# There is no Reliable Way to Detect Hacked Ballot-Marking Devices

Philip B. Stark

University of California, Berkeley

21 August 2019

**Abstract.** Election system vendors are marketing ballot-marking devices (BMDs) as a universal system, and some states are deploying them for all voters, not just those who need a BMD to vote independently. Like all devices with CPUs, BMDs can be hacked, misprogrammed, or misconfigured. BMD printout might not reflect what the BMD screen or audio confirmed. If a voter complains that the BMD altered votes, officials have no way to tell whether there was a BMD malfunction, the voter erred, or the voter is attempting to cast doubt on the election. Pre-election logic and accuracy (L&A) tests have little ability to detect outcome-changing problems, in part because BMDs know the time and date, and in part because it is in practice impossible to simulate the full range of voter interactions with the device. *Parallel* or *live* tests of BMDs on election day address the first problem but not the second. In practice, neither L&A nor parallel tests can probe all combinations of voter preferences, device settings, ballot language, duration of voter interaction, input and output interfaces, and other variables that could include a enough votes to change election outcomes. Even if less parallel testing sufficed, it would still require extra BMDs, new infrastructure for creating test patterns, new infrastructure for reporting test results, additional polling-place staff, and additional staff training. And if parallel testing discovers an error, the only remedy is to hold a new election: there is no way to reconstruct the correct election result from an untrustworthy paper trail. Minimizing the number of votes cast using BMDs is prudent election administration.

**Keywords**: elections, logic and accuracy testing, parallel testing, live testing

> In theory there's no difference between theory and practice, but in practice there is.
>
> — Jan L.A. van de Snepscheut

## 1 Introduction

Voting system vendors are promoting the use of ballot-marking devices (BMDs) for all voters, and several states are currently contemplating purchasing enough BMDs for all voters. While this option is more profitable for vendors, deploying BMDs for all voters is more expensive and less reliable than using BMDs as needed for accessibility (Perez 2019) and letting voters who are able to hand mark a ballot mark their ballots by hand.

It is commonly argued that security problems in BMDs are not an issue for election integrity because voters have the opportunity to inspect the BMD printout before casting it, and to spoil the printout and start over if the printout does not match their intended selections. It is also commonly argued that since voters can make mistakes marking a

ballot by hand, hand-marked ballots are no more secure or reliable than BMD printout. Both arguments are myopic: they do not take the overall security of the election into account, and confuse holding a voter responsible for the voter's own errors with holding voters responsible for the overall security of a vulnerable electronic system.

Current BMDs and the protocols for their use make voters responsible for testing BMD security, but do not give them the tools they need to do it. In particular, (Stark 2019) and (Appel, DeMillo, and Stark 2019) point out an intrinsic security gap in current ballot-marking devices: if what the device prints on the paper differs from what the voter expressed to the device or confirmed on a review screen, only the voter knows for sure. The system does not generate any evidence the voter can present to a pollworker or election official to prove there was a discrepancy.[1] There is no way for an election official to tell whether a complaint that a BMD altered selection indicates a machine error, a voter error, or a cry of "wolf" intended to cast doubt on election results.

As a result, error or malfeasance could change a large percentage of votes without inducing election officials to act, beyond possibly offering voters who notice errors another chance to mark a ballot. Worse, if a problem is detected, the only remedy is to hold a new election: there is no way to figure out which printed votes were affected, nor what should have been printed.

Pre-election "logic and accuracy" (L&A) testing should be a routine precaution, but it is not well suited to discovering malware or subtle bugs. It is not generally possible to probe many of the possible vote combinations during L&A testing: to do so would take far too long. There are a number of ways a BMD could "know" that it is being tested, for instance, if the test is not during polling hours on election day. Even if one resets the internal clock on a BMD, the way L&A testers interact with the device is unlikely to closely match how voters interact with the device. The difference can be used as "cryptic knock" like that at the root of the VW "dieselgate" scandal: the device could behave one way when it is being tested, and differently when used by actual voters.

In to address some of the shortcomings of L&A testing, some computer scientists have proposed using "parallel testing" or "live testing." In parallel testing, pollworkers or trusted agents (*testers*) use a BMD to mark ballots on election day, but do not cast those ballots. Their input selections are compared to what the BMDs printed. Even a single discrepancy is evidence of a serious problem: that BMD alters votes, and others might, too.

Because BMDs within a jurisdiction are expected to be running the same software and to be configured identically, a discrepancy on one machine casts doubt on an entire election.[2]

How much testing is required? Juan Gilbert suggests testing once per hour.[3] Dan Wallach suggests testing approximately at random, with no detail about how and when to test (Wallach 2019). While "enough" suitably designed random testing can provide assurance,

[1] Voters could take "ballot selfie" videos of the interaction with the BMD, the review screen, and the printed output, but that would compromise voter privacy and is illegal in many jurisdictions. Moreover, it would not be hard for malicious voters to fake problems by revising their selections before printing, or by constructing a "deep fake." That opens elections to attacks on public confidence by spreading unwarranted fear, uncertainty, and doubt (FUD).

[2] However, BMDs cannot be *assumed* to be programmed and configured identically: if one BMD behaved correctly, that does not mean every BMD behaved correctly. An attacker might hack only some of the machines, and hacking might be designed to affect different machines differently. Indeed, examples below show how that can help an attacker avoid detection.

[3] http://www.juangilbert.com/BallotMarkingVerificationProtocol.pdf, last visited 15 August 2019.

