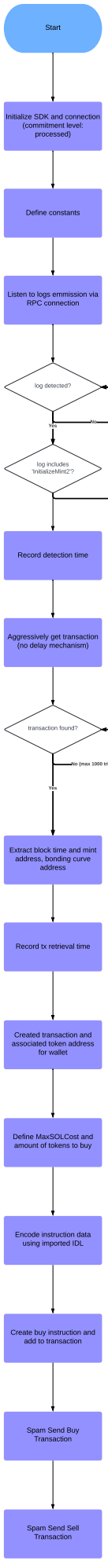
VERSION 1



Latency of connection (Simple ping):

1. School wifi + main net default RPC: 28.64ms

2. School wifi + helius free RPC: 25.56 ms

3 different methods:

1. connection.onlogs, parse transactions -> fetch transactions with retry + helius RPC (mint address, mint time available) = 1225 ms (11 detections average)

2. event listener pumpdotfunsdk + helius RPC, latency = 14460 ms (7 detections average)

3. gRPC Shyft.io (probably quickest) (company stopped posting on twitter and discord is dead, dont want to pay just to try)

Rate of request depends on:
- The speed of your internet connection
- The response time of the Solana RPC endpoint
- The processing power of your machine
- Any rate limiting implemented on the RPC server side

Thought about directly getting mint address from logs (doable, look for 'pump' string) and mint time from logs (harder to get block time, dont know if this is possible?) if possible dont need to gettransaction, can reduce latency.

Emitted logs does not contain mint address or bonding curve address, so need to use get transaction.

	Mint-detection	Detection-tx-retrieval	Retrieval-Transaction-Sent	Total
	1384	229	5451	7064
	983	609	8730	10322
	1018	2994	1432	5444
	700	2070	4824	7594
	1716	3286	2323	7325
	2155	3483	4038	9676
	1282	314	3883	5479
	1750	276	3008	5034
	457	518	3694	4669
	462	541	6228	7231
Average	1767.571429	2369.698413	3375.125	7512.394841

Mint-detection	Detection-tx-retrieval	Retrieval-Transaction-Sent	Total
1312	652	104	2068
1481	533	141	2155
660	320	84	1064
875	706	186	1767
1296	9297	173	10766
687	2927	638	4252
459	1005	639	2103
851	771	149	1771
810	2826	761	4397

Now I am using Jupiter API to send transaction, but takes too long (average 3300ms latency). What if I send transaction directly?

Sending transaction directly which avoids the rate limit issue, but latency is still a problem, average around 20 blocks behind, no priority, jito, bloxroute, etc. check CSV logs. Clearly the latency is much lower. However, alot of transactions do fail when they are sent. so cant get into every con.

Mint to Detection Latency (ms)	Detection to Retrieval Latency (ms)	Retrieval to Buy Start Latency (ms)	Total Buy Latency (ms)	Buy Spam Duration (ms)	Sell Spam Duration (ms)
420	620	0	1061	802	795
981	426	0	1407	808	735
499	985	0	1424	815	749
829	429	0	1358	808	795
1208	272	1	1481	814	827
1271	135	0	1406	840	982
1223	210	1	1434	817	980
1042	480	0	1442	749	668
680	685	0	1364	666	641
1048	282	0	1250	853	588
1002	320	0	1412	573	685
635	137	0	772	739	674
585	377	0	962	702	677
1193	291	0	1359	812	618
1055	189	1	1165	719	920

Currently, the maxSOLCost is 0.002 SOL and amount of token is 39225 tokens. There has to be a better way of defining this instead of hard coding.

First block price: 2.88 * 10^-8

What can i do is: Tune maxsolcost and amount of tokens to a point where I can consistently get 3 * 10^-8 price , or else the transaction will fail due to slippage. I only lose networking fee (~ 0.000005 SOL)

If I have this price limit mechanism in place do I still need the latency limit?

Now Im spamming each token 5 buys 5 sells; can optimize tx parameters to make them higher prob of landing and quicker.

Sell logic:
1. Send sell transaction immediately after sending buy transaction
2. Sell on Jupiter API (Extra fees, API latency, but easy to send, rate limit, not preferred)
3. Send based on PnL, TP/SL using token price. How? Track bonding curve?