Security & Privacy of Internet Voting in U.S. Federal Elections

Michael A. Specter // @mspecter // specter@mit.edu
MIT EECS PhD Candidate

Feb 19, 2021
Encryption & Surveillance

Keys Under Doormats: Mandating Insecurity by Requiring Government Access to All Data and Communications
Harold Abelson, et. al. (List is alphabetical) // Oxford Journal of Cybersecurity & Communications of the ACM

Practical Deniability & Cryptography

KeyForge: Mitigating Email Breaches with Forward-Forgeable Signatures
Michael A. Specter, Sunoo Park, and Matthew Green // To appear, USENIX Security 2021

Election Security

The Ballot is Busted Before the Blockchain: A Security Analysis of Voatz
Michael A. Specter, James Koppel, and Daniel Weitzner // USENIX Security 2020

Security Analysis of the Democracy Live Online Voting System
Michael A. Specter, and J. Alex Halderman // To appear, USENIX Security 2021

Going from Bad to Worse: From Internet Voting to Blockchain Voting
Sunoo Park, Michael A. Specter, Neha Narula, Ron Rivest // To appear, Oxford Journal of Cybersecurity
Systems Security & Applied Crypto // Policy

Encryption & Surveillance

Keys Under Doormats: Mandating Insecurity by Requiring Government Access to All Data and Communications
Harold Abelson, et. al. (List is alphabetical) // Oxford Journal of Cybersecurity & Communications of the ACM

Practical Deniability & Cryptography

KeyForge: Mitigating Email Breaches with Forward-Forgeable Signatures
Michael A. Specter, Sunoo Park, and Matthew Green // To appear, USENIX Security 2021

Election Security

The Ballot is Busted Before the Blockchain: A Security Analysis of Voatz
Michael A. Specter, James Koppel, and Daniel Weitzner // USENIX Security 2020

Security Analysis of the Democracy Live Online Voting System
Michael A. Specter, and J. Alex Halderman // To appear, USENIX Security 2021

Going from Bad to Worse: From Internet Voting to Blockchain Voting
Sunoo Park, Michael A. Specter, Neha Narula, Ron Rivest // To appear, Oxford Journal of Cybersecurity

This talk
Outline

I. Why voting is hard
II. The curious case of Voatz
III. Reaction and impact
IV. Bigger picture
Outline

I. Why voting is hard
II. The curious case of Voatz
III. Reaction and impact
IV. Bigger picture
What do we want from a voting system?

- **Correctness & Usability**
  - Counted as cast, cast as intended, accessible to all eligible voters

- **Privacy**
  - An attacker cannot learn a voter’s selections

- **Receipt Freeness**
  - No voter can prove the way they voted after the fact

- **Coercion Resistance**
  - Voter cannot cooperate with an attacker to prove the way they voted

- **End to end verifiability (E2E-V)**
  - Voters can prove to themselves that a vote was counted correctly
How is voting “secured”? 
The Dream: Replace Trust
MODERNIZE VOTING

Considering the advances in technology, there's no reason why we should still be waiting in line at polling stations to cast votes. The machines being used in most locales are also as vulnerable to tampering and hacking as modern technologies. Americans should be able to vote via their mobile device, with verification done via blockchain. This would dramatically increase participation in all elections, whether local, state or federal.

HELP MAKE THIS IDEA A REALITY.

It's ridiculous that in 2020 we are still standing in line for hours to vote in antiquated voting booths. It is 100% technically possible to have fraud-proof voting on our mobile phones today using the blockchain. This would revolutionize true democracy and increase participation to include all Americans - those without smartphones could use the legacy system and lines would be very short.
Can blockchains be useful for voting?

<table>
<thead>
<tr>
<th>Going from Bad to Worse: From Internet Voting to Blockchain Voting</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sunoo Park</td>
</tr>
<tr>
<td>MIT &amp; Harvard*</td>
</tr>
<tr>
<td>April 8, 2020</td>
</tr>
</tbody>
</table>

Abstract

Voters are understandably concerned about election security. We examine the suggestions that "vote-online seems tantalizingly convenient: just a few taps on a phone from anywhere, without breaking your daily routine, taking off from work, or waiting in line. However, voting online has a fatal flaw.