"enough" turns out to be too much to be practical, both for parallel testing and for L&A testing.[4]

## 2 A Naive Calculation

Gilbert and Wallach seem to have the following calculation in mind. (Indeed, the calculations in section 5.1 of (Wallach 2019) are exactly of this form.)

Suppose that a BMD alters one or more votes on a ballot with probability $p$, independently across ballots—regardless of the selections made on the BMD, the time of day, or any of the BMD settings (e.g., whether the audio interface is being used, the language the ballot is displayed in, the font size, etc.) and without regard for aspects of voter behavior that the BMD can monitor (e.g., how long the voter takes to make selections, whether the voter changes any selections, whether the voter backs up to previous pages, etc.). Then if testers make $n$ tests, the chance that the BMD will alter at least one of the votes on one of the test ballots—and thus that parallel testing would detect the problem—is $1-(1-p)^n$.

For $p = 0.01$ (i.e., a 1% chance that the BMD alters votes on each ballot, independently across ballots), marking $n = 50$ test ballots would give a 39% chance of catching the BMD. To have a 95% chance of catching the problem, the sample size would need to be $n = 300$ test ballots.

I understand that a BMD can handle about 140 voting transactions on election day. Testing enough to have a 95% chance of detecting a 1% problem, i.e., 300 tests, would leave no time for voters to use the BMD. Even for pre-election L&A testing, where capacity for actual voters is not an issue, conducting 300 tests would take about 25 hours.

(Wallach 2019) argues that considering $p = 0.01$ is unrealistic, that a malicious actor would always attempt to "cheat" by a large amount ($p$ rather greater than 1%). But an attacker seeking to avoid detection would not try to alter more votes than necessary (with some room for error). Changing votes on 1% of ballots can alter the margin of a jurisdiction-wide contest by 2%, if there are no undervotes, and by more than 2% if there are undervotes.[5]

Many contests are decided by less than that. Indeed, the margin in the 2016 U.S. presidential election was 0.22% in Michigan, 0.37% in Rhode Island, 0.72% in Pennsyvania (one of the states poised to buy BMDs to serve all voters), and 0.76% in Wisconsin. If the results had been different in Michigan, Pennsylvania, and Wisconsin Trump would have received fewer than 270 electoral votes, the number required to win.

Moreover, altering the votes on 1% of ballots overall can change the margin of smaller contests by far more than 2%. For instance, if a contest is only on 10% of the ballots,

[4] Those who claim that L&A testing is a reliable means of discovering malware, bugs, and misconfiguration do not seem to have accounted for the vast number of combinations of things that could trigger a problem, as illustrated below.

[5] The usual way of reporting margins ignores undervotes: it is the difference between the number of votes the winner and runner-up received, divided by the number of valid votes in the contest. The "diluted margin" is the difference between the number of votes the winner and runner-up received, divided by the total number of *ballots* cast in the contest, rather than the total number of *votes* cast in the contest. The diluted margin is never larger than the margin, but it can be much smaller. Changing the votes on $x$% of the ballots can change the diluted margin by up to $2x$%, but can change the ordinary margin more. For instance, if the undervote rate is 30%, then changing votes on 1% of the ballots can change the margin by $0.02/0.7 = 2.9\%$.

altering votes on 1% of all ballots could change the margin in that particular contest by 20% if there are no undervotes. If the undervote rate in the contest is 30%, altering the votes on 1% of all ballots could change the margin in that particular contest by nearly 29%.

The sample size calculations above are based on an unrealistic model for how a malicious adversary would alter votes. An adversary has better strategies to avoid detection than these pro forma calculations—and those in (Wallach 2019)—assume. We now explore some of the options available to an attacker, illustrating that effective parallel testing requires sample sizes that are impossible in practice.

# 3 What is the threat model?

Suppose Alice, Bob, Carol, and Dan are running against each other in a plurality (whoever gets the most votes wins) contest. A malicious actor, Mallory, wants Alice to win. Mallory is not going to alter ballots at random. Mallory is not going to turn votes for Alice into undervotes or votes for a different candidate: it would only make Mallory's job harder. Mallory would need to alter even more votes, which would increase the risk of detection. Rather, Mallory is going to turn some undervotes or votes for other candidates into votes for Alice.

Is Mallory going to cause every machine to alter some votes? Only some machines? Only machines in precincts where Alice is already expected to win, to keep the precinct totals unsuspicious? Will the programming cause the machine to alter votes all day, or only, say, in the last hour of voting, or when the lines for voting are predicted to be longest (when testers might be reluctant to conduct tests, because it would slow down voting)? Will the programming only alter votes if the voter is using the audio interface?[6] Only if the voter takes a long time to vote?[7] Only if the voter changes selections repeatedly?[8]

Will Mallory hack the BMD to make every vote for other candidates equally likely to be altered into a vote for Alice? Or will Mallory make the machine alter votes only when some other pattern of votes occurs in other contests on the ballot, e.g., only when the voter does not make any selections in some set of contests?[9]

To have a large chance[10] of detecting a particular problem, testers need to use "enough" tests under the conditions that trigger the vote alterations. That might mean testing particular machines using particular vote pattern or patterns at particular times of day when the machines have been used in a particular way, and using the machines in a particular way.[11]

The malware might or might not change votes at random: if we knew what the malware did, we wouldn't need testing. To talk about the probability of detecting a problem, we

[6] That might signal that the voter is less likely to check the printed selections.
[7] That might signal that the voter is elderly or has trouble reading.
[8] That might signal that the voter is less likely to remember the final selections, and more likely to assume that any discrepancy is their own fault.
[9] That might signal that the voter is not very interested in those contests, and thus might not be alert to changes.
[10] Here, "chance" is meant in a non-technical sense: see below.
[11] For instance, using the audio interface, changing the font size, displaying the ballot in a language other than English, taking a short or a long amount of time to vote, etc. The possible strategies are endless.

need to assume that the *testers* are testing at random. For instance, testers could test randomly selected machines at times randomly selected during the day, using randomly selected test patterns. They could choose randomly whether to use the touchscreen interface, the audio interface, or the sip-and-puff interface; whether to change the font size, whether to display the ballot in a language other than English; whether to vote slowly or quickly; whether to revise selections and how many times; and so on.

