Implementation of a Web Bulletin Board for e-voting applications

Roland Krummenacher

Software and Systems

Hochschule für Technik Rapperswil, Schweiz

Abstract

Many studies for the practical implementation of e-voting base on the existence of a digital bulletin board. Such a bulletin board allows the voters to verify that their votes are counted correctly and makes the (encrypted) accumulation of the votes possible. In 2008, Heather and Lundin elaborated a proposal for such a bulletin board.

This paper first discusses the append-only web bulletin board of Heather and Lundin. It is found that in practice, a single bulletin board will not be sufficient. To prevent failures and manipulation, several boards need to be distributed on different stakeholders and the messages between these boards have to be replicated. It is further explained, why the definition in the publication of Heather and Lundin for the *distributed* web bulletin board is not sufficient.

The problems in the practical implementation of the distributed bulletin board are identified and various approaches are discussed. Finally, a proposal for the implementation of such a distributed web bulletin board is made, which meets the identified security features and can be scaled arbitrarily.

Table of contents

1. Introduction

In 2006, Ben Adida wrote in his dissertation that „Cryptographic voting protocols revolve around a central, digital bulletin board. As its name implies, the bulletin board is public and visible to all, via, for example, phone and web interfaces. All messages posted to the bulletin board are authenticated, and it is assumed that any data written to the bulletin board cannot be erased or tampered with." (Adida, 2006).

This work is based on the publication of Header and Lundin from the year 2008 (Heather, et al., 2008) which made an approach for the practical implementation of such a digital bulletin board. They fill out a gap that was left open by several publications on the topic: In their paper, they list some of the authors of publications that have the practical implementation of e-voting applications to issue and assume in these publications the existence of a digital bulletin board, but the implementation of the bulletin board, however, is not discussed by any of them further. Heather and Lundin identify in their study the basic requirements for such a board and define their "Append-Only Web Bulletin Board", abbreviated WBB.

1.1. Overview

In the first part of this work the requirements to a single WBB are explained. This section discusses and complements the work of Heather and Lundin. Therefore, it is recommended to the reader to first become familiar with their paper "The Append-Only Web Bulletin Board". In the second part of this paper a proposal is made for a distributed web bulletin board. Such a distributed WBB is explained by Header and Lundin only as a sketch. The discussion of the problems in the practical implementation and the solutions represent the value of this work.

1.2. Motivation and definition

The paper discusses solely web bulletin boards which are used in e-voting systems. Other applications such as auctions are not considered in this work. The security of the overlying e-voting application and the security of the bulletin board should be independent of each other, i.e. the WBB only guarantees the correctness of its history and the authenticity of the messages published, but not the correctness of the content of the messages in the context of the vote. The e-voting application can, in turn, rely on a bulletin board which offer the properties listed in this paper.

1.3. The Web Bulletin Board

Lundin and Heather define a web bulletin board as follows. There are three agents: The bulletin board itself, readers and writers. The board is public; everybody can read the published messages. The board allows predefined writers to publish messages. New messages can only be attached to the history of the board, once a message is published, it cannot be changed or deleted in any way. Inserting, changing or deleting messages is a violation of the protocol and can be detected and proved by any reader.

The board itself may not generate a message, but only the writers. Each message is provided with an identification and a signature of the respective writers, so that the message may be clearly assigned to its writer. The board is responsible for validating the messages before publication (e.g. checking the authenticity of the message) and making sure that the content of the messages after the publication does not change anymore.

1.4. Preconditions

Heather and Lundin put ahead an "adequate public key infrastructure", where the public keys of the writers and the boards are known by all agents, but the private keys are only known to their respective owners. It is under these conditions assumed that digital signatures can be performed only by the owner of the private key.

2. The single Web Bulletin Board of Header and Lundin

The definition of an independent WBB and the reasoning for the required properties are described in the work of Heather and Lundin. I omit the repetition of the above, but complement the definitions of the authors where necessary with regard to the proposal for the implementation in Chapter 3.5.

