Analyzing CRFIDs in Simulation

Ting-You Wang and Ting-Yen Wang
Advised by Michael Buettner

Background

RFIDs have almost become ubiquitous such that one can find RFID readers installed in almost any location. These RFIDs follow a well-established protocol, Fig. 1, that is built on the fact that the RFID tags perform very simple processing that require very little power. Thus, the RFID tags do not have to perform any complex storage of energy and it is assumed it can perform all of the necessary computations with the power received from the reader.

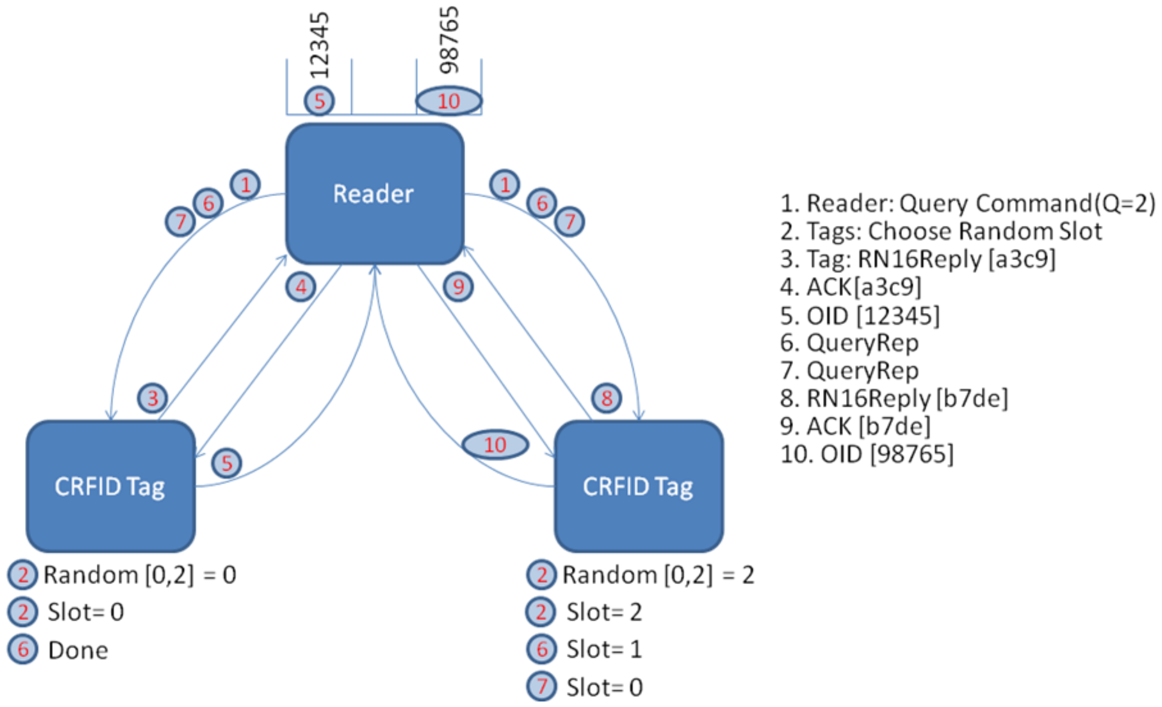

Figure 1: *Example of communication between one reader and two tags*

The above image shows the RFID protocol. Typically a Select command is issued to begin inventorying the RFIDs. Following the Select comes a ***Query command*** from the reader that contains the number of slots tags can respond in. Tags will ***randomly choose*** a slot to respond in. ***QueryRepeats*** are sent to decrement each tags slot count. When a tag's slot number decrements to 0, it sends an ***RN16 reply*** to the reader to notify that it wants to send its ID. The reader responds with an ***ACK*** containing the same RN16, and finally the tag sends its ***OID***. There are other complications that could arise, such as collisions or lost packets, but the above is the general protocol and explaining the special cases is beyond the scope of this paper.