- Came up with a few schemes
  - E.g. Tokens + ZCash-like Shielded Transactions
- Result was "no, not really"
- Internet Voting in general → Scalable & Remote attacks
Outline

I. Why voting is hard

II. The curious case of Voatz
   A. Motivation
   B. Methodology and reverse engineering
   C. Attacks and results

III. Reaction and impact

IV. Bigger picture
Outline

I. Why voting is hard
II. The curious case of Voatz
   A. Motivation
   B. Methodology and reverse engineering
   C. Attacks and results
III. Reaction and impact
IV. Bigger picture
WEST VIRGINIA LEGISLATURE
2020 REGULAR SESSION

Enrolled
Committee Substitute
for
Senate Bill 94

SENATORS TRUMP, WELD, AZINGER, BALDWIN, BEACH, CLEMENTS, CLINE, HARDESTY, JEFFRIES, LINDSAY, MAYNARD, PITSENBERGER, ROMANO, RUCKER, SMITH, TAKUBO, WOELFEL, HAMILTON, STOLLINGS, IHLENFELD, AND SYMPHONY, original sponsors

[Passed January 24, 2020; in effect from passage]

AN ACT to amend and reenact §3-3-1, §3-3-2, §3-3-2b, §3-3-4, §3-3-5, and §3-3-6 of the Code of West Virginia, 1931, as amended; and to amend said code by adding thereto a new section, designated §3-3-1a, all relating generally to absentee voting; clarifying that voters with disabilities prevented from voting in person may vote by mail-in absentee ballot; providing that voters with physical disabilities may vote by electronic absentee ballot; clarifying that certain overseas military members and citizens may vote by electronic absentee ballot; defining terms; providing that a voter with a physical disability may electronically submit an application to vote by absentee ballot; and making other changes.
22% of adults have “Serious Difficulty Walking or Climbing Stairs” - CDC
(https://www.cdc.gov/ncbddd/disabilityandhealth/impacts/west-virginia.html)
Questions

- How well does Voatz achieve the security desiderata?
  - Correctness, Privacy, Receipt Freeness, Coercion Resistance

- Advertised use of cryptography:
  - Hardware-backed key storage!
  - Mixnets!
  - And, of course, the Blockchain!
  - Is this End to end verifiable (E2E-V)?
A quick glance looks good!

- Bug bounty via HackerOne
- Security Audits
- There’s even some documentation!

https://jeffleejohnson.com/site/Horror.html
Documentation
Documentation

- No formal description of their system
- Security reviews
  - Not public
  - No list of fixed vulns
  - Done by National Cybersecurity Center & ShiftState Security
FAQ’s interesting crypto claims:

- “doubly anonymized”
- “end-to-end vote encryption”
- “anonymized voter-verified digital receipts”
- “voter-verified audit trail”
Bug Bounty
• “Special” version of the app via HackerOne
  ○ No documentation of differences with production

• Limited scope
  ○ No MITM attacks
  ○ Nothing requiring physical access
    ■ No definition of physical access

• Limited test infrastructure
● “Special” version of the app via HackerOne
  ○ No documentation of differences with production

● Limited scope
  ○ No MITM attacks
  ○ Nothing requiring physical access
    ■ No definition of physical access

● Limited test infrastructure
Both apps are obfuscated!
Android RE:

- Android App’s are APK’s
- Zip files w/ java classes & XML
- Decompile to Source!
38

```java
184 public final class C0036 {
    // renamed from: os reason: contains not printable characters */
    private static C1668 f3263 = new C1668<>(Object.class);
    // renamed from: os reason: contains not printable characters */
    private Map<Type, C0734> f3264;
    // renamed from: os reason: contains not printable characters */
    private boolean f3265;
    // renamed from: os reason: contains not printable characters */
    private C1938 f3266;
    // renamed from: os reason: contains not printable characters */
    private C3695 f3267;
    // renamed from: os reason: contains not printable characters */
    private List<C0943> f3268;
    // renamed from: os reason: contains not printable characters */
    private List<C0943> f3269;
    // renamed from: os reason: contains not printable characters */
    private ThreadLocal<Map<C1668, C0734>> f3270;
    // renamed from: os reason: contains not printable characters */
    private boolean f3271;
    // renamed from: os reason: contains not printable characters */
    private boolean f3272;
    // renamed from: os reason: contains not printable characters */
    public final boolean f3273;
    // renamed from: os reason: contains not printable characters */
    private boolean f3274;
    // renamed from: os reason: contains not printable characters */
    }
Functions, classes, variables → all renamed into unicode!
There’s more: String Obfuscation

// Normal Code
KeyAgreement instance = KeyAgreement.getInstance("ECDH");

// Code from Voatz:
KeyAgreement instance = KeyAgreement.getInstance(DeobfuscateSubString(29, 0, 4));
// Not shown: Roughly 200 lines of obfuscation

// Obfuscated string including all crypto parameters.
// E.G. “AESGCM” is somewhere in this byte array
public static char[] obfuscatedByteString = new char[]{
    12425, 28648, 36421, 12031, 19775, 60832, 3082, 44217, 52223, 27254,
    35524, ...
};

// Deobfuscates string.
public static String DeobfuscateSubString(int startIndex, char c, int len) {
    char[] cArr = new char[len];
    for (int i = 0; i < len; i++)
        cArr[i] = obfuscatedByteString[startIndex + i] ^ (i * 2859717209460793199L) ^ c;
    return new String(cArr);
}
Remember:
This is the bug bounty app.
“Provide us with a **reasonable amount** of time to resolve vulnerabilities prior to any disclosure to the public or a third-party.”

“Voatz **retains the authority** to determine what issues and / or vulnerabilities can and should be remediated and **within what time frame**.”
Voatz can hold on to bugs indefinitely.
"We just noticed a certain group of people from a certain part of the country tried to access the system. We stopped them, caught them and reported them to the authorities"

-- Voatz’s Nimit Sawhney, CEO
Red Flags:

- No formal documentation
- Obfuscated code
- Dodgy bug bounty

They’re trying to make it really hard to do analysis!
This is a huge problem!
This is a huge problem!
Asymmetric Information & Market Failure
Outline

I. Why voting is hard
II. The curious case of Voatz
   A. Motivation
   B. Methodology and reverse engineering
   C. Attacks and results
III. Reaction and impact
IV. Bigger picture
Methodology

● Challenge:
  ○ Can’t touch server infrastructure (legal & ethical concerns)
  ○ Must make assumptions about the backend

● Solution:
  ○ Manually reverse engineer the Voatz Android app
  ○ Iteratively reimplement the server to understand protocol
  ○ Assume the best possible situation for the backend
A tour of Voatz
What is Voatz?

Voatz is an elections platform to vote securely and remotely on your smartphone.

- Enter the code we just texted:

111111

Choose an 8-digit PIN to secure your account. Your pin cannot contain 3 or more repeating numbers.

Enable Fingerprint Log In: [ ]

Log In

Verify your fingerprint to continue

Touch sensor
Ready to verify your vote?

After you finish voting, an encrypted ballot receipt will be sent to the email address you used to sign up for Voatz.

In order to access your ballot receipt, you'll need to enter the following 16-digit password:

(Press and hold the box to view.)

............... 

Need assistance? Please contact us.
Behind the scenes
Voatz is a REST app.
Voatz is a REST app.
Voatz is a REST app.
Network:

VOATZ

voatzapi.nimsim.com

OOB Device Verification?

Alert on malware

HTTPS get/put requests

ID Info

OOB ID verification

JUMIO
Outline

I. Why voting is hard
II. The curious case of Voatz
   A. Motivation
   B. Methodology and reverse engineering
   C. Attacks and results
III. Reaction and impact
IV. Bigger picture
# Attacks

<table>
<thead>
<tr>
<th></th>
<th>Stop ballot</th>
<th>Learn ballot</th>
<th>Alter ballot</th>
<th>Learn Voter PII</th>
</tr>
</thead>
<tbody>
<tr>
<td>On-Device Malware</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Company Servers</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Network Adversary</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3rd Parties</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Not E2E-V, receipt free, or coercion resistant
