Wireless Routing in Hostile Environments

Jake Garrison (1126716), James Goin (1137690), Tyler Williamson (1265614)
December 13, 2014
EE 418 Network Security and Cryptography
Dept. of Electrical Engineering, University of Washington

Introduction

Wireless Routing

Initially a network of size N = 250 nodes was deployed over a square area of width 2500m. Each of theses nodes must be within 400m in order to form a link. An example of this is shown in the figure below.

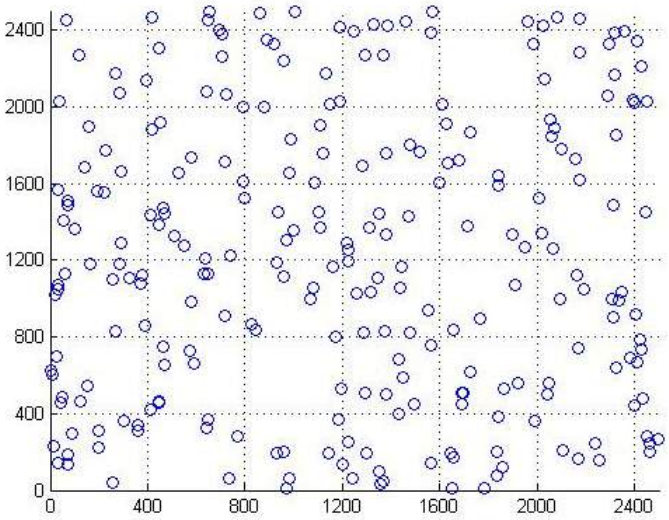

Figure 1: Random Node Distribution

Keys were then given with a key pool size of 1000 and 30 keys per node. For each pair of nodes the linked key vulnerability metric was calculated to quantify the effect of security loss due to duplicate shared keys in the network. The above plot and initial parameters were generated in the "init graph" matlab script.

$$V(i,j) = \sum (i = 1T \subseteq k_{ij} \frac{(-1)^{|T|+1} N}{N_T}) \tag{1}$$

where N_T is the number of nodes with at least one key in set of keys T. Since, during a node capture attack, keys that appear with great frequency in the network will be captured first by an adversary, this metric is needed to help identify frequent keys. After running nodes through this metric (see LKVM script), 30 pairs of nodes where randomly chosen and their shortest paths with respect to the threshold were computed.

In order to find the shortest path between nodes so the signal doesn't drop out, a shortest path algorithm is utilized to link two nearby nodes with a common key (see "setup paths" script). Additionally, nodes further than 400m apart will not be considered as a pair since 400m is the limit on the distance for reliable communication[3]. In order to ensure the paths are resilient to attack, two separate security metrics are

observed and presented below; Threshold Performance Vulnerability Metric (TPVM) and the Weighted Linear Performance Vulnerability Metric (WLPVM).

$$g(i,j) = \begin{cases} 1, & V(i,j) > \tau \\ \infty, & \text{else} \end{cases} \qquad h(i,j) = 1 + \lambda \frac{1}{V(i,j)}$$

TPVM WLPVM

In TPVM, links with the vulnerability metric, $V(i,j)$, exceeding a certain threshold, τ, are considered by the routing protocol, while links below the threshold are given infinite cost weight, and will therefore be ignored. This metric is based on the idea that, since compromise of a single link will lead to the capture of all traffic passing through that link, the overall security of a path will be governed by the security of its weakest link. Guaranteeing a certain security level for a path is therefore equivalent to placing a lower bound on the security of the weakest link [3].

In WLPVM, links are all scaled by multiplied by the inverse of $V(i,j)$, the output of the link vulnerability matrix. Unlike TPVM which is a binary threshold, WLPVM doesn't exclude any nodes by scaling them to infinity. This results in a much softer, less predictable weight.

The plots below show how sweeping either τ, or affects capture cost and hop count (generated from the "TPVM plot" and "WLPVM plot" scripts). The graphs were generated plotting the average change in hop count or cost over ten randomly generated trials. Capture cost being defined as the average amount of node captures to compromise a path, and hop count being average number of hops in a shortest path between random points. These plots are similar to figures 2(b,d) from [3].

