ECE 4607 Final Project

Paras Chetiwal
Alex Davidson
Tej Koindala
Pratik Sharma
Kenji Tanaka

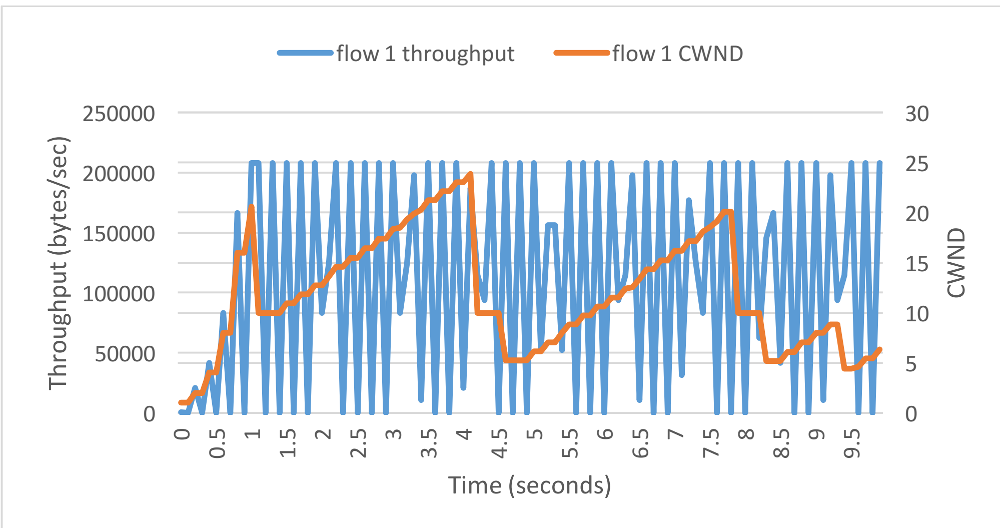
Part 1:

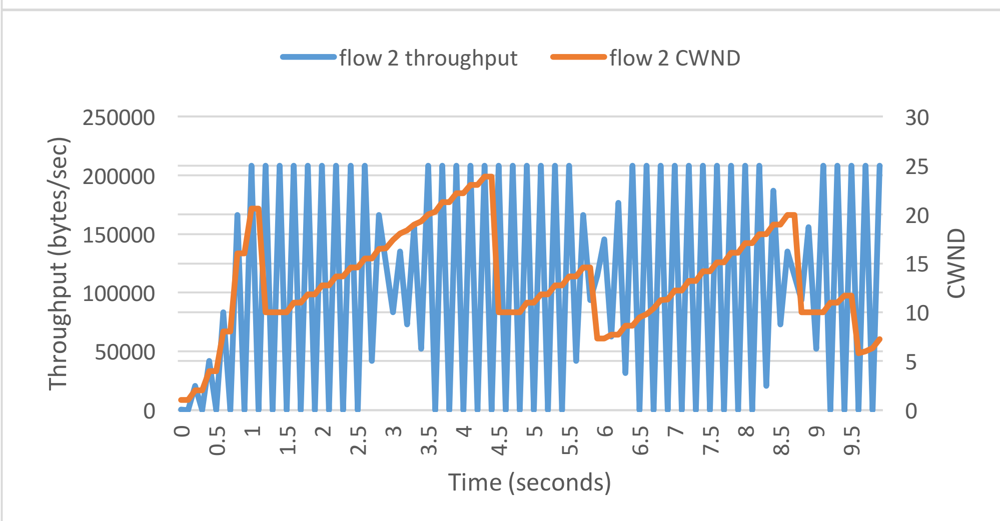
Total number of TCP segments received and end-to-end throughput:

	Flow 1		Flow 2	
	Segments Received	End-to-End Throughput	Segments Received	End-to-End Throughput
Tahoe	10911	113464 bytes/sec	10291	107016 bytes/sec
Reno	10911	113464 bytes/sec	10291	107016 bytes/sec
NewReno	10911	113464 bytes/sec	10291	107016 bytes/sec
Sack	10911	113464 bytes/sec	10291	107016 bytes/sec

