ALTERING ELECTRONIC BALLOTS USING PDF SCRIPTING

Henry D. Herrington
Princeton University
henryherrington@gmail.com

April 7, 2022

I. INTRODUCTION

In recent decades, it has become increasingly common to transmit absentee ballots electronically as PDFs in American elections. From a security perspective, PDF ballots may seem analogous to physical paper ballots because both are used to display voter preferences. However, PDFs have many additional capabilities, including the ability to run JavaScript code, that make them insecure for ballot usage. In this article, I will explain current PDF ballot usage in the US, and then demonstrate why PDF is an insecure file format in this context.

Current PDF Ballot Usage in the US

All states permit some form of absentee voting, but laws on which voters are eligible to receive and return electronic ballots vary from state to state [7]. Military and overseas voters, for example, can receive their ballots electronically in all states, and are permitted to return their ballots as PDF email attachments in over half of all states [1], [5]. In addition to the electronic distribution and submission of ballots, some states also provide electronic means of marking ballots, such as Nevada's EASE system. This service allows certain voters to receive and fill out their ballots electronically, and then download a PDF of their completed ballot to submit as an email attachment [8]. In Nevada, EASE is accessible for both military and overseas voters as well as voters with disabilities [8]. Similar commercial products exist that allow voters to receive and mark their ballots electronically, and then download their completed ballots to submit as PDFs. One such product is Democracy Live's Omniballot, which has been used in elections in over 20 states [3], [4].

PDF Capabilities

Although there are clear accessibility and convenience benefits associated with PDF ballots, sometimes less apparent are the security tradeoffs of this technology. Unlike paper, PDF files have many additional capabilities outside of displaying images that make them more exploitable as ballots. PDFs can, for example, play video and audio, open webpages, launch other files, and, most relevant to this article, execute internal JavaScript code. I will now demonstrate how this scripting capability of PDFs makes them insecure for ballot usage.

II. DEMONSTRATION

The following page contains a demonstration PDF ballot that was adapted from an existing sample ballot from Cass County, Missouri [2]. The ballot has already been marked electronically, and could easily be the downloaded product of some ballot marking software. At this point, take note of which candidates are voted for. Once you do, you might feel comfortable closing this document and sending it off to election officials to have your vote counted. A non-technical voter may assume that once they stop interacting with this document, their vote is essentially secure. However, this ballot will actually alter the candidates it votes for based on the current time. In a real election, this ballot would be designed to alter its votes after the ballot is transmitted. For demonstration purposes, this ballot will conveniently flip its votes every minute. Specifically, in even minutes, it will vote for Michael R. Brown, Bryan Cribbs, Emily Stone, and "No" on the middle proposition, and in odd minutes, it will vote for Chris Benjamin, Ryan P. Miller, Jenny Wagoner, and "Yes" on the middle proposition. Importantly, this type of PDF ballot hacking can occur even if the voter uses a legitimate ballot marking software that was not originally designed to produce corrupted ballots. So long as an election hacker compromises a voter's machine, they can corrupt this software to generate hacked PDF ballots like the one demonstrated. Please see Section VII on troubleshooting if you encounter any difficulties with the demonstration.

OFFICIAL BALLOT

GENERAL MUNICIPAL ELECTION
CASS COUNTY, MISSOURI
TUESDAY, APRIL 5, 2022

NOTICE OF ELECTION

Notice is hereby given that the General Municipal Election will be held in the County of Cass on Tuesday, April 5, 2022 as certified to this office by the participating entities of Cass County. The ballot for the Election shall be in substantially the following form.