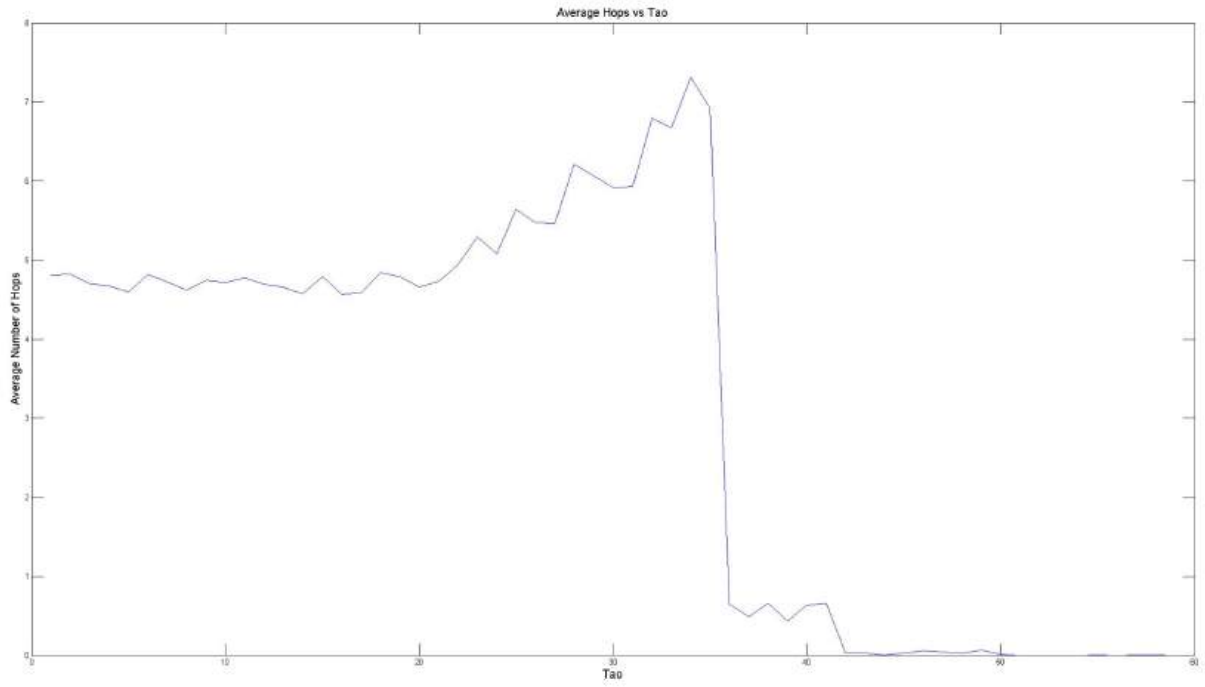

Figure 2: Average Number of Hops Vs. τ for TPVM Metric

As shown in the figure above the average number of hops is stable at approximately five for τ from zero to twenty then increases to about seven from twenty to thirty-six then drops to zero. Based off of this, the optimal τ is between 20 or 30, which is right before the curve drops to 0. Choosing the threshhold becomes a compromise between security and efficiency. With more hops taking longer but offering a more secure path.

Since the threshold, τ, determines how many links are ignored due to key frequency, it makes sense that at too high of a threshold, the hops drop to 0 since communication is no longer possible.