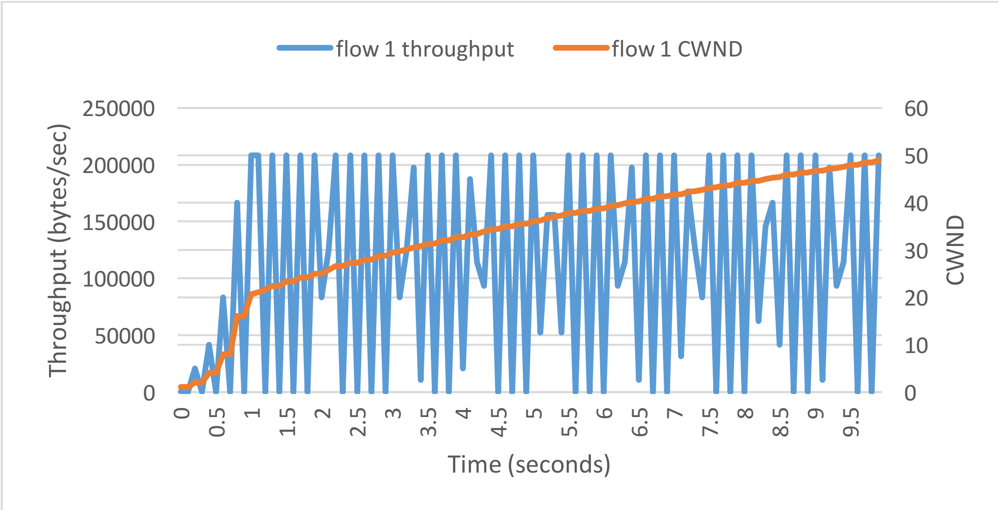
Instantaneous Throughputs (graph of first 10 seconds):

Tahoe:

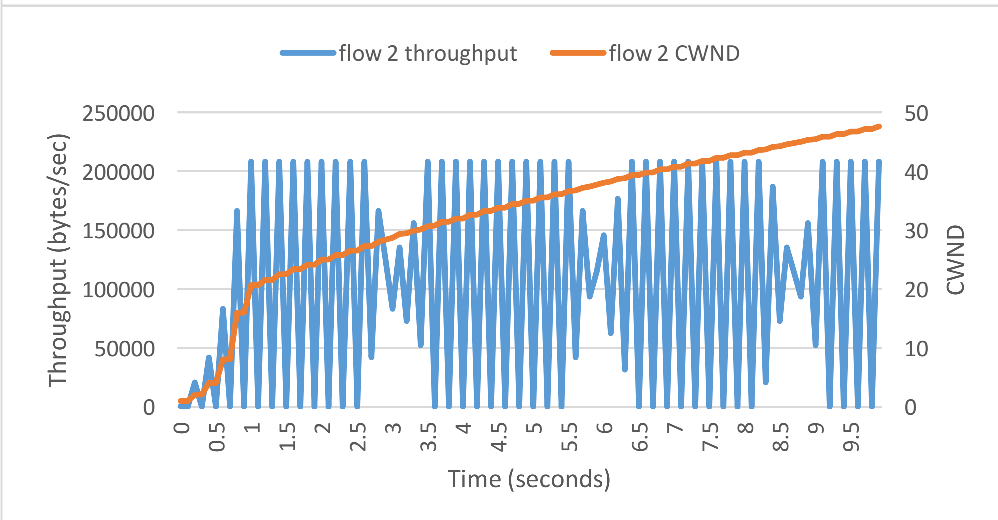




Reno:

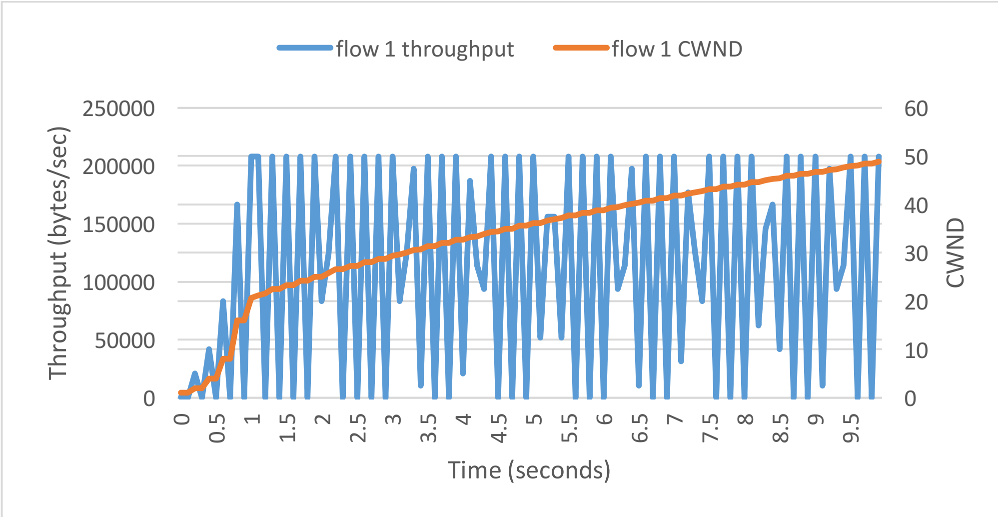


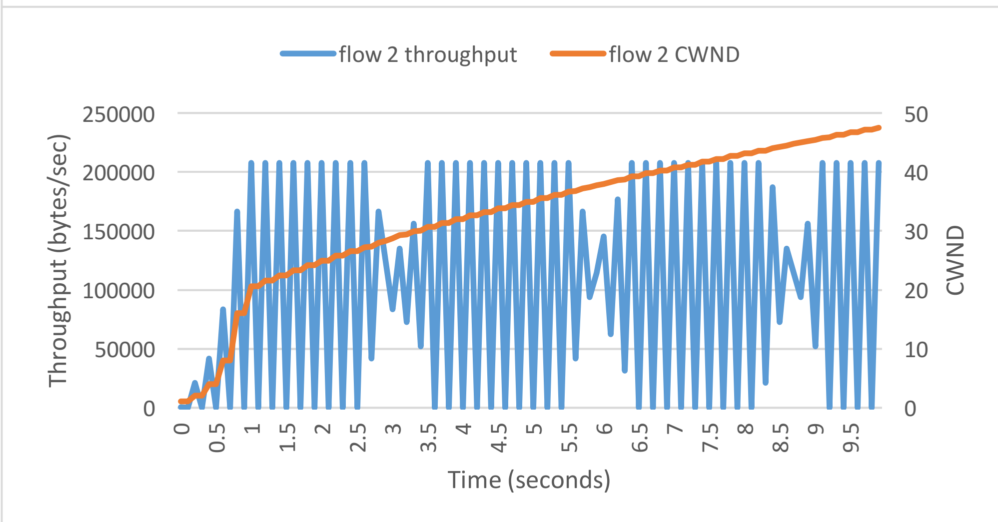
flow 1 throughput
flow 1 CWND
250000
200000
150000
100000
50000
0
60
50
40
30
20
10
0
Throughput (bytes/sec)
CWND
0
0.5
1
1.5
2
2.5
3
3.5
4
4.5
5
5.5
6
6.5
7
7.5
8
8.5
9
9.5
Time (seconds)



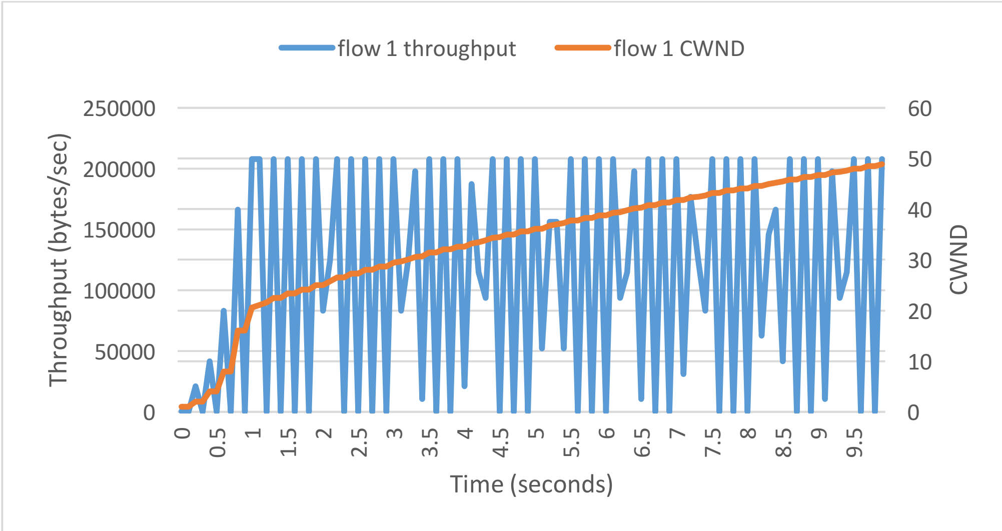
flow 2 throughput
flow 2 CWND
250000
200000
150000
100000
50000
0
50
40
30
20
10
0
Throughput (bytes/sec)
CWND
0
0.5
1
1.5
2
2.5
3
3.5
4
4.5
5
5.5
6
6.5
7
7.5
8
8.5
9
9.5
Time (seconds)