Computational RFIDs (CRFIDs) take advantage of existing RFID architecture but do much more processing. Instead of a small CPU as in the RFID case, it is has fairly powerful one to do computation. Furthermore, this computation could be done "offline," that is not part of the normal RFID-Reader communication, often called its "workload". In order to power such a CPU, a CRFID has an onboard capacitor to store energy and once enough is stored, the CRFID turns ON and burns energy quickly when performing tasks – much more quickly than an RFID. In this paper, we use the approximation of about 0.9mW consumption. Thus the problem arises that a CRFID could burn up its energy before finishing its communication with a reader.

Applications of CRFIDs range from just taking sensor readings to performing authentication for Human Implantable devices [1].

Problem

The general assumption that an RFID tag has enough power once it is "Awake" and can always respond/process messages to/from the reader is no longer valid in CRFIDs. A CRFID can run low on power and go to sleep in the middle of any step. If the tag chooses a slot count too far out, it will burn out before it can respond, when it might have succeeded with a smaller slot number. Furthermore, certain tags will have more power than others depending on their distances from the reader, so they can handle larger slot counts.

Currently, RFID simulators model the normal RFID setup with the "unlimited" power assumption once on. However, this does not work for CRFIDs. Therefore, to simulate CRFIDs and to see the effects of a high energy consuming CPU, a simulator needs to be extended to model the CRFID, taking into account shutting down due to energy burn out. Furthermore, we need to see the effect of using the random slot count choice and investigate how it compares to a variation on the protocol that takes into account the current energy stored to generate a slot number.

In the real usage of a CRFID, there is the "workload" that it performs upon start-up. This is typically a static amount every time the tag wakes up and in the project we are not interested in this fixed amount, but rather the effects of the power usage in the MAC Protocol. Therefore, we do not have a workload in this iteration of the simulator.

In the project we will be using the WISP as the CRFID that we will model.

We will not make any adjustments to the Reader's behavior because the common setup in the real world today is to use the currently deployed RFID readers and add in CRFIDs with no change to the reader.

Approach

In order to evaluate the effectiveness of a power heuristic for CRFIDs, we adapted an RFID simulation [2] to a CRFID simulation and measured the performance of the tags when they were treated as: Normal RFIDs, CRFIDs without any power heuristics, and CRFIDs with a heuristic. We will first discuss the changes that had to be made to the RFID simulation to make it into a CRFID simulation and then we will discuss the power heuristics that were added to simulation to improve performance.

First, we picked up the RFID simulation and had to get it to run. Upon getting the simulation to run correctly, we needed to begin by introducing a model for the tag's capacitor. Basically, the capacitor object keeps track of the number of Joules stored up by the tag. Using measurements provided by Michael Buettner, the capacitor has a maximum of 5Volts. As long as the tag is getting a signal from the reader, the tag accumulates energy up to the maximum. Once the tag performs tasks, such as processing a message or transmitting it will deduct from the capacitor.

As part of the capacitor model, a CRFID also has certain energy levels for when it is "ON" and "OFF". The following is a summary of its behavior:

1. The tag begins with no energy
2. When the tag hits 2V, the tag is considered "ON" and can perform tasks
3. When the tag drops to 1.5V, the tag goes to sleep in "OFF" mode and drops all tasks
4. Repeat 2 and 3

This behavior was added into the Tag model so that when performing any task, we first check if we are "ON" or "OFF" to decide if we drop a task (because there is not enough energy) or we let the task run.

When the capacitor and tag states were modeled correctly, we needed to model the addition and subtraction of energy appropriately. In order to make the project possible within the quarter, we made the following approximations:

- Between signals from the reader and tag, the power level is constant (no attenuation)
- The energy in the capacitor grows and shrinks linearly

The simulator provided a *Power Signal* which was not a message but a way for the tag to record the current power level we are getting from the reader. These signals, in addition to actual messages, are used to perform the accumulation of energy in between messages and transmits. This is done by simply taking the power level indicated by the signal and using that power over the duration from the last event (transmit, message, or power signal). Next, when a real message is being processed, we accumulate power as well as use it up as the message is processed. Therefore, we still add in the energy, but if the energy usage drops us to 1.5V, we drop the message (even if we are midway through the process). A similar thing is done for transmits.