2.1. History

The web bulletin board consists of a sequence of entries, with an entry consisting of a message, and other, security-related parts. Such a sequence of messages is called history and marked <wbb1, ..., wbbn>, with wbb_i corresponding to the entry i. Such an entry consists of the elements in Table 1.

m_i	Message i.
T_i	Time of the writing of the message m_i by the writer
W_i	Identification of the writer of the message m_i.
H_i	Hash on $H(m_i, T_i, W_i, H_{i-1})$, where $H_0 = 0$.
$WSign_i$	Digital signature of the writer W_i with its private key of the hash H_i.
T_i'	Time of the signature by the board.[1]
$BSign_i$	Signature of the board with its private key on $(WSign_i, T_i')$.

Table 1: Complete list of elements of a single board entry.

It is important to note that the hash H_i is created over the current entry and over *the hash of the previous entry* in the history and then is signed by the writer and indirectly by the board. This leads to a linked history. It is easy to see that this construction describes an "Append-Only Web Bulletin Board", allowing only additions to a WBB. This history has the following characteristics:

- The history can be reviewed by anyone at any time for consistency by calculating the hash chain and comparing the result with the hash value of the currently last entry.
- If the history is consistent, every prefix of this history is consistent too.
- If wbb_λ is the last entry in the history, then H_λ is the Hash of wbb_λ and thus the current state hash of the board. H_λ must be used by the next writer as the last component in the formation of the hash[2].

Next, the authors introduce a security parameter ε. ε defines the maximum allowed time between the constitution of the message and publication on the WBB, i.e. between T_i and T_i' in Table 1. A message which is written on T_i, and processed by the WBB later as $T_i + \varepsilon$ will be rejected. The author is forced to retry to publish the message on the WBB.

The security parameter ε expands the features of the history as follows:

- When a reader reads the history of the board on time T and later a message with creation date before T - ε is published, then the reader is able to prove that the board was forged.

In the context of e-voting this is important because it gives the reader the assurance that the WBB cannot withhold any messages to publish them later (after T + ε). ε should therefore be kept as small as possible so that the margin of the WBB for making a decision whether to publish a particular message or not is kept as small as possible, but large enough that the writers can publish their messages on the assumption of normal latency of network connections. Heather and Lundin recommend a value of a few milliseconds for ε.

[1] In contrast to the definition of Heather and Lundin, T_i' is defined as part of the entry too, because in their publication it is not clear whether it is visible in clear text as part of the signature $BSign_i$. However, to verify the signature $BSign_i$ this value is essential.
[2] Heather and Lundin specify in their Definition 27 H_λ as the third component. This is a mistake.

2.2. Definitions

To remain constant with the labeling of Heather and Lundin, the abbreviation S is used in the following for digital signatures. Thus, $S_W(x)$ denotes a signature by the writer on a value x, $S_B(x)$ a signature by the board, each with the corresponding private key. Further, T_B' corresponds to T_i' from Table 1.

2.3. Reading messages from the board

Table 2 shows the protocol for reading the messages that are published on the WBB.

No.	Time	WBB		Reader
M0			<-	Read
M1	T_B	$<wbb_1, ..., wbb_\lambda>, H_\lambda, T_B, S_B(H_\lambda, T_B)$	->	

Table 2: The protocol to read the history of the WBB.

In the context of a *web* bulletin board, the message M0 is a simple HTTP-Request to the board. The WBB returns the current history at the time T_B, along with the status hash H_λ, the current time stamp T_B and a signature on these last two elements. The signature serves as a receipt for the reader, making it impossible for the board to revoke the history later or generally to change the history prior to the time T_B. If the reader is interested in the correctness of the received history, she must actively replicate and compare the hash chain (of H_0=0 up to and including H_λ).

2.4. Publishing messages on the board

Writing messages is more complex than the reading process. First, the protocol will be explained. Then, various scenarios are discussed, which can occur when publishing messages on the WBB.

2.4.1. The protocol

Table 3 shows the protocol for writing messages on the WBB.

