they will be refused.

2.2.4 *Record Keeping*
The server needs an estimate of the number of users connected as well as bytes per second as per design requirements. Since each AP keeps a list of connected users, we can easily calculate the number of users connected. Specifically, the number of bytes transferred since last reset is known, so the server can calculate number of bytes transferred by keeping two numbers in memory and subtracting them: the current number of bytes transferred and the number of bytes transferred from a second ago. These two metrics will be added to an append-only log, and each AP will flush out to the server every time the IS&T server contacts an AP. To allow this, the system utilizes a record request RPC.

Record Request Remote Procedure Call
The server sends the AP the last timestamp on record continuously. The AP goes to log and sends all records after that timestamp. If the server is down, it doesn't matter since it will be polled when it comes back up. This results in less overhead for AP; link between AP and IS&T is fast, so it is better to send one big log occasionally than many smaller ones at smaller intervals. As explained in section 3.1.1, this RPC will take place every 14 minutes, generating around 10KB of logs for each AP which can be fit easily within the 64MB of storage.

3. EVALUATION
In order to prove our system meets the design goals we have specified, we will make evaluations on the system's performance for expected and worst use cases.

3.1 Communication Overhead

We will be evaluating three ways modules communicate with each other. The IS&T Server communicates with APs, APs communicate with other APs, and clients communicate with APs. Clients never communicate with the IS&T Server directly. In addition, the client to AP communication only happens during client connection, which we will analyze in section 3.2.

3.1.1 Between IS&T and APs

There are two types of communication that occur between APs and the IS&T server. These are AP information requests sent by an AP to the IS&T server, and Record Request RPCs from the IS&T server to an AP.

When an AP requests information about another AP from the IS&T server, it will be because either the information does not exist on the AP, or the AP has stale information with an expired TTL (Time to Live). In the case where the information does not exist, the AP will either ask for a max of 11 AP IP addresses (during the delegation phase of connection) or a list of APs within 500 feet (in the case no nearby AP has sufficient capacity during delegation). This should be around 44 APs (section 3.1.2). For each AP, the server will send a MAC address (6 bytes) and IP address (4 bytes) pair. This means that an AP will use at most .44 kBytes of data. If every AP requests nearby AP data at the same time, the total data usage would be .44 kBytes x 4000 APs = 1.76 MB. This is easily handled by our Gigabit link to the server, so it does not pose a congestion issue. Furthermore, the actual usage will be much less since each AP will store IP addresses in a cache. With a TTL of say 24 hours, the network impact on this address data will be miniscule.

The network impact of record request RPCs will also be relatively negligible. Each AP will store the following every second: users (32 bit integer), number of bytes transferred per second (32 bit integer), timestamp (4 bytes - 32 bits) for a total of 96 bits = 12 bytes per second. Assuming we have 4000 APs, and the IS&T server contacts five APs every second to provide their logs, each AP will be contacted around every 14 minutes. This turns into around 10KB of logs every 14 minutes, so this easily fits in 64 MB of available storage. We can also delete earlier logs as space runs out. As for network usage, five APs will provide around 10KB of logs every second, so this turns into 50KB per second, which again is negligible given our Gigabit link to the IS&T server.

3.1.2 Between APs and APs

The primary communication happening between APs is the sharing of free capacity, as explained in section 2.2.2. Each AP sends a request for free capacity from nearby APs in a round robin fashion. First we will estimate the number of APs expected to be in this table, then we will estimate how frequently a free capacity request is sent, and finally we will estimate how much bandwidth this utilizes.

We estimate that there is an average AP density of about 11 APs per 125 feet, due to the average AP range and there being 11 APs across the channels, and we assume that an AP will only learn about other APs at most 500 feet from it. With this estimate, any given AP should expect to see at most 44 other APs.

Assuming an average round trip time of 20ms and the APs update their free capacity completely sequentially, it will take about 880ms for an AP to update the entire free capacity table. A single AP would be sending at most 50 free capacity requests per second.

The specific free capacity request should be about 106 bits, assuming 48 bits for a source address, 48 bits for a destination address, and the remaining 8 bits for the procedure identifier. The free capacity response should be 184 bits total, including the source and destination addresses, an 8 bit identifier, 16 bits for the free user count, and 64 bits for the remaining bandwidth capacity. So during a single round trip, there are about 300 bits/s being exchanged between a pair of APs, and about 15 Kb/s being sent in and out from a single AP to other APs within a single second. APs can handle 54 to 96 Mb/s, so the 15 Kb/s overhead is insignificant.

3.2 Client Connection

We will estimate the amount of time it takes for a client to connect to the network. There are three stages in the client connection process, as explained in section 2.2.2. In the first stage, the client scans all of the data channels which APs broadcast on. Since there are 11 channels and it takes 30ms to switch between channels, this phase will take 330ms at most. The second stage, delegation, will take a variable amount of time. Assuming the RTT between IS&T and each AP is 25ms, and the RTT between APs are also 25ms, and that the AP will need to populate its nearby AP table and retrieve free congested info from all 10 discovered APs, the time to perform a single delegation request will be at most 275ms. Finally, the connection phase will take 30ms for the user to switch to that channel. Over all, a conservative estimate is that it will take the client 635ms to connect.