Next, a CRFID also will turn OFF upon sending its OID and wait until it is back up to 2V before participating in query rounds again. This was added into the model by modifying the possible *Tag States*. When a tag sends off its OID, it is in the *Acknowledged* state and stays there until another query or query repeat comes in. At this point, instead of performing a read or generating another slot number to participate in another round, it goes to the *Unpowered* state and recharges back up to 2V.

Once we had a simulator modeling a CRFID, we examined its performance against normal RFIDs to show that there was, indeed, a problem with CRFID performance. We attempted to determine an optimal performance for CRFIDs, but this turned out to be a nontrivial problem that we leave for future research. What makes this difficult is that a tag could only best schedule itself into a slot if it could actually keep track of the energy of *all* of the tags as well as everyone's accumulation rate (not just its own). After this, it is very difficult to determine whether it might be better for a tag to *not* participate in a query round even though you have energy to do so (ex. the tag estimates it might collide with another tag), or if a tag should just abandon the current round in order to choose a better (ex. non-colliding) position for the next round. To add to the complication, the schedule might be different depending on the layout and number of tags.

We primarily tried to improve the performance of CRFIDs (more about what 'performance' means in the results section) by making simple modifications to the slot number algorithm. We did this in two ways. The first was to take the amount of energy available in a capacitor to process tasks and compute based on that the number of QueryRepeats the tag can process before it will not have enough energy to complete the "transaction" (send OID). This is a conservative estimate, as we are assuming no future accumulated power. We choose a random slot number no more that this maximum number of QueryRepeats. If we do not have enough energy for even one QueryRepeat, we do not participate in the round. We called this *SlotPredict*.

The second algorithm tries to keep collisions down and not be overly pessimistic about the number of QueryRepeats that can be handled by spreading the tags across a larger slot range, however it does so probabilistically. The algorithm first divides up the entire slot range down the middle – the higher slots and lower slots. Then, based on the number of QueryRepeats a tag believes it can handle, it probabilistically chooses a random slot from either the upper range or the lower range. We performed two variations of this, one where the higher the QueryRepeat count, the more likely the tag will be in the top, and vice versa. The former is to give the low energy tags a better chance to send before running out of energy – we call this *SlotPartition*. The latter is based the observation that sometimes choosing farther slots actually give a tag time to gain more energy as other tags are transmitting – we call this *ReverseSlotPartition*.

There were also certain features not implemented in the simulator that we had to adapt to. For example, to get some meaningful results, we wanted the reader to continue to issue queries and have the tags respond even after all tags have been inventoried – the simulation used to shut down after each tag responded just once. Typically using session one would do this, but this did not seem to work, so instead we had to adjust the tags to always respond and the reader to never shut down. There were many other tweaks made to bend the simulator to our model (many thanks to Michael for helping us understand what could and could not be removed/modified).

Results

In this section we will present our results. To measure performance, we first see the success rate of tags (given by the formula: |OIDs sent|/|RN16s sent|) as well as the overall throughput of OIDs sent through the system (given by the formula: # OIDs sent/second). We vary the distance a tag is from the reader as well as the number of tags and their arrangements. Each experiment was conducted over 10 simulated seconds.

First, as a proof of concept that, indeed, we do have major performance degradation when the battery is introduced; as shown in Figure 2, which shows the performance of a cluster of 36 tags at various distances, we see a large gap in throughput and a drop in success rate when comparing RFIDs against a basic CRFID.

At the 2 to 3 meters, we begin to see that processing messages costs more than the amount of energy accumulated from the reader. At 5 to 6 meters, we see that the accumulation of energy so low that many tags are not even able to participate in rounds and if they can participate, they can only handle a small number of slots.