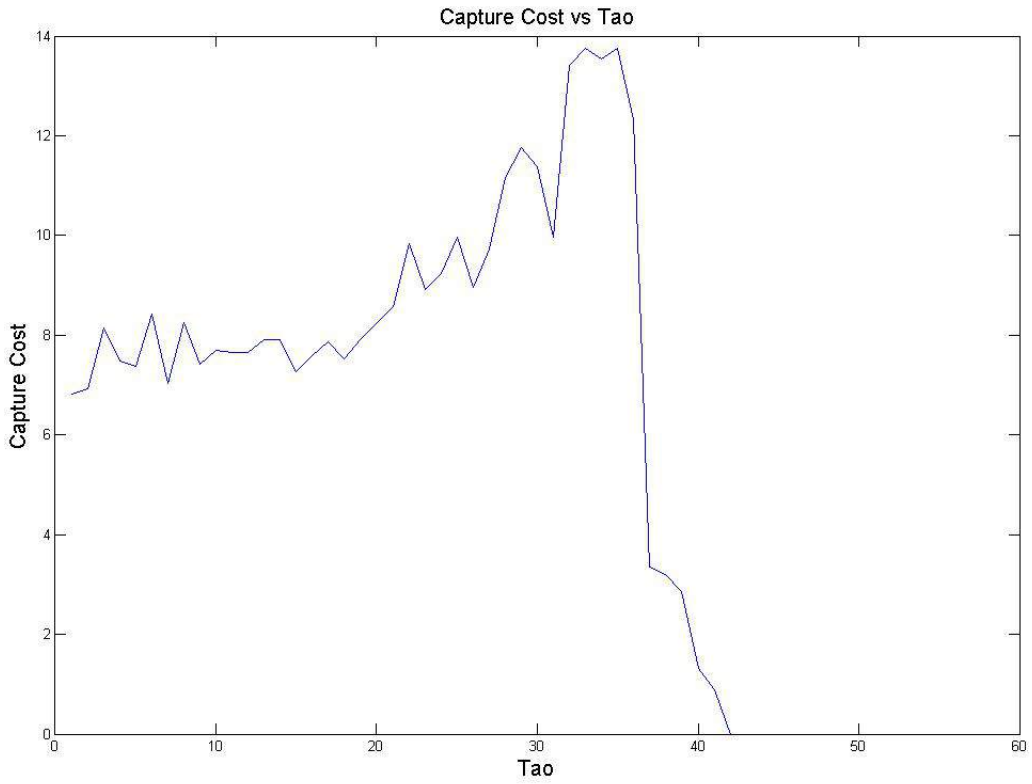

Figure 3: Capture Cost Vs. τ for TPVM Metric

Similar to comparing number of hops to τ, the capture cost is stable around 8 for τ between 0 and 20, then drops off as τ approaches 40. The fact that both parameters drop off around $\tau = 40$ further enforces optimizing τ to around 25. This tells us that the average number of compromises to compromise a route increases with τ, but the drops to 0 after τ exceeds 40. This means for optimal security, one must choose a τ just before the curve falls to zero. This plot turns out to be nearly identical to figure 2(d) from [3].

The plots for TPVM have shown that as the amount of hops increases, the security of the path is enhanced as well, thus increasing the capture cost. Raising τ will ignore some insecure links, in turn making the hop count and capture cost increase as well and resulting in a more secure system. At larger values of τ it becomes harder to make nodes with neighbors. Eventually, there will be no pair-able nodes within the 400m radius and no path will be made so the hop count goes to zero. In our analysis, this occurs around when $\tau = 40$. At this point, too many links are ignored resulting in little or no communication, shown as a 0 on our plots.