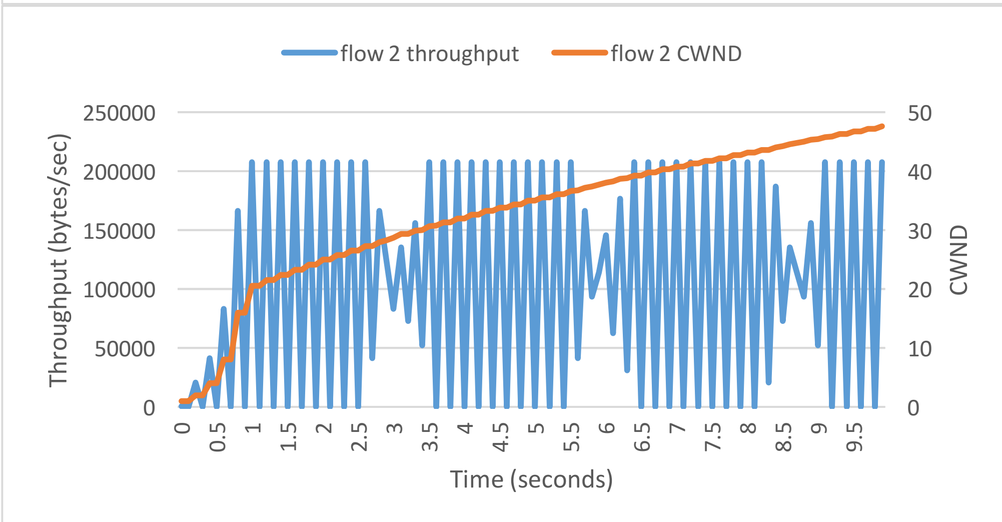
NewReno:


flow 1 throughput
flow 1 CWND
Throughput (bytes/sec)
CWND
Time (seconds)
250000
200000
150000
100000
50000
0
60
50
40
30
20
10
0
0 0.5 1 1.5 2 2.5 3 3.5 4 4.5 5 5.5 6 6.5 7 7.5 8 8.5 9 9.5


flow 2 throughput
flow 2 CWND
Throughput (bytes/sec)
CWND
Time (seconds)
250000
200000
150000
100000
50000
0
50
40
30
20
10
0
0 0.5 1 1.5 2 2.5 3 3.5 4 4.5 5 5.5 6 6.5 7 7.5 8 8.5 9 9.5

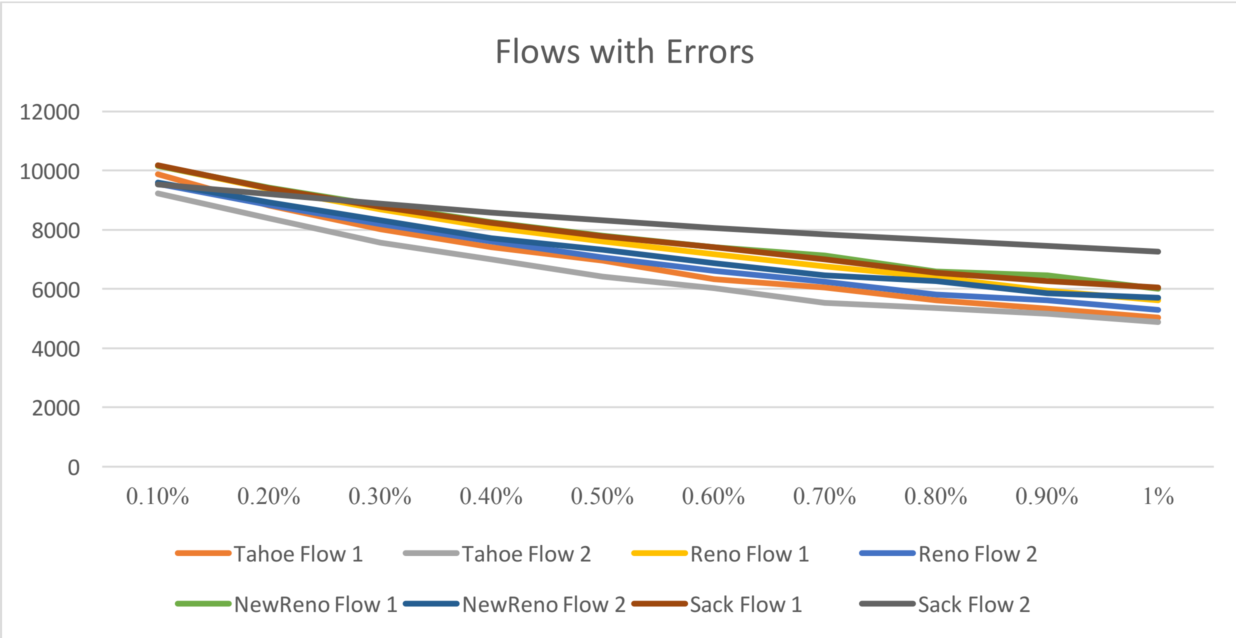
Sack:





Averaged Throughput with Loss Module:

Loss	Tahoe		Reno		NewReno		Sack	
	Flow 1	Flow 2	Flow 1	Flow 2	Flow 1	Flow 2	Flow 1	Flow 2
0%	113464	107016	113464	107016	113464	107016	113464	107016
0.1%	9877	9224	10146	9560	10157	9587	10182	9532
0.2%	8824	8388	9383	8837	9423	8915	9399	9206
0.3%	8008	7551	8675	8197	8819	8313	8779	8887
0.4%	7404	6998	8075	7601	8246	7705	8236	8585
0.5%	6955	6408	7609	7072	7796	7318	7787	8312
0.6%	6328	6035	7179	6618	7403	6865	7407	8065
0.7%	6055	5527	6763	6247	7135	6467	7007	7848
0.8%	5613	5353	6386	5820	6591	6263	6541	7644
0.9%	5328	5167	5950	5616	6461	5861	6265	7457
1%	5029	4879	5616	5301	5998	5698	6056	7269



Discussion:

Without loss, the CWND can increase indefinitely. This leads for extremely high throughputs for all protocols. As loss is added, we can see the results of TCP Reno and NewReno making direct improvements on TCP Tahoe when packet loss is added: in the case of packet loss, TCP Reno and NewReno simply half the CWND whereas TCP Tahoe has to set CWND to one and go back to the slow start algorithm. One can therefore see an increase in throughput throughout the error spectrum.

One can also see the improvements of the TCP NewReno in this simulation. As the probability of errors increases, the fast recovery algorithm is again improved, so throughputs are even higher as compared to TCP Reno.

Finally, for TCP SACK, we can see performance about equivalent to NewReno for Flow 1, which is not entirely surprising as TCP SACK and NewReno essentially set out to quickly fill holes in sequences. However, TCP SACK's performance for flow 2 is considerably higher than any other flow at high error rates. This likely because TCP SACK has more explicit information about which packets are lost, allowing it to better adapt to higher error rates whereas TCP NewReno only knows for sure the first packet lost.

Part 2:

1. **IP Addresses for:**
    a. Local EH: 192.168.1.187
    b. Remote EH: 74.125.212.177
2. **Port numbers for:**
    a. Local EH: 4070
    b. Remote EH: 80
3. **Maximum segment size used in either direction:**
    a. Max up: 1260 bytes
    b. Max down: 1260 bytes
4. **Total and unique bytes sent by Remote EH:**
    a. **Total bytes sent by Remote EH:**
        i. Actual data bytes: 38785361 bytes (Data packets = total packets – 15)
    b. **Unique bytes sent by Remote EH:**
        i. 38719841 bytes
5. **Average downlink throughput:**
    a. Unique packets = actual packets – retransmit packets
    = 30785 – 52
    = 30733
6. **Average downlink throughput (calculated value by hand):**
    a. Total (actual) data bytes sent: 38719841
    b. Data transmit time: 111.603s
    c. Actual bytes/sec = 347529.73 bytes/sec = 347.5 KBps
7. **Average RTT in either direction:**
    a. Avg. RTT @ Local EH: 30.8ms or 30.9ms
    b. Avg. RTT @ Remote EH: 49.9ms or 49.8ms
8. Minimum end-end RTT:
    a. Local Host: 30.6ms
    b. Remote Host: 49.1ms
9. TCP flavors used by both end-hosts:
    a. TCP SACK
    b. Both end-hosts ‘req sack’ so we can safely assume that both are using TCP SACK
    c. A paper by Jaiswal et al.[1] suggests observing how the CWND changes after losses to determine TCP flavors. The CWND simply inferred by counting ACKs and estimating the initial CWND by counting the number of packets before the first observed ACK. Therefore, in general, it is possible to identify TCP flavors by how they respond to losses.
10. Maximum congestion control window size used by the the remote end-host:

[1] S. Jaiswal, G. Iannaccone, C. Diot, J. Kurose and D. Towsley, "Inferring TCP Connection Characteristics Through Passive Measurements", 2004.

a. 'max win adv' = 8896 bytes = 8.896 KB
b. Max owin (@ remote) = Max adv. Window (@ local) = 65521 bytes

Part 3:

1.)

- 802.11 standards supported by your Wi-Fi card: **6.802.11 a/b/g**
- SSID: GTwifi
- MAC ID of wireless access point: 3c-08-f6-22-10-00
- MAC ID of Wi-Fi Card: AC-2B-6E-AA-7B-89
- Bit Rates Supported by the Access Point: 1, 2, 5.5, 11, 6, 9, 12, 18, 24, 36, 48 and 54 Mbps
- RTS/CTS is enabled
- Security is enabled. It uses WPA2-Enterprise

2.)

Klaus

	Access Points	Security Enabled?	Different Channels Used	# of APs using each Channel (Channel - # of APs)
1	SETUP	None	11	11-5
2	dd-wrt	None	6	6 - 5
3	GTwifi	Yes	161, 153, 149, 140, 112, 11, 108, 104, 1, 100, 64, 56, 6, 48, 44, 36, 40, 157, 116, 165	161 - 4 153 - 4 149-4 140-4 112-4 11-5 108-4 104-4 1-4 100-4 64-4 56-4 6-5 48-4 44-4 36-4 40-4 157-4 116-4 165-4
4	GTVisitor	None	161, 11, 157, 153, 149, 140,	161-4 11-5

			112, 165, 36, 108, 104, 100, 64, 6, 1, 48, 44, 40, 116, 56	157-4 153-4 149-4 140-4 112-4 165-4 36-4 108-4 104-4 100-4 64-4 6-54 1-4 48-4 44-4 40-4 116-4 56-4
5	GTRI-Device	WPA2-Personal	161, 153, 149, 140, 112, 108, 36, 104, 100, 11, 64, 48, 1, 44, 6, 165, 40, 116, 56, 157, 165	161-4 153-4 149-4 140-4 112-4 108-4 36-4 104-4 100-4 11-5 64-4 48-4 1-4 44-4 6-5 165-4 40-4 1164-4 56-4 157-4
6	eduroam	WPA2-Enterprise	161, 153, 149, 140, 112, 108, 36, 104, 11, 100, 64, 6, 48, 1, 44, 165, 40, 116, 157, 56	161-4 153-4 149-4 140-4 112-4 108-4

				36-4 104-4 11-5 100-4 64-4 6-5 48-4 1-4 44-4 165-4 40-4 116-4 157-4 56-4

Home

	Access Points	Security Enabled?	Different Channels Used	# of APs using each Channel (Channel - # of APs)
1	ATT2q4JTN7	WPA2-Personal	6	6-1
2	TMBHJM	WPA2-Personal	36, 165, 11	36-1 165-1 11-2
3	marmar	WPA2-Personal	1	1-1
4	HOME-DB5C	WPA2-Personal	11	11-2
5	ATT4Ine4QM	WPA2-Personal	9	9-1
6	ATT7Wqs85z	WPA2-Personal	10	10-1

I did not find more than one AP in Klaus having the same ESSID. However, it is not surprising that this is common because it is useful when the clients are roaming across various points in the building. It would be inefficient and downright troublesome to have to deal with various ESSIDs.

In a residential setting, since there is not too much space a couple access points will be more than sufficient to cover the zone of the residence, so the clients don’t experience a low signal. This of course is possible if there is a seamless transition between the access points with a low handoff time which consists of scanning, authentication, and re-association.

In a commercial setting, this becomes a much more complicated issue. Wifi deployment in high density needs to address the capacity of the area, estimate the numbers of access points, and also be concerned with access point placement. So obviously there are many more constraints to factor in a high density setting where there could be numerous access points with 40-50 devices connected to each.