No.	Time	WBB		Writer
M0			<-	$Read_{ForAppending}$
M1	T_B	$H_\lambda, T_B, S_B(H_\lambda, T_B)$	->	
				Verification: If $T-T_B > \varepsilon$ M1 is discarded.
M2	T		<-	$m, T, W, H(m, T, W, H_\lambda), S_W(H)$
	T_B'	Verification: If $T_B'-T > \varepsilon$ the message is rejected		
	T_B'	Publication of the message on the board (acc. to Table 1).		
M3	T_B'	$S_W(H), T_B', S_B(S_W(H), T_B')$	->	
				Verification: If $T_B'-T > \varepsilon$ the WBB is faulty.
				Storing messages M1 and M3 as receipt.

Table 3: The protocol to publish messages on a WBB.

At the beginning, the writer sends a read request, where he is only interested on H_λ and T_B. The writer checks the signature of the response. Is T_B older than ε, the message is discarded and a new request is made. It does not emerge from the work of Heather and Lundin why exactly this procedure is security relevant. I will discuss this in the next chapter "Scenario 2: The WBB denied the publication of messages".

Now the writer generates its actual message with content m, the current time stamp T, his identification W, the hash over the elements m, T, W and the just received H_λ and the signature of the generated hash value. He sends this to the WBB.

The WBB validates whether the latency falls within the security parameter ε[3]. If not, the message is discarded. The writer must create a new message. Otherwise, the message is published together with a time stamp at the time of publishing T_B' and a signature of the board over the signature of the writer and T_B'.

[3] Heather and Lundin make in Chapter 2.5 with respect to message M2 the following restriction: $T-T_B > \varepsilon$. This is a mistake. The restriction should be $T_B'-T > \varepsilon$. After M0, T_B is no longer known to the board as the WBB is stateless for read requests.

The signature of the writer, the time stamp T_B', and the signature of the boards over these elements are sent to the writer. He checks whether the board has made the verifications and if the signature is correct. The signature of the board serves as receipt so he can prove at any time that his message was added on the appropriate place to the history of the WBB. To demonstrate this evidence, the writer must keep both the receipt and the original message.

Delivering $S_W(H)$ and T_B' in message M3 was appended to the protocol specification of Heather and Lundin (see Heather, et al., Chapter 2.5). T_B' must be in plaintext part of the receipt so the writer can verify the signature of the board. This time stamp is published on the board too, it is however necessary to complement the receipt with it, for the case that the board erroneously not releases the message. Otherwise the receipt as evidence is incomplete.

2.4.2. Discussion of scenarios when writing messages

There are two scenarios that I think are particularly relevant to be discussed at this point. For the analysis of other security-specific characteristics, including proofs, see the work of Heather and Lundin, Chapter 2.6.

Scenario 1: Two writers compose the message of $m_{\lambda+1}$ and send them for publication to the WBB.
According to the definition, only one of the two messages can be accepted, the other must be rejected and the appropriate writer must write a new message $m_{\lambda+i,}$ $i \geq 2$. I assume a value for $\varepsilon = 5$ minutes for the following variants. This is only illustrative and can be arbitrarily scaled. The variations are not limited to two messages. Having n messages, n-1 messages will be rejected (marked with the result "X").

Variant 1: messages arrive consecutively.
If both messages arrive consecutively the first received message is accepted. Which message first arrives at the WBB can be random if the time differences due to the latency of the Internet are small enough.

Writer	Time of writing	Time of arrival	Result
W1	12.00	12:03	X
W2	12:01	12:02	OK

Table 4: Numerical example for variant 1.

Variant 2: Latency exceeded.
The latency of message 1 has been exceeded. On 12.06, the message has been checked and rejected. On 12.07, another message was published.

Writer	Time of writing	Time of arrival	Result
W1	12.00	12:06	X
W2	12:05	12:07	OK

Table 5: Numerical example for variant 2.