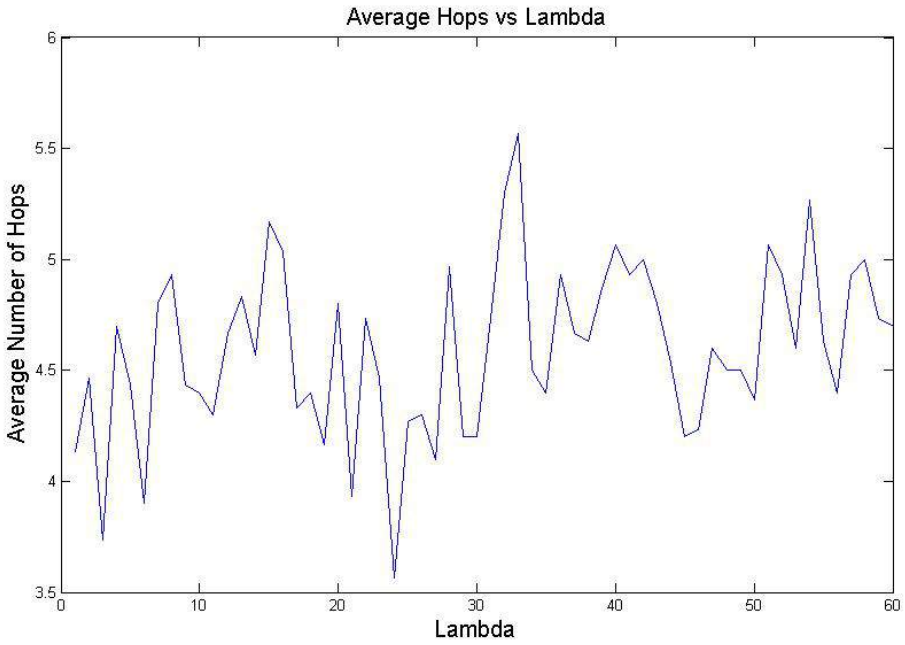

Figure 4: Average Number of Hops Vs. λ for WLPVM Metric

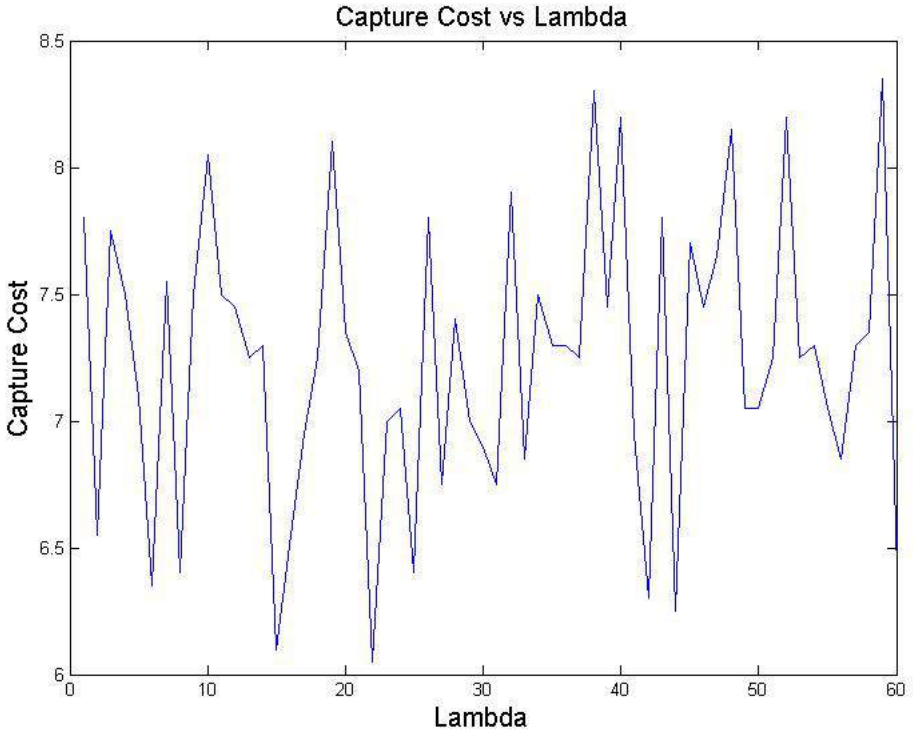

Figure 5: Capture Cost Vs. λ for WLPVM Metric

The Weighted Linear Performance-Vulnerability Metric as shown above, had very unpredictable behavior as λ increased. Based off our data there is no trend to altering λ, which suggests it is an unreliable metric when compared to TPVM. Averaging out the noise, however, does reveal a slight linear upward trend as

lambda increases in both hops and capture cost, which implies security does indeed go up. Plotting a larger range of λ may reveal an eventual drop off point, similar to TPVM, but our data was so variable in every trial, that this wasn't further explored. Due to the fact that WLPVM isn't mentioned in any papers and is new to this assignment, it may have some kinks to work out before it is reliable enough to draw conclusions from.

Questions:

1. There exist other key predistribution schemes. Read Chapter 10, Section 4 of Stinson, 3rd ed, which describes the Fiat-Naor and Mitchell-Piper key predistribution schemes. Then complete Chapter 10, Exercise 6 of Stinson.

