Punchscan: Introduction and System Definition of a High-Integrity Election System

Kevin Fisher, Richard Carback and Alan T. Sherman
Center for Information Security and Assurance (CISA)
Department of Computer Science and Electrical Engineering
University of Maryland, Baltimore County (UMBC)
May 2006

Punchscan is a unique hybrid paper/electronic voting system concept. As a receipt-based system, Punchscan provides high voter privacy and election integrity, yet it does not rely on the complex and fragile electronic voting machines found in many current implementations. In this paper, we define the Punchscan system and voting protocol, including the people, objects and events involved and the ways they interact. We also trace the flow of data throughout the election process. This definition will aid those implementing the Punchscan system, but also lays a foundation for critical analysis and discussion within the voting research community.

In December 2005, David Chaum presented Punchscan, his latest concept for a receipt-based voting system that combines paper ballots and a cryptographically secure electronic tabulation process. As a hybrid paper/electronic system, it seeks to combine the best of both worlds. The paper ballot is intuitive and familiar to the average voter, who can cast their vote and understand the basic security model with little effort. At the same time, voting and security experts can inspect every step of the open yet cryptographically secure electronic tabulation process.

Since the initial announcement, a team of researchers from the University of Maryland, Baltimore County (UMBC) and George Washington University (GWU) has worked to refine and implement the Punchscan concept. The first step in this process is to more formally define the concept in terms of the people, technology and processes involved. This paper provides a clear definition of the Punchscan concept, with an eye toward the practical implementation of the system with currently available technology.

In the next section, we survey the current state of the art in electronic voting systems. We then introduce the Punchscan ballot along with the Punchboard, the core component of the electronic tabulation system. We continue with a discussion of the remaining key components, followed by a description of the Punchscan election protocol in terms of those components. A discussion of future work leads to the conclusion of this paper.

Voting systems have evolved from the once-ubiquitous hand-counted paper ballot. Many modern systems present ballots in electronic form, and some offer the voter a receipt used to verify their vote.

Direct Recording Electronic (DRE) voting systems are characterized by the use of electronic screens to display the choices available for each ballot question. Newer systems (such as the Diebold AccuVote TS) employ a touch-screen to register the voter's choice, while others use buttons or keypads. The vote is encoded and stored on some

medium, paper or electronic, for transport to a counting authority. Some DREs print a paper record of each vote, while others automatically count the votes and transmit only the final tally. In all cases, the voter must trust the DRE to faithfully record, protect and transmit their vote, however there is no basis for this trust.

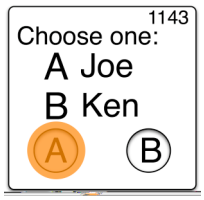
Few DRE manufacturers provide adequate documentation of their hardware or software, or allow the public to inspect their source code. While many states have a procedure for certifying DREs for use in their elections, test procedures and acceptance thresholds vary widely. Much of the security and privacy of DRE-based voting systems rely on the policies and procedures in place to manage the devices throughout the election process. Further, there are as yet no standard metrics to gauge the security and usability of DREs, or to compare their performance to that of other voting systems

As the name implies, receipt-based voting systems generate a physical receipt the voter can use to verify their vote was recorded as cast and counted as recorded. Some employ clever encryption techniques to provide this functionality without revealing the ballot contents, protecting voter privacy. However, this is often a difficult trick to manage. Despite efforts to make cryptographic protocols a natural part of the voting experience, these systems often suffer considerable usability issues [2,3,4].

The VoteHere Sentinel [5,6] is a leading product in this field, and provides mathematically provable integrity through a cryptographic protocol developed by Andrew Neff. The Sentinel is often used to add vote verification and receipt generation capability to existing touch-screen DREs.

In 2004, David Chaum proposed [7] an unnamed receipt-based system that allowed the voter to inspect the digital form of their ballot as printed on a two-layer plastic receipt tape. This system, based on the concept of "visual cryptography", proved difficult to implement and was not developed into a usable product, it did spark discussion within the voting community. Many sought to improve the system, adapting the central concepts to a simpler and more usable form. Peter Ryan [8] proposed one such improvement, replacing the expensive and exotic plastic receipt with a simpler, two-column perforated paper receipt.