## 3.1 Attacks using vote patterns and time of day

There is an enormous number of possible voting patterns. For instance, if there are 5 contests, each with 4 candidates, there are $5^5 = 3125$ possible vote patterns, including the possibility of undervotes. In our hypothetical example where the contest between Alice, Bob, Carol, and Dan is one of the 5 contests on the ballot, there are $4 \times 5^4 = 2500$ patterns that do not already have a vote for Alice. Mallory will alter votes when the voter selects some subset of those patterns, possibly at random, possibly only at particular times of day, and possibly only when the BMD is being used in a particular way, e.g., when it is busiest or when the voter takes a long time. The malware might never alter votes if the BMD has been idle for more than 5 minutes, on the assumption that testing is more likely to take place when the polling place is "calm." The malware might never alter votes unless the user takes more than 10 minutes to mark the ballot, on the assumption that testers will be in a hurry and will only rarely tie up a BMD for that long for a single test. There is a very large space of strategies available to Mallory.

To illustrate the difficulty of detecting problems, suppose that Mallory alters *every* ballot on which the voter selected Bob, but *only* if the the voter also selects the candidate from Bob's party in the next two contests on the ballot. And Mallory only alters votes for 30 minutes a day—at the time predicted to be busiest, say 12:15-12:45pm.

Suppose that 10% of voters vote during that half hour. Since many people vote along party lines, Mallory might expect a large fraction of voters who choose Bob will pick someone in Bob's party in the next two contests, too. For simplicity, let's suppose everyone who votes for Bob does just that.

Suppose that 10% of all voters vote during that half hour, and that 40% of those voters vote for Bob (and for the candidates in Bob's party in the two following contests). Then Mallory will flip votes on $10\% \times 40\% = 4\%$ of the ballots, changing the margin between Alice and Bob by 8% for ballots marked using *that* BMD. Hacking 25% of all BMDs in the same way would change the overall margin by 2%.

## 3.2 What is the chance Mallory gets caught?

Suppose the jurisdiction has 400 BMDs in all, of which 100 were hacked.

In the scenario described above, tests of the 75% of machines that were not hacked will not find any error whatsoever. Any tests using 60% of vote patterns that don't trigger the hack will not find any error whatsoever. Any tests run outside the critical 12:15–12:45 window will not find any error whatsoever.

Since that is (by assumption) the busiest time, the testers might not run any tests at all in that half hour. Let's assume that they run one test between 12:15 and 12:45, on an affected machine, using a random test pattern.

What is the chance that any of the tests finds a discrepancy? Of the 3125 possible test patterns, only $5^2 = 25$ will trigger the malware. That is, there is only a $25/3125 = 0.8\%$

chance that the testers will catch that BMD, no matter how many times they test it outside the window 12:15–12:45.

Suppose every machine is tested in the same way, using test patterns selected independently at random, and all happened to be tested between 12:15 and 12:45. Even though 100 machines have been infected and the margin has been changed by 2%, the chance that *none of the tests of any of the 400 machines* finds a discrepancy is $(1 - 0.008)^{100} = 45\%$.

This strategy for altering votes is *not* a worst-case scenario, just one of countless possible attacks. In particular, this attack mentioned does not use information about BMD settings or voter interaction with the BMD.

## 3.3 Attacks using BMD settings and voter interaction

BMDs are a perfect device to disenfranchise voters with disabilities, older voters, and voters whose preferred language is not English. That is because the BMD "knows" whether the voter is using the audio interface or the sip-and-puff interface, whether the voter has changed the font size, whether the BMD is displaying the ballot in English, how long the voter is taking to vote, and whether the voter is changing selections or reviewing ballot pages. It can therefore be programmed—deliberately or accidentally—to alter the selections of just such voters.

Thus, parallel tests need to probe not only vote patterns and time of day, but also BMD settings and voter interactions with the BMD: anything known before the election or that the BMD can monitor on election day can be used to target votes to alter.

For instance, suppose that 5% of voters take more than 2 minutes per page to vote. Then a substantial number of the live tests in every polling place need to take more than 2 minutes per page. Otherwise, if the malware only alters votes when the voter takes longer than 2 minutes, it could change some margins by up to 10% with no possibility of detection.[12] Note that this means testing cannot (always) be fast. Gilbert's proposed testing protocol[13] assumes that tests will take about 4 minutes each, and does not contemplate the possibility that an attack could be targeted using the duration of the voting session. The attack described here has no possibility of being detected by his proposed protocol.

Similarly, suppose that 5% of voters change at least two of their selections. Then a substantial number of the live tests in every polling place need to change at least two selections. Otherwise, if the malware only alters votes if the voter changes two or more selections, it could change some margins by 10% with little or no possibility of detection.