   Question 10.6 - In Section 10.4.2, we describe a probabilistic approach which provided an existence result for (t,w)-CFF(v, n). It is in fact possible to modify the approach to yield a practical algorithm to construct certain (t,w)-CFF(v,n) with high probability. We explore this approach in this exercise. Throughout this exercise, let X be a randomly constructed v by n matrix, having entries in 0, 1, where each entry is defined to be 1 with probability p. Finally, define $p_{t,w}$ as in Section 10.4.2.

   (a) Prove that the probability that M is not the incidence matrix of a (t,w)-CFF(v,n) is at most Exp[X].
   First Note: $Exp[X] = \sum_{i=0}^{\infty}(Pr[X = i] * i)$ which is given in the book
   Therefore it can be clearly shown that: $Exp[X] \geq \sum_{i=1}^{\infty}(Pr[X = i]) = Pr[X > 0] = Pr[$M is not a CFF$]$

   (b) Prove that $Exp[X] < 2^{-s}$ if $v > \frac{(t+w)log_2 n+s}{-log_2(p_{t,w})}$
   First rearrange the formula given for v to get:

$$\begin{aligned} -log_2(p_{t,w})v &> (t+w)log_2 n + s \\ -log_2(p_{t,w})v - (t+w)log_2 n &> s \\ log_2(p_{t,w})v + (t+w)log_2 n &< -s \\ n^{t+w}(p_{t,w})^v &< 2^{-s} \end{aligned} \tag{2}$$

   This can be compared to the equation given in the book: $Exp[X] < n^{t+w}(p_{t,w})^v$ if $v > \frac{(t+w)log_2 n}{-log_2(p_{t,w})}$ where $p_{t,w} = 1 - \frac{t^t w^w}{(t+w)^{t+w}}$

   Which proves: $Exp[X] < n^{t+w}(p_{t,w})^v < 2^{-s}$

   (c) Suppose t = 2, w = 1 and n = 100. How big should s and v be in order that there is at least a 99% chance that a randomly constructed M is a (2, 1)-CFF(v, 100)?

   $p_{t,w} = 1 - \frac{t^t w^w}{(t+w)^{t+w}} = p_{2,1} = 1 - \frac{2^2 1^1}{(2+1)^{2+1}} = \frac{23}{27}$
   $n^{t+w}(p_{t,w})^v < 1$ Given in book
   $log_2(p_{t,w})v + (t+w)log_2 n < 0$ $v > \frac{(t+w)log_2 n}{-log_2(p_{t,w})} = v > \frac{(2+1)log_2 100}{-log_2(\frac{23}{27})} \approx 86$

   *How would you expect the performance of this scheme to differ under an attack when compared to the random key predistribution scheme?

   By using a (t,w) cover-free family distribution, such as the Mitchell-Piper key distribution, there is a key for every group of t users, and each such key is secure against any disjoint coalition of at most w users. When using the random key distribution it is possible for an adversary to compromise more than one node by capturing a single node's key set due to overlap. This is due to the fact that the random key distribution has no restricitons on on assigning two users the same set of keys.

2. The performance of the random key predistribution scheme depends upon the key distribution parameters (i.e. the number of keys given to each node and the total number of nodes). Using the results of

[1], estimate how many keys should be given to each node under the following conditions:

This can be computed through combining equations from [1] and [3] for probability that two nodes are in the same region and that two nodes share at least one key. Where N = number of nodes, P = size of pool of keys, k = number of keys, c is computed from Equation 3, r = the maximum distinct between two nodes, A = the area deployed over. By using the given information, plugging into the following set of equation and solving for k the folling values were computed.

$$P_c = lim_{n\to\infty} Pr[G(n,p) - is - connected] = e^{-e^{-c}} \tag{3}$$

$$P = \frac{ln(N)}{N} + \frac{c}{N} = Pr[two{-}nodes{-}are{-}connected] = Pr[same{-}region] * Pr[share{-}at{-}least{-}1{-}key] \tag{4}$$

$$Pr[same - region] = 1 - e^{(\frac{-N * pi * r^2}{A})} \tag{5}$$

$$Pr[share - at - least - 1 - key] = 1 - \frac{(1 - \frac{k}{P})^{2(P-k+.5)}}{(1 - \frac{2k}{P})^{P-2k+.5}} \tag{6}$$