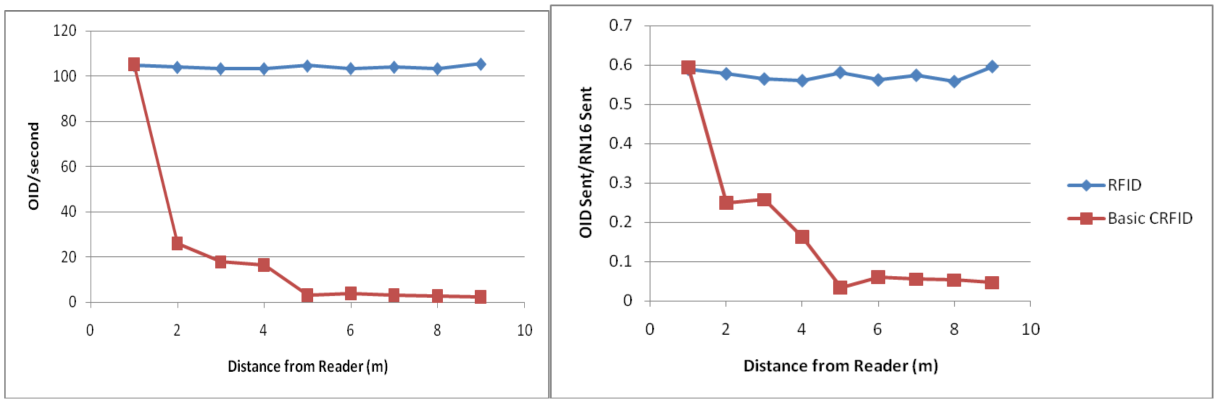

Figure 2: *Throughput (left) and success rate(right) of 36 tags clustered together at various distances*

Note that we also tested with clusters of 4 tags, 6 tags, and 72 tags, which had very similar graphs. (An interesting note is that with fewer tags, the drops are less severe because the number of QueryRepeats that need to be handled are very small.)

We looked at the collisions and they are actually lower than regular RFIDs because of the lack of participation. Therefore, we deemed the drop in success rate to be the main problem and attempt to improve this metric using *SlotPredict*. Figure 3 shows that we see a significant improvement in throughput and success rate when there are only a few tags. However, once we increase the tag number, there is actually no performance improvement, or worse performance. The problem is that we introduced collisions, shown in Figure 4.

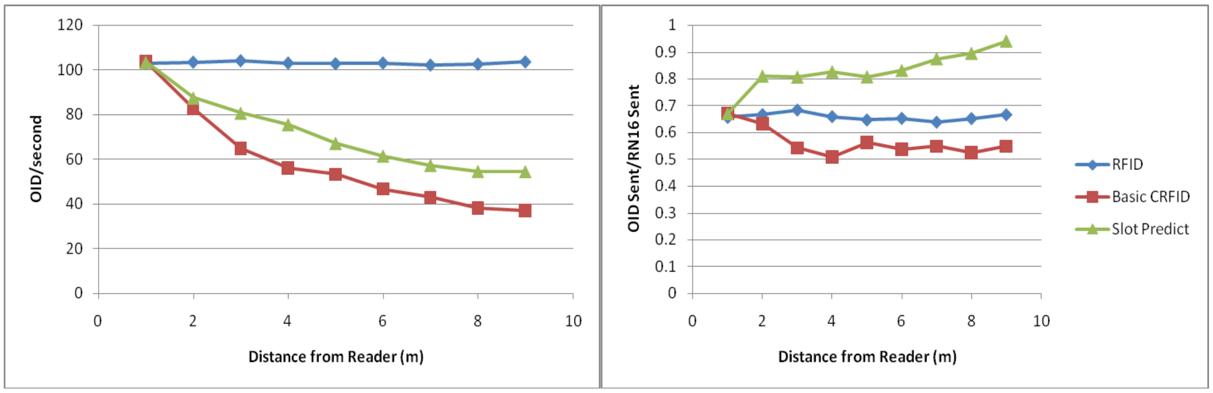

Figure 3: *Throughput (left) and success rate (right) of 4 tags clustered at various distances*