Similarly, if 0.8% of voters use the audio interface,[14] a substantial number of tests in every polling place need to use the audio interface. Otherwise, if malware only alters votes when the voter uses the audio interface, it could change some margins by 1.6% with little or no possibility of detection.

[12] In this example and the ones that follow, the maximum change depends in part on what the true voter preferences in each contest are. For instance, if all such voters voted for Bob, changing their votes to votes for Alice would alter the margin by 10%. If votes in that group were evenly split between Alice and Bob, changing the votes for Bob into votes for Alice votes would alter the margin by 5%. If none of them voted in the contest in question, changing their votes to votes for Alice would alter the margin by 5%. If all of them voted for Alice in the first place, no change to those votes could increase Alice's margin.

[13] http://www.juangilbert.com/BallotMarkingVerificationProtocol.pdf, last visited 18 August 2019.

[14] Approximately 0.8% of the U.S. population is legally blind. https://www.nfb.org/resources/blindness-statistics, last visited 15 August 2019.

If 25% of voters display the ballot in a language other than English,[15] a substantial number of tests in every polling place need to set the BMD to use a language other than English. Otherwise, the BMD could change some margins by 50% with little or no possibility of detection.

Live tests need to probe *every* subset of voter preferences, BMD settings, and voter interactions with the BMD that could alter *any* contest outcome, and they need to probe every such subset enough to have a high probability of detecting any changes to selections in that subset.

For illustration, imagine a single contest expected to be neck-and-neck across all four candidates, Alice, Bob, Carol, and Dan. Mallory knows from past experience that about 5% of voters in those precincts will display the ballot in English, take more than 10 minutes to vote, and change at least two of their selections before printing, and that this behavior is not associated with voter preferences.[16]

Mallory rigs the machines that contain the contest identically: they will all randomly flip votes for Bob into votes for Alice with 50% probability, but only if the voter displays the ballot in English, takes more than 10 minutes to vote, and changes at least two selections. That will make Alice's margin over Bob 2.5%, and her margin over Carol and Dan 1.25%.

To have a 95% chance of finding this problem, there need to be 5 tests in those precincts where the testers display the ballot in English, select Bob in that contest, take more than 10 minutes to vote, and change their selections at least twice ($1 - (1 - 0.5)^5 = 0.968$). This alone would tie up BMDs for more more than 50 minutes. Since testers would not know ahead of time that Mallory is trying to rig the election in favor of Alice, to protect just that single contest against the same hack in favor of the other candidates, they would also need to do those tests but with votes for Alice, votes for Carol, votes for Dan, and undervotes. That would tie BMDs up for $5 \times 50 = 250$ minutes, i.e., 4 hours and 10 minutes.

Of course, Mallory might have hacked the BMDs to change votes for Carol into votes for Alice, but only for voters who display the ballot in English, take 5-7 minutes to vote, and do not change any selections. If voters who display the ballot in English, take less than 5-7 minutes to vote, and do not change any selections comprise 10% of voters, then Mallory would only have to flip 25% of the votes for Carol in that group to votes for Alice to get the same margins. To have 90% chance of detecting the problem, there need to be 11 tests in those precincts where testers selected Carol, took 5-7 minutes to vote, and did not change any selections ($1 - (1 - 0.25)^1 1 = 0.958$). Those 11 tests take at least 55 minutes, and again, since the testers would not know in advance that Mallory was using votes for Carol to benefit Alice, they also need to do those tests, using votes for Alice, for Bob, for Dan, and undervotes. That would tie BMDs up for $5 \times 55 = 275$ minutes, i.e., 4 hours and 35 minutes.

Because the conditions that trigger those two attacks are mutually exclusive, testing for one does not test for the other: we have to test for both. This brings us to 8 hours and

[15] In 2013, roughly 26% of voters in Los Angeles County, CA, spoke a language other than English at home. http://rrcc.lacounty.gov/VOTER/PDFS/PUB/2013_Multilingual_Access_Elections.pdf, last visited 15 August 2019.

[16] Such things could be measured by malware on BMDs deployed in previous elections, or possibly inferred from BMD internal logs from previous elections. It is not likely to be something that an election official knows or could measure legally or ethically: taking such measurements would likely compromise vote anonymity.

45 minutes of testing, just for these two possible attacks on that single contest, among many other possible attacks.

For instance, Mallory might instead have changed votes for Bob into votes for Alice, but only when the voter displays the ballot in a language other than English and takes 10 minutes or more to vote. If such voters comprise 10% of all voters, the calculation two paragraphs above shows that to have a 95% chance of detecting this attack requires at least another 4 hours and 35 minutes of testing. We are now at 13 hours and 20 minutes of testing.

But we have not exhausted the possible attacks, even on that one contest. For instance, Mallory could have hacked the BMDs to change votes for other candidates into votes for Alice only under *other*, mutually exclusive conditions, which could depend on *other* variables as well, such as votes in other contests, time of day, recency of the last voter's session, BMD load, etc. The 13 hours of tests already described cannot not reveal such hacks—even in principle—because the attacks involve mutually exclusive conditions.

And all of this testing is just for one contest. Once additional contests are on the ballot, there need to be tests for attacks against them, too. In practice, live testing cannot provide much protection against malware.