(a) A network of 100 nodes with range 500m, deployed over an area of 1500x1500m, with a key pool of size 1000.
21 keys

(b) A network of 1000 nodes with range 100m, deployed over an area of 1000x1000m, with a key pool of size 1200.
32 keys

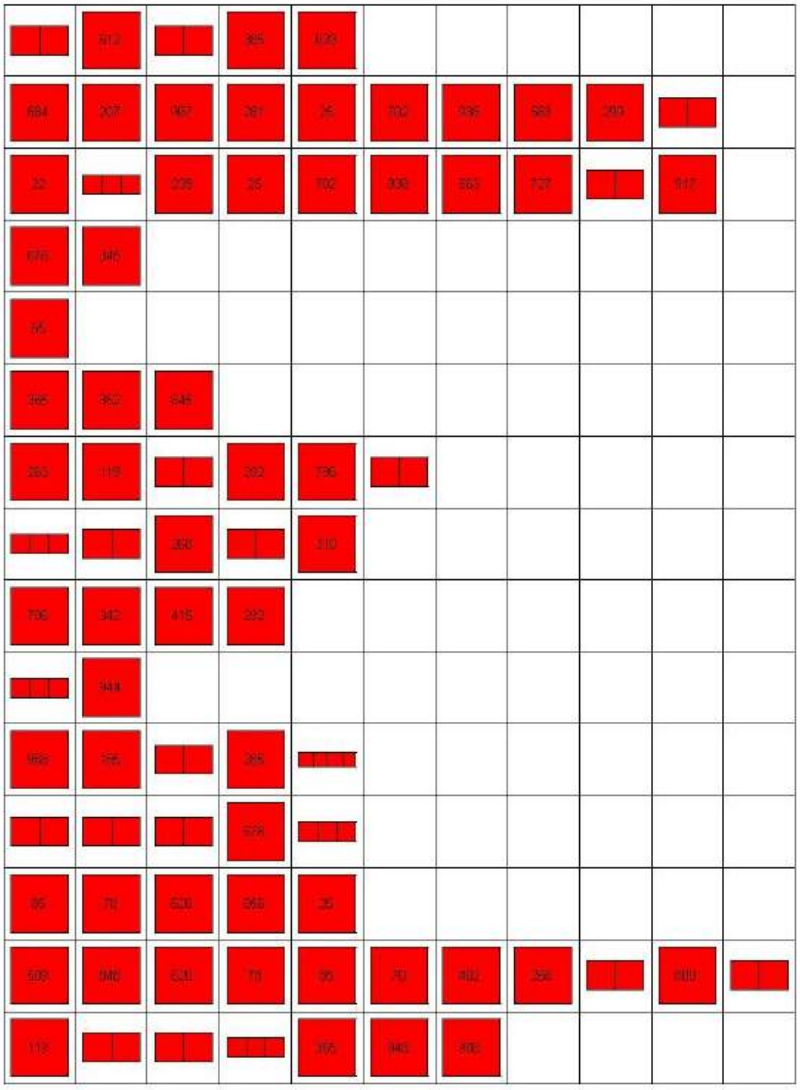

Figure 6: Path Visualization

The path visualization above maps out shortest path for different node links, and also reveals the set of shared keys between any two given nodes. This is generated in out "setup paths" script.

Network Vulnerability Analysis

Questions:

1. For the network diagram in Figure 1 compute the effective resistance metric between nodes s and d by hand. Show all steps.

