ALTERING VOTES IN ELECTRONIC BALLOTS USING PDF SCRIPTING

Henry Herrington, advised by Jennifer Rexford
Princeton University, Princeton – New Jersey, USA
e-mail: henrydh@princeton.edu

I. INTRODUCTION

Recent decades have seen an increased use of digital technology in American elections. In all states, some voter group, such as military and overseas voters, has the right to receive and return their absentee ballots electronically, often as PDFs.

It may seem intuitive that from a security perspective, PDF ballots are analogous to physical paper ballots that simply display a voter's preferences. However, what many don't realize is that PDFs are capable of much more than simply displaying images. For example, they can play video, launch applications, and, most relevant to this article, execute JavaScript code. I will demonstrate how this internal scripting capability of PDFs makes them inherently insecure for ballot usage, and discuss the consequences of this insecurity.

II. DEMONSTRATION

The following page contains a PDF ballot, altered to include interactive PDF forms. This ballot is meant to represent how a real interactive PDF ballot might behave. At this point, feel free to mark your vote on this PDF ballot. Once you do so, you can even imagine 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, as you may notice, this ballot is able to change the vote it contains, even after it has been manually marked. Specifically, every minute, the ballot will alter itself to vote for Michael R. Brown, Bryan Cribbs, Emily Stone, and "No" on the middle proposition.

III. TECHNICAL DETAILS

The basis of this vulnerability is the use of PDF form fields, which can be altered programatically 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 buttons, to radio buttons, and more. All of these form fields are manipulable using internal JavaScript. While native radio button forms match the functionality of ballot bubbles, their visual display is not flexible, and so this demonstration uses the button form field to display custom ballot bubbles. Each ballot bubble is created using two PDF form field buttons – one button displaying an empty bubble and another displaying a full bubble.

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 bubble forms are named "vote4_cd2_empty" and "vote4_cd2_full", because she is the second candidate in the fourth vote.

B. Internal JavaScript

Each of these form fields is programmed to run a JavaScript once clicked to toggle that bubble's appearance. All form fields have a display property that can be set to hidden or visible, among other options, according to the fields of the Adobe-defined display object. So as an example, Emily Stone's ballot bubble could be filled with the following code:

```
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 and candidate number, essentially does the work of selecting that candidate and deselecting all other candidates in that vote.

Altering votes without explicit user input, then, is just a matter of calling SelectBubble based on different non-user triggers. In this demonstration, the non-user trigger is a JavaScript interval that is started by a document-level script called when the document is opened. This interval continuously runs a function checkMinuteChange to check if the minute has changed, and when it has, is executes a function UpdateForm which calls SelectBubble with predetermined arguments and clears all write-in text form fields.

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

IV. EXTENSIONS

This is a simple demonstration used to show a highly-detectable attack on a specific type of ballot. However, the internal scripting capabilities of PDFs make them security vulnerabilities in many other voting contexts. For example, more sophisticated scripting attacks could manipulate ballots only after the voting deadline, could execute based on 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 programatically delete evidence of tampering after the fact. Furthermore, because PDF scripting allows malicious code to hide and display images, a hidden image of a pre-filled ballot could also be programatically revealed to "overwrite" the entirety of a legitimate ballot. This means that even PDF ballots that do not use form fields to record voter preferences can still be manipulated in a compromised PDF. Demonstrations and a more rigorous discussion of many of these attack extensions are explored in the full version of this article.

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' personal machines. If either party has a corrupted PDF reader, or if voters are phished, PDF ballots with malicious JavaScript code are liable to enter into the system, compromising the integrity of the election. Because it is infeasible to expect all voters and election officials to maintain secure personal machines and follow secure technical voting protocols, it is advisable to minimize the usage of PDF ballots in official elections.

VI. ACKNOWLEDGEMENTS

Thank you to Jennifer Rexford for her many thoughtful contributions to this project, and to Andrew Appel, who initially suggested investigating PDF ballot manipulation and who regularly provided helpful guidance. Thank you also to Arvind Narayanan, Kart Kandula, Michael A. Specter, and Matt Bernhard for taking the time to meet with me and offer their expertise on both information security and online voting practices. This article formatting adapted from a template by the Brazilian Power Electronics Journal.

VII. TROUBLESHOOTING

If the included demonstration appears to not be working, please consider the following. This document must be a PDF file, and not some other image file. Additionally, it must be viewed in a relatively advanced PDF viewer like Adobe Acrobat, or even in a browser like Chrome or Firefox. Other readers like Apple's Preview restrict much PDF functionality such as the execution of JavaScript code, and will not run the demonstration. An original version of this file can be found at: https://github.com/henryprinceton/senior-thesis-demos.