# 4 A little math

## 4.1 Mallory's goal

Let $i$ denote a set of conditions that could apply to a voter's interaction with a BMD, including patterns of possible preferences in the elections on the ballot, time of day, duration of voting session, BMD user settings such as language and font size, whether the voter revised selections and which were revised, etc. Let $f_i$ be the frequency with which actual voter sessions meet those conditions; we assume there is a complete enumeration of conditions, so that $\sum_i f_i = N$, the total number of ballots marked by voters. To alter the margin between Alice and Bob by $m$ votes, Mallory needs to alter votes on a fraction $\pi_i \in [0, 1]$ of the sessions that satisfy the set of conditions $i$, in such a way that $\sum_{i \in \mathcal{I}} \pi_i f_i \geq m/2$. (The bound is sharp only if all the conditions $i$ for which $\pi_i > 0$ already have a vote for Bob and not for Alice; otherwise, more than $m/2$ ballots need to be altered.)

Suppose that, given the fraction $\pi_i$, Mallory picks the particular $\pi_i f_i$ sessions that satisfy condition $i$ to alter randomly, independently of anything the tester does, and independently across $i$. Then, if the tester tries condition $i$, the conditional probability that Mallory will alter any votes in that session—and hence that the tester will detect that Mallory interfered—is $\pi_i$.

Suppose the testers know $\{f_i\}$, i.e., they know how many sessions satisfy each set of conditions.[17] If the testers test condition $i$ with probability $f_i/(\sum_j f_j)$, the probability

[17] If this were literally true, the testers would already know the correct outcome of every contest, and thus could detect any interference by Mallory simply by looking at the overall totals: there would be no need to audit—or even to conduct the election. Moreover, to estimate or measure $f_i$ would require capability that is not currently (supposed to be) part of voting systems, because it would compromise vote anonymity. Mallory could, however, estimate $f_i$ well using malware previously installed on BMDs.

that a given test will reveal Mallory's attack is

$$\frac{\sum_i \pi_i f_i}{\sum_i f_i} \geq \frac{m}{2N}.$$

In general Mallory will be able to alter outcomes by changing votes randomly for sessions that share some set of characteristics that any given auditing strategy is unlikely to probe enough to detect.

## 4.2 Auditing as an adversarial game

The auditing problem can be viewed as a two-player game, tester versus Mallory: the tester picks an auditing strategy, which must be public. With knowledge of that strategy, Mallory picks a (possibly random) rule for altering votes to in a way that has a large chance of changing the outcome of one or more contests. If Mallory can keep the probability that the audit will find any changed votes low, Mallory wins. If the auditing strategy ensures that if Mallory altered the outcome of one or more contests, the audit has a large chance of sounding an alarm, the tester wins.

Nothing in this paper should be construed to imply that parallel tests *only* need to detect malicious attacks. Bugs, misconfiguration, and other problems can also alter votes. Parallel testing does not solve the BMD security problem unless it has a sufficiently large chance of detecting outcome-changing faults, whatever their cause.

## 4.3 The curse of dimensionality

Several features of this problem make random testing require large sample sizes to be effecctive:

1. The testers do not know which contest(s) may be affected, nor which candidate(s) in those contests.
2. The outcome of a small contest or a contest with a narrow margin can be changed by altering only a small fraction of votes—depending on actual voter preferences, an arbitrarily small fraction of votes.
3. Malware or errors can be triggered by a vast number of combinations of many variables.

The high dimension requires exponentially more sampling, in general. As a statistical problem, auditing election outcomes using a trustworthy paper trail (e.g., checking outcomes using a risk-limiting audit (Stark 2008) (Stark 2009) (Lindeman and Stark 2012)) is easier than parallel testing of BMDs, in part because such audits only need to detect errors that favored the reported winner(s), and in part because whether anything affected the tabulation of the cast ballots is evident by looking at the cast ballots—provided the paper trail is trustworthy. In contrast, parallel BMD testing needs to detect errors that favor *any* candidate and to catch the malware in the act, by trying the right input to the right BMD at the right time.

## 4.4 An oracle bound

Suppose the tester could ask an oracle whether a particular cast BMD printout had an error. This would obviate the need even to know what variables characterize voters'

interactions with the BMD, much less to know how often voters interact with the BMD in any particular way (vote preferences, time of day, speed of voting, interface, language, etc.).

Suppose a contest was on the ballot on 20 BMDs, which printed a total of $20 \times 140 = 2800$ ballots on election day. Suppose that the BMDs altered 14 of the ballots, which could change the outcome of the contest in question by 1% if there were no undervotes, and by more if there were undervotes. (For instance, if the undervote rate is 50%, then changing votes on 0.5% of the ballots could change the margin by 2%.)

The tester will ask the oracle whether a set of randomly selected cast BMD printouts had any errors. How big would that set need to be in order to have at least a 95% chance of finding at least one printout with an error? The answer is the smallest value of $n$ such that

$$\frac{2800-14}{2800} \cdot \frac{2799-14}{2700} \cdots \frac{2800-(n-1)-14}{2800-(n-1)} \leq 0.05,$$

i.e., $n = 539$ voters, about 20% of the capacity of those machines. This would mean testing each BMD about 27 times on election day, on average: several times an hour, even under these counterfactual, optimistic assumptions.

We can turn the oracle bound around to ask how large a contest would need to be so that oracle-based testing would have at least a 95% chance of detecting a change to 0.5% of the ballots cast in the contest using not more than 13 tests per day per machine, on average. (That corresponds to testing a machine once per hour for a 13-hour day, and to using roughly 9.2% of BMD capacity for testing rather than voting.) The answer is that it would need to be on at least 47~BMDs, i.e., at least 6,580 votes, even under these impossibly optimistic assumptions.