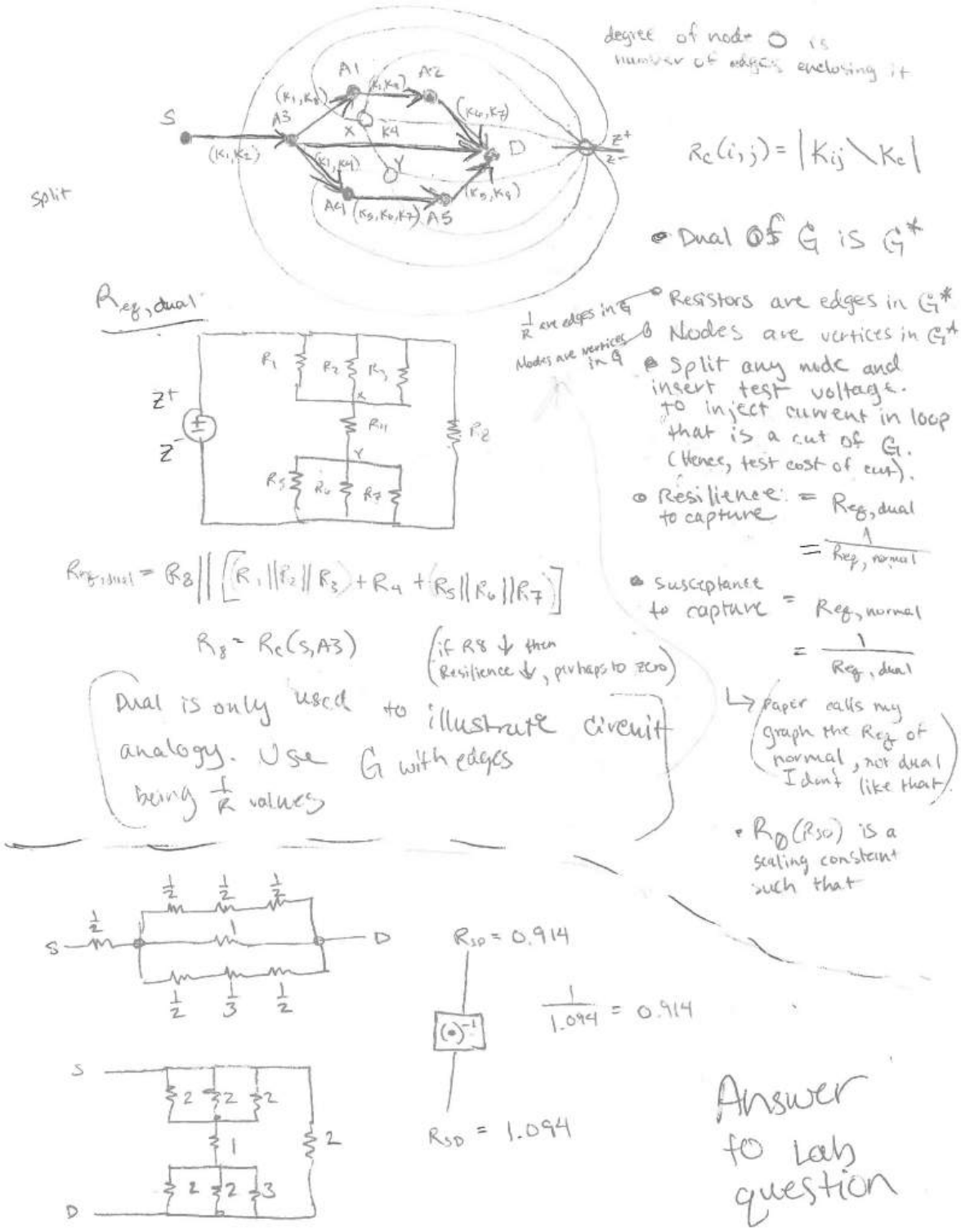

Figure 7: Effective Resistance Metric By Hand

2. Using the simulation parameters from Part 1, generate a network and choose paths between 10 pairs of nodes. Simulate an attack on the network using the following strategies:

(a) Random node capture:

(b) The GNAVE algorithm using the set-theoretic route vulnerability metric, described in Section 4.1 of [4]

(c) The GNAVE algorithm using the circuit-theoretic route vulnerability metric, described in Section 4.2 of [4].

The Random Node capture strategy captures nodes in random order (demonstrated in the attached script random_node_capture). As nodes are captured their keys are stored and used to determine which paths are compromised. The attack succeeds once all paths are compromised. A link is compromised when all keys protecting the link are compromised. A path is compromised if any of its links are compromised. For both the set and circuit theoretic vulnerability strategies use metrics for determining nodes to compromise. These are demonstrated and explained through the scripts gnave_circ.m and gnave_set.m. The results are shown in Figure 9.