Variant 3: The messages arrive simultaneous.
In this case, Heather and Lundin define "early rejection", i.e. the message that was *written at last*, is accepted and all other, simultaneously arrived messages that were *written before* this message will be rejected. It does not matter which message is accepted in principle. But early rejection has the advantage that if the last written message falls outside the maximum latency ε and therefore must be rejected, all other messages received at the same time must be rejected too (since they must have exceeded the latency too). Thus, the probability that the last written message is valid is higher than for all other messages and no list of alternate messages have to be kept, in case that it is invalid.

Writer	Time of writing	Time of arrival	Result
W1	12.00	12:03	X
W2	12:02	12:03	OK

Table 6: Numerical example for variant 3.

Scenario 2: The WBB denies the publication of messages
As we have seen a WBB can no longer change its history. If the WBB was acquired by an attacker, all fraud attempts may be discovered. As the only attack scenario for the WBB remains rejection of messages. This is done by "starving" one or several writers, i.e. by constant preferring other writers or by delivering old H_λ in message M0.

Variant 1: The WBB starve a writer.
The WBB rejects all messages of the writer. A writer can check if he is starved by looking whether another message was published on the board in the given time. If this is not the case, the writer can prove that the WBB was falsified. But this is difficult on high frequenting WBBs: If this is the case a WBB can deliberately starve a writer, without that the writer can prove this. Using a small value for ε or a distributed WBB can be corrective.

However, if ε is too small the WBB may constantly say the message of the writer was too long on the road and therefore reject it. It is very difficult to prove this wrong behavior of the WBB. Remedy provides again a distributed WBB as it is presented in the next chapter.

Variant 2: The WBB delivers permanently obsolete status hashes to a writer.
The WBB can answer requests from a particular writer with an old H_λ and the corresponding timestamp T_B. This can be for example, $H_{\lambda-1}$. The writer can never publish his message although the protocol is respected by the writer and the WBB from message M2 on. The writer must therefore check if the received H_λ is not older than ε (cf. Chapter 2.4.1). If the writer receives regularly outdated H_λ, the WBB could be spoofed. This suspicion can be proved with a reader who receives the correct history.

3. Distributed Web Bulletin Board

Basically, it can be distinguished whether voters vote directly via the Internet on an e-voting web application of their residential community, as in pilot tests in Switzerland, or whether they are using electronic voting machines as described in the work of Ben Adida (Adida, 2006). For the definition of the web bulletin board this does not matter, because dial machines as well as e-voting web servers behave to the WBB in the same way, i.e. the protocols defined in Chapter 2 will not change in this respect.

The following assumes an infrastructure, as shown in Figure 1. The individual voters perform their voting process on an e-voting web server. After the electoral process, the voice of the voter should be published encrypted on the WBB. The e-voting web server is used to identify of the voter and to receive her voice. It has its own key pair and occurs as a writer to the WBB.

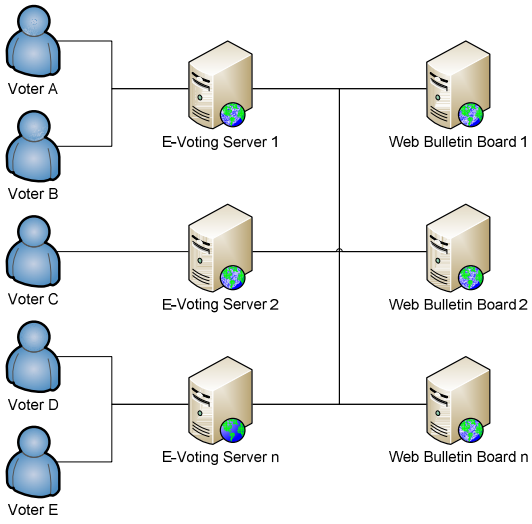

Figure 1: E-Voting over the Internet

3.1. Definitions

The definitions listed in Table 7 apply to chapter 3.

Wildcard / term	Definition
n	Total number of web bulletin boards.
k	Maximum number of web bulletin boards which could fail or be controlled by a single attacker.
Threshold-Set	A group of web bulletin boards that redundant publish a message m.
$l \geq k+1$	Size of a threshold set. Generally $l = k+1$.
p	Number of parties which operate a WBB. Such as political parties, the government or international election observation organizations.