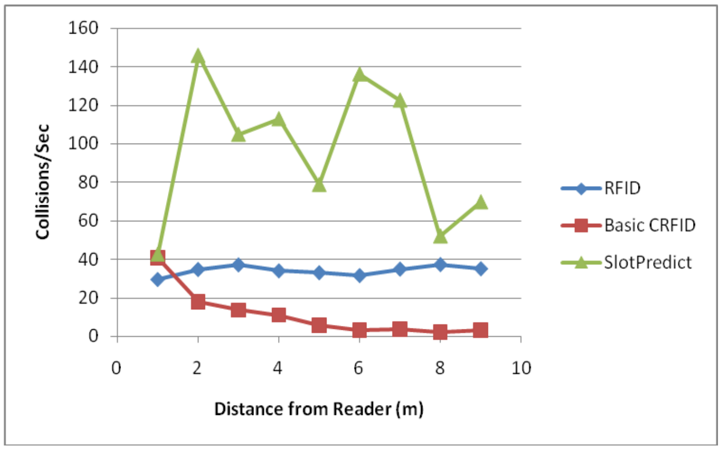

Figure 4: *Collisions per second of 72 tags clustered at various distances*

To combat the high collisions and still try to improve the success rate, we try the *SlotPartition* algorithm, the idea being, with less energy we still choose the lower range, but spread it out a little and occasionally choose large slots. For large number of tags, this had the desired effect of dropping collisions, but failed to raise the success rate, as can be seen in Figure 5. The problem was that once tags have little energy, waiting any longer than what they can handle is essentially a failure. We note that there are some improvements when the tags were farther away since low energy tags are pushed to the earlier slots and have a higher likelihood of succeeding, but the improvement is very minimal since most tags are still in slots that are too far out.

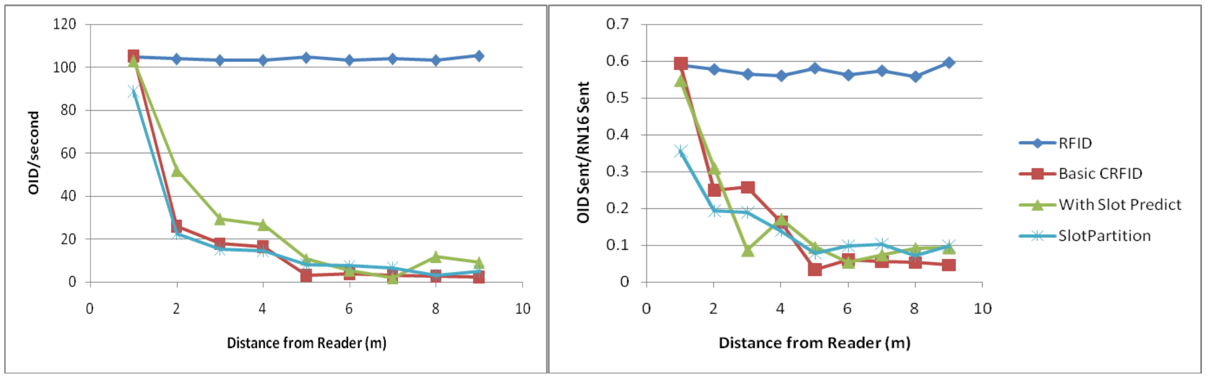

Figure 5: *Throughput (left) and success rate (right) of 36 tags clustered at various distances*

To see if *RevSlotPartition* indeed give low energy tags more time to gather energy by taking advantage of transmitting high energy tags first, we measured its performance. Figure 6 shows that we are not able to see advantages with this algorithm; we continue to see low throughput and even though the success rate improve for closer tags, the farther tags have no change and the overall success rate is still poor. The problem was that the high energy tags operated as normal, but the time gap in between the high energy tags and the low energy tags was too great for the low energy tags to take advantage of the extra time to gather energy – the extra energy was drained away during this inactive period.