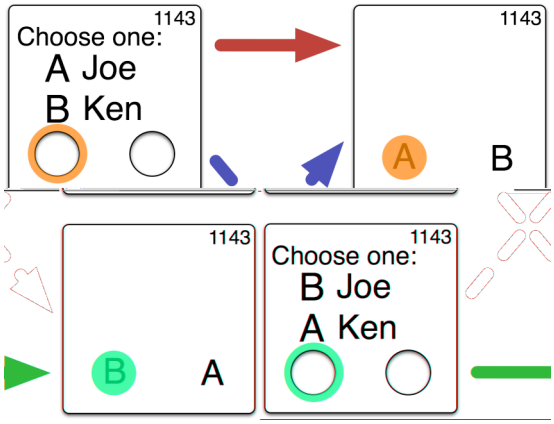
Punchscan is in many ways another such improvement of David Chaum's earlier concept. It employs a two-layer ballot and receipt and a sophisticated cryptographic tabulation system called a Punchboard. This section introduces both concepts using a simple example: an election with one question and two candidates.



A ballot.

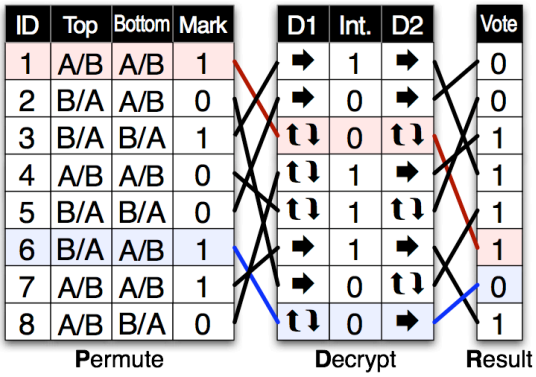
The Punchscan ballot, pictured at left, consists of two paper layers. The top layer contains the ballot choices matched with randomly chosen characters. The bottom layer contains the same characters in random order, visible through holes in the top layer. A voter marks the letter matching their choice with an ink dauber and separates the layers.

One layer is destroyed, and the other is scanned at the polling place and returned to the voter as their receipt. Voters may choose to keep either layer. When the ballot layers are separated, we find that neither reveals the original vote. In Figure 2, any layer can represent a vote for either candidate. The voter may show anyone their receipt without revealing their vote.



. Neither half reveals the original vote, whether for Joe (solid arrows) or Ken (dashed arrows).

To determine the original vote, election officials must know the order of the symbols on the destroyed ballot half. This information is stored and processed on the Punchboard, a set of three linked tables. Each row of the Punchboard contains the information needed to construct a single printed ballot, record the voter's ballot mark and translate the mark to a concrete vote.



. This Punchboard shows Ken has won the sample election, 5 votes to 3.

The Permute (P) table stores the order of the symbols on both ballot halves and the ballot position marked by the voter. For example, in Figure 3, row 4 of the Permute table corresponds to the ballot in Figure 1. Symbols on both layers follow the order (A, B), and the first position (position zero) is marked.

The Result (R) table holds the final votes, stored as a number representing a candidate or choice. In this case, 0 denotes a vote for Joe, the first listed candidate, and 1 a vote for Ken. The Decrypt (D) table performs the translation of each mark to a vote. In two stages named D1 and D2, the mark is either preserved (straight arrow) or inverted (circular arrows), reversing the effect of the random ordering of the symbols on both halves. Between the stages, an intermediate value is stored, and the ordering of votes is randomized before and after the Decrypt phase. This is represented by lines connecting rows in one table to those in the next.

For example, ballots 1 and 6 were both marked in position 1, however the top layer symbol order of ballot 6 is opposite that of ballot 1. Following the lines between tables, the votes appear correctly in the Result table as votes for Ken (row 6) and Joe (row 7), respectively.

The Punchboard embodies the fundamental tradeoff between voter privacy and election integrity. If the Punchboard is provided to the public as shown in Figure 4, it becomes trivial to link each voter to his or her vote. However, if the Punchboard remains secret, votes may be altered by arbitrary changes in the decrypt stage.

Instead, the entire Punchboard is made available to the public. Initially, all cells and connecting lines are encrypted and therefore unreadable. Though a series of audits and challenges[8], enough information is revealed to make significant deviations infeasible. Information that is not revealed can still be protected against arbitrary changes through zero knowledge bit commitments [10]. Changes to committed data can be detected by observers, though they do not know the original or changed value.