JUNIOR COLLEGE DISTRICT OF METROPOLITAN KANSAS CITY, MISSOURI

FOR BOARD OF TRUSTEES
SUBDISTRICT NUMBER 6
SIX YEAR TERM

Vote for ONE

- [✓ **MICHAEL R. BROWN**]
- **CHRIS BENJAMIN**
- ______ **WRITE IN**

EAST LYNNE SCHOOL DISTRICT NO. 40

FOR BOARD MEMBER
THREE YEAR TERM

Vote for TWO

- **RYAN P. MILLER**
- [✓ **BRYAN CRIBBS**]
- ______ **WRITE IN**
- ______ **WRITE IN**

PROPOSITION STUDENTS, GROWTH, AND SAFETY

"Shall the Board of Education of the East Lynne School District No. 40, Missouri, borrow money in the amount of Five Hundred Thousand Dollars ($500,000) for the purpose of providing funds for site development, construction, equipping, and furnishing the reconfiguration of current spaces to address recent growth within the district, to replace and/or repair roofs; to implement safety and security improvements, including secure entrances; to the extent funds are available, complete other repairs and improvements to the existing facilities of the District; and issue general obligation bonds for the payment therof resulting in an estimated increase to the debt service property tax levy of $0.2400 per one hundred dollars of assessed valuation?

If this proposition is approved, the adjusted debt service levy of the School District is estimated to increase from $0.0000 to $0.2400 per one hundred dollars of assessed valuation of real and personal property."

- **YES**
- [✓ **NO**]

HARRISONVILLE R-IX SCHOOL DISTRICT

QUESTION:
To choose by ballot two (2) directors who shall serve as members of the Board of Education of said School District for a term of three (3) years each.

Vote for TWO

- **BRITTNEY SEXTON**
- [✓ **EMILY STONE**]
- **JENNY WAGONER**
- **DAVID W. REECE**
- **DAVID PETERMAN**
- ______ **WRITE IN**
- ______ **WRITE IN**

III. TECHNICAL DETAILS

The basis of this vulnerability is the use of PDF form fields, which can be altered programmatically using internal PDF scripting. I now will describe the specific form fields and PDF scripts in this demonstration.

A. Form Fields

PDFs support interactive form fields ranging from text boxes, to dropdowns, to checkboxes, to radio buttons, and more. All of these form fields are manipulable using internal JavaScript. This demonstration uses the simple button form field to display custom ballot bubbles. Button form fields can take the appearance of an image, and in this demonstration, each ballot bubble is created using two buttons – one button displaying an empty bubble and another displaying a full bubble. At any given time, one of these buttons is hidden and the other is visible.

These form fields are named to reflect which of the four votes they count towards and which candidate within that vote they represent. For example, Emily Stone's button form fields are named "vote4_cd2_empty" and "vote4_cd2_full", because she is the second candidate in the fourth vote.

B. Internal JavaScript

All form fields have a display property that can be set to hidden or visible, among other options, according to the fields of an Adobe-defined display object. So as an example, Emily Stone's ballot bubble could be filled with the following document-level JavaScript:

```
this.getField("vote4_cd2_empty").display = display.hidden;
this.getField("vote4_cd2_full").display = display.visible;
```

We have inserted a document-level JavaScript function called SelectBubble that, given a vote number and candidate number, essentially does the work of selecting that candidate and deselecting all other candidates in that vote.

The trick to altering votes in this seemingly static ballot, then, is just a matter of calling SelectBubble on non-user triggers. In this demonstration, the non-user trigger is a JavaScript interval that is started by a document-level script when the document is opened. Every second, this interval will run a function to check if the minute has changed, and when it has, it will execute a function UpdateForm which in turn makes calls to SelectBubble with predetermined arguments based on the minute parity.

IV. EXTENSIONS

This is a simple demonstration used to show an easily-detectable attack on a non-interactive native PDF ballot. However, the scripting principles underlying this demonstration apply to a much broader range of PDF ballots, and can be used in much sneakier attacks. I will briefly describe how PDF scripting can be used to manipulate interactive PDF ballots, as well as non-native PDF ballots such as scanned images.

This demonstration naturally applies to interactive PDF ballots, which are built using the same form fields that this malicious JavaScript targets. In the above demonstration, all button form fields have been set to read-only, so that they can no longer be interacted with by user clicks. This was done to mimic how a downloaded, marked, non-interactive ballot might look. However, by removing this read-only setting, this document could quickly be turned into an interactive ballot that allows users to mark their votes, with all of the demonstrated vulnerabilities.