For a worst case analysis, let us assume that a delegator AP's client queue has a large number clients. An AP should have at most 44 other APs in its nearby AP table as shown in section 3.1.2. For an AP to completely populate its nearby AP free capacity table, it will need to request IPs from the IS&T Server at most 44 times each taking 20ms, and then get free capacities 44 times each taking 20ms ms, which will take 1760ms total. After this process, processing a client should be almost immediate, as the delegator AP simply needs to scan its nearby AP free capacity table. Because of this, we can expect a fresh AP with an unpopulated nearby AP table to be able to serve a large queue of delegation requests within 1-2 seconds.

3.3 Load Balancing

Since our systems uses average bandwidth usage as the congestion metric, this means that our system will attempt to keep all APs operating at near max capacity. That is, we won't refuse a connecting client unless an AP is actually congested. There will not be much unused capacity.

Our system prioritizes satisfying more users, as well as existing connections over new ones. This has a number of implications. First, our system is biased against heavy network users. These are the users that are disconnected first when an AP becomes congested. Again, we believe this an acceptable tradeoff since we would like to serve more users. Second, a user might be required to perform the client connection protocol multiple times before finding an acceptable AP to connect to, even if we could move users to other APs in order to make room for this user. Again, our goal of prioritizing existing connections means that this an acceptable tradeoff. In addition, designing our system to shuffle users to other APs optimally would require significant complexities in our system, going against our design goal of simplicity.

3.4 Scalability

Our system is able to scale well when encountered in situations of both increased users and increased APs.

3.4.1 Increased Users

Our system is able to adapt to changing requirements. If our system were deployed at a larger school with more users, our load balancing mechanisms and client connection protocol would distribute the heavier load of users evenly among APs. This is due to the randomization that occurs in the discovery phase and the fact that an AP will accept users up until it is congested.

3.4.2 Increased APs

A larger school might also opt to use a larger number of APs. Our system is able to handle this as well. Since user density should not change with school size, the only increase in load will be to the IS&T server. Since the IS&T server is contacted infrequently due to AP caching of information, this does not present a major bottleneck. However, a linear increase in time between data log collection is observed, since the IS&T server has to contact each AP individually. This could result in APs running out of memory. Each AP stores around 10kB of logs every 14 minutes, so with 64 MB of available storage, we could handle a 6000x increase in the number of APs when looking at data log collection. AP to AP communication likewise will not be a bottleneck since an AP will only contact APs within 500 ft. Thus, the network usage from AP to AP communication will remain fairly constant as the system increases in scale. If IS&T were to upgrade the hardware used, our system can adapt. In the case of APs with increased bandwidth, our system can take advantage of the updrages easily, since free bandwidth is the metric used for congestion. From the viewpoint of other APs, these updgraded APs will simply appear to be able to handle more users before becoming congested.

3.5 Fault Tolerance

Our system is robust against AP failures and server failures.

3.5.1 AP Failures

When an AP fails, clients will detect that the AP is no longer sending heartbeats. Once this happens they will perform the client connection protocol once again, in order to find an available nearby AP, or to get a recommendation if none are available. Due to the randomization and queue used in the discovery and delegation phase, these displaced users will be distributed evenly among the remaining nearby APs.

3.5.2 IS&T Server Failures

Due to the possibility of IS&T server failures for up to two minutes, our system must be resilient against such failures. Since each AP caches information it receives from the IS&T server, it can simply use these cached values whenever the IS&T is unable to be contacted. Additionally, AP to server communication will be infrequent; only when an AP requires information it does not currently have cached or the cached information has expired will an AP contact the IS&T server. As for record keeping, the IS&T server will contact individual APs, so this will not be affected by server failures. We currently estimate the server will contact each AP every 14 minutes to provide their logs. At an estimated 12 bytes per second, this amounts to 10 kB every 14 minutes. Utilizing the onboard 64 MB of flash storage, each AP could store many hours worth of logs, so two minute failures will not be an issue in this regard. If the IS&T server were to go down for longer, say a couple of hours, our system still tolerates this fault. Since we don't expect cached data to go stale very often, the system can still function. The cached data includes IP and MAC addresses of nearby APs, which should not change, unless APs are moved or new APs are added. The only issue that could arise is when the network is first being bootstrapped, and an AP does not have any cached data. If the server goes down, this AP is unable to communicate to other APs, and thus unable to perform its load balancing duties. However, we believe this will rarely be the case, since the network is only bootstrapped whenever there is a complete restart of the entire network.

3.6 Security

One potential issue is that an attacker can track users geographically across campus if they were to compromise the system. This is due to the fact that MAC addresses for clients do not change. In order for this to happen, the attacker would require access to every AP, since APs keep a table of connected users, but do not broadcast it to the IS&T server or other APs. If an attacker were to compromise the IS&T server, this would not grant the attacker the required info. We believe since an attacker obtaining access to every AP is unlikely, our system is justified in keeping connected user records in-memory.

4. CONCLUSION

Through our evaluation we've shown that our system is performant and scales well. In addition, it is simple due to the clear separation of responsibilities between modules and processes. Our system is not only straightforward to implement, but also simple to understand. Our system is also fault tolerant, since it can handle failed APs as well as periodic outages by the IS&T server. Lastly, our system addresses user happiness by maintaining an even load and minimizing the risk of surges and needing to kick users off.

5. ACKNOWLEDGEMENTS AND REFERENCES

We would like to acknowledge Matei Zaharia, our recitation instructor, and Cathie Yun, our T.A., for their guidance throuhgout this project. In addition, we would like to thank Michael Trice and Juergen Schoenstein for their advice on technical writing and presentations. Finally, we would like to thanks Katrina LaCurts and the remaining members of the 6.033 staff for designing this project and their instruction this semester.