Table 7: Definition of terms

3.2. From the single to the distributed Web Bulletin Board

A system for a single WBB was discussed in Chapter 2. In practice however it must be reckoned with disasters, attacks and manipulation of WBBs. Examples include server crashes, DOS attacks, loss or theft of the private key of the board or manipulation by conscious filtering or interruption of communication by the operator of the board (such as the estimated loser party)

To make a board resistant to such events, Heather and Lundin suggest a distributed Web Bulletin Board which is based on the replication of messages on several WBBs and a distributed signature by the collective of the associated WBB (Threshold-Cryptography). Distributed signatures are described in (Shoup, 2000). The authors sketch their idea as follows: A writer sends his message to a WBB of his choice. The board assembles a threshold set with other WBBs that publish the message and sign with the shared key. Finally, the message is replicated on the remaining WBBs. The writer receives a receipt that proves that l out of n WBBs have published the message $l \geq k + 1$. A synchronous and an asynchronous model are possible where the asynchronous model is better performing, but where in the end not all WBBs hold all messages.

3.3. The synchronous distributed Web Bulletin Board

Heather and Lundin loosely describe the synchronous model. The Table 8 below shows the protocol for the publishing, analog to the single model, as it would be in my view, based on a two phase commit protocol for redundant databases. The synchronous model uses a lock on l boards. For making it impossible that two messages can be published at the same time $2l > n$.

No.	**Time**	**Board l+1...n**	**Board 2...l**		**Board 1**		**Writer**
M0						<-	Read
M1	T_B				H_λ, T_B, $S_B(H_\lambda, T_B)$	->	
M2	T				Verification of the message. Locking of board 1	<-	m, T, W, $H(m, T, W, H_\lambda)$, $S_W(H)$
M2a	T_B‘		Verification of the message. Locking of board 2...l	<-	m, T, W, $H(m, T, W, H_\lambda)$, $S_W(H)$, T_B‘		
			Agree	->			
				<-	Commit		
			Publication on board 2...l		Publication on board 1		
M3a			Acknowledge, $S_{Bi}(S_W, T_B')$	->			
M2b		Publication on all other boards	<-	<-	m, T, W, $H(m, T, W, H_\lambda)$, $S_W(H)$, T_B‘		
					Assembling the Signature $S_{B1...l} = S_B$		
M3					$S_W(H)$, T_B', $S_B(S_W, T_B')$	->	Verification, Storing

Table 8: The protocol for the synchronous model of Heather and Lundin. S_B = signature with distributed key, S_{Bi} = partial signing with sub key of board i.

The only advantage compared with the single WBB is the redundancy of data storage. The receipt is signed by each board with its key. The board 1 (the one which was addressed by the writer) assembles the signature and exhibits the receipt to the writer, signed by the threshold key.

3.3.1. Discussion of the synchronous model

The proposed protocol of the synchronous model in Table 8 is applicable, with the following restrictions:

- Avoiding deadlocks requires careful implementation with additional communication complexity. Both, accidental and malicious forced deadlocks need to be handled. A single malicious board could exploit implementation bugs and endanger the whole system.
- The protocol is locking, which means that the synchronous version for a distributed WBB scale with the amount of requests. For e-voting applications one must expect a large amount of write requests at the same time arriving at the board. The load depends on the size of the electorate, but the synchronous model will not meet the requirements but for small communities.

3.4. The asynchronous distributed Web Bulletin Board

In the asynchronous model, l must not be greater than n anymore and thus multiple threshold sets may exist simultaneously. Therefore multiple messages can be written at the same time. The consequence is that not all WBBs have the same history and the same value for the state hash H_λ. To get the entire history for the accumulation of the votes in the asynchronous model, at least the messages of n-k+1 boards must be combined.

The description of the asynchronous model in the work of Heather and Lundin has in my opinion the following mistake: while in the synchronous model H_λ is the same value over all boards at the same time, this is no longer the case in the asynchronous model. The problem occurs in the message M2: Over what value H_λ should the writer build the hash H?