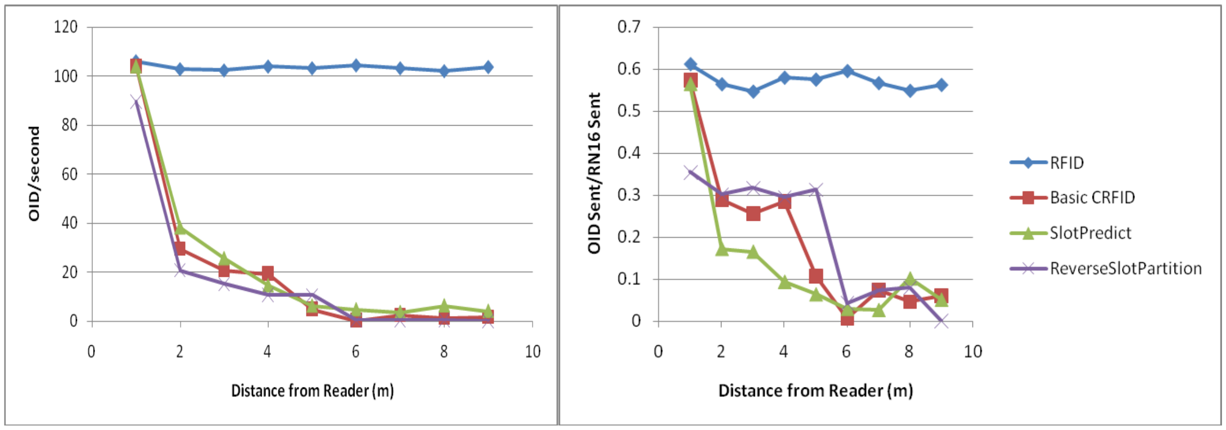

Figure 6: *Throughput (left) and success rate (right)of 72 tags clustered at various distances*

One more interesting thing we discovered in our testing was that distance was a key factor in a tag's ability to succeed. In Figure 7, we spread 36 tags across the space with 4 tags at every meter. This is opposed to clustering all of the tags at one single distance, as we have been doing previously. As you can see in the figure, all of the tags 1 meter away have a very high success rate, but all of the tags from 2 meters and farther away have a very low success rate. We called this phenomenon the *curse of distance.*

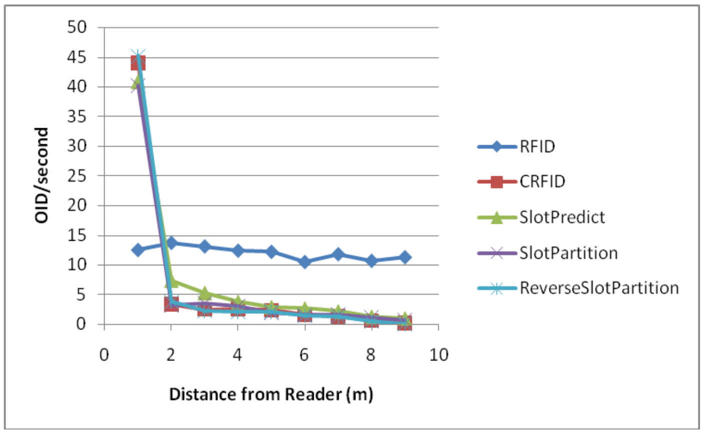

Figure 7: *Throughput of all 36 tags where 4 tags are placed at each distance from 1meter to 9meters*

Learnings

The primary learnings that the authors attained throughout this project was an understanding of RFIDs and CRFIDs. Both authors began this project with virtually no knowledge about RFIDs or CRFIDs and chose to work on this project in order to gain some understanding about the technology, with the hope of being to use the knowledge and work with RFIDs or CRFIDs in the future. Concerning this goal of learning more about the technology, we have to give a warm thanks to Michael Buettner who advised us at least one day a week, guiding us with our project and our understanding step by step.