Scenario 1: ‘Attacker w/ root on-device’

voatzapi.nimsim.com

HTTPS
get/put requests

voatz

Jumio
Zimperium’s Malware Detection

- Initialized on app-creation
  - Scans immediately & after every “app resume”
- Looks for:
  - Known exploits / malware
  - Any indicators of jailbreak, debugging, or modding
- It’s a snitch!
  - If caught, will alert both the Voatz API server & Zimperium’s servers!
Defeating Zimperium

You can’t catch me if you never run!

@NgModule
class z_threatDetector = lpparam.classLoader.loadClass("Zimperium.DetectionStateCallback");
Class z_threatCallback = lpparam.classLoader.loadClass("Zimperium.ThreatCallback");

findAndHookMethod("zimperium.ZDetection", lpparam.classLoader,"addDetectionStateCallback",
    z_threatDetector, new XC_MethodHook() {
        @Override
        protected void beforeHookedMethod(MethodHookParam param) throws Throwable {
            param.setResult(null);
        }
    });

findAndHookMethod("zimperium.ZDetection", lpparam.classLoader, "detectCriticalThreats", Context.class, z_threatCallback, new XC_MethodHook() {
    @Override
    protected void beforeHookedMethod(MethodHookParam param) throws Throwable {
        param.setResult(null);
    }
});
Without Zimperium:

- Attacker has complete control of the user
- Receipts are meaningless, user’s email is likely compromised as well
- There’s not much that can be done to prevent this
  - Voatz is not E2E-V
We can do better.
We can do better. What if we only get access after the fact?
On-disk encryption

- Encrypted DB contains:
  - User’s Vote history
  - Everything used to authenticate the user
- Secret Key = $KDF(PIN + salt)$
  - Salt stored on disk, unencrypted.
- **PIN** is stored on-disk, but encrypted
  - Using Android keystore!
  - Keys stored in hardware enclave!
  - Requires user’s fingerprint to decrypt!
  - This is actually kind of OK!
But, PIN = 8 digits. Numeral only.

- ~$10^8$ combinations = 100,000,000 PINs
- Copying from device → laptop to brute force
  - ~.05 ms per attempt = max ~1.6 hours
Pin = 8 digits. Numeral only.

- ~10^8 combinations = 100,000,000
- ~.05 ms per attempt = max ~1.6 hours

Classes:
- ChoiceRealm 1
- LoginInfo 1
- NotificationRealm 1
- QuestionRealm 1
- StatementRealm 1
- VoteHistoryInfo 1

LoginInfo:
- customerId (Primary Key): int 952804346
- nextKey: int 1337
- auditToken: string SomeAuditTokenValue
Anyone with forensic access can brute force the pin, control the voter, and see how they voted.
## Attacks

<table>
<thead>
<tr>
<th></th>
<th>Stop ballot</th>
<th>Learn ballot</th>
<th>Alter ballot</th>
<th>Learn Voter PII</th>
</tr>
</thead>
<tbody>
<tr>
<td>On-Device Malware</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Company Servers</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Network Adversary</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3rd Parties</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Not E2E-V, receipt free, or coercion resistant
Scenario 2: Attacker controls the API server
Custom Crypto Protocol!
Gen 100 ECDSA Key Pairs. Discards all secret keys, except #57

Perform key agreement & decrypt AESGCM_{sk}

ECDSA_{Enc}(\text{Key}_{57}, \text{AESGCM}_{sk}), 100 PubKeys

All comms Enc(AESGCM_{sk'}, *)

Gen 100 ECDSA key pairs
Key_{57} \leftarrow \text{Key agreement with the sender’s 57th key}
AESC_{sk} \leftarrow \$ R
...And that was 100% of the crypto.

- No communication with the blockchain.
  - “Proofs of inclusion”
- No verification of server’s public key
  - Active MITM still possible
- No non-ephemeral public key from device
- Nothing is ever signed

Standard HTTPS

100 ECDSA Pubkeys

ecdsa(key_{SECP256K1}, AESGCM_{sk}), 100 public keys

All comms Enc(AESGCM_{sk^*})
The API server can do anything it wants.
Why?
## Attacks

<table>
<thead>
<tr>
<th></th>
<th>Stop ballot</th>
<th>Learn ballot</th>
<th>Alter ballot</th>
<th>Learn Voter PII</th>
</tr>
</thead>
<tbody>
<tr>
<td>On-Device Malware</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Company Servers</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Network Adversary</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3rd Parties</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Not E2E-V, receipt free, or coercion resistant**
Scenario 3: ‘Passive Network Attack’
Short_Candidate = {
    "choiceDetails": {
        "imageUrl": "short.com/img.jpg",
        "webUrl": "short.com/desc"
    },
    "choiceId": "1",
    "description": "Short",
    "description 1": "^",
    "description 2": "^",
    "isWriteIn": False,
    "nonSelectable": False
}

Long_Candidate = {
    "choiceDetails": {
        "imageUrl": "www.LONG_IMG_URL.info/LONG_IMG_URL.jpg",
        "webUrl": "www.LONG_IMG_URL.info/Long_Candidate_Info"
    },
    "choiceId": "2",
    "description": "Long Description !",
    "description 1": "See? It’s super long .REALLLY long.111111",
    "description 2": "EPICALLYLOOOOOOOOONG....",
    "isWriteIn": False,
    "nonSelectable": False
}
When the user submits their ballot

- You might expect it to just send id #’s
- Nope
- Sends *all* metadata of the voter’s choice
- But only that candidate’s metadata