But in reality, testers cannot look over voters' shoulders to see whether the BMD printed what it was supposed to. Testers do not know what variables characterize voters' interactions with BMDs, nor how often votes fall into any range of values of those variables: they could not measure these things without collecting data that would compromise the anonymity of votes. There would need to be more tests to compensate for that lack of information—many more. In reality, tests have to be *in addition to* actual voters' use of the BMDs, so the jurisdiction would need more machines (at least 24 machines instead of 20 in the first example; at least 52 instead of 47 in the second example) to accommodate the testing. In reality, contests can be decided by less than 1% of the ballots, so there would need to be more checks. And in reality, some contests are on fewer machines, which also increases the amount of testing required.

# 5 Complications

## 5.1 The only remedy is a new election

If testing catches a BMD altering votes, it is clearly appropriate to remove that BMD from service and conduct a forensic investigation of that BMD—and all the other BMDs used in the jurisdiction.[18] But what about *this* election? There is no way to figure out which ballots were affected, nor even how many ballots were affected. There is not even a way to tell how many BMDs were affected, especially because malware could erase itself after election day, leaving no forensic evidence.

[18] It's also appropriate to contact DHS and the FBI.

Thus, there is no way to know whether the election outcome reflected by the printed output is the same as the outcome according to what the BMDs told voters on confirmation screens (or through audio confirmation). Conducting a new election is the only way to rectify the damage.

## 5.2 Margins are not known

Margins are not known until the election is over, when it is too late to do more testing if contests have narrower margins than anticipated. Testing to a pre-defined threshold, e.g., to ensure a 95% chance of detecting errors in 0.5% or more of the votes in any contest, will sometimes turn out not to suffice. See below.

## 5.3 Tests have uncertainty

Relying on parallel testing to detect BMD faults has intrinsic uncertainties that should not be neglected.

For instance, suppose it were possible to design practical parallel tests that had a 95% chance of sounding an alarm if BMDs alter 0.5% or more of the votes in any contest. A reported margin of 1% or less in a plurality contest[19] would be below the "limit of detection"[20]: morally, the outcome of such a contest would be a tie. Would we revise election law to require automatic runoff elections whenever a reported margin is below 1%? If not, elections cease to be democratic and become an arbitrary means of deciding political contests.

The principle of evidence-based elections (Stark and Wagner 2012) says that election officials should not only report the winners, but also provide persuasive evidence that the reported winners really won—or admit that they cannot, in which case there should be a new election. In this hypothetical, that would mean holding a new election if the reported margin were below 1%.

And if a reported margin is greater than 1% (or the relevant limit of detection), we would still have only 95% "confidence" that BMD faults (bugs, misconfiguration, or hacking) did not cause the wrong candidate to appear to win. If the BMD output reflects the wrong electoral outcome, a perfect full manual tally, recount, or risk-limiting audit based on BMD printout will (with high probability) confirm that wrong outcome.

## 5.4 Asymmetry of information

The testing strategy will be public: it is a matter of regulation or law. But the attacker's strategy is private. Thus, an attacker can adapt to the announced testing strategy, but the attacker does not have to announce its strategy. Even if the tests contain random or adaptive elements, testing must be limited to a small number of tests per machine per day, to leave time for actual voting. That limits the ability of testing to find problems that affect only a small fraction of votes.

[19] For non-plurality social choice functions such as ranked choice or instant runoff, there is more than one definition of the margin, and the margin can be difficult to calculate. The relevant question is whether a change to 0.5% of the votes in a contest could cause the wrong candidate to appear to win, given the social choice function in that contest. If so, then the contest outcome is in doubt, and a new election seems like the appropriate remedy.

[20] If there are undervotes, the "limit of detection" measured as a margin would be even higher, as discussed above. That is, changing 0.5% of the votes could change the margin by more than 1%.

### 5.5 Malicious attacks are not the only risk

Misconfiguration, malfunction, and bugs can also alter votes. One cannot assume, as (Wallach 2019) does implicitly, that the only problems parallel testing needs to detect are those that a clever opponent would engineer.

### 5.6 Randomness is key

The test patterns need to be unpredictable, or malware could simply leave the votes alone if the input pattern matches a known or likely test pattern. Similarly, the test times need to be unpredictable (not, for instance, during the first 5 minutes of each hour), or malware could simply leave votes alone when tests are expected.

It is impractical to test at uniformly distributed random times, or after a random number of voters have used each BMD. For instance, testers will probably not mark 5 test ballots in a row on the same BMD. More likely, the sampling plan would involve testing hourly or at other regular intervals. For instance, Gilbert suggests testing once an hour (http://www.juangilbert.com/BallotMarkingVerificationProtocol.pdf), which makes it easier for malware to evade the test. If the polling place is busy when the test is supposed to occur, there will be pressure on the testers not to slow down voting by testing the BMD. If there is less testing when BMDs are busy, it's easier for malware to avoid the test.

### 5.7 New systems and extra hardware would be needed

Parallel testing requires new infrastructure to generate the random test patterns "just-in-time" and to signal the testers to conduct tests at random times. Infrastructure to log and report test results and to respond to discrepancies would also be needed.

Parallel testing also requires deploying *more* BMDs, because some (or all) of the capacity will be taken up by the live testing, reducing (or completely exhausting) the capacity for actual voters.