Furthermore, even non-native PDF ballots, such as photos or scanned images of filled ballots, are liable to PDF scripting attacks. Because PDF scripting allows malicious code to hide and display images, a hidden image of an entire pre-filled ballot could easily be programmatically revealed to "overwrite" the entirety of a legitimate scanned ballot. This means that even non-native PDF ballots can still be manipulated in a compromised PDF.

In all of these ballot types, there also exist far more sophisticated scripting attacks. These attacks could, for example, manipulate ballots only after a specific date, could execute based on the machine's time zone, could remove candidates from blank ballots sent to voters, could allow hackers to target only a small percentage of corrupted ballots, and could programmatically delete evidence of tampering after the fact. Demonstrations and further discussion of many of these attack extensions are explored in the full version of this article [6].

V. CONCLUSIONS

In the domain of voting security and ballot technology, PDF files possess often overlooked capabilities that make them inherently at-risk for manipulation attacks. Although online voting might increase accessibility and voter turn out, it also makes the election process more dependent on the security of voters' and election officials' machines. If a voter is phished, if an election official or voter is using a corrupted PDF reader, or if a voter is using a once-legitimate ballot marking software that has been hacked, PDF ballots with malicious JavaScript code are liable to enter into the system, compromising the integrity of the election. It is particularly difficult to trust that third-party ballot marking software such as Omniballot is sufficiently protected against potential corrupting attacks, considering that many of these software manufacturers don't disclose their software to the public. Because it is infeasible to expect all voters and election officials to maintain secure machines and secure software, it is advisable to minimize the usage of PDF ballots in official elections.

VI. ACKNOWLEDGEMENTS

Thank you to my advisor Jennifer Rexford for her encouragement and countless contributions to this project. Thank you also to Andrew Appel, who initially suggested investigating PDF ballot manipulation and who regularly provided helpful guidance. Thank you also to Arvind Narayanan, Kartikeya Kandula, Matthew Bernhard, and Michael A. Specter for taking the time to meet with me and offer their expertise on both information security and online voting practices.

VII. TROUBLESHOOTING

If the included demonstration appears to not be working, please consider the following. This document must be in its original PDF form, and not in some other image file format. Additionally, it must be viewed in an advanced and up-to-date PDF viewer like Adobe Acrobat or the Google Chrome browser. Currently, the most up-to-date working versions of these applications are 22.001.20085 (Acrobat Reader DC and Acrobat Pro DC), and 99.0.4844.83 (Chrome). Other readers, like Apple's Preview, will restrict much PDF functionality such as the execution of JavaScript code, and will not run the demonstration even if up-to-date. An original version of this file can be found at: `https://github.com/henryprinceton/senior-thesis`.

REFERENCES

[1] 111th Congress, "Military and Overseas Voter Empowerment Act," 2009. [Online]. Available: https://www.fvap.gov/uploads/FVAP/Policies/moveact.pdf

[2] Cass County, Missouri, "April 2022 Sample Ballot," 2022. [Online]. Available: https://www.casscounty.com/DocumentCenter/View/2969/April-Sample-Ballot

[3] Democracy Live, "OmniBallot Portal." [Online]. Available: https://democracylive.com/omniballot-portal/

[4] ——, *OmniBallot Paper Return*, Oct 2020. [Online]. Available: https://www.youtube.com/watch?v=NFz5asVo5Qo

[5] Federal Voting Assistance Program, "Voting Assistance Guide." [Online]. Available: https://www.fvap.gov/guide/chapter2

[6] Henry D. Herrington, "Ballot Acrobatics: Altering Electronic Ballots using Internal PDF Scripting," Undergraduate Thesis, Princeton University, April 2022.

[7] National Conference of State Legislatures, "Voting Outside the Polling Place: Absentee, All-Mail and other Voting at Home Options," Mar 2022. [Online]. Available: https://www.ncsl.org/research/elections-and-campaigns/absentee-and-early-voting.aspx

[8] Office of the Nevada Secretary of State, "EASE - Overview." [Online]. Available: https://www.nvsos.gov/sos/elections/voters/uniformed-overseas-citizens/ease-overview