The solution is on the following page.

■	■	1 G	2 D	3 P	■	4 B	5 R	6 A
■	7 S	E	E	S	8 T	O	I	T
■	9 C	O	M	E	A	T	M	E
■	10 A	M	O	U	N	T	■	■
11 B	L	E	N	D	■	12 L	13 A	14 B
15 O	P	T	■	16 O	17 T	E	R	I
18 P	E	R	■	19 R	O	O	M	S
20 S	L	Y	■	21 A	S	P	S	■
■	■	■	22 E	N	T	E	R	■
23 I	24 N	25 E	E	D	A	N	A	26 P
27 C	A	R	G	O	D	E	C	K
28 E	T	A	■	29 M	A	R	E	S

Pravan Chakravarthy
"Never Let Them Know Your Next Move" (Lil AVC X 7/25/23) spoilery notes

The theme idea came to me when solving some of my Lil colleagues' asymmetric themeless puzzles and wondering how they decided to place the black squares where they did. I started thinking about what a completely random assortment of black squares would look like. Initially, I simply used Google's random number generator to pick numbers from 1-100 to place on a 10x10 grid, but we decided not to go with that because we didn't want 1-2 letter entries in the final grid.

The next thing I tried was using this random number generator to give me a bunch of numbers between 1 and 120. These numbers were translated to a 10x12 grid (12 to accommodate PSEUDORANDOM), where 1 was the top-left corner, 2 was immediately to its right, and so on. I'd go down the list, manually ruling out any number that left a 1- or 2-letter answer — though in the rare case when a number appeared later that would enable it, it was allowed. (For example, if 61 was a square on the left edge, and I got both 61 and 62 at any point regardless of what order I got them in, I would use them both.) Unfortunately, the resulting grids were quite difficult to fill: the 3-letter word minimum rules out roughly half the squares in the grid from being selected, and creates lots of open space around the edges. I ran into the same problem with my next attempt, which was going over each available square one at a time with a binary random number generator (one with outputs of either 0 or 1). Perhaps I could have found okay fill in one of those grids if I'd stuck with it — maybe something to come back to later.

What we finally settled on was a 9x12 grid with PSEUDORANDOM running down the middle. (Funny to note that building the black squares around the revealer makes the process even more pseudorandom.) This allowed me to independently put in the black squares on either side of the revealer. I used two random number generators, both of which I'd roll one at a time: the first (from 1-48, because there are 48 squares in each half of the grid) to select a square, the second (from 0-1) to decide whether to add black squares around that square, in the event that it was a disallowed square. For example, if the first generator put me 2 squares away from the edge, I'd run the binary generator twice — if both times I got 1, I'd add black squares filling in the gap; if not, I'd ignore the pick. This worked out because it was exponentially harder for more unlikely placements (i.e. a square that created multiple 1- or 2-letter entries) to work out.

Again, all of this could have been done in probably 10% of the time by anyone who knows a bit of coding (heck, I could have brushed up on my programming knowledge and done it myself if I wasn't so lazy). I'm curious to see what kind of grid would have come out of a completely coded approach — probably one that better adheres to the principles of pseudorandomness than what I did.