The Punchscan protocol involves an array of people, hardware and software that interact with Punchscan ballots and the Punchboard. Becoming acquainted with each entity will aid a detailed discussion of the protocol.

Voters are, of course, responsible for casting ballots. Because Punchscan is a receipt-based system, voters keep half of their ballot as a receipt. They are encouraged to use a website to verify the correctness of the information representing their ballot in the Punchboard.

Election Officials (EOs) are the key election authorities, responsible for setting up and running the election. As a group, they are trusted to handle all election data, including the Punchboard, in encrypted and unencrypted forms. Only Election Officials can link a single voter to their ballot. Though they are trusted with voter privacy, that trust is not blind.

Independent Audi

Punchboard's Permute table. Future work will explore the implications of this trust and methods of ensuring the correctness of printed ballots.

Within the polling place, Voters mark their ballot and separate its layers. One layer is destroyed by a cross-cut paper **Shredder** with a battery backup. Shredded ballot layers are properly disposed of using standard procedures for handling sensitive documents. The remaining layer is scanned using an optical **Scanner** with battery backup attached to a computer workstation. The workstation includes software to detect marks made by the Voter and a screen to allow for verification and corrections. Once verified, the vote is encoded in an XML file as a list of marks on a specified ballot layer. The file is transmitted to the Web Server or stored on removable storage for later hand delivery. The Scanner must be properly calibrated to recognize all possible valid marks on each ballot. This can be done using software algorithms or by calibrating the Scanner with a sample ballot with all positions marked.

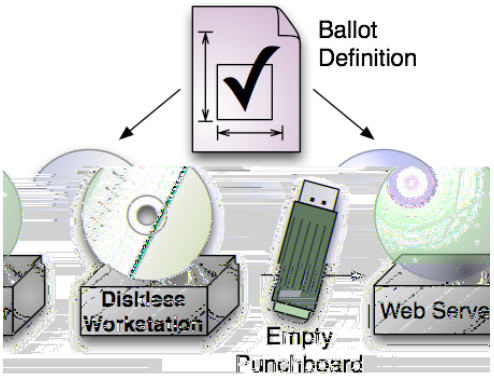
As the central communications hub for election participants, the Web Server performs many important functions. When voters enter the Ballot ID from their receipt, the server's **Web Application Software** accesses mark and permutation data from the public Punchboard to render a virtual copy of the receipt. Voters can inspect this virtual receipt to ensure it is identical to their original copy. Observers can download all public election data, including the Punchboard, from the server in an open data format for automated processing or manual inspection. At the appropriate times, the server will accept challenges and audit requests from authenticated election Auditors. In response, Election Officials must be able to log onto the Web Server to securely upload updated election data. Only Auditors and Election Officials require authenticated access to the server; all other users may remain anonymous. All data and software on the Web Server are public, therefore there is no risk a malicious user obtaining sensitive data. Since the Diskless Workstation is the only computer to process election data in unencrypted form, a high threshold is set on its security and integrity. Its hardware configuration limits its ability to store or transmit sensitive information, and its **Verified Trusted Software** must faithfully process all data according to the algorithms introduced by Hosp, et al. [9].

All source code for the Workstation's operating system and user applications are open and published on the Web Server along with any derivatives, including compiled binaries and optical disk images. All published code and binary data are accompanied by their public hash value and the steps necessary to reconstruct any derivative from the original source code. This allows anyone to use publicly available tools to examine, build, test

recordable storage device. This output data is hand-carried to its destination, often the public Web Server.

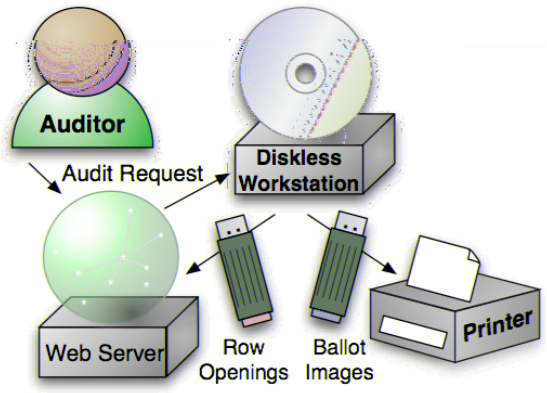
In the first of four phases, Election Officials use Ballot Authoring Software to define critical ballot and election parameters, posting the resulting ballot definition file on the Web Server. Once the public has inspected the ballot definition file, the first Election Officials' meeting is called.