Compare the results of each of these attack strategies. You may assume that the cost w_i of capturing nodes is uniform and that the source and destination nodes cannot be captured.

Random Node Capture

Number of nodes captured to compromise 10 paths

Graph	Attack Qty	Mean	Std. Dev
1	1000	34.197	19.645
2	1000	33.652	19.786
3	1000	33.221	19.943
4	1000	32.561	17.992
5	1000	34.892	19.818

GNAVE - Set Theoretic RVM

Number of nodes captured to compromise 10 paths

Graph	Attack Qty	Results	Mean	Std. Dev
1	5	4,4,6,4,4	4.4	0.894
2	5	5,4,4,4,4	4.2	0.447
3	5	4,4,4,6,4	4.4	0.894
4	5	5,5,4,4,3	4.2	0.837
5	5	4,6,4,4,4	4.4	0.894

GNAVE - Circuit Theoretic RVM

Number of nodes captured to compromise 10 paths

Graph	Attack Qty	Results	Mean	Std. Dev
1	5	3,5,3,4,4	3.8	0.837
2	5	5,4,3,4,4	4.0	0.707
3	5	4,2,4,5,4	3.8	1.095
4	5	3,5,5,4,3	4.0	1.000
5	5	4,3,5,5,4	4.2	0.837

Figure 9: Comparison of Attack Strategies

It can clearly be shown through the above table that both GNAVE Set and Circuit attack strategies were immensely more effective than random node capture. Both GNAVE attack methods only need four nodes on average captured in order to compromise ten paths, while the random node capture attack needed thirty four nodes on average. The random node capture not only takes an immense number more nodes captured in order to compromise the paths but also was unreliable with a standard deviation close to twenty which can drastically change the effectiveness. Both GNAVE attacks had a standard deviation of close to

one, demonstrating high reliability. One major draw back of the GNAVE attacks however is the amount of time taken in order to compute. Both of these took about a minute to compute per cycle, while the random attack was seconds. Both GNAVE attacks had very similar results with Circuit attack being just slightly more effective, however with the small set of results taken due to immense time needed it is hard to say which is truly more effective.

Conclusion

In conclusion, we generated a network of nodes based of the given parameters, computed LKVM for each node pair, found the shortest path for a smaller random subset of the network, then simulated a node compromise attack on the network to expose the keys of each link. In an attempt to increase the security of the network, metrics such as the TPVLM and WLPVM were employed to observe the effect they had on average hop count and capture cost during the attack simulation. For TPLVm we found as in paper [3] raising τ increases both the hop count and capture cost, resulting in a more secure network. There was however a upper limit to τ that resulted in both capture count and average hops approaching 0 due to the threshold being too strict and nodes being unable to pair. Analysis of the WLPVM metric yielded less obvious results. Due to the large amount of noise in the plot (even after averaging the plot over ten trials) no substantial result or trend was obtained. Further research could go into sweeping a larger scope of λ, and developing a metric more optimized for this type of attack. A comparison of multiple attack strategies showed that both GNAVE metrics greatly improved the effectivness of nodes being compromised, however also greatly increased the ammount of time to compute.

References

1. L. Eschenauer and V. Gligor. A Key-Management Scheme for Distributed Sensor Networks. In ACM Conference on Computer and Communications Security (CCS), 2002.

2. A. Clark and R. Poovendran A Metric for Quantifying Key Exposure Vulnerability in Wireless Sensor Networks. In IEEE Wireless Communications and Networking Conference (WCNC), 2010.

3. A. Clark, R. Hardy, and R. Poovendran . A Joint Performance-Vulnerability Metric Framework for Designing Ad Hoc Routing Protocols. In IEEE Military Communications Conference, 2010.

4. P. Tague, D. Slater, J. Rogers, and R. Poovendran. Evaluating the Vulnerability of Network Traffic Using Joint Security and Routing Analysis. In IEEE Transactions on Dependable and Secure Computing, 2009.