### 5.8 Extra staff and training would be needed

Who will conduct these tests? Pollworkers or elections staff would need additional training. Tests would probably need to involve two people, to ensure that the tester did not make a mistake in inputting selections. Additional pollworkers or other staff would be needed in every polling place to perform the additional work. The most important time to test is when the polling place is busiest, because that's when the most votes could be affected in the shortest time. Thus, effective testing is almost guaranteed to slow down voting.

## 6 Can the number of spoiled ballots be used to detect problems?

(Wallach 2019) also suggests comparing the rate of ballot "spoilage" to the expected background rate as a statistical test for BMD malfunction. Again, while this idea has heuristic appeal, numbers matter.

The Poisson distribution is a standard model for independent rare events. Without worrying about whether it is a good model for voting errors, let's use it as a guide to intuition.

Suppose that the background rate of ballot spoilage, based on previous elections, is 1%; that is, about 1% of voters request a fresh ballot. We suppose that the spoilage rate is the same for all machines in all precincts. We also suppose that about 100 voters use each BMD on election day.

Imagine a contest that appears on 2,000 BMDs, collectively marking 200,000 ballots. If things are functioning properly, we would expect about $200000 \times 0.01 = 2000$ spoiled ballots. What observed number of spoiled would give 95% confidence that something unusual is going on? The answer is 2074: 2073 spoiled ballots would not sound an alarm. If 10% of voters whose selections are mis-recorded notice the error and spoil their ballots,[21] that would correspond to 730 BMD errors. That means that malware, bugs, or other misconfiguration could alter the margin in this contest by $2 \times 730/200000 = 0.73\%$. But it could alter the margin in a smaller contest by far more without creating a statistically suspicious number of spoiled ballots.

The numbers depend strongly on the number of BMDs that contain the contest. For instance, if a contest includes 50 BMDs (5,000 voters), then under the same assumptions we would expect 50 spoiled ballots if everything is functioning correctly. It would take 62 spoiled ballots to sound an alarm at 95% confidence, i.e., a 5% chance of a false alarm if everything is actually working properly. Finding 61 spoiled ballots would not sound an alarm. The $61 - 50 = 11$ extra spoiled ballots signal 110 errors, which could shift the margin by $2 \times 110/5000 = 4.4\%$. Smaller contests can be altered by even larger amounts without raising an alarm.

Even under ideal circumstances, passive monitoring based on spoiled ballots does not produce direct evidence of malfunction; it does not identify which ballots, if any, have errors; and it does not provide any evidence about whether the errors changed any outcomes. It seems implausible that an election official would hold a new election simply because the number of spoiled ballots was a bit larger than expected.

# 7 Can BMDs ever be secure?

Current BMDs as currently deployed have an insurmountable security flaw described in (Stark 2019) and (Appel, DeMillo, and Stark 2019): the protocols make voters responsible for checking whether BMDs are performing correctly, voters cannot get any evidence to prove to others that a malfunction occurred. As a result, election officials will always be in a bind if voters complain of problems: there is no way to tell the difference between a misbehaving machine, voter error, and a cry of "wolf."

Some protocols developed for end-to-end cryptographically verifiable (E2E-V) elections can allow voters to prove that machines misbehaved in particular ways.[22] While those protocols do not apply to BMDs, similar ideas might make it possible for a voter to obtain evidence of MBD malfunction that could be presented to others. None has been proposed.

The design and use of BMDs needs to be re-thought from the ground up so that either

a. voters get evidence of malfunctions that can be used to demonstrate to others that a malfunction occurred, or

[21] The work of (DeMillo, Kadel, and Marks 2018) suggests this is very optimistic.

[22] E.g., the "Benaloh challenge," as described in (Benaloh 2006). The Benaloh challenge can provide proof that a machine encrypted a plaintext vote incorrectly, but it does not address what happens if the voting machine prints the wrong plaintext vote.

b. some other mechanism is in place that has a high probability of catching all BMD malfunctions that could alter election outcomes.

Parallel testing is an attempt at (b). Unfortunately, it requires far more testing than is possible in practice. "Passive" testing by monitoring the number of spoiled ballots is also an attempt at (b). Unfortunately, the signal-to-noise ratio makes passive testing ineffective, especially for smaller contests. In any event, the only remedy if any BMD is discovered (or suspected) to have altered votes is a new election. Voting systems should instead be designed to be *resilient*, so that they can recover from errors without re-running the election. This is inherent in "strong software independence" (Rivest and Wack 2006).

# 8 Conclusion

While in theory parallel testing or logic-and-accuracy testing might help protect against malfunctioning or hacked BMDs, in practice, there are so many attack strategies (and possibilities for subtle bugs) that no reasonable amount of testing could guarantee even a modest chance of finding an outcome-changing problem. An attack (or bug) could depend on any number of variables, including, among others:

- the current voter's selections (including undervotes) in every contest
- selections made by previous voters who used that BMD
- time of day
- BMD load
- BMD user settings, such as font size, ballot language, whether the audio interface is in use, whether the sip-and-puff interface is in use
- voter interaction with the BMD, such as how quickly the voter completes each page, whether and how often the voter revises selections, and whether the voter chooses to ignore/override undervote warnings

Because there are so many possible outcome-changing problems that involve mutually exclusive sets of conditions and different contests, it's effectively impossible to protect against malware or subtle bugs by parallel testing or logic and accuracy testing. The number of tests required to get a reasonable level of assurance is simply too high.

Moreover, even if it were possible to get, say, 95% confidence that no more than 0.5% of the votes were changed in any contest, fairness would then demand a runoff election in any (plurality) contest with a diluted margin[23] of 1% or less.[24]

No jurisdiction has conducted parallel testing of this kind. Any parallel testing requires more BMDs (to leave enough capacity for actual voters), new infrastructure, new procedures, and more pollworker training. Testing when the polls are busiest is especially important—and especially disruptive. The sample sizes required for testing to be effective against outcome-changing errors are prohibitively large (the same is true for logic and accuracy testing): considerable time and labor are required. And if a problem is detected, the only safe response is to hold a new election: there's no way to tell which votes were changed, nor even how many votes were changed.

[23] See footnote 5.

[24] For non-plurality social choice functions such as ranked choice or instant runoff, the margin can be quite difficult to calculate. The relevant question would be whether a change to 0.5% of the ballots could cause the wrong candidate to appear to win, whatever the social choice function. If so, then the contest outcome is in doubt, and a new election seems like the appropriate remedy.

Absent some clever new design for a BMD-like device that enables voters to provide public evidence of any malfunctions they experience—without making elections vulnerable to false accusations and without compromising voter privacy—there is no way to ensure that BMDs have not misprinted enough votes to change election outcomes.

Nonetheless, well-designed BMDs[25] are arguably the best technology for voters with some kinds of disabilities to mark and cast a paper ballot independently. To my knowledge, existing commercial BMDs do not provide a means for blind voters to check whether the printed output matches their intended selections. It would not be technically challenging to add "verification stations" that speak a printed ballot to the voter, so the voter can spoil the ballot and start over if there is a discrepancy.

If BMDs are deployed, they should undergo some pre-election logic and accuracy testing to screen for gross configuration errors. BMD printout should be designed to make it as easy as possible for voters to check whether the printout matches their intended choices. Voters who use BMDs should be urged to check whether the printout matches their intended choices, and to request another chance to mark a ballot if there's a discrepancy. Voters with disabilities should be provided an accessible way to check BMD printout. And election officials should pay attention to the number and distribution of spoiled BMD printouts.

But for the foreseeable future, prudent election administration also requires keeping the number of ballots marked by machines as small as possible—while ensuring that every voter is provided a means to mark, verify, and cast a paper ballot independently.

# Bibliography

Appel, A.W., R.A. DeMillo, and P.B. Stark. 2019. "Ballot-Marking Devices (Bmds) Cannot Assure the Will of the Voters." https://papers.ssrn.com/sol3/papers.cfm?abstract_id=3375755.

Benaloh, J. 2006. "Simple Verifiable Elections." *Proceedings of the USENIX/Accurate Electronic Voting Technology Workshop and Electronic Voting Technology Workshop* Vancouver, Canada.

DeMillo, R., R. Kadel, and M. Marks. 2018. "What Voters Are Asked to Verify Affects Ballot Verification: A Quantitative Analysis of Voters' Memories of Their Ballots." https://ssrn.com/abstract=3292208. https://doi.org/http://dx.doi.org/10.2139/ssrn.3292208.

Lindeman, M., and P.B. Stark. 2012. "A Gentle Introduction to Risk-Limiting Audits." *IEEE Security and Privacy* 10: 42–49.

Perez, E. 2019. "Georgia State Election Technology Acquisition: A Reality Check." https://trustthevote.org/wp-content/uploads/2019/03/06Mar19-OSETBriefing_GeorgiaSystemsCostAnalysis.pdf; OSET Institute Briefing.

[25] There are many poorly designed BMDs on the market, including designs with blatant, easily exploited security flaws. See, e.g., https://tyt.com/stories/4vZLCHuQrYE4uKagy0oyMA/5YIEQxHW5qmWayG0kYCsy2 or https://freedom-to-tinker.com/2018/09/14/serious-design-flaw-in-ess-expressvote-touchscreen-permission-to-cheat/ (both visited 20 August 2018). Moreover, recent testing in Pennsylvania suggests that some current BMDs do not enable voters with disabilities to vote independently (Secretary of the Commonwealth of Pennsylvania 2018, pp68–90)

Rivest, R.L., and J.P. Wack. 2006. "On the Notion of Software Independence in Voting Systems." http://vote.nist.gov/SI-in-voting. pdf.

Secretary of the Commonwealth of Pennsylvania. 2018. "Report Concerning the Examination Results of Election Systems and Software EVS 6012 with DS200 Precinct Scanner, DS450 and DS850 Central Scanners, ExpressVote HW 2.1 Marker and Tabulator, ExpressVote XL Tabulator and Electionware EMS."

Stark, P.B. 2008. "Conservative Statistical Post-Election Audits." *Annals of Applied Statistics* 2: 550–81. http://arxiv.org/abs/0807.4005.

———. 2009. "Risk-Limiting Post-Election Audits: *P*-Values from Common Probability Inequalities." *IEEE Transactions on Information Forensics and Security* 4: 1005–14.

———. 2019. "Ballot-Marking Devices (BMDs) Are Not Secure Election Technology: Letter to Subcommittee on Voting Technology of Government Affairs Committee, Georgia House of Representatives." https://www.stat.berkeley.edu/~stark/Preprints/bmd19.pdf.

Stark, Philip B., and David A. Wagner. 2012. "Evidence-Based Elections." *IEEE Security and Privacy* 10: 33–41.

Wallach, D. 2019. "On the Security of Ballot Marking Devices." https://arxiv.org/pdf/1908.01897.pdf.