Officials load the ballot definition file on the Diskless Workstation, which outputs a Punchboard with the specified number of ballots, questions and choice permutations. At this stage, all data is encrypted, but commitments to each data value prevent their alteration by Election Officials. The Punchboard is copied from the recordable media to the Web Server.



Ballot parameters are specified in the Election Definition phase.

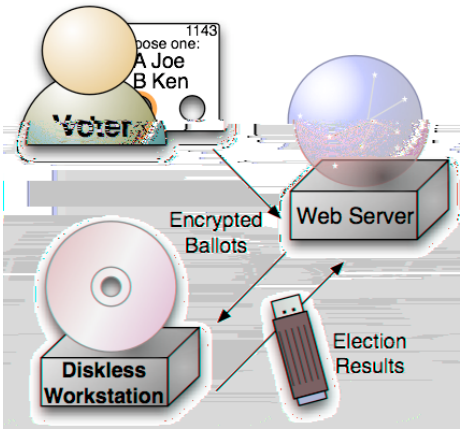
Once the Punchboard is published, Auditors perform a Pre-Election Audit by choosing half the ballot ID numbers listed in the Punchboard. At their second meeting, Election Officials use the Diskless Workstation to fully decrypt the rows of the Punchboard corresponding to the chosen ballot ID numbers. The partially decrypted Punchboard is transferred to the Web Server. For each of the decrypted rows, Auditors and Observers can verify that the operations specified in the Decode table would correctly decode and count each ballot given the symbol ordering in the Permute table. Moreover, the commitments for each opened data value are recomputed to prove they match the commitments in the first edition of the Punchboard.



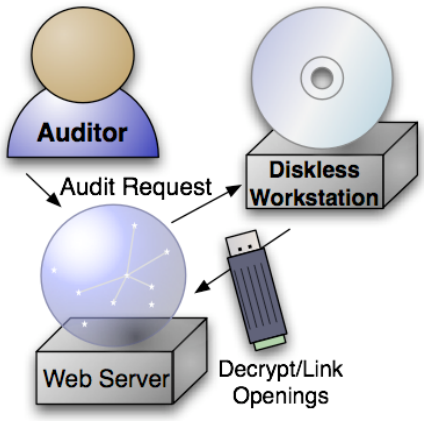
Pre-Election phase: Punchboard rows become spoiled or printed ballots.

Also during the second meeting, the Diskless Workstation renders print-ready ballot images for each ballot ID number not chosen for the audit. These ballot images are stored on a separate storage device and transferred to the Printer. Printed ballots are placed in envelopes and transported to each polling place.

On Election Day, each Voter marks their ballot and separates its layers. One layer is destroyed, the other scanned. After the Voter verifies the Scanner has correctly detected the marks on their ballot, the ballot is returned to the Voter and an electronic copy is prepared, encrypted and transmitted to the Web Server. A second copy is retained on a removable storage device in case the Web Server or its Internet connection fails. After the polls close, any votes not already transmitted to the Web Server are copied from the removable storage device from each polling place. Voters can visit the election website to verify their ballot is correctly posted and included in the batch of tallied votes.



Ballots are cast and counted during the Election phase.



Auditor requests the opening of half the Decrypt table in the Post-Election phase.

Election Officials copy each ballot onto a removable storage device and meet for a third time. The Diskless Workstation fills the Punchboard with data obtained from each encrypted ballot. Each ballot mark is processed through the Punchboard's Decrypt table and stored as a single vote in the Results table. The updated Punchboard and preliminary vote totals are transferred to the Web Server.

After election results are posted online, Auditors perform a Post-Election Audit, choosing either the left or right half of the Punchboard's Decrypt table. Election Officials meet for a final time and use the Diskless Workstation to decrypt the chosen half of the Decrypt table. This step reveals half of the ballot mark translation process, all intermediate values from this process and the links from each row of the Decrypt table to rows of the Permute or Results table.

From this, Auditors and Observers can verify each calculation in the Decrypt table and that each row of the Decrypt table links to exactly one row of the Permute or Results table (and vice versa). Each commitment can be recomputed and compared with earlier editions of the Punchboard to verify the links and data values released to the public have not been altered.

Although the Pre and Post-Election Audits do not reveal all the data and calculations within the Punchboard, they are an effective guard against corruption among Election Officials. Any area of the Punchboard may be opened in response to either audit. In addition, voters may choose either ballot layer as their receipt, and any attempt to modify the chosen layer is detected when the voter verifies their receipt online. Therefore a corrupt Official must risk detection to alter any aspect of the ballot or the tabulation process. Established formulae for parallel testing and probability theory ensure any significant corruption of the Punchboard is almost certainly detected. [11]

This work is intended as a first introduction to the Punchscan voting system, with an eye toward its implementation with currently available hardware and software. The concepts introduced here will be more formally expressed in a system definition document. Researchers at UMBC and George Washington University will also build a prototype election system and test its security and usability in mock elections.

While the core Punchscan concept is well-defined, many peripheral issues remain unexplored. The Punchscan ballot concept

these voters to use the same ballot as those voting at a standard polling place, providing equal levels of security and privacy for all members of the electorate. We must also consider the implications of trusting a third-party printer agent to manufacture printed ballots.

By formally introducing this new and interesting voting system concept, we intend to provoke discussion among experts in the electronic voting research community. Starting from a common set of concepts and definitions, researchers with diverse talents can analyze the system, explore its qualities and suggest improvements.

We thank Dr. David Chaum for sharing his deep understanding of the Punchscan concept and remarkable zeal for solving implementation issues. Ben Hosp and Stefan Popoveniuc graciously provided early copies of their papers, helping us grasp the mathematics behind the Punchscan protocol. We also thank our advisor, Dr. Alan Sherman, for his role in bringing Dr. Chaum and our research group at UMBC together, and for sharing his passion for high-integrity voting systems.

[1] Chris Karlof, Naveen Sastry, David Wagner. .
Proceedings of the 14th USENIX Security Symposium, August 2005. pp. 33-50.

[2] Alan T. Sherman, Donald F. Norris, Andrew Sears, Aryya Gangopadhyay, Stephen H. Holden, George Karabatis, A. Gunes Koru, Chris M. Law, John Pinkston, and Dongsong Zhang,
accepted for
USENIX/Accurate Electronic Voting Technology (EVT'06) Workshop.

[3] Donald F. Norris, Andrew Sears, Charles Nicholas, Anne V. Roland, Aryya Gangopadhyay, Stephen H. Holden, George Karabatis, A. Gunes Koru, Chris M. Law, John Pinkston, Andrew Sears, Alan T. Sherman, and Dongsong Zhang, , prepared for the
Maryland State Board of Elections, National Center for the Study of Elections of the Maryland Institute for Policy Analysis and Research, University of Maryland, Baltimore County (February 2006).
Available online: http://umbc.edu/mipar/documents/VoteVerificationStudyReport-FINAL_001.pdf

[4] Paul S. Herrnson, Benjamin B. Bederson, Charles D. Hadley, Richard G. Niemi, Michael J. Hanmer.

Center for American Citizenship and Politics, University of Maryland College Park (2006).
Available online: http://elections.state.md.us/citizens/voting_systems/MarylandReport2-15-06.pdf

[5] C. Andrew Neff. . Available online:
http://votehere.net/vhti/documentation/egshuf-2.0.3638.pdf October 2004.

[6] C. Andrew Neff. . Available online:
http://votehere.net/vhti/documentation/vsv-2.0.3638.pdf October 2004.

[7] David Chaum. . IEEE Security and Privacy,
2(1), pp. 38-47. January-February 2004.

[8] Peter Y. A. Ryan . Technical Report CS-TR 864,
University of Newcastle. October 2004.

[9] Ben Hosp, Stefan Popoveniuc. Available online:
http://punchscan.org/Documents/PunchscanSummary.pdf May 2006.

[10] Gilles Brassard, David Chaum, Claude Crèpeau.
Journal of Computer and Systems Sciences, vol. 37 no. 2, pp. 156-189. 1988.

[11] American National Standards Institute. ANSI/ASQC Z1.4-1993.
. 1993.