No.	**Time**	**Board l+1…n**	**Board 2…l**		**Board 1**		**Writer**
M0						<-	Read
M1	T_B				H_λ, T_B, $S_B(H_\lambda, T_B)$	->	
M2	T					<-	m, T, W, H(m, T, W, ???), S_W(H)

Table 9: Problem with the asynchronous model: H_λ can (and will) vary from board to board.

Since Heather and Lundin leave this question open, the conceivable variations are discussed in the following chapters. The relevant protocols are adapted from the protocols in Table 3 respectively Table 8. The validation routines of signatures, the temporal parameters T_B, T and T_B‘ and the reference to the obligation of the writer to preserve the receipts are always part of the protocol and no longer listed explicitly.

3.4.1. Variant 1: H_λ matches H_λ of board 1

As in the single WBB model, the writer does not cares about the structure of the distributed WBB but refers H_λ by an arbitrary board 1, calculates the hash on this value and sends the message to board 1. Board 1 locks l-1 other boards and sends the message to these boards. However, these other boards cannot sign the message because the value H_λ of the message most likely does not match the value H_λ in their histories. This variant is therefore not possible.

3.4.2. Variant 2: Board 1 acts as writer on board 2…l

In this option, the writer communicates again only with board 1 and uses its H_λ (see Table 10). Board 1 publishes the message on its history, assembles a set of l-1 other WBBs, snatches their $H_{\lambda 2\ldots l}$ and forwards the message to each individual board. The signature over the hash must be made by the board 1 because the Hash of the message no longer matches. The receipt to the writer can only be made by board 1 too, because the other boards can not verify the signature of the writer and thus cannot acknowledge the original message.

This leads to the following non-solvable problems:

- If board 1 fails the receipt of the writer is worthless.
- Since the message on board 2…l is not signed by the writer but only by board 1, board 1 can generate its own messages on behalf of any writer, which then are published on the other boards.

Therefore this variant is not possible either.

No.	Time	Board 2...l Index i: 2...l		Board 1		Writer
M0					<-	Read
M1	T_B			$H_{\lambda 1}$, T_B, $S_B(H_{\lambda 1}, T_B)$	->	
M2	T				<-	m, T, W, H(m, T, W, $H_{\lambda 1}$), $S_W(H)$
M0a	T_B‘		<-	Read		
M1a		$H_{\lambda i}$, T_B, $S_{Bi}(H_{\lambda i}, T_B)$	->			
M2a			<-	m, T, W, H(m, T, W, $H_{\lambda i}$), $S_{B1}(H)$, T_B‘		
M3a		$S_{B1}(H)$, T_B´, $S_{Bi}(S_{B1}, T_B$´)	->	Publication on board 1		
		Publication on board i				
M3				$S_W(H)$, T_B´, $S_{B1}(S_W, T_B$´)	->	

Table 10: Protocol of variant 2.

3.4.3. Variant 3: The writer publishes on several boards in parallel

We have seen that if the writer communicates with only one board a single point of failure arises. Table 11 shows the protocol if the writer himself publishes the message to k boards.

No.	Time	Board 1...l		Writer
$M0_1$			<-	Read 1
$M0_2$			<-	Read 2
			<-	...
$M0_l$			<-	Read l
$M1_1$	T_{B1}	$H_{\lambda 1}$, T_B, $S_{B1}(H_{\lambda 1}, T_{B1})$	->	
$M1_2$	T_{B2}	$H_{\lambda 2}$, T_B, $S_{B2}(H_{\lambda 2}, T_{B2})$	->	
		...	->	
$M1_l$	T_{Bl}	$H_{\lambda l}$, T_B, $S_{Bl}(H_{\lambda l}, T_{Bl})$	->	
M2	T		<-	m, T, W, H1(m, T, W, $H_{\lambda 1}$), H2(m, T, W, $H_{\lambda 2}$), ... Hl(m, T, W, $H_{\lