COMPUTER SECURITY LAB

RICE UNIVERSITY

Cool Crypto Tricks: Homomorphisms, Zero-Knowledge Proofs

Dan S. Wallach
Rice University

Back to basics: Elgamal Crypto

Elgamal encryption

Non-deterministic cryptosystem (different *r* every time)

$$E(g, g^a, r, M) = < g^r, (g^a)^r M >$$

$$D(g^r, a, g^{ar}M) = \frac{g^{ar}M}{(g^r)^a}$$

$$= M$$

g	group generator
M	plaintext (message)
r	random (chosen at encryption time)
a	(private) decryption key
g^a	(public) encryption key

Homomorphic property

Anybody can combine two ciphertexts to get a new one.

$$
\begin{aligned}
E(M_1) \oplus E(M_2) &= \; < g^{r_1}, (g^a)^{r_1} M_1 > \oplus < g^{r_2}, (g^a)^{r_2} M_2 > \\
&= \; < g^{r_1} g^{r_2}, (g^a)^{r_1} M_1 (g^a)^{r_2} M_2 > \\
&= \; g^{r_1 + r_2}, g^{a(r_1 + r_2)} M_1 M_2 \\
&= \; E(M_1 M_2)
\end{aligned}
$$

g	group generator
M	plaintext (message)
r	random (chosen at encryption time)
a	(private) decryption key
g^a	(public) encryption key

Violation of encryption semantics?

If I know M_1 **and** M_2 **and** $E(M_1) \oplus E(M_2) = E(M_1 M_2)$
then I can find other messages where
I know their encryption!

Solution: Padding

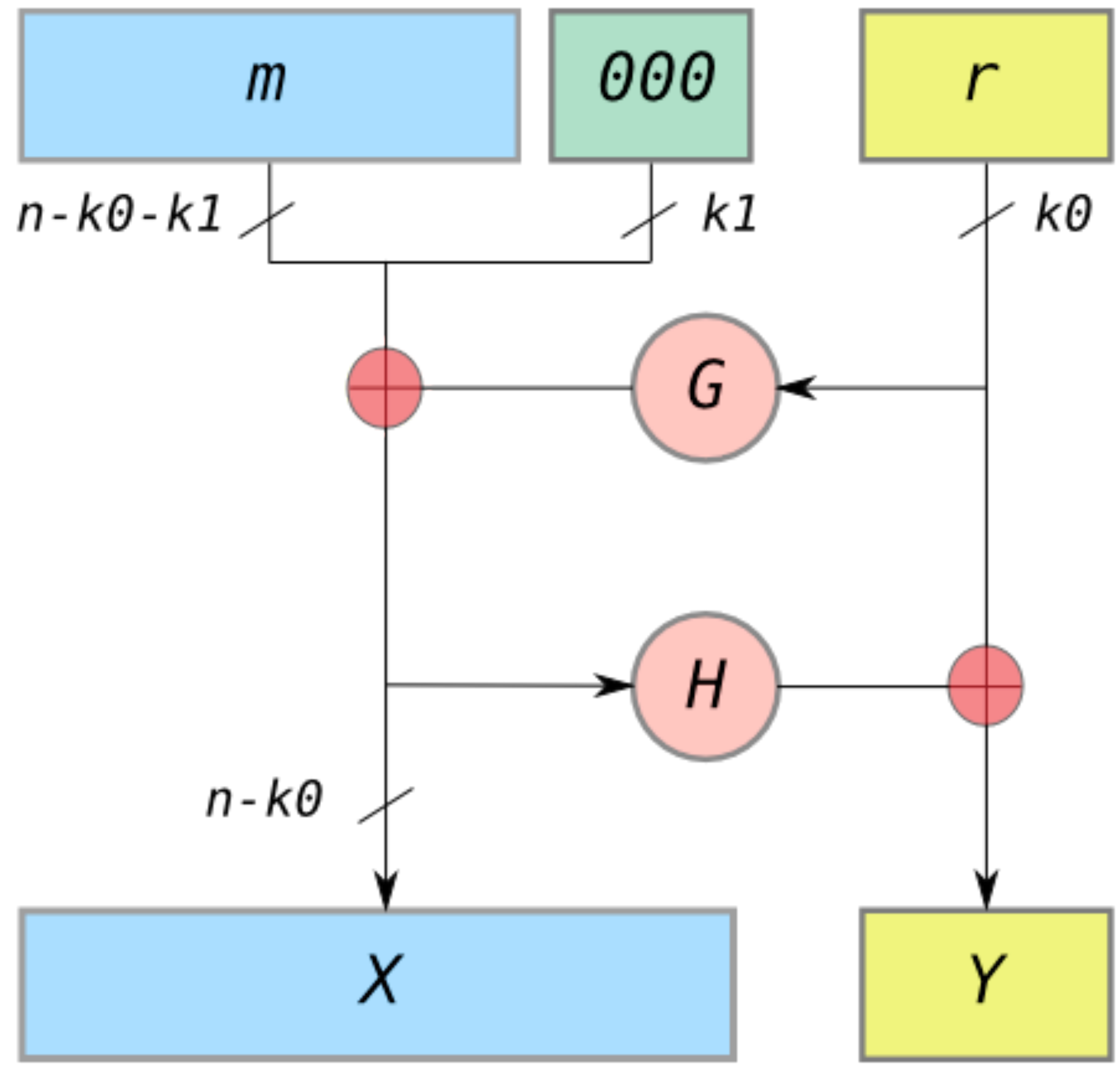
Optimal Asymmetric Encryption Padding (OAEP) - *Belare and Rogaway (1995)*

m - message (plaintext)

r - random number

G, H - cryptographic hash functions

X, Y - the message that gets encrypted



Cool trick: reencryption

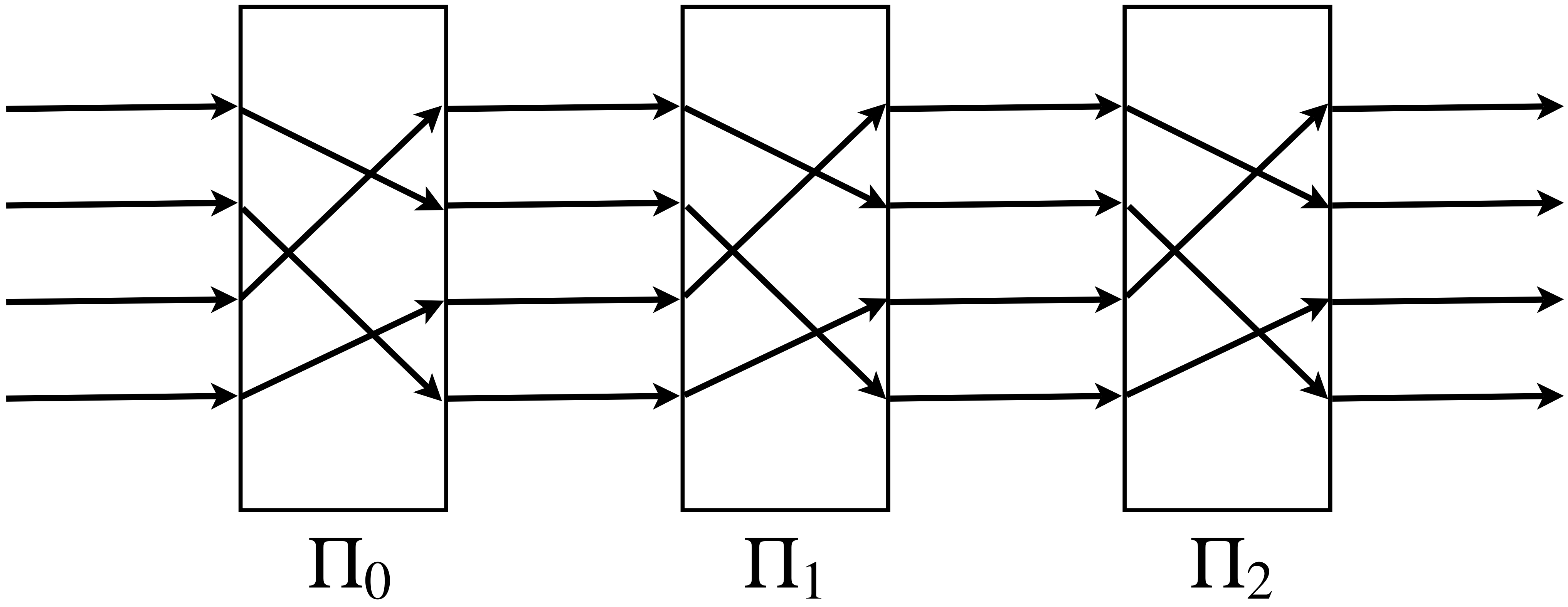
$$E(M) \oplus E(0) = E(M)^*$$

Anybody can "reencrypt" a message.

(New random number introduced from *E*(0).)

Reencryption mixnets

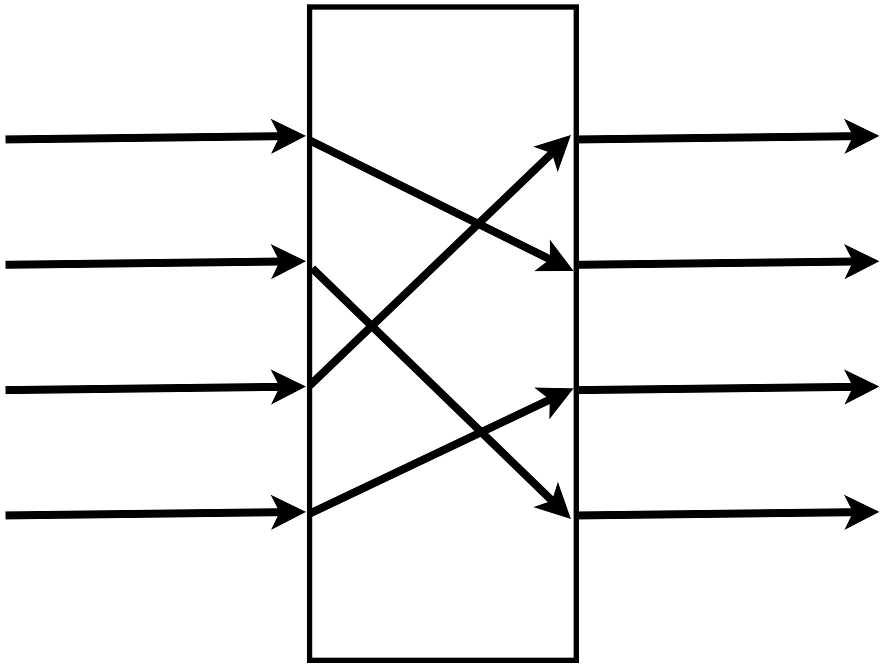
Permutations Π_i, where output is reencrypted.



Each mix permutes/reencrypts.
Must prove output corresponds to input.

Non-solution: reveal the mix

Publish the random numbers and the permutation.

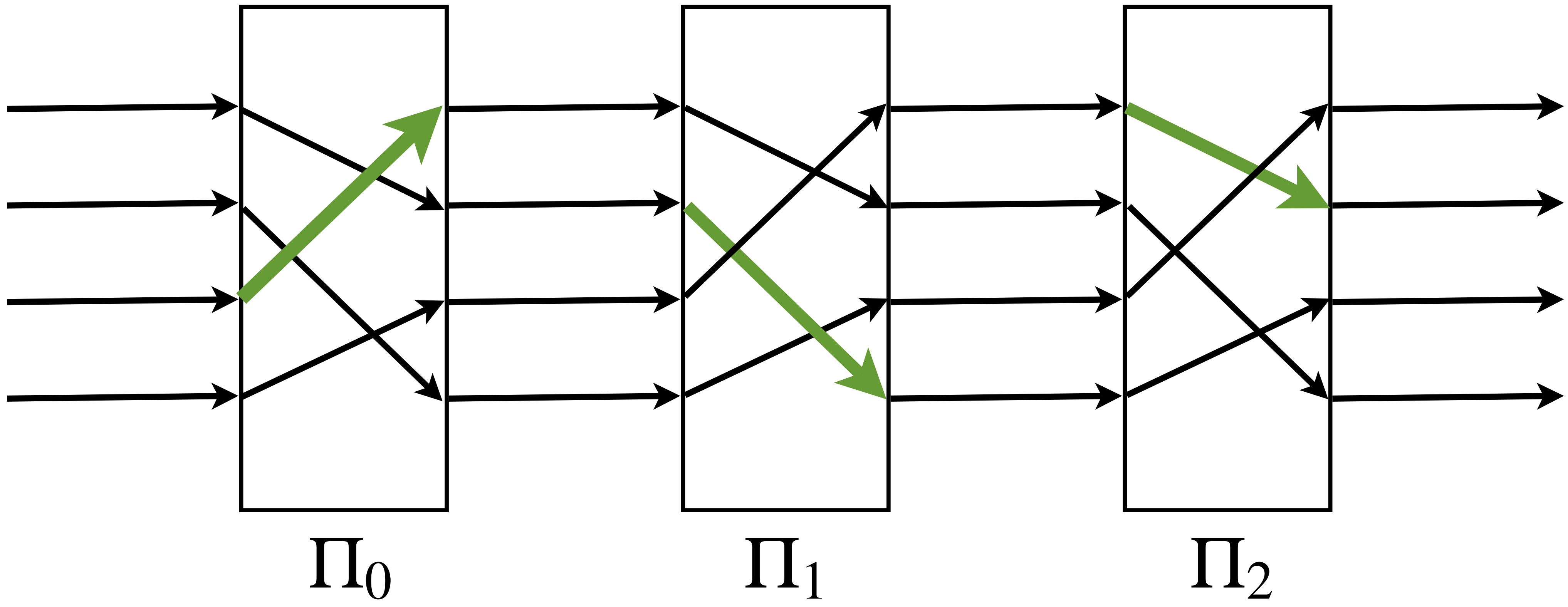


Π_0

Eliminates benefit of randomization.

Randomized partial checking

Effective across larger mixes.



(Jakobsson, Jules, Rivest '02)

Zero-knowledge proofs (ZKP)

want to prove you know something

while revealing nothing

generalized format

prover: commit to something (e.g., reencryption mix output)

verifier: *challenge* the prover

prover: respond to the challenge

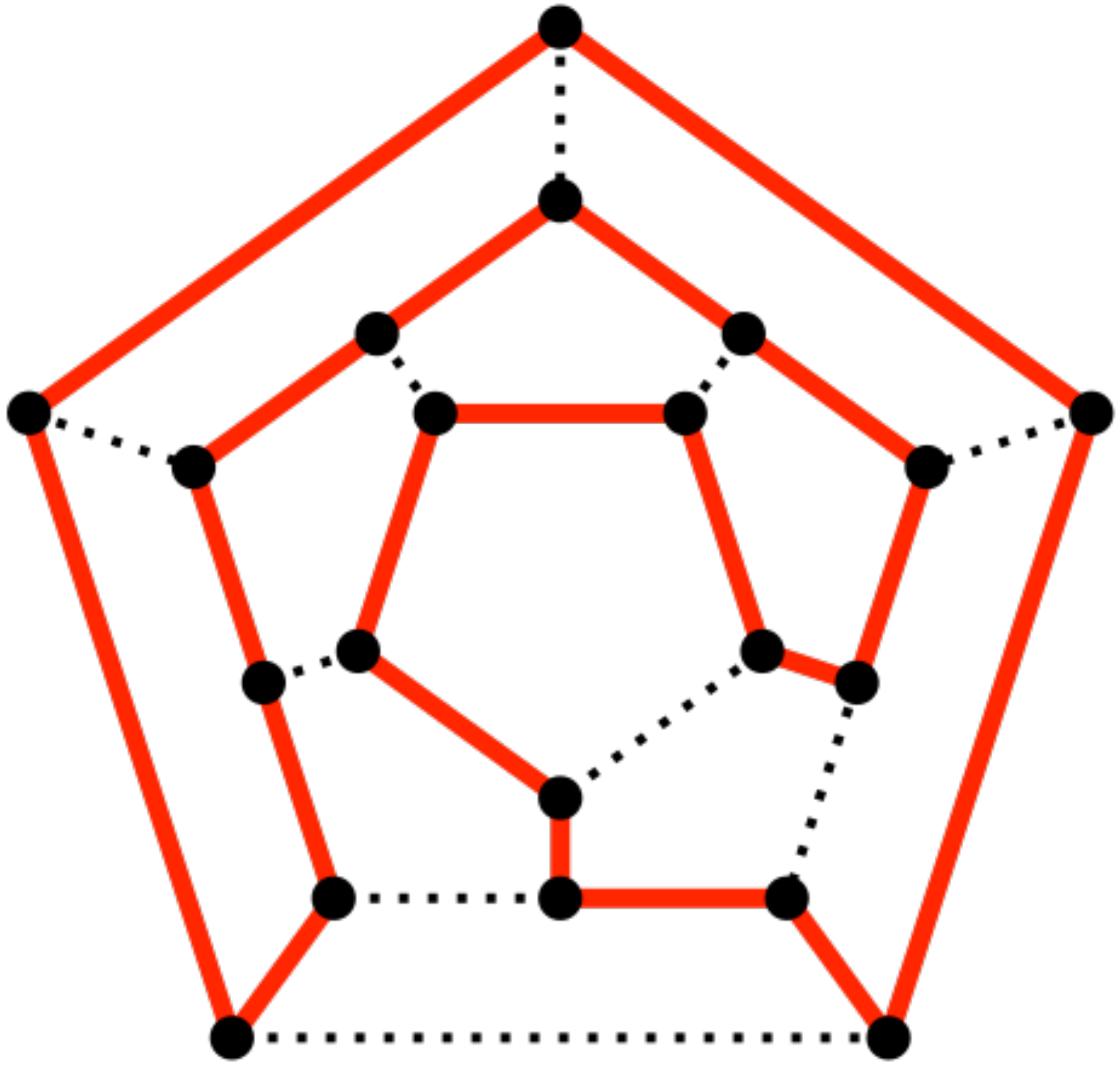
Example: Hamiltonian paths

Prover: "I know a HP over graph *G*." Compute graph isomorphism *H*. Publish *G*, *H*.

Verifier: Coin toss. Heads: tell me HP over *H*. Tails: tell me isomorphism *G* to *H*.

(Repeat *N* times.)

If prover doesn't know HP, verifier catches with high probability.

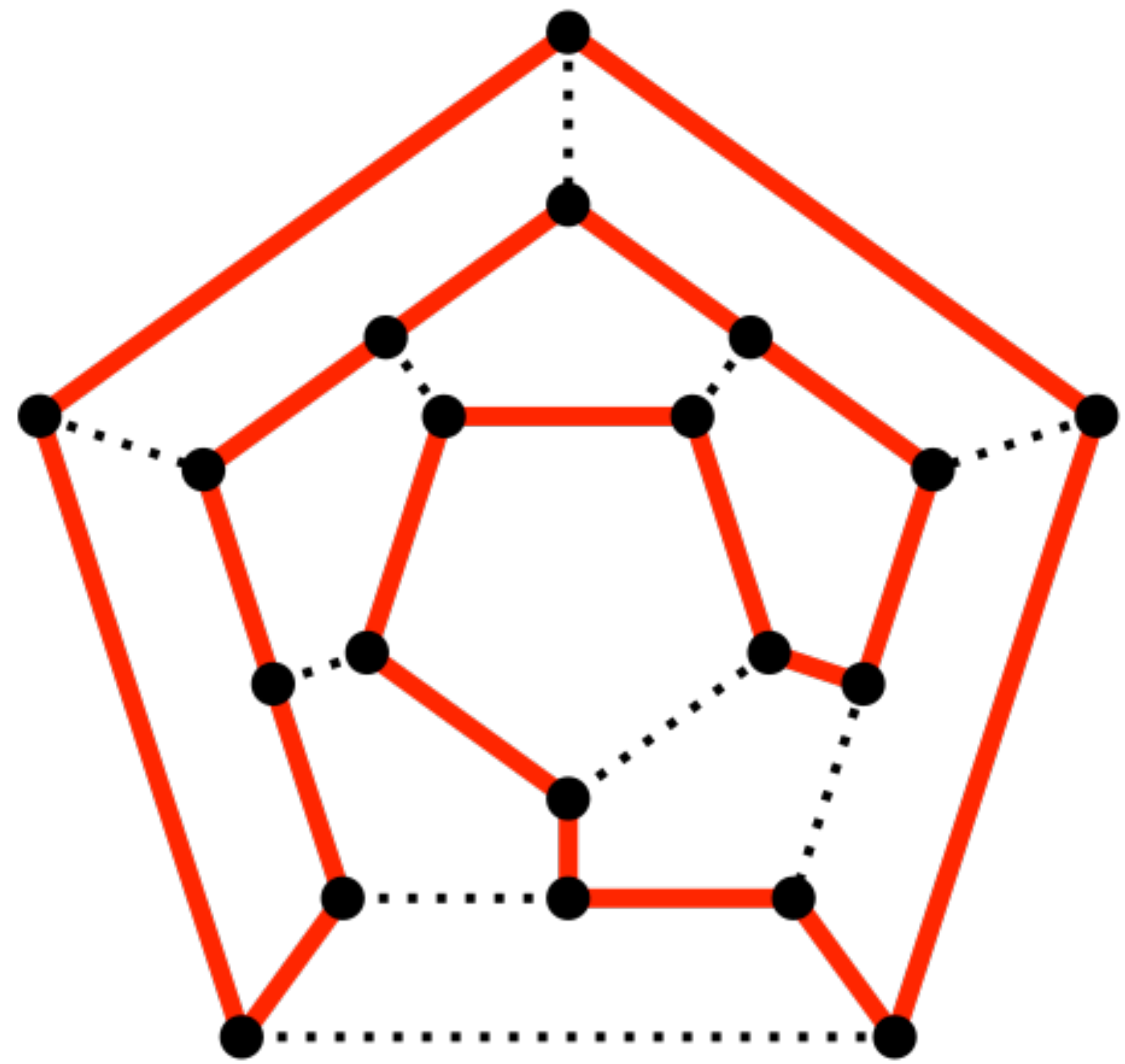


Non-interactive ZK proofs

Prover: Precompute *N* isomorphisms (H_1 to H_N) and hash them. Hash function yields coin tosses for virtual challenger. Then output the results.

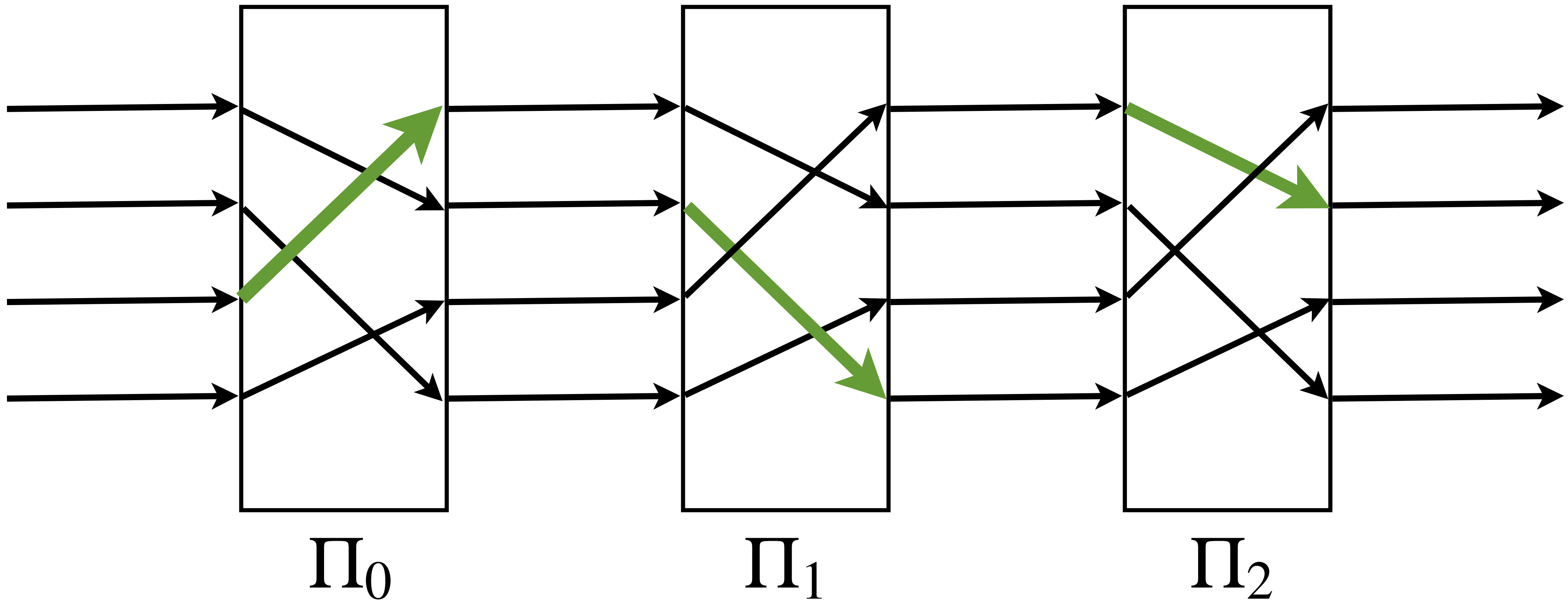
(Assumes good hash functions.)

This is an example of the *Fiat-Shamir heuristic* (1986).



NIZK variant for mixes

Hash the output of the permutation/reencryption. Use those bits to select which edges get revealed.



Say we're mixing 1 million ballots, each mix reveals 1%. After five mixes, 99.99% chance that all ballots reencrypted at least once.

Homomorphic vote tallying

Change messages to counters, additive in exponent of *g*.

$$\begin{aligned} E(v_1) \oplus E(v_2) &= < g^{r_1}, (g^a)^{r_1} g^{v_1} > \oplus < g^{r_2}, (g^a)^{r_2} g^{v_2} > \\ &= < g^{r_1+r_2}, g^{a(r_1+r_2)} g^{v_1+v_2} > \\ &= E(v_1+v_2) \end{aligned}$$

g	group generator
v	plaintext (counters)
r	random (chosen at encryption time)
a	(private) decryption key
g^a	(public) encryption key

Evil machine: E(bignum)?

Must prove ciphertext corresponds to well-formed plaintext. (Example, prove counters are zero or one.)

We need another ZK tool: Chaum-Pedersen proofs.

Prover knows: $(g, g^x), (h, h^x)$

Wants to prove that these two tuples share x

Chaum-Pedersen proofs

Goal: demonstrate $(g, g^x), (h, h^x)$

P: choose random $w \in \mathbb{Z}_p^*$, compute $(A = g^w, B = h^w)$

Send (A, B) to V

V: pick a random number c (challenge), send to P

P: compute $R = w + xc$

send R to V

V: Compute

$$\begin{aligned} A(g^x)^c &= g^w g^{xc} \\ &= g^{w+xc} \\ &= g^R \end{aligned}$$

$$\begin{aligned} B(h^x)^c &= h^w h^{xc} \\ &= h^{w+xc} \\ &= h^R \end{aligned}$$

Fake C-P proofs?

Goal: demonstrate $(g, g^x), (h, h^x)$

P: choose random $w \in \mathbb{Z}_p^*$, compute $(A = g^w, B = h^w)$

Send (A, B) to V

V: pick a random number c (challenge), send to P

P: compute $R = w + xc$

send R to V

V: Compute

$$\begin{aligned} A(g^x)^c &= g^w g^{xc} \\ &= g^{w+xc} \\ &= g^R \end{aligned}$$

$$\begin{aligned} B(h^x)^c &= h^w h^{xc} \\ &= h^{w+xc} \\ &= h^R \end{aligned}$$

Fake C-P proofs?

Goal: demonstrate $(g, g^x), (h, h^x)$

P: choose random $w \in \mathbb{Z}_p^*$, compute $(A = g^w, B = h^w)$

Send (A, B) to

P choses fake c, R s.t. $A = g^R (g^{xc})^{-1}$.

V: pick a random number c (challenge), send to P

P: compute $R = w + xc$

send R to V

V: Compute

$$\begin{aligned} A(g^x)^c &= g^w g^{xc} \\ &= g^{w+xc} \\ &= g^R \end{aligned}$$

$$\begin{aligned} B(h^x)^c &= h^w h^{xc} \\ &= h^{w+xc} \\ &= h^R \end{aligned}$$

Fake C-P proofs?

Goal: demonstrate $(g, g^x), (h, h^x)$

P: choose random $w \in \mathbb{Z}_p^*$, compute $(A = g^w, B = h^w)$

Send (A, B) to

P choses fake c, R s.t. $A = g^R(g^{xc})^{-1}$.

V: pick a random number c (challenge), send to P

P: compute $R = w + xc$

send R to V

Observer can compute $A(g^x)^c$...

V: Compute

$$\begin{aligned} A(g^x)^c &= g^w g^{xc} \\ &= g^{w+xc} \\ &= g^R \end{aligned}$$

$$\begin{aligned} B(h^x)^c &= h^w h^{xc} \\ &= h^{w+xc} \\ &= h^R \end{aligned}$$

Fake C-P proofs?

Goal: demonstrate $(g, g^x), (h, h^x)$

P: choose random $w \in \mathbb{Z}_p^*$, compute $(A = g^w, B = h^w)$

Send (A, B) to

P choses fake c, R s.t. $A = g^R (g^{xc})^{-1}$.

V: pick a random number c (challenge), send to P

P: compute $R = w + xc$

send R to V

Observer can compute $A(g^x)^c$...

V: Compute

$$\begin{aligned} A(g^x)^c &= g^w g^{xc} \\ &= g^{w+xc} \\ &= g^R \end{aligned}$$

$$\begin{aligned} B(h^x)^c &= h^w h^{xc} \\ &= h^{w+xc} \\ &= h^R \end{aligned}$$

ZK protocols only work when "live" (or use Fiat-Shamir heuristic for non-interactive)

C-P for vote testing

Can I prove a vote is zero or one? First, how about proving it's zero using C-P.

Want to verify $< g^r, g^{ar} g^v >$ where $v = 0$

Do C-P protocol where $(g, g^x), (h, h^x)$ becomes

$$(g, g^r), \left(g^a, \frac{g^{ar} g^v}{g^v}\right)$$

We could also do this for $v = 1$

Challenge is to do them together, at the same time.

(Note: the original slides had a typo in the math, fixed here.)

Cramer-Damgård-Schoenmakers ('96)

Can run two Chaum-Pedersen (or any two ZK proofs like this) simultaneously, one "real" and one "simulated".

First, fake a proof (e.g., for v=1) in advance.

Then, announce the first message for both protocols. Challenger sends *c*, prover announced a split c_0, c_1 where $c_0 + c_1 = c$, then executes both ZK protocols

Verifier cannot tell which one was real vs. simulated, but knows that **one** of them was real.

Crypto summary

At the end of the day, **any** election observer can now:

- verify every single ballot for being “well-formed” (valid Elgamal tuple, encrypted zero-or-one, etc.)
- add together all the ballots (homomorphically)
- verify a proof of the tally (Chaum-Pedersen again) (only the election authority can generate this)

But we have no idea if the original ciphertext corresponded to the **intent of the voter** (versus evil machine flipping votes).

The California Top-To-Bottom Study

Summer 2007

Biggest study of its kind, ever

40+ researchers (source code, “red team,” documentation, accessibility)

three vendors (Diebold, Sequoia, Hart InterCivic)

http://www.sos.ca.gov/voting-systems/oversight/top-to-bottom-review.htm

Significant flaws found with each vendor

Viral attacks possible!

Diebold and Sequoia "conditionally recertified"

Only one machine per precinct for accessibility

Other votes on paper

Hart InterCivic has comparable sanctions

Revised conditions announced later

(e.g., reboot inventory computer from CDROM after every DRE machine connected)

Hart eSlate architecture

Local network in the polling place

Controller sees all machines,
collects all votes together

Cryptography?

HMAC-SHA1 for integrity checking of cast ballots

Single shared key for the entire election

OpenSSL in some places, but incorrect cert checking

No crypto on voting-machine local network

Network protocol?

Messages that directly read and write to memory

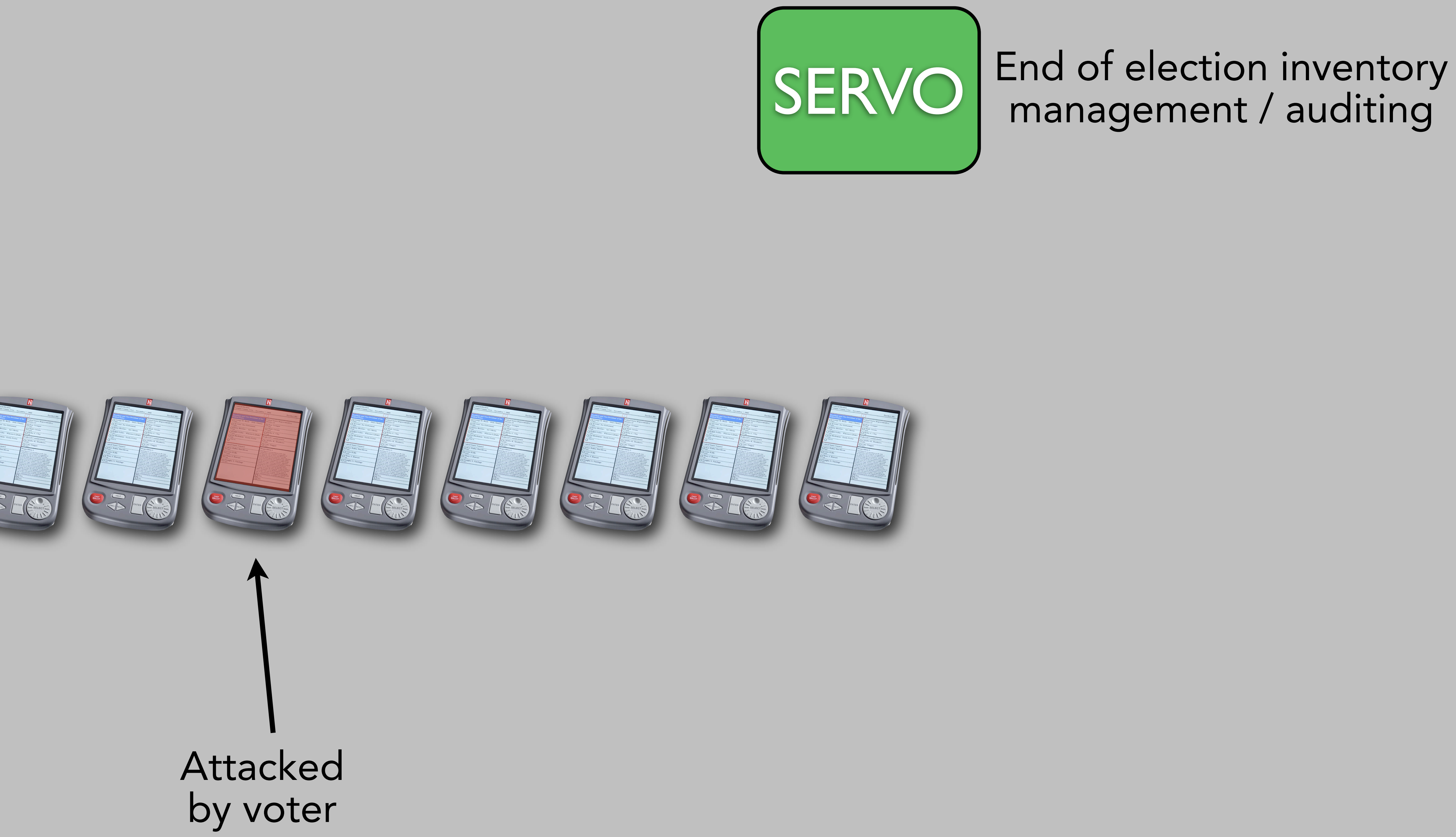
Officially used to test whether code is authentic

Also allows votes to be extracted or changed

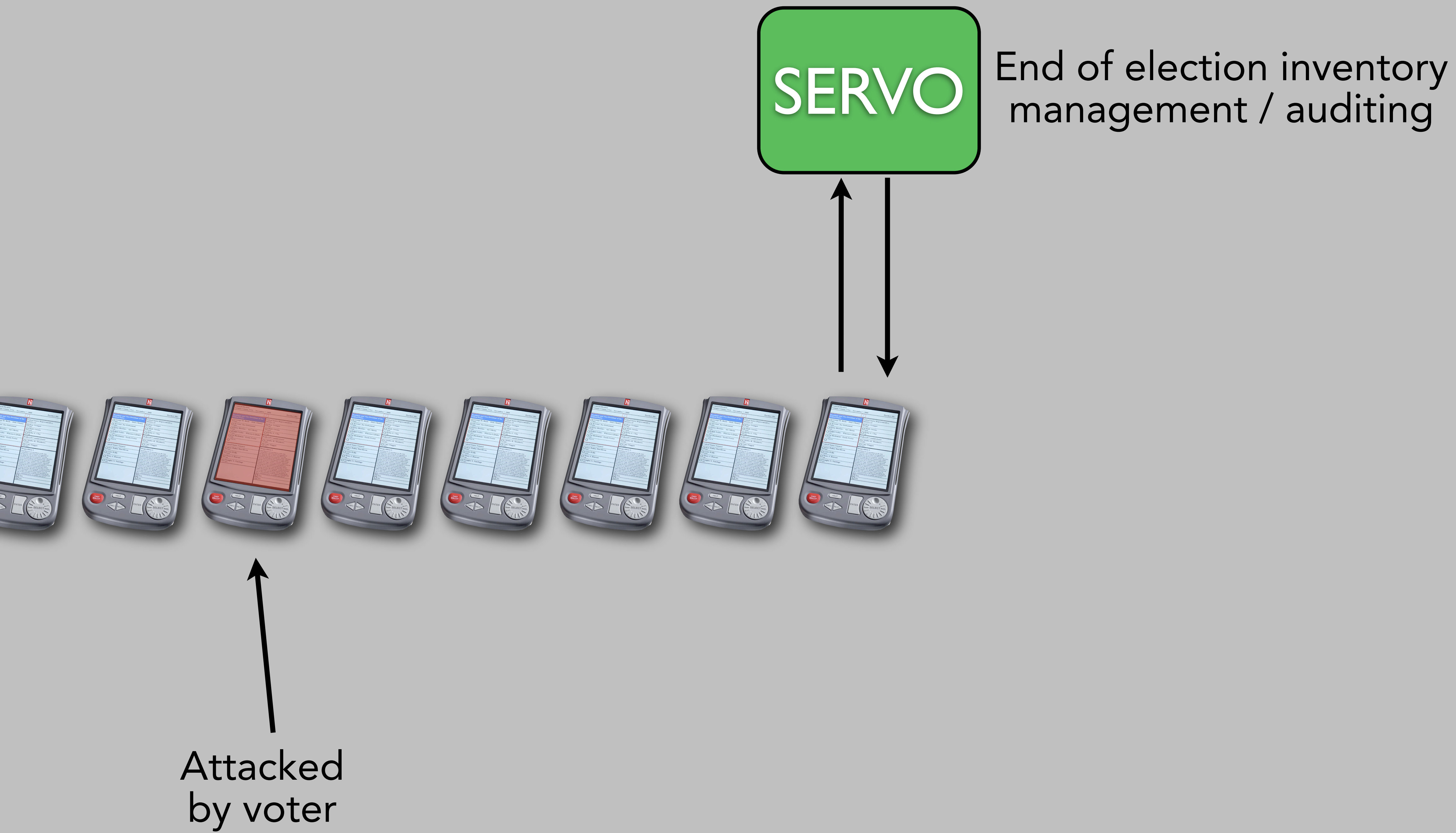
Enables virus injection

Regular voters have access to the network port

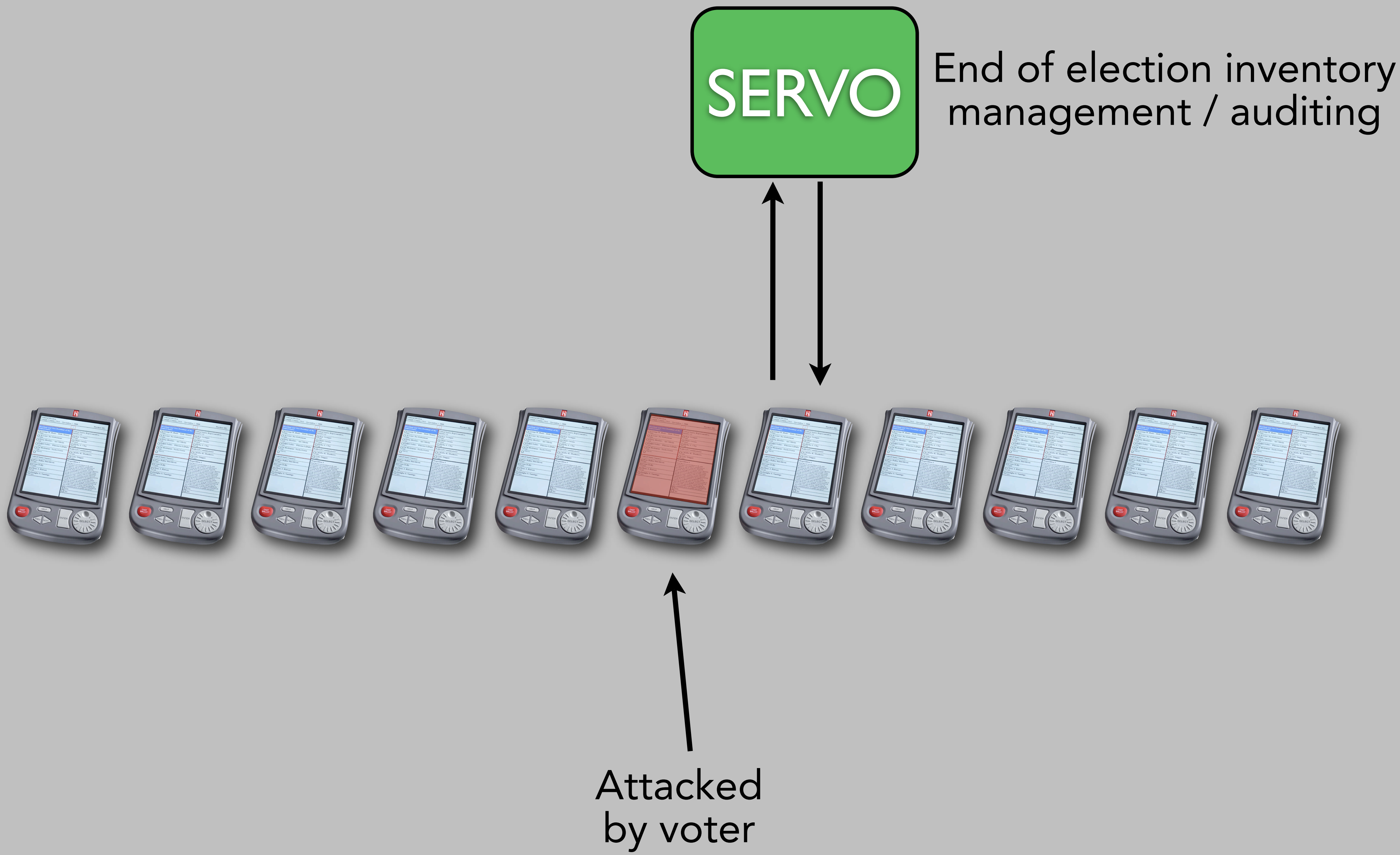
Viral attacks?



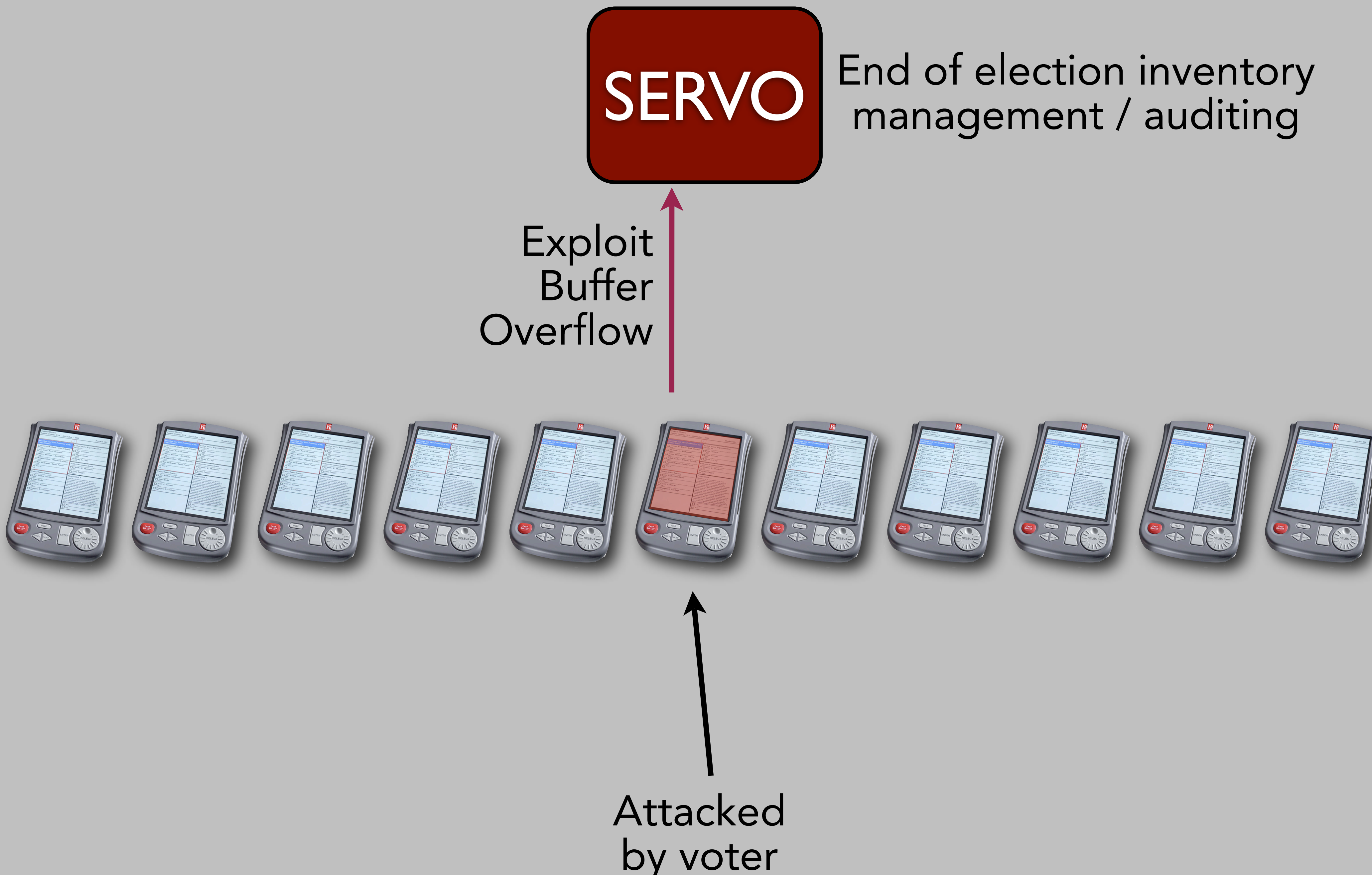
Viral attacks?



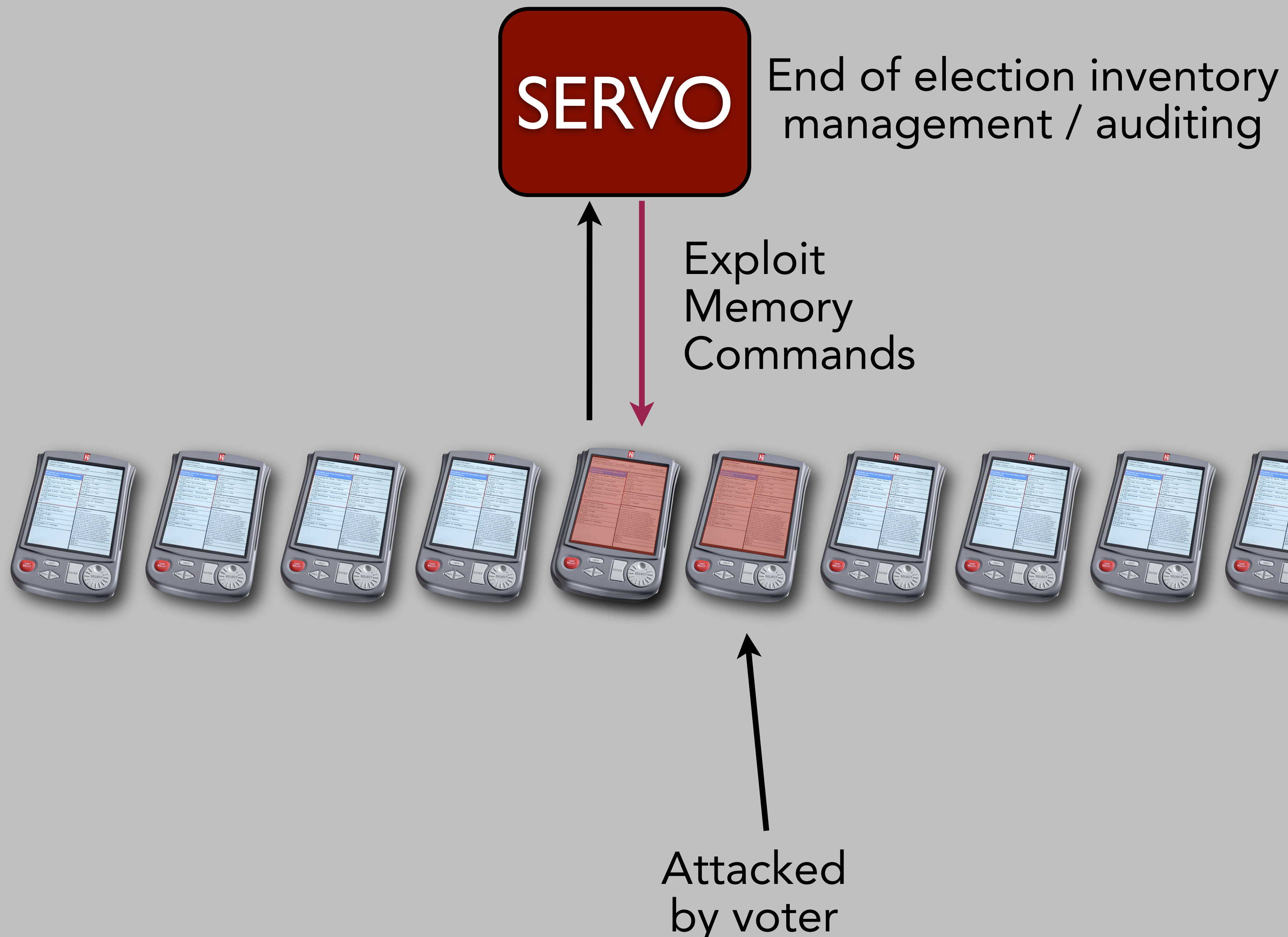
Viral attacks?



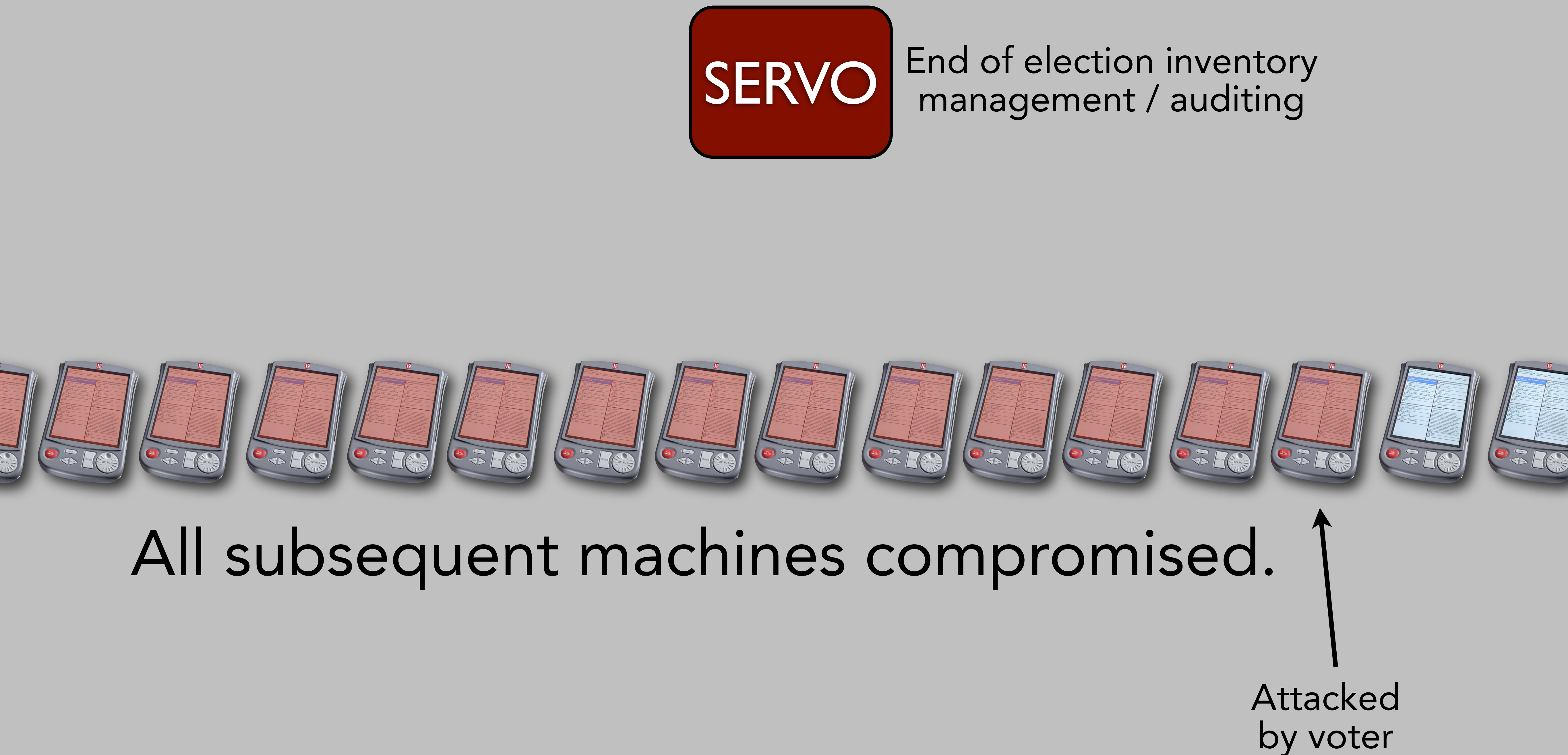
Viral attacks?



Viral attacks?



Viral attacks?



No easy way to clean a compromised machine

Must replace internal chips by hand

No easy way to detect compromised machines

Hacked machine can correctly answer network queries

Other Hart problems

Audio unit can be overheard with a short-wave radio

“Adjust votes” feature in tabulation system

Premier (née Diebold, now part of ES&S) and Sequoia had similar problems.

(Results confirmed by follow-on study in Ohio.)

What’s next?

What's next?

Some states following California's lead (but not Texas)

Limit use of DREs to one per precinct

Mandatory audits to compare paper to electronic records

What's next?

Some states following California's lead (but not Texas)

Limit use of DREs to one per precinct

Mandatory audits to compare paper to electronic records

Vendors will (hopefully) engineer better products

What's next?

Some states following California's lead (but not Texas)

Limit use of DREs to one per precinct

Mandatory audits to compare paper to electronic records

Vendors will (hopefully) engineer better products

Optical scan paper ballots growing in popularity

Example: Travis County (Austin, TX) dropping eSlate after 2012

Research goals

Research goals

Make it easier to audit results after the election

every vote included is valid; every valid vote is included

Research goals

Make it easier to audit results after the election

every vote included is valid; every valid vote is included

Make it harder to make mistakes on election day

tolerate accidental loss/deletion

Research goals

Make it easier to audit results after the election

every vote included is valid; every valid vote is included

Make it harder to make mistakes on election day

tolerate accidental loss/deletion

How?

Connect the machines together.

VoteBox's approach

D. Sandler and D. S. Wallach. **Casting Votes in the Auditorium.** In Proceedings of the 2nd USENIX/ACCURATE Electronic Voting Technology Workshop (EVT'07).

D. Sandler, K. Derr, and D. S. Wallach, **VoteBox: A Tamper-Evident, Verifiable Electronic Voting System**. 17th USENIX Security Symposium (USENIX Security '08).

VoteBox's approach

Store everything everywhere

Massive **redundancy**

Stop trusting DREs to keep their own audit data

D. Sandler and D. S. Wallach. **Casting Votes in the Auditorium.** In Proceedings of the 2nd USENIX/ACCURATE Electronic Voting Technology Workshop (EVT'07).

D. Sandler, K. Derr, and D. S. Wallach, **VoteBox: A Tamper-Evident, Verifiable Electronic Voting System**. 17th USENIX Security Symposium (USENIX Security '08).

VoteBox’s approach

Store everything everywhere

Massive **redundancy**

Stop trusting DREs to keep their own audit data

Link all votes, events together

Create a **secure timeline** of election events

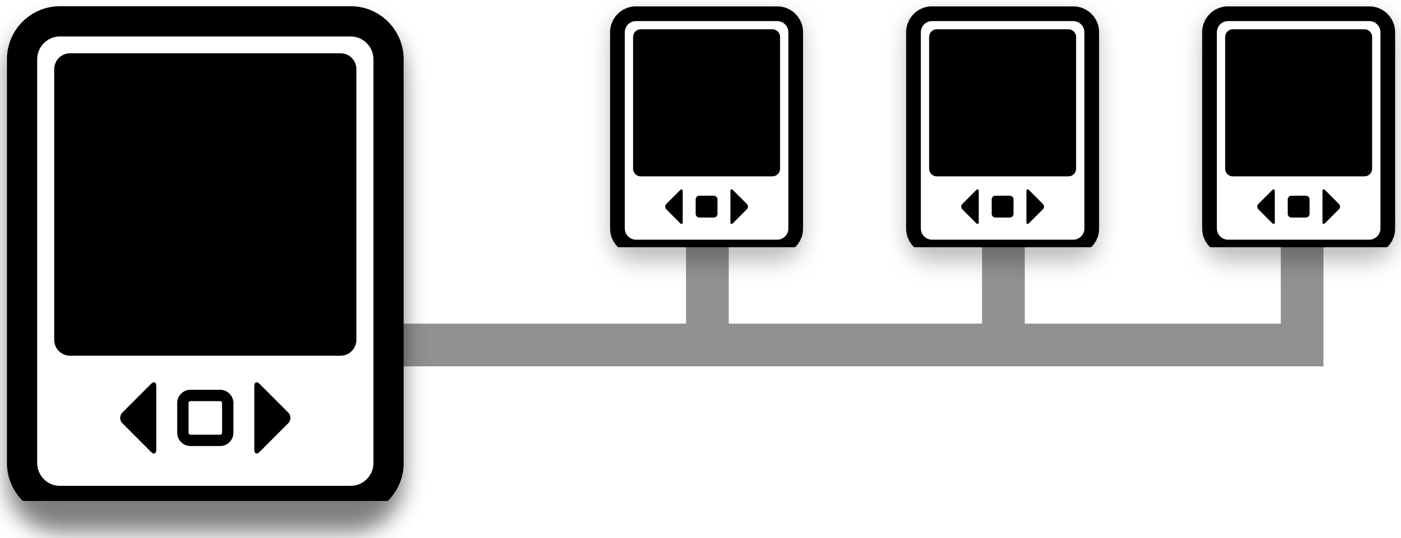
Tamper-evident proof of each vote’s legitimacy

D. Sandler and D. S. Wallach. **Casting Votes in the Auditorium.** In Proceedings of the 2nd USENIX/ACCURATE Electronic Voting Technology Workshop (EVT’07).

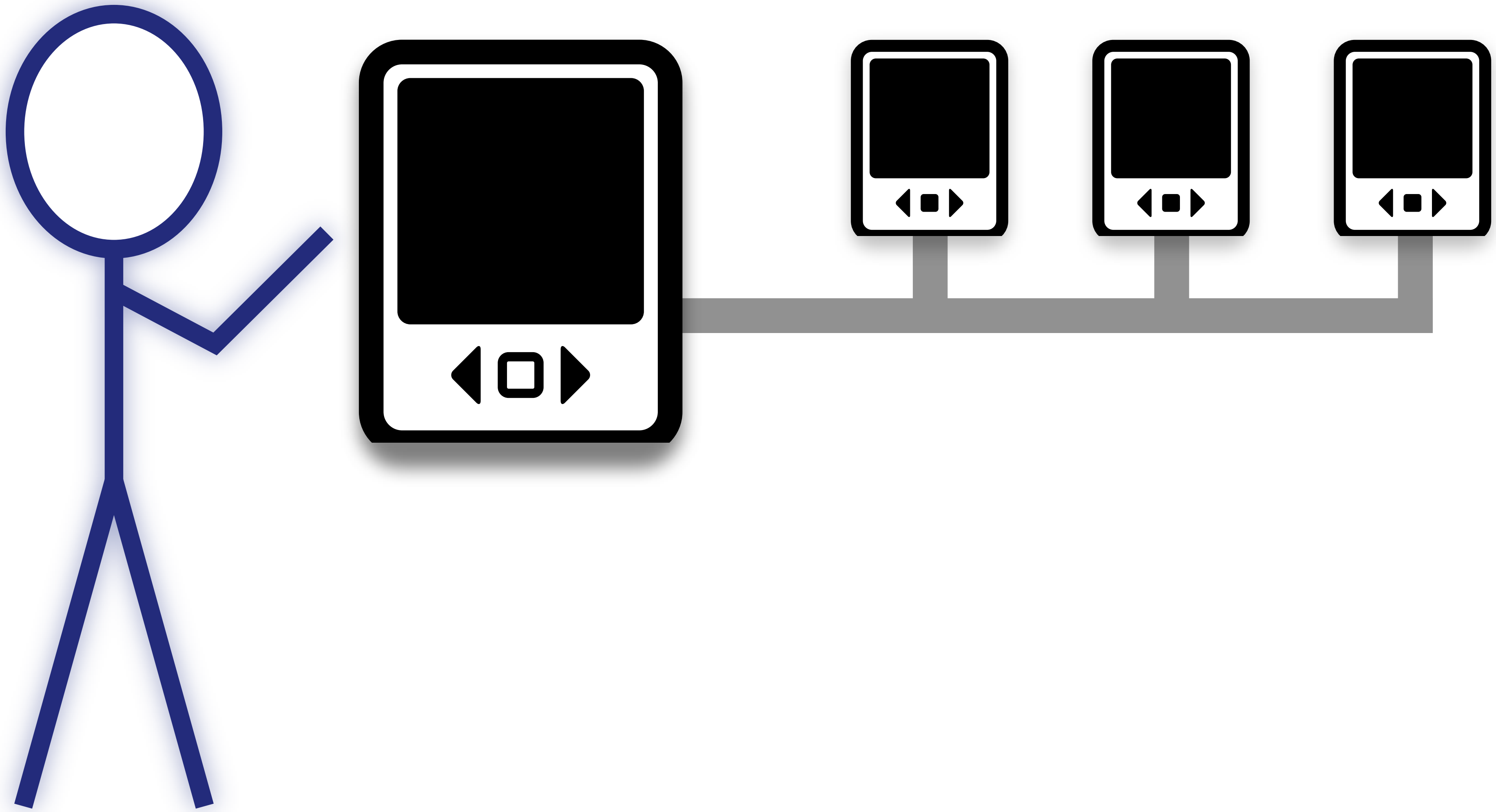
D. Sandler, K. Derr, and D. S. Wallach, **VoteBox: A Tamper-Evident, Verifiable Electronic Voting System**. 17th USENIX Security Symposium (USENIX Security ’08).

How can I be sure my vote is faithfully captured by the voting machine?

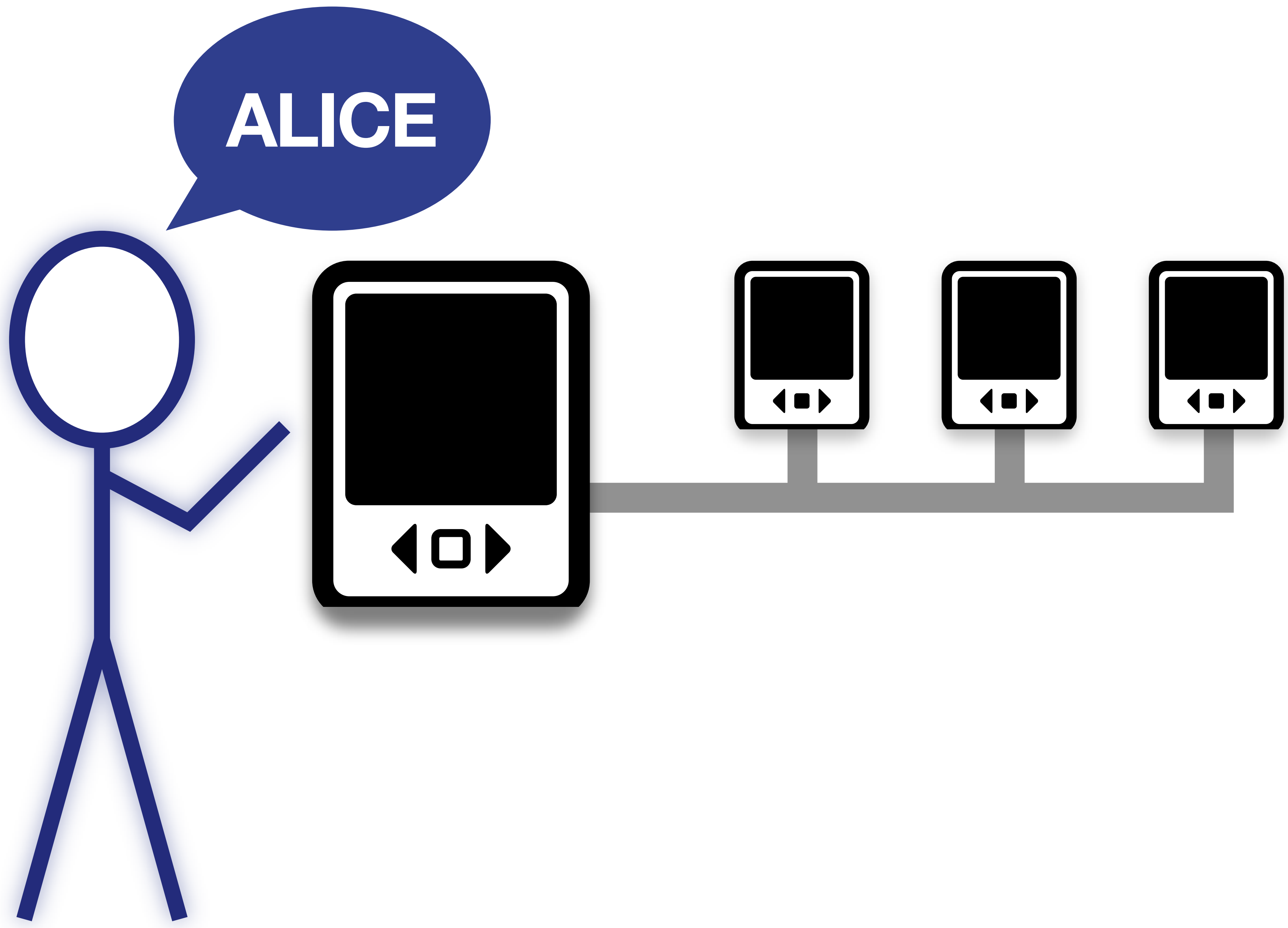
polling place



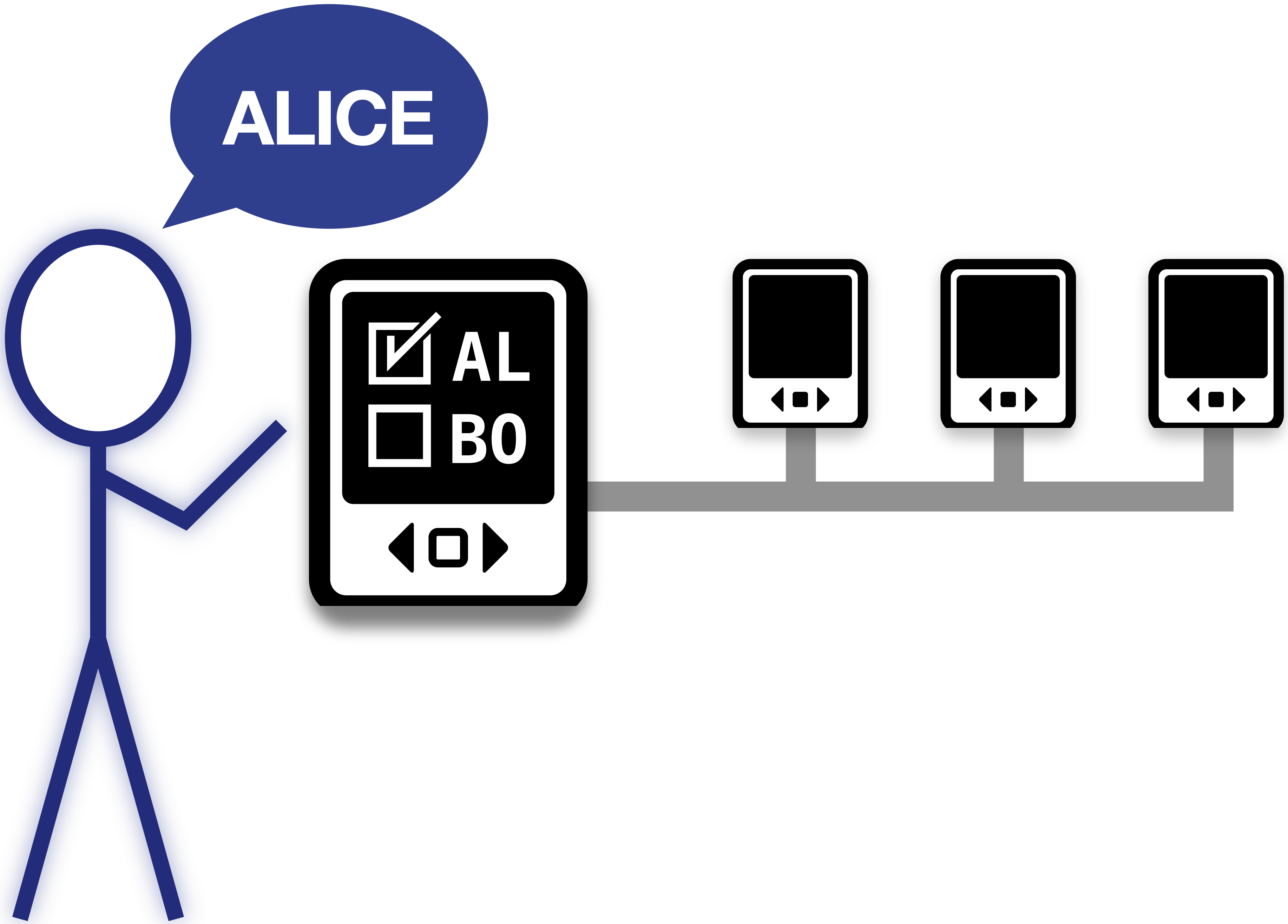
polling place

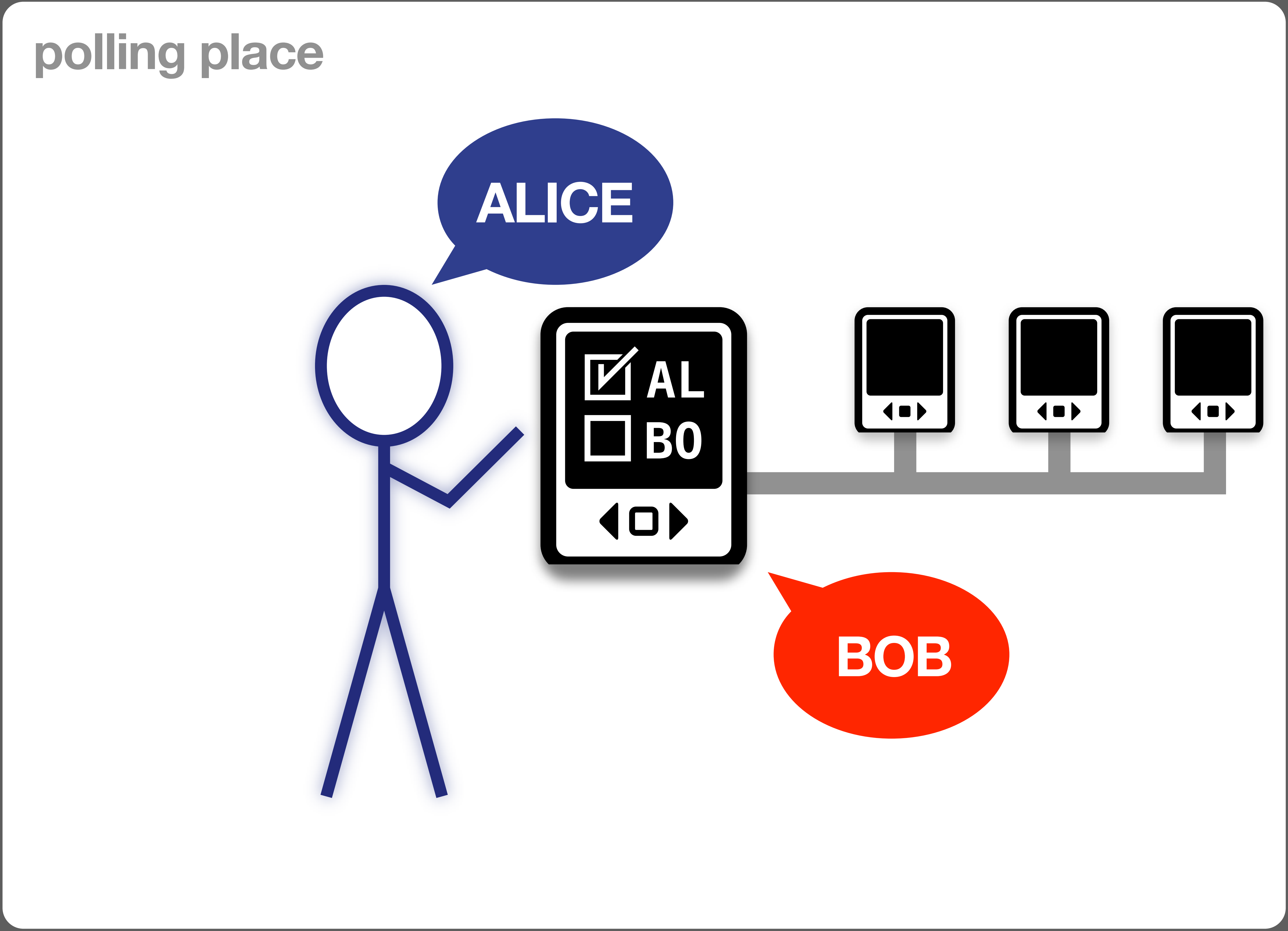


polling place

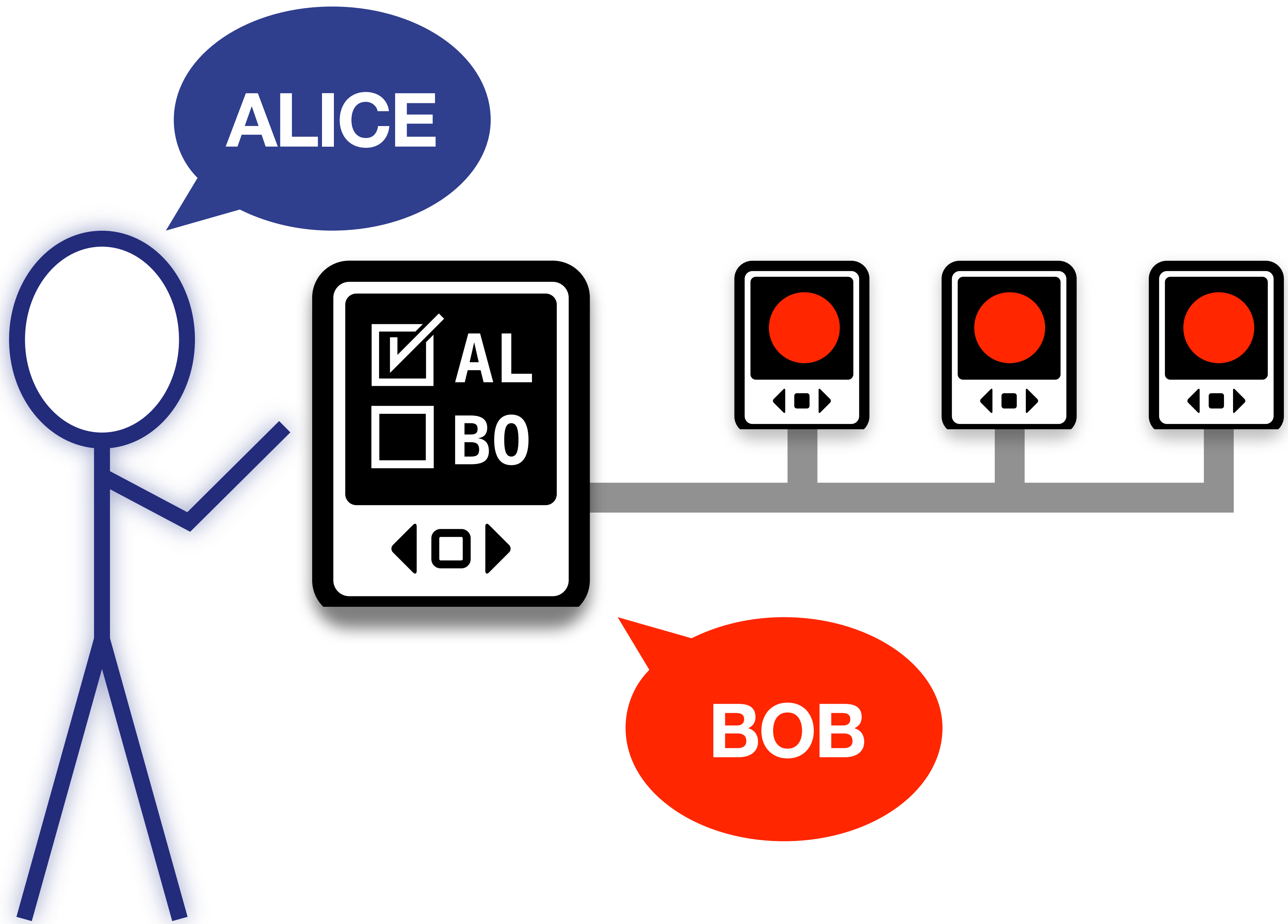


polling place




polling place
ALICE
AL
BO
BOB

polling place



VoteBox's approach:

ballot challenge

ballot challenge

ballot challenge

a technique due to [Benaloh '07]

ballot challenge

a technique due to [Benaloh ’07]

at the end, instead of casting your ballot:

force the machine to **show it to you**

ballot challenge

a technique due to [Benaloh ’07]

at the end, instead of casting your ballot:

force the machine to **show it to you**

this happens on election day

no artificial testing conditions (viz., “L&A tests”)

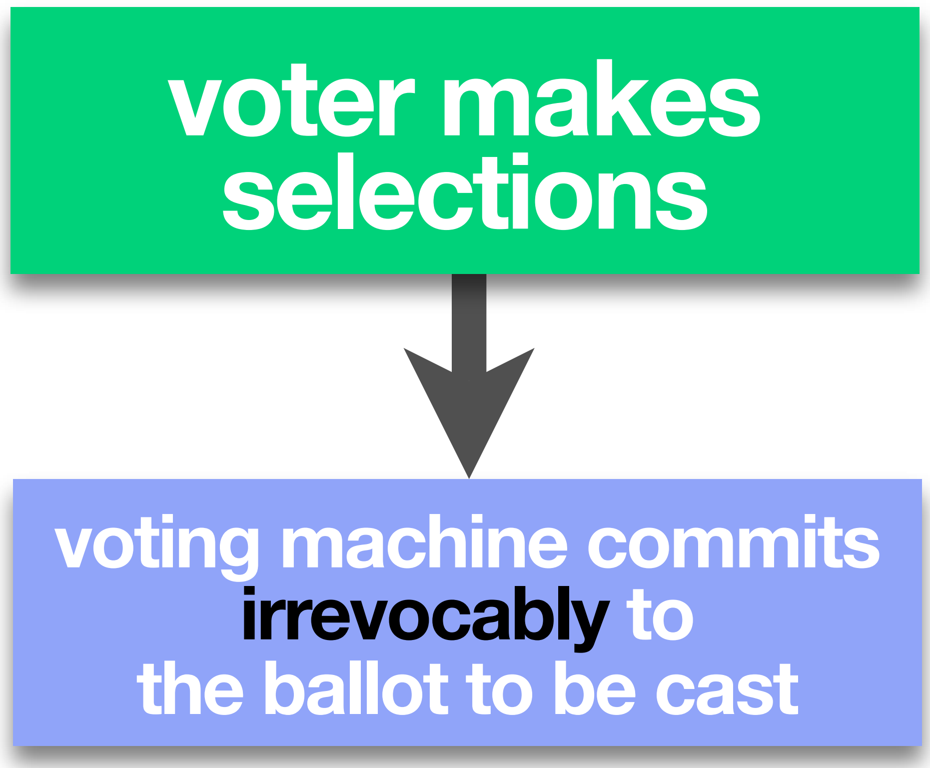
the voting machine cannot distinguish this from a real vote until the challenge

ballot challenge

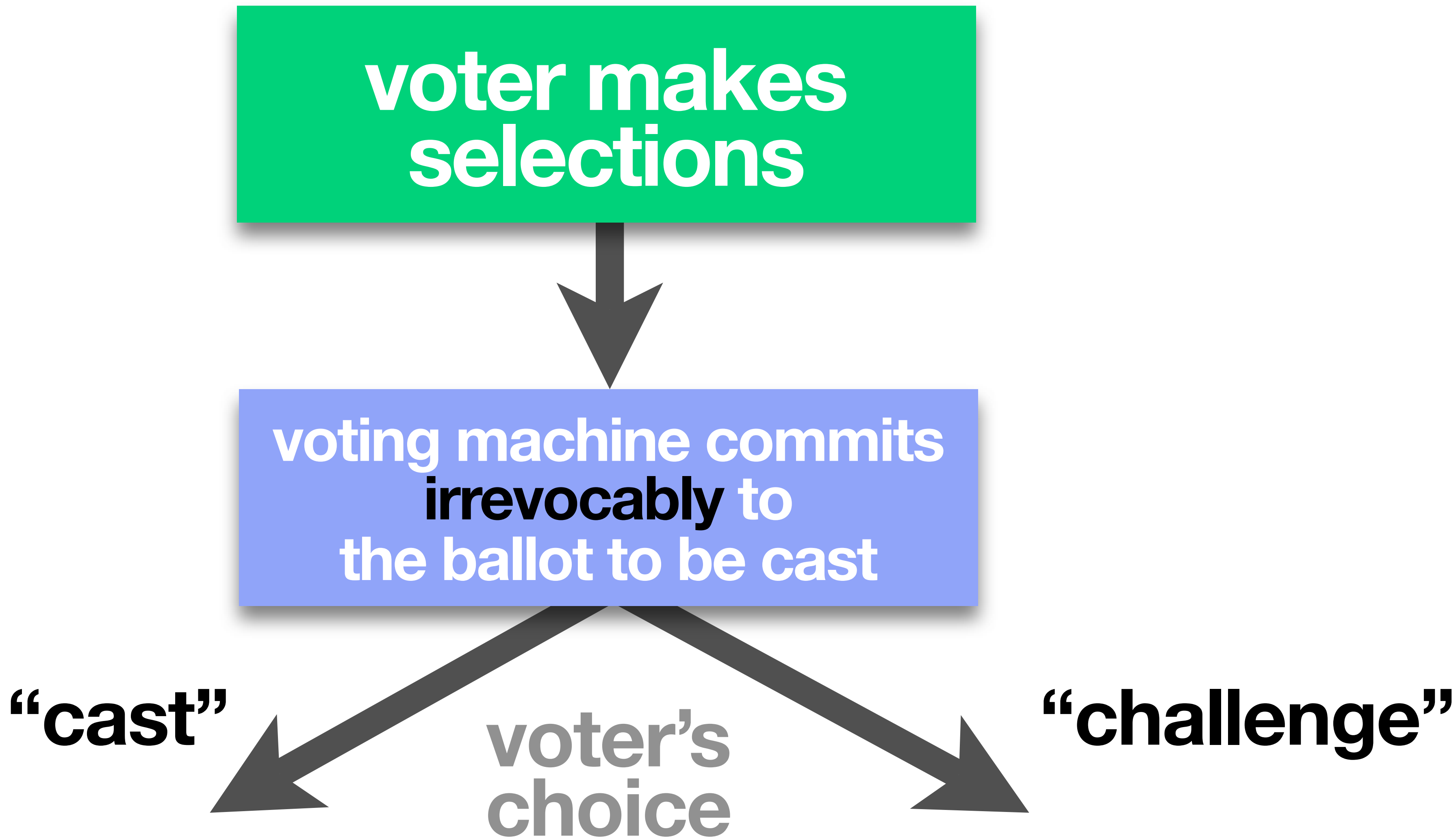
ballot challenge

voter makes selections

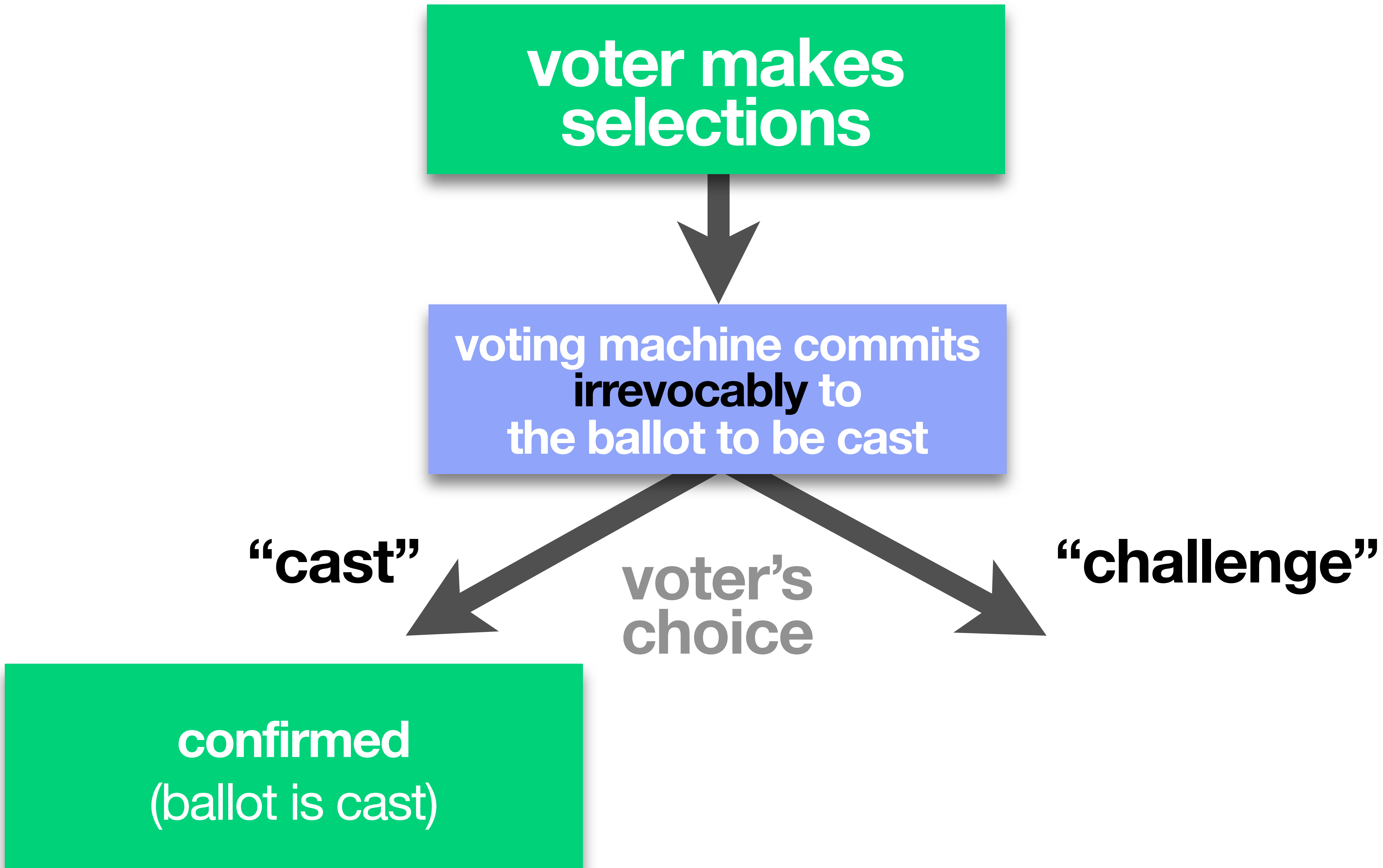
ballot challenge



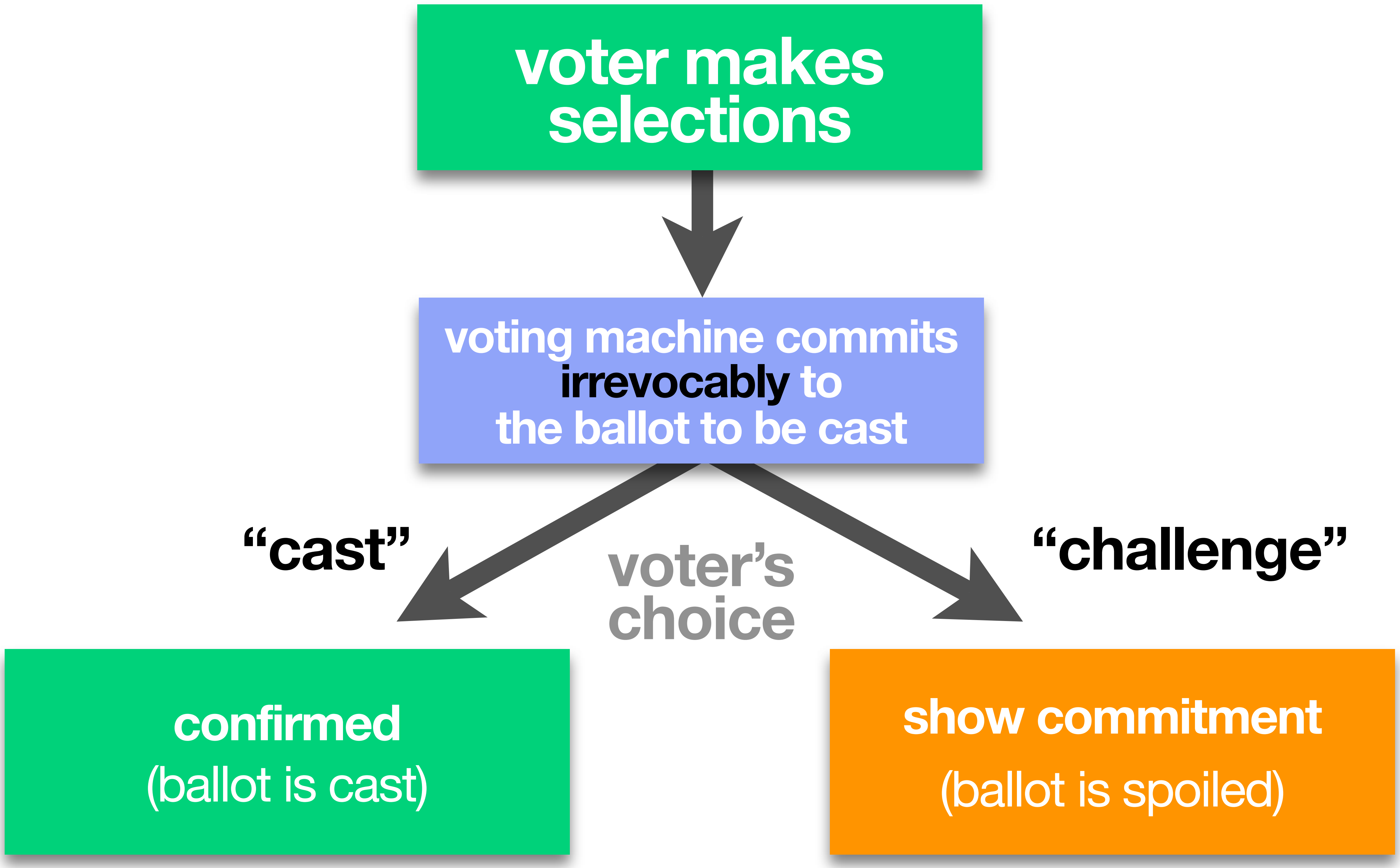
ballot challenge



ballot challenge



ballot challenge



ballot commitment

ballot commitment

What is the commitment?

How do we force the machine to produce proof of what it’s about to cast on the voter’s behalf?

ballot commitment

What is the commitment?

How do we force the machine to produce proof of what it’s about to cast on the voter’s behalf?

Benaloh’s proposal

print the encrypted ballot behind an opaque shield

You can’t see the contents, but you can see the page

the computer cannot “un-print” the ballot

ballot commitment

What is the commitment?

How do we force the machine to produce proof of what it's about to cast on the voter's behalf?

Benaloh's proposal

print the encrypted ballot behind an opaque shield

You can't see the contents, but you can see the page

the computer cannot "un-print" the ballot

How do you test the commitment?

ballot commitment

What is the commitment?

How do we force the machine to produce proof of what it's about to cast on the voter's behalf?

Benaloh's proposal

print the encrypted ballot behind an opaque shield

You can't see the contents, but you can see the page

the computer cannot "un-print" the ballot

How do you test the commitment?

Decrypt it.

But decryption requires the private key for tabulating the whole election!

Elgamal reminder

Two ways to decrypt:

$$
\begin{aligned}
E(c, r, g^a) &= \; < g^r, (g^a)^r M > \\
D(g^r, g^{ar}M, a) &= \frac{g^{ar}M}{(g^r)^a} \\
D(g^r, g^{ar}M, r) &= \frac{g^{ar}M}{(g^a)^r} \\
&= M
\end{aligned}
$$

g	group generator
M	plaintext (message)
r	random (chosen at encryption time)
a	(private) decryption key
g^a	(public) encryption key

challenging the machine

challenging the machine

When challenged, the machine must reveal *r*

We can then decrypt this ballot (only) and see if it's what we expected to see

In Benaloh, the encrypted ballot is on paper

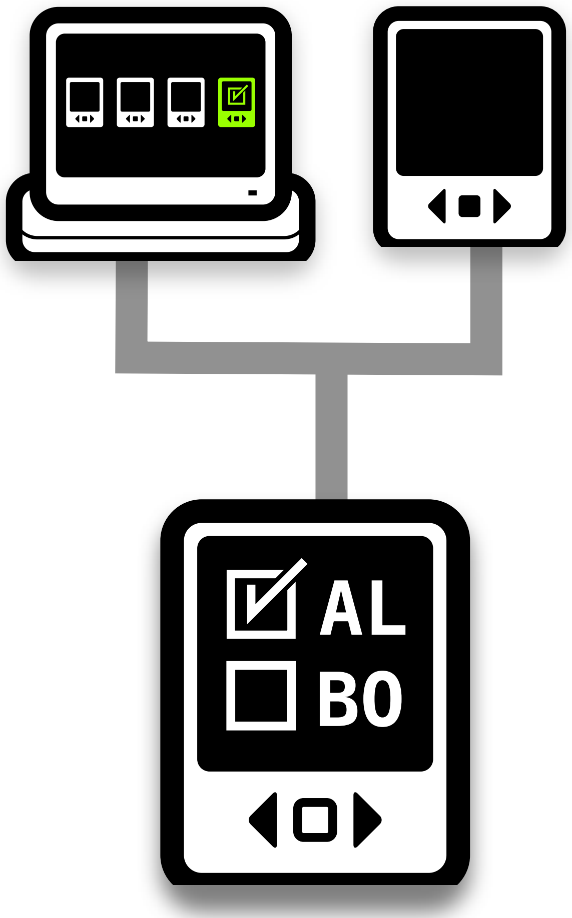
An **irrevocable** output medium

decrypting requires additional equipment

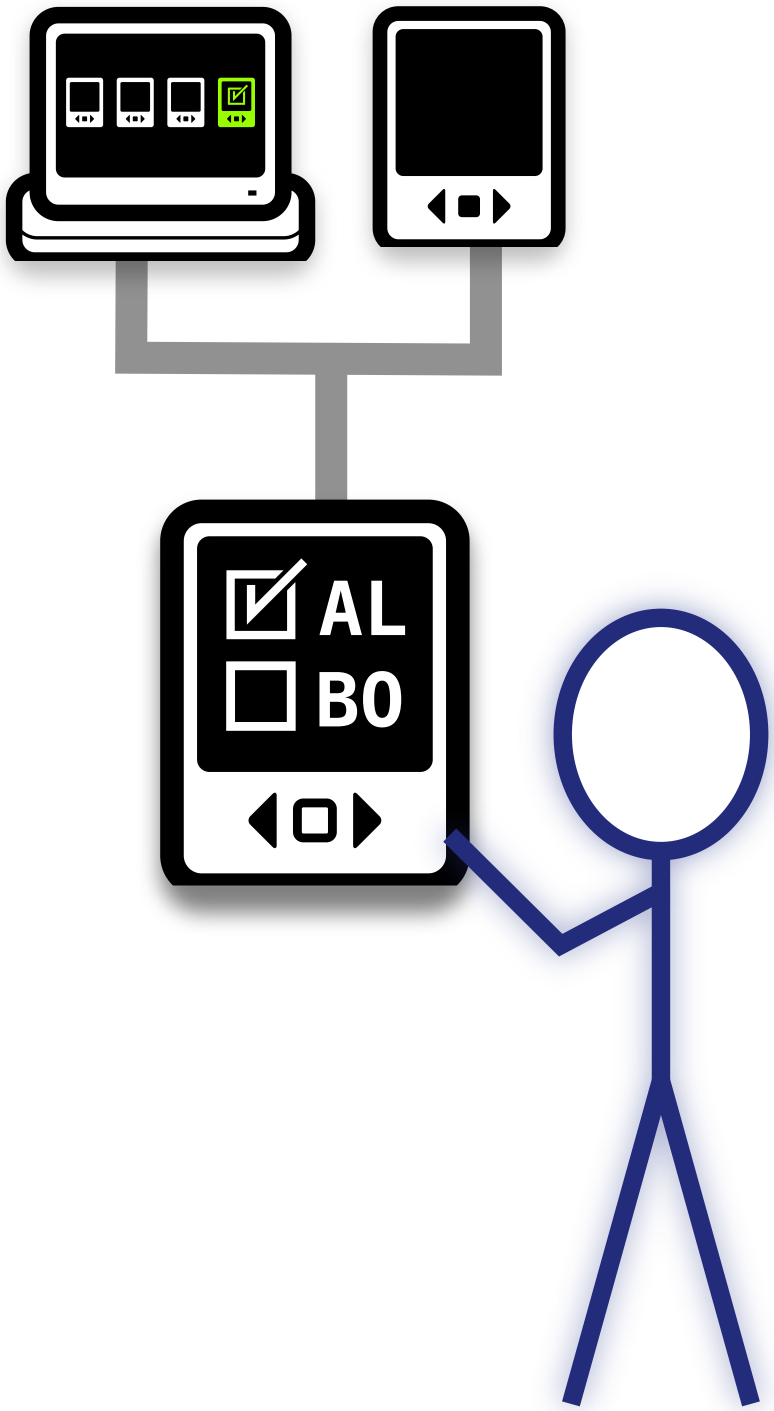
VoteBox happens to have its own irrevocable publishing system

(Its in-precinct LAN, where all machines replicate everywhere.)

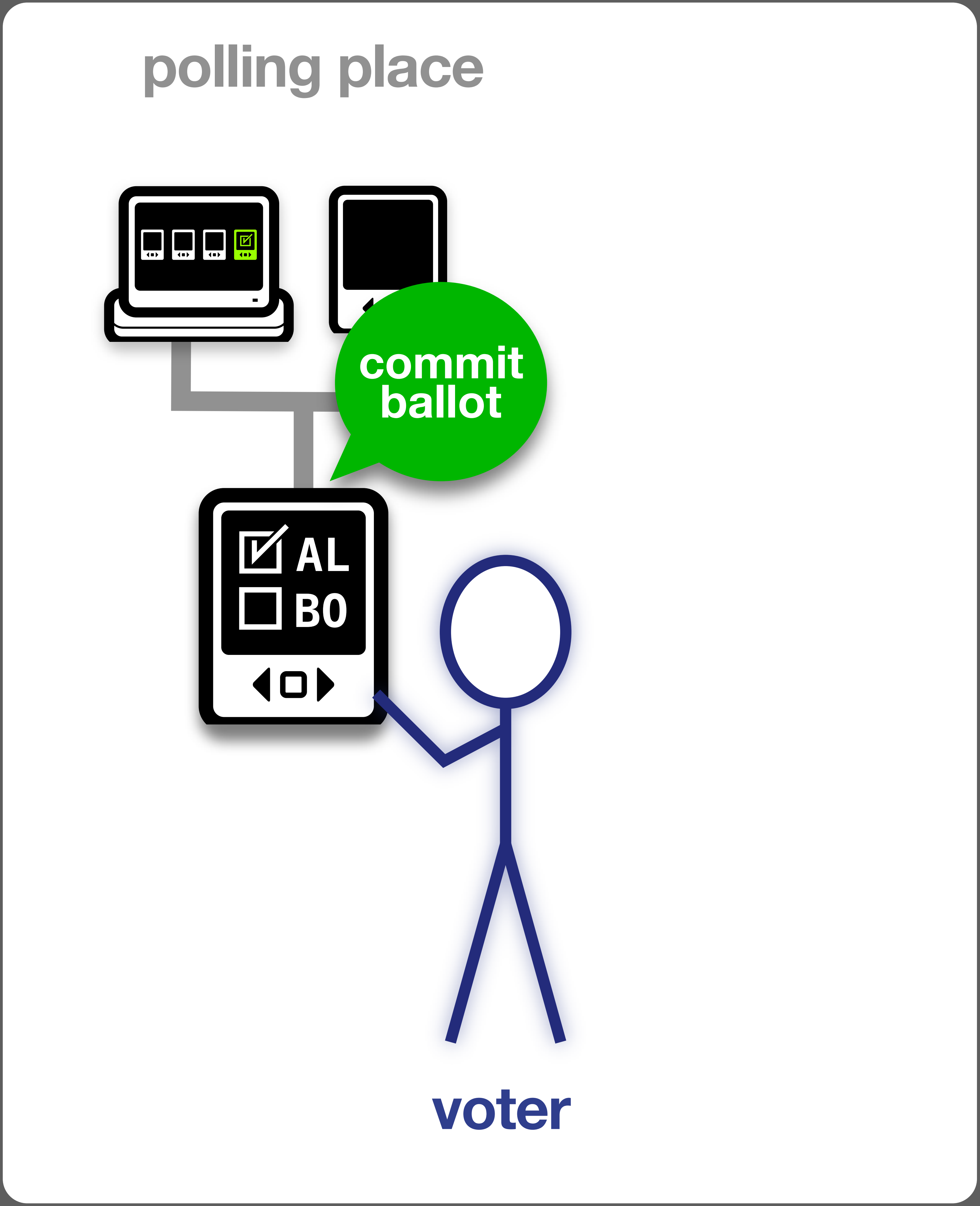
polling place



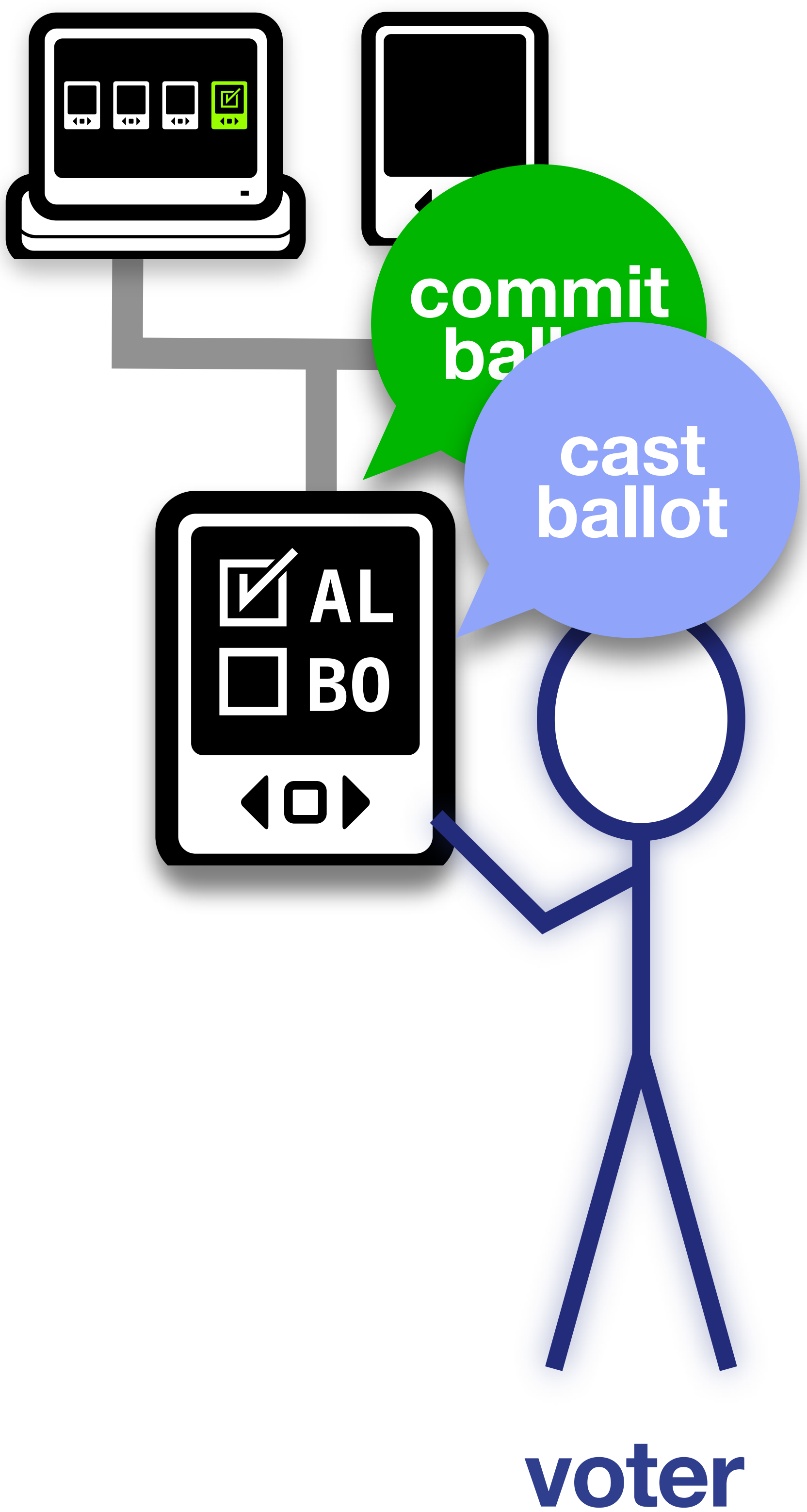
polling place



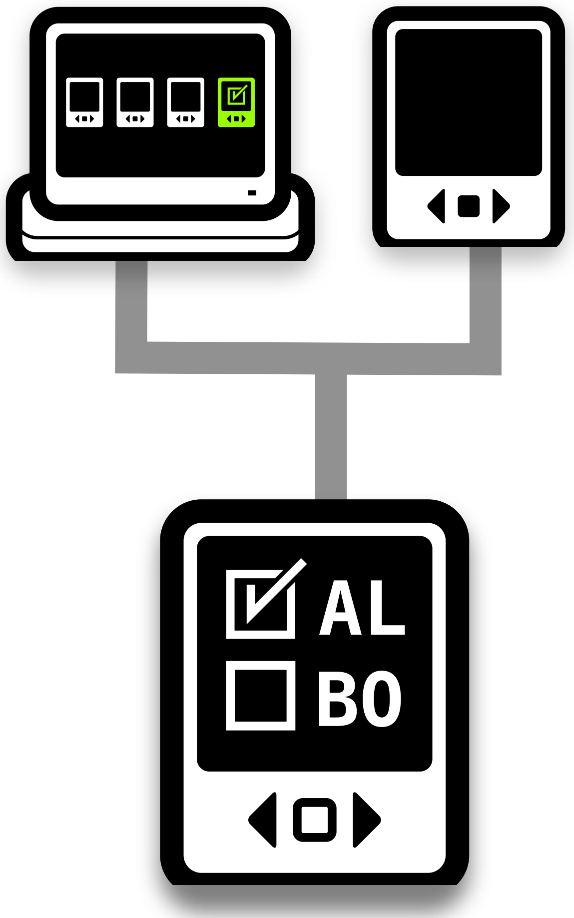
voter

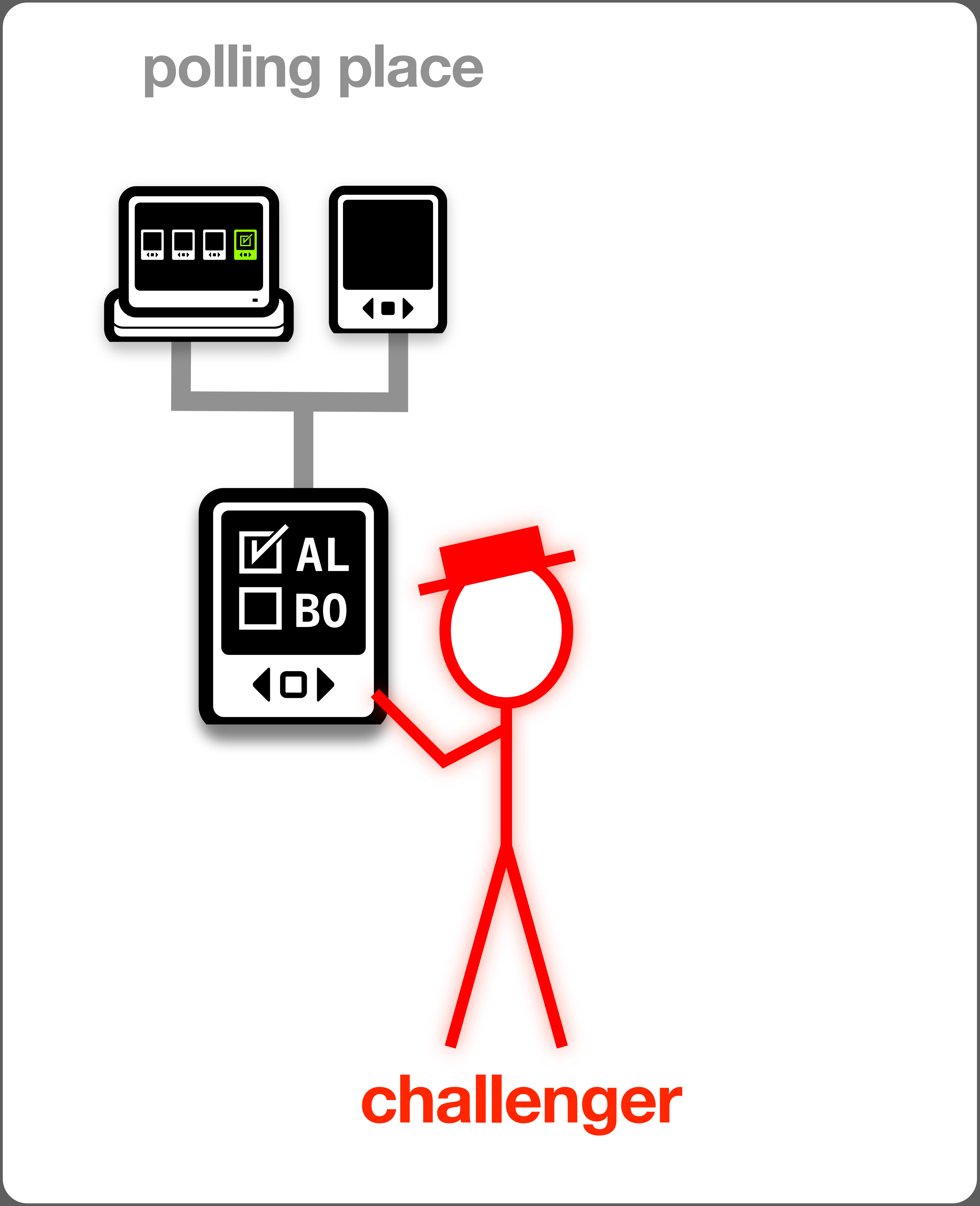

polling place
AL
BO
commit ballot
voter

polling place

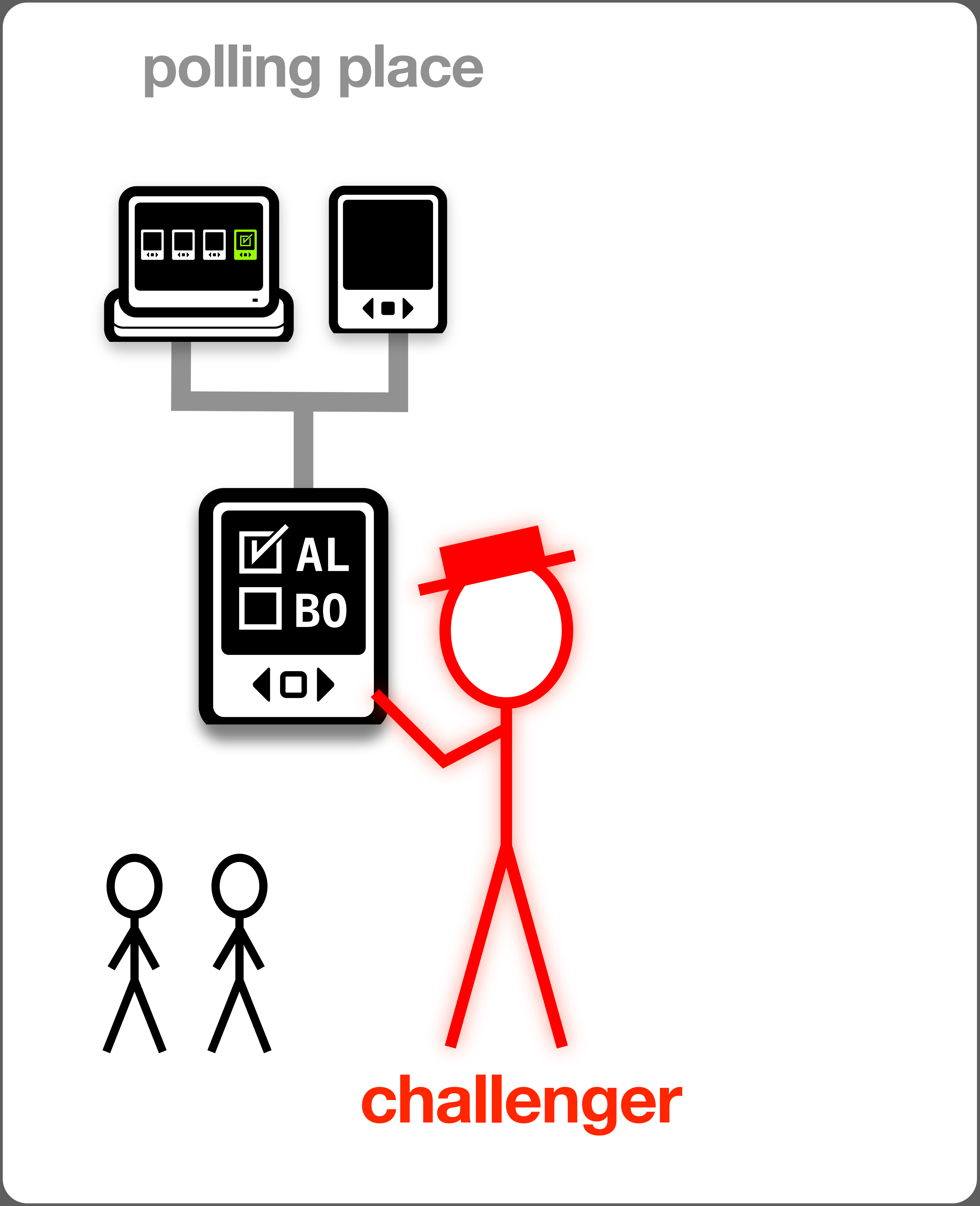


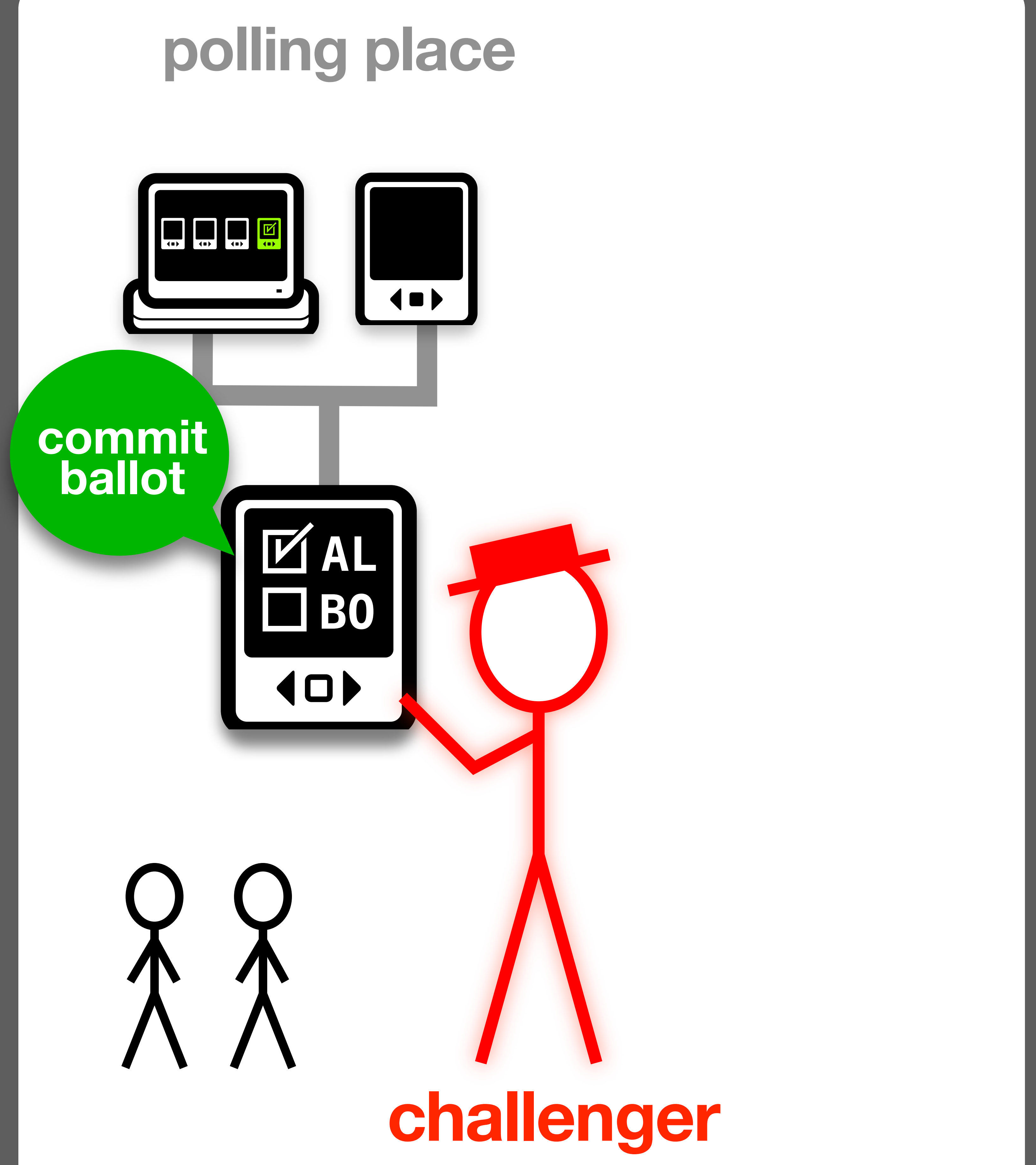
polling place

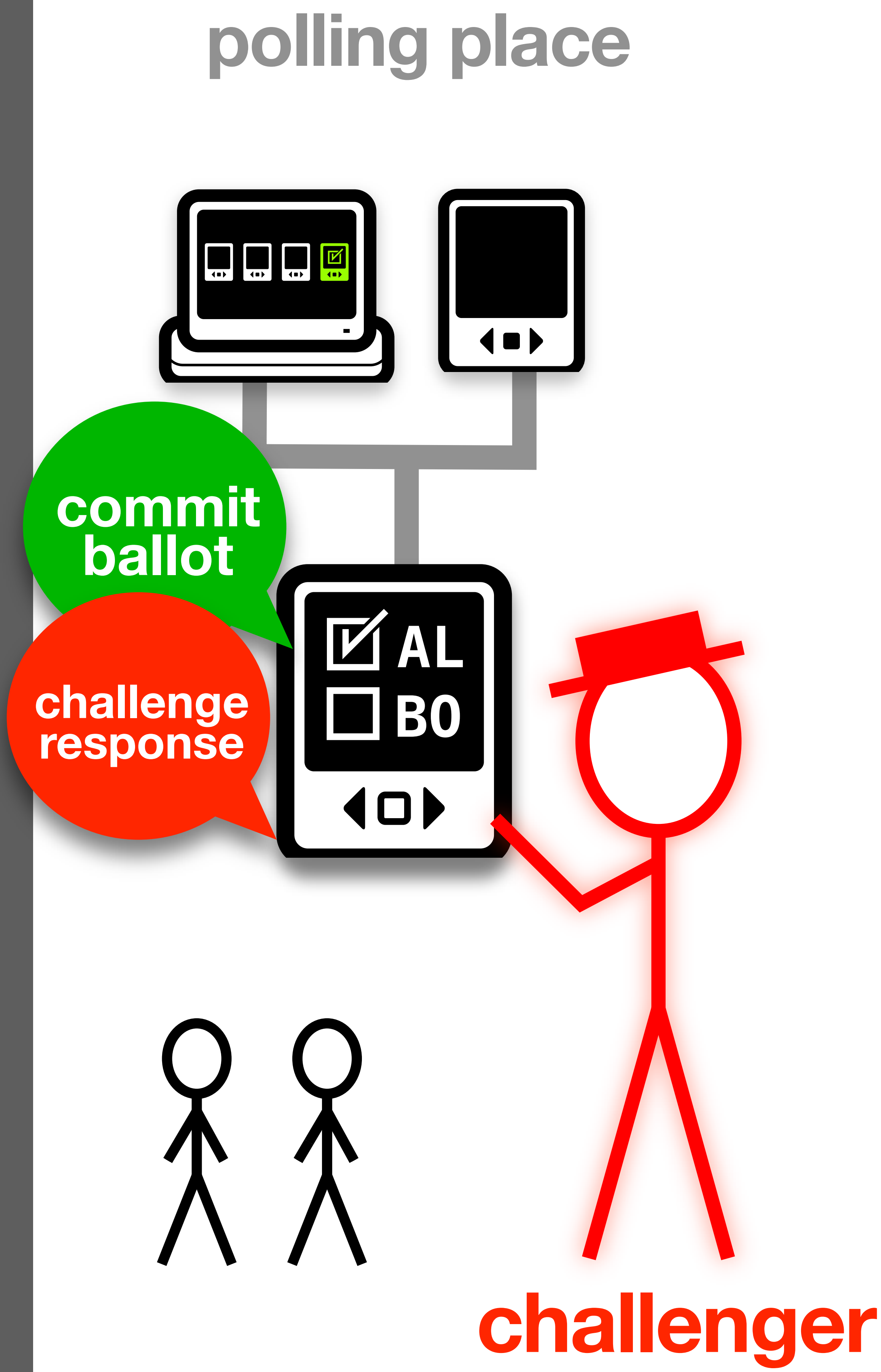


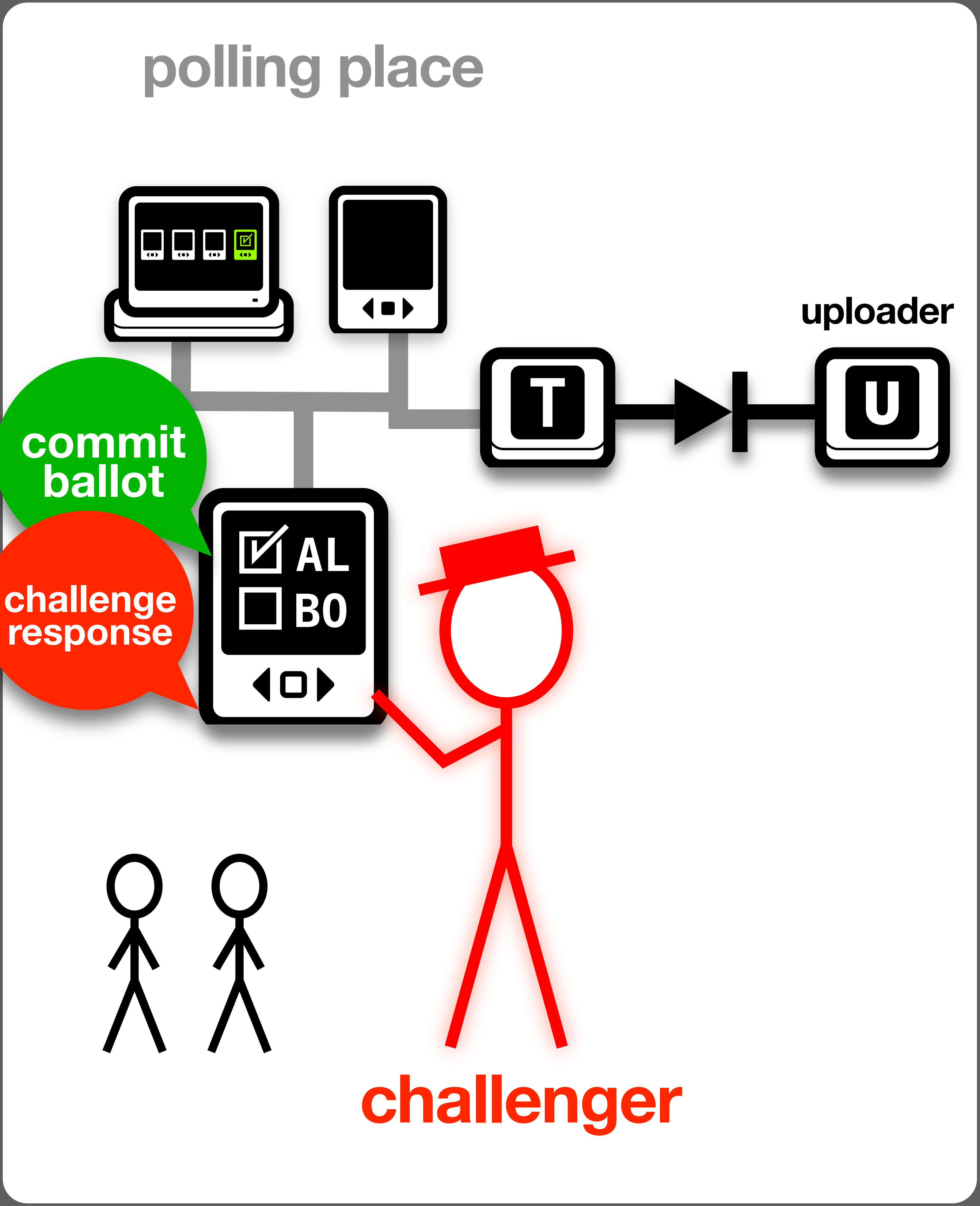


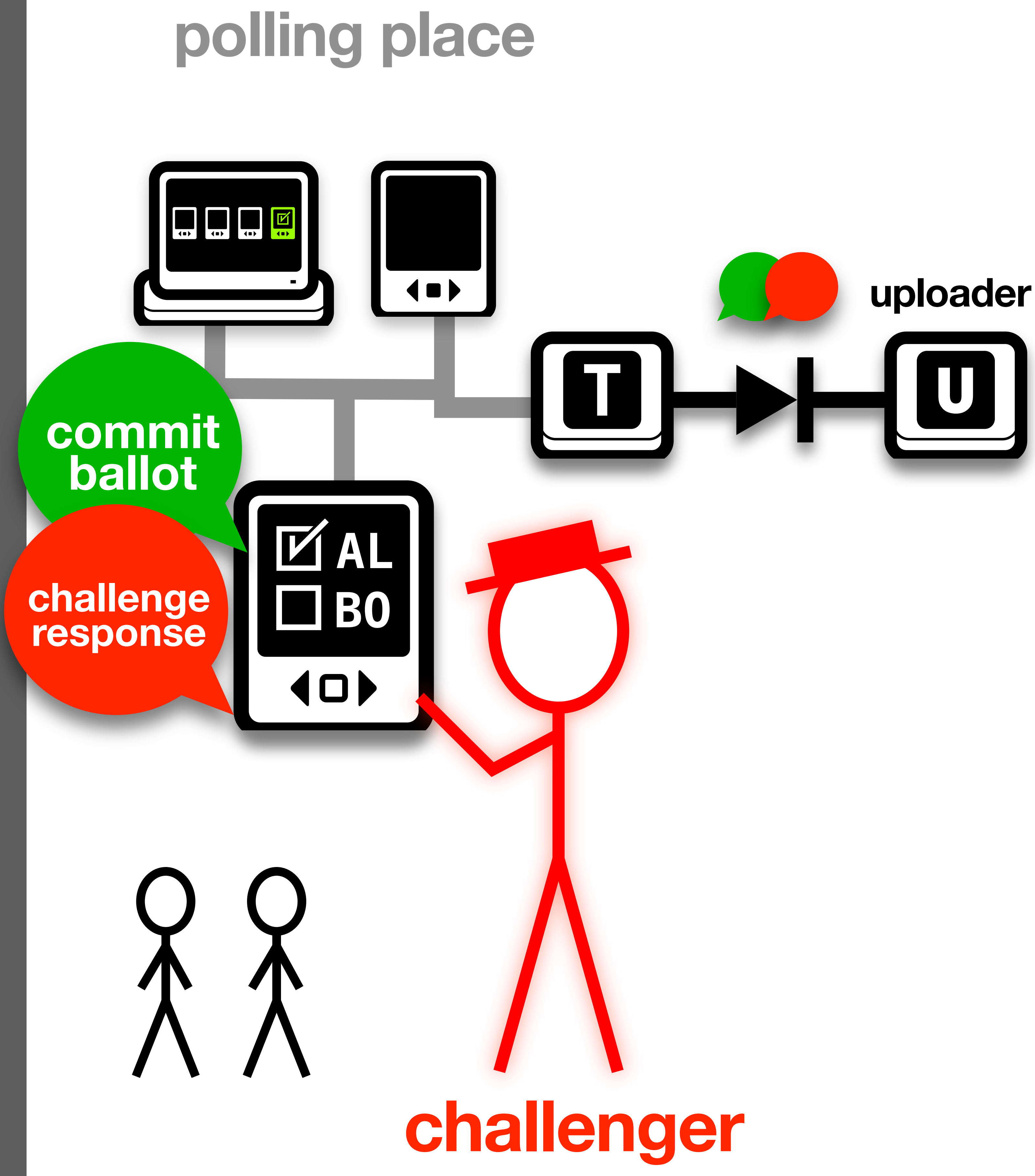
polling place
AL
BO
challenger

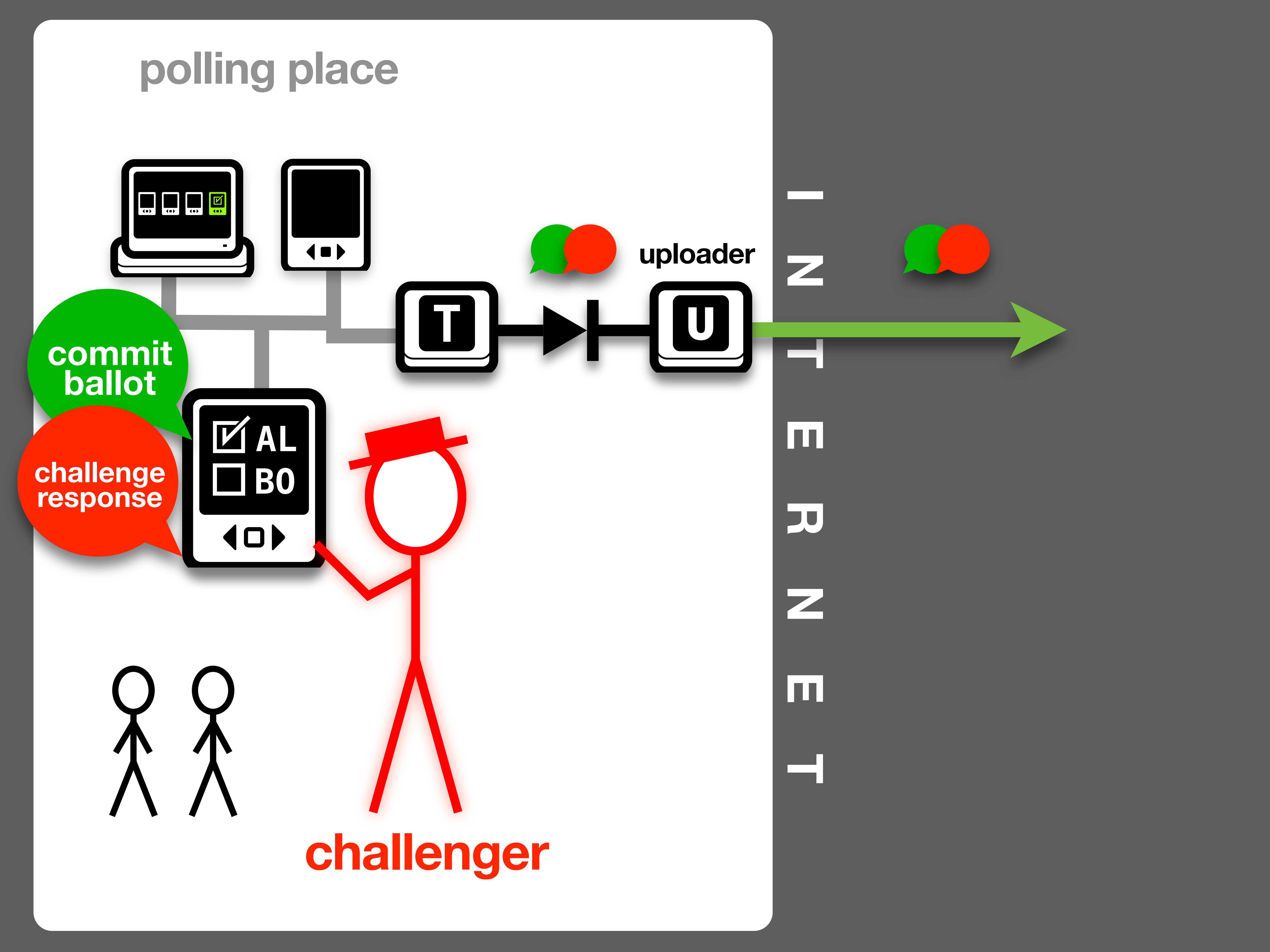

polling place
AL
BO
challenger

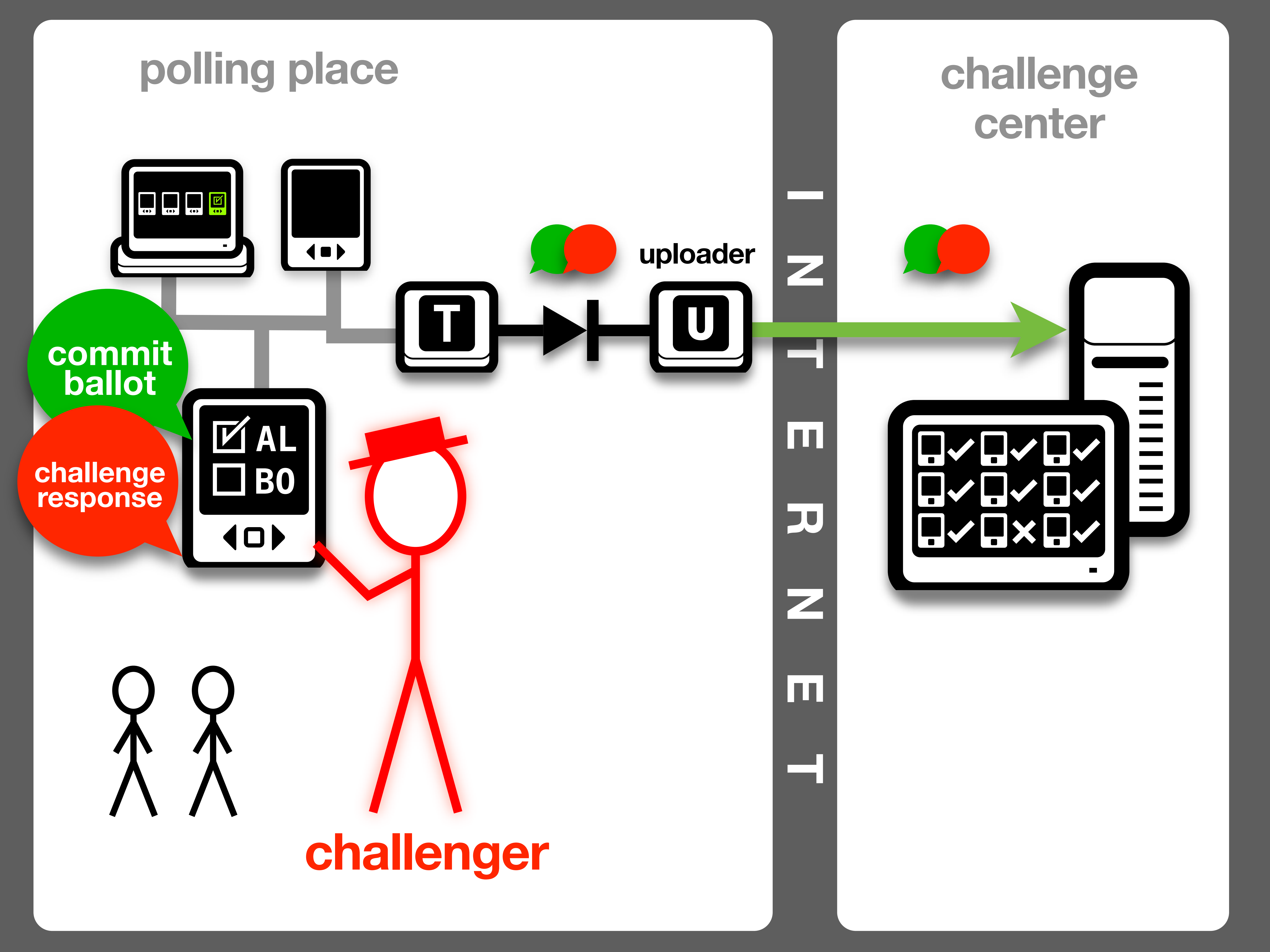

polling place
commit ballot
AL
BO
challenger

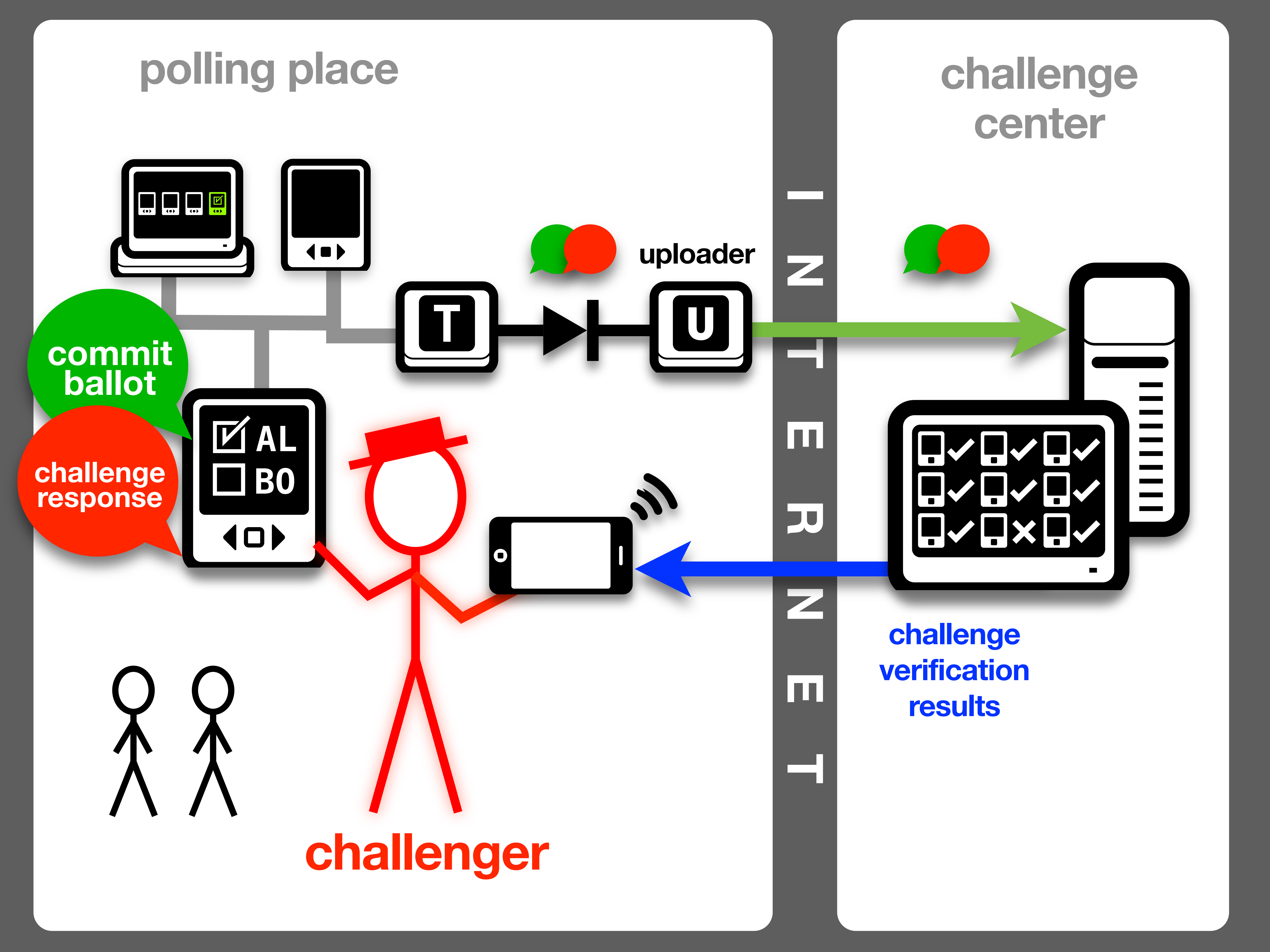

polling place
commit ballot
challenge response
AL
BO
challenger

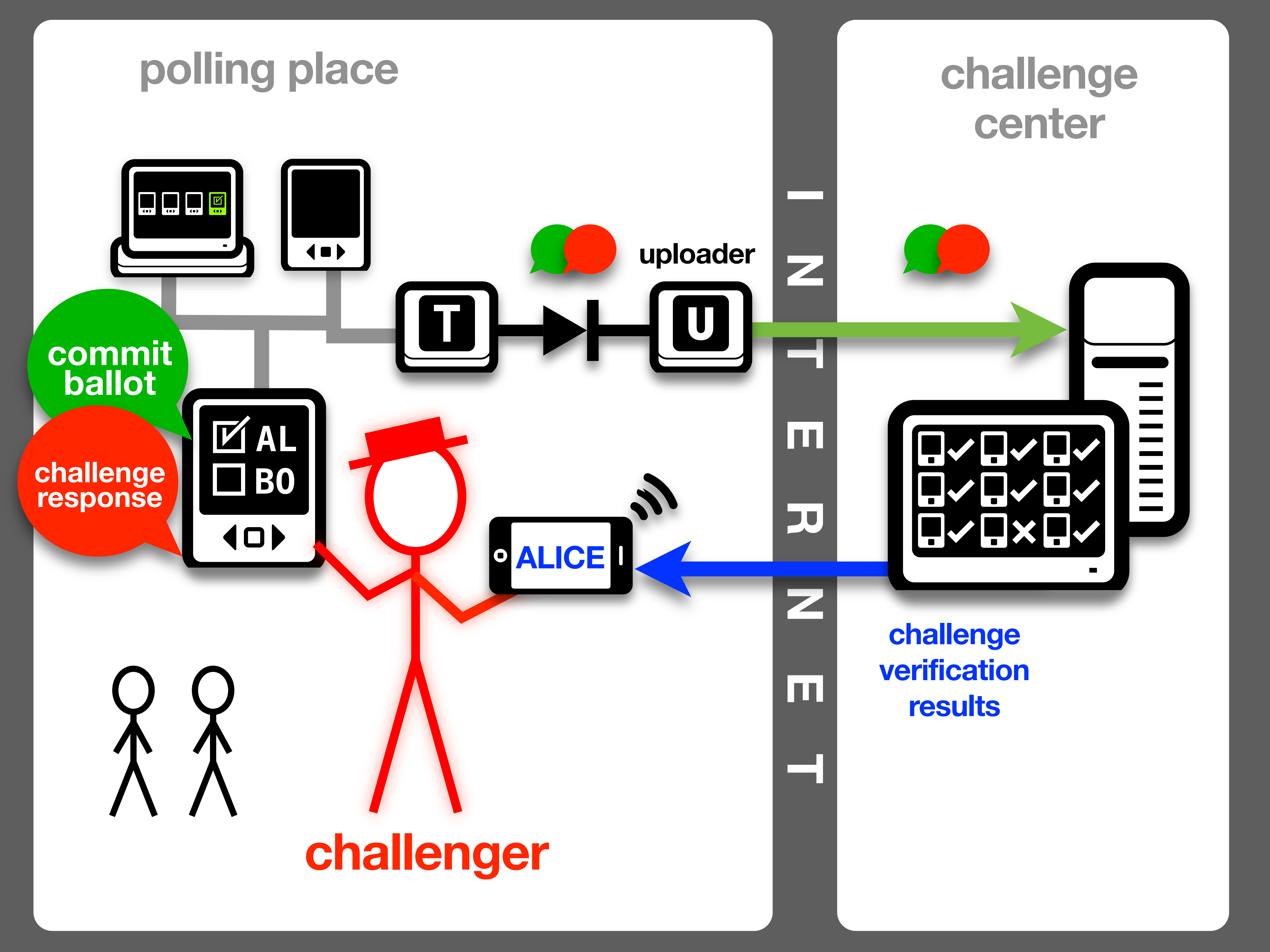

polling place
uploader
T
U
commit ballot
challenge response
AL
BO
challenger


polling place
uploader
T
U
commit ballot
challenge response
AL
BO
challenger


polling place
uploader
commit ballot
challenge response
AL
BO
T
U
INTERNET
challenger


polling place
challenge center
commit ballot
challenge response
AL
BO
T
U
uploader
INTERNET
challenger


polling place
challenge center
uploader
T
U
INTERNET
commit ballot
challenge response
AL
BO
challenger
challenge verification results



polling place
challenge center
commit ballot
challenge response
AL
BO
T
U
uploader
INTERNET
ALICE
challenge verification results
challenger