We also learned much about how RFIDs are simulated. The library we adopted included actual libraries used in real systems, such as the LLRP library to communicate with the readers. But, in learning through this simulator, we learned there is a major risk with adopting research code. The code was broken at first, unable to even compile. Furthermore, there were many configurations and files that were undocumented and many hours were spent trying to interpret what the simulation designers meant. Furthermore, many of expected behaviors of the system were not even implemented, partially broken, or implemented in a very unintuitive manner.

Through reading about (C)RFIDs and discussing with others about how they operate, we also learned the various uses of CRFIDs. For example, using it to get various sensor readings in a building or tracking objects.

Regarding the project results, themselves, we learned that there is, indeed, a major problem with basic CRFIDs using the RFID protocol, primarily due to the poor choices by tags of when to transmit. This leads us to believe that adding a power heuristic to the standard RFID protocol when using CRFID tags benefits the operation of CRFID tags significantly and should be evaluated with real devices (as opposed to a simulation) for use in the real world. The methods that showed the greatest promise were to have low-energy tags purposefully choose lower slots improve their success rates, but this is not to say that we cannot take some calculated advantage of moving high-energy tags to earlier slots. All of these methods were really a balance between three main tradeoffs: improving collisions per second, improving success rate, and increasing the aggressiveness of tag participation. For example, if you wanted high participation with a high per tag success rate, this comes at the cost of more collisions and a lower overall success rate when using more tags. Lastly, we learned that the environment also has to be taken into account, as it can have a severe impact on the protocol and one's ability to find an ideal protocol. For example, the performance of each of the algorithms implemented in this project was impacted by the *curse of distance* and the number of tags (ex. *SlotPredict* works pretty well with a low number of tags, but not a high number).

Future Directions

One of the future directions is to model the capacitor more accurately. In reality, accumulation actually begins super-linear and then slows down as it approaches the max. Similarly, when discharging, it drops faster than linear than slows down as it approaches zero. We believe that the simplification made in this project should not affect the results much, but without an actual implementation it is hard to tell.

In this project, we tested three possible adjustments to the protocol and they showed that making these types of changes can definitely help avoid unnecessary transmits and increase success rate. However, by doing so could cause the overall throughput to drop. With some careful calibrations and possibly a different way to even choose the slot count, we believe we can get performance much closer to the normal RFID case. For example, some things to also consider include the energy accumulation rate where a next step with *SlotPredict* is to have it factor the accumulation rate into its slot calculation. Another thing to consider is that instead of a random number generator, we could use a pseudo random number which can help avoid collisions between tags the same distance away.

With some more simulation, a natural follow-up step is to take one of the successful protocols and implement it into real CRFIDs and see how they perform in reality. In simulation, things could look just fine, but in reality there are many more factors that could affect the performance, such as noise, interference of objects, etc.

One piece we did not consider was what we can do with the reader. Currently, in most setups, people are adding CRFIDs to existing readers (readers are very expensive, so replacing is undesirable), but there could be a time when new ones will be ushered in. Research into how readers could aid in increasing the efficiency of the system would beneficial. The readers have much more power to use for computation than tags.

Lastly, there is much to be said about the actual simulator. We had to make many “hacks” to bend the system to perform the way we expected. This simulator needs to be cleaned up much more before it can be really used well for simulating a CRFID setup and provide very deep insights into the behavior and mimic reality as close as possible.

References:

[1] Pacemakers and Implantable Cardiac Defibrillators: Software Radio Attacks and Zero-Power Defenses. Halperin, Heydt-Benjamin, Ransford, Clark, Defend, Morgan, Fu, Kohno, and Maisel. IEEE Symposium on Security and Privacy 2008.

[2] RFIDSim. http://trac.assembla.com/RFIDSim/.

[3] An Empirical Study of UHF RFID Performance. Buettner, Michael and Wetherall, David. MobiCom 2008.