```json
Short_Candidate = {
    "choiceDetails": {
        "imageUrl": "short.com/img.jpg",
        "webUrl": "short.com/desc"
    },
    "choiceId": "1",
    "description": "Short",
    "description 1": "^",
    "description 2": "^",
    "isWriteIn": False,
    "nonSelectable": False
}

Long_Candidate = {
    "choiceDetails": {
        "imageUrl": "www.LONG_IMG_URL.info/LONG_IMG_URL.jpg",
        "webUrl": "www.LONG_IMG_URL.info/Long_Candidate_Info"
    },
    "choiceId": "2",
    "description": "Long Description !",
    "description 1": "See? It's super long.REALLLY long.111111",
    "description 2": "EPICALLYLOOOOOOOOOONG....",
    "isWriteIn": False,
    "nonSelectable": False
}
```
Ciphertext Size Leaks the Voter’s Selections

Regular HTTPS:
- Ciphertext $= \text{AES(gzip(data))}$
- $\text{len(ciphertext)} \approx \text{len(gzip(data))}$

Voatz Protocol:
- Ciphertext $= \text{AES(gzip(AES(data))}$
- $\text{len(ciphertext)} \approx \text{len(data)}$
# Attacks

<table>
<thead>
<tr>
<th></th>
<th>Stop ballot</th>
<th>Learn ballot</th>
<th>Alter ballot</th>
<th>Learn Voter PII</th>
</tr>
</thead>
<tbody>
<tr>
<td>On-Device Malware</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
</tr>
<tr>
<td>Company Servers</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
</tr>
<tr>
<td>Network Adversary</td>
<td>✔</td>
<td>✔</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3rd Parties</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Not E2E-V, receipt free, or coercion resistant
Scenario 4:
‘Privacy, and Informed Consent’
Jumio ID Verification

- Liveness Detection
- ML match “selfie” & Voter’s ID
- OCR

- ...Not done on device.
Privacy & Jumio

- Jumio’s servers get:
  - Both of these images →
  - + GPS location
Privacy & Jumio

- Jumio’s servers get:
  - Both of these images →
  - + GPS location

- The only place Jumio is mentioned in-app →
Privacy & Jumio

- Jumio’s servers get:
  - Selfie photo
  - Photo of your ID
  - Your location & IP

- 100% of the times Jumio is mentioned in-app

---

Pr
## Summary of Discoveries

<table>
<thead>
<tr>
<th></th>
<th>Stop ballot</th>
<th>Learn ballot</th>
<th>Alter ballot</th>
<th>Learn Voter PII</th>
</tr>
</thead>
<tbody>
<tr>
<td>On-Device Malware</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Company Servers</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Network Adversary</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3rd Parties</td>
<td>✓</td>
<td></td>
<td></td>
<td>✓</td>
</tr>
</tbody>
</table>

Not E2E-V, receipt free, or coercion resistant
Outline

I. Why voting is hard
II. The curious case of Voatz
III. Reaction and impact
   A. Disclosure Process
   B. Public Reaction
   C. Another Case Study
IV. Bigger picture
Responsible Disclosure
How do you begin to report this?
“Given that our elections infrastructure is classified as critical infrastructure under the Department of Homeland Security, we will continue to report any such attempts in the future.”

- Voatz CEO Nimit Sawhney

OK, then.
The Ballot is Busted Before the Blockchain:
A Security Analysis of Voatz, the First Internet Voting Application Used in U.S.
Federal Elections*

Michael A. Specter

James Koppel

Daniel Weitzner

Abstract

In the 2018 midterm elections, West Virginia became the first state in the U.S. to allow select voters to cast their ballot on a mobile phone via a proprietary app called “Voatz.” Although there is no public formal description of Voatz’s se-

The company has recently closed a $7-million series A [22], and is on track to be used in the 2020 Primaries.

In this paper, we present the first public security review of Voatz. We find that Voatz is vulnerable to a number of attacks that could violate election integrity (summary in Table 1). For
Voatz smartphone voting app has significant security flaws, MIT researchers say

Researchers did not say they found evidence that the app had been hacked, but they said the vulnerabilities could have been exploited.

Security experts raise concerns about voting app used by military voters

By Brian Fung, CNN
Updated 5:29 PM ET, Fri February 14, 2020
E-voting is a bad idea. A team of MIT researchers finds that Voatz pairs a bad idea with bad execution.

The Ballot is Busted Before the Blockchain: A Security Analysis of Voatz, the First Internet Voting Application Used in U.S. Federal Elections

Abstract

In the 2018 midterm elections, West Virginia became the first state in the U.S. to allow voters to cast their ballots using a mobile voting app called “Voatz.” Although there is no public formal description of Voatz’s security model, the company claims that election security and integrity are maintained through the use of a permissioned blockchain. However, a team of researchers explored the app's vulnerabilities.

Not to in any way diminish this (excellent) work, but the fact that an online mobile voting scheme has serious security flaws is ultimately unsurprising. Every serious expert has warned against Internet voting. The people making the extraordinary claims here are Voatz, not MIT.

Today computer scientists at @MIT issued a security analysis of the Voatz voting app. Spoiler alert — they found numerous security and privacy vulnerabilities. An app is used in several states. #ElectionSecurity

Today, myself and co-authors @jimmykoppel and @djweitzner released a paper discussing a slew of vulnerabilities we found in Voatz, a blockchain voting app that's been used in US federal elections. You can read about it in the @nytimes! nytimes.com/2020/02/13/us/...
"I commend the team from MIT for showing yet again, that internet voting is dangerous."

--- Senator Ron Wyden, Keynote @ DEFCON
February 21, 2020

Andre McGregor
Chief Security Officer
ShiftState Security
2826 Hudson Street
Denver, CO 80207

Dear Mr. McGregor:

I write to seek information about ShiftState Security’s cybersecurity audit of the mobile voting app Voatz.
"...[MIT’s] true aim is to deliberately disrupt the election process, to sow doubt in the security of our election infrastructure, and to spread fear and confusion."

https://blog.voatz.com/?p=1243
“...version of the Voatz mobile voting app that was at least 27 versions old at the time of their disclosure and not used in an election.”

“...the researchers fabricated an imagined version of the Voatz servers, hypothesized how they worked...”

“The unit has security software that was two generations old, and to our knowledge is not used anywhere in the country.”

“By any standard – academic or common sense – the study is unrealistic and inaccurate.”

https://blog.voatz.com/?p=1243

Trail of Bits report

- Validated our Methodology
- ToB confirmed our vulnerabilities & severity
  - *Before Voatz spoke to the press*
- Zimperium wasn’t running during pilots
- Confirmed *not* E2E-V
- +40 other bugs/vulnerabilities.
In the
Supreme Court of the United States

NATHAN VAN BUREN,

Petitioner,

v.

UNITED STATES,

Respondent.

ON WRIT OF CERTIORARI TO THE UNITED STATES COURT OF APPEALS FOR THE ELEVENTH CIRCUIT

BRIEF OF AMICI CURIAE COMPUTER SECURITY RESEARCHERS, ELECTRONIC FRONTIER FOUNDATION, CENTER FOR DEMOCRACY & TECHNOLOGY, BUGCROWD, RAPID7, SCYTHER, AND TENABLE IN SUPPORT OF PETITIONER
No. 19-783

In the
Supreme Court of the United States

NATHAN VAN BUREN, Petitioner,
v.
UNITED STATES, Respondent.

ON WRIT OF CERTIORARI TO THE
UNITED STATES COURT OF APPEALS FOR THE ELEVENTH CIRCUIT

BRIEF OF VOATZ, INC., AS AMICUS CURIAE
IN SUPPORT OF RESPONDENT IN
AFFIRMANCE OF THE DECISION BELOW

JARED L. HUBBARD
(Counsel of Record)
FITCH LAW PARTNERS LLP
One Beacon Street
Boston, MA 02108
(617) 542-5542
jh@fitchlp.com
Counsel for Amicus Curiae Voatz, Inc.

Dated: September 3, 2020
Response to Voatz’s Supreme Court Amicus Brief

September 14, 2020

On September 3, 2020, Voatz — a blockchain voting company with a publicly documented track record of hostility towards security research — filed an amicus brief with the U.S. Supreme Court in Van Buren v. United States arguing in favor of a broad interpretation of the Computer Fraud and Abuse Act (CFAA), the federal anti-hacking law enacted in 1986. Voatz’s amicus brief repeatedly refers to independent good-faith security research as a threat to cybersecurity and glosses over harmful effects to security research that would result from an overbroad CFAA.

As representatives of the security community, including pioneers of coordinated vulnerability disclosure, bug bounties, and election security, it is our opinion that Voatz’s brief to the Court fundamentally misrepresents widely accepted practices in security research and vulnerability disclosure, and that the broad interpretation of the CFAA threatens security research activities at a national level. We stand in support of the petitioner, reiterating arguments made by the Computer Security Researchers, Electronic Frontier Foundation, et al., and many others advocating a narrow interpretation of the CFAA in which contractual violations do not constitute CFAA violations.

Security research is vital to the public interest.

We benefit from security research in nearly every aspect of our lives. From crucial work exposing vulnerabilities in technologies ranging from election systems to medical devices and automobiles, it is clear security research has tangibly improved the safety and security of systems we depend upon. It is not a given that this vital security work will continue. A broad interpretation of the CFAA would magnify existing chilling effects, even when there exists a societal obligation to perform such research.

Jared Fisch
(Fish & Richardson)
One Beacon Street
Boston, MA 02108
(617) 573-2500
jfish@fish.com

Dated: September 3, 2020
Response to Voatz’s Supreme Court Amicus Brief

September 14, 2020

On September 3, 2020, Voatz — a blockchain voting company with a publicly documented track record of hostility towards security research⁵ — filed an amicus brief with the U.S. Supreme Court in Van Buren v. United States arguing in favor of a broad interpretation of the Computer Fraud and Abuse Act (CFAA), the federal anti-hacking law enacted in 1986.²³ Voatz’s amicus brief repeatedly refers to independent good-faith security research as a threat to cybersecurity and glosses over harmful effects to security research that would result from an overbroad CFAA.

As representatives of the security community, including pioneers of coordinated vulnerability disclosure, bug bounties, and election security, it is our opinion that Voatz’s brief to the Court fundamentally misrepresents widely accepted practices in security research and vulnerability disclosure, and that the broad interpretation of the CFAA threatens security research activities at a national level. We stand in support of the petitioner, reiterating arguments made by the Computer Security Researchers, Electronic Frontier Foundation, et al.,⁴ Orin S. Kerr,⁵ Technology Companies,⁶ and many others advocating a narrow interpretation of the CFAA in which contractual violations do not constitute CFAA violations.

Security research is vital to the public interest.

We benefit from security research in nearly every aspect of our lives. From crucial work exposing vulnerabilities in technologies ranging from election systems⁷ to medical devices⁸ and automobiles,⁹ it is clear security research has tangibly improved the safety and security of systems we depend upon. It is not a given that this vital security work will continue. A broad interpretation of the CFAA would magnify existing chilling effects, even when there exists a societal obligation to perform such research.

Dated: September 3, 2020
The State’s Reaction, and a New System

- West Virginia (+4 other jurisdictions) drop Voatz, citing security concerns
  - Still used in Utah County, Utah
- West Virginia, Delaware, and New Jersey announce adoption of new system
  - DemocracyLive’s OmniBallot
  - Delaware enabled for *all* voters in the 2020 primary
  - Also various counties in WA & OR
How would you like to return your ballot?

Select how you would like to return your ballot from the buttons shown below.

PLEASE NOTE: Any marked ballots submitted via Electronic Return produce a printed paper ballot that is in the same format all other paper absentee ballots received by mail or submitted by in-office absentee voting. These ballots are scanned for tabulation with other absentee ballots. No votes are cast online under any circumstances.
How would you like to return your ballot?

Select how you would like to return your ballot from the buttons shown below.

PLEASE NOTE: Any marked ballots submitted via Electronic Return produce a printed paper ballot that is in the same format all other paper absentee ballots received by mail or submitted by in-office absentee voting. These ballots are scanned for tabulation with other absentee ballots. No votes are cast online under any circumstances.
Submitting Bug or Vulnerability Reports

Bug reports can be sent to security@democracylive.com with the subject line of Bug Report – DD-MM-YYYY

By submitting a report, you agree to not disclose that bug to any third parties without the approval of Democracy Live. We ask that bug reports follow the standard Github format:
Security Analysis of the Democracy Live Voting System

Michael A. Specter\textsuperscript{1} and J. Alex Halderman\textsuperscript{2}

\textsuperscript{1} MIT
specter@mit.edu

\textsuperscript{2} University of Michigan
jhalderm@eecs.umich.edu

June 7, 2020
Quick Summary:

- **Same methodology:**
  - Reverse Engineer & Reimplement

- **Results:**
  - Not E2E-V (no crypto other than HTTPS).
  - Sends voter’s private selections to their server *even if you print and physically mail the ballot.*
  - Collects voter’s identity, ballot selections, and browser fingerprint
  - Uses Google Analytics, and Google gets your voter ID.

- **No privacy policy.**
Attacks on Democracy Live’s OmniBallot:

<table>
<thead>
<tr>
<th></th>
<th>Stop ballot</th>
<th>Learn ballot</th>
<th>Alter ballot</th>
<th>Learn Voter PII</th>
</tr>
</thead>
<tbody>
<tr>
<td>On-Device Malware</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Company Servers</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>3rd Parties</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Google, Cloudflare, &amp; Amazon</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
</tbody>
</table>

Not E2E-V, receipt free, or coercion resistant
Outline

I. Why voting is hard
II. The curious case of Voatz
III. Reaction & Impact
IV. Bigger picture
How did we get here?
It’s not that the cybersecurity people are bad people, *per se*.

I think it’s that they are solving for one situation, and I am solving for another. They want zero technology risk in any way, shape, or form. [...] I am solving for the problem of turnout.

- Bradley Tusk, Voatz Backer & Mobile Voting Project founder
Argument:
Solve a **Policy Problem** with more **Technology**
Problem: Asymmetric Information → Market Failure
What can we do?
Elections are Complex
Sociotechnical Systems
Future Work

● Security Analysis
  ○ Voter registration databases, Pollbooks, Election Night Reporting systems

● Cryptography & Systems
  ○ Many of the definitions in cryptography and voting are *bad*
  ○ E2E-V for mail-in ballots
  ○ Dispute resolution for E2E-V
  ○ Authenticated Data Structures & Voter registration databases

● HCI & Usability
  ○ How do you create a Ballot Marking Device (BMD) that voters will actually verify?
  ○ How usable are the current BMDs for Poll Workers?
  ○ How can we protect voters in the case of intimate partner abuse?
Don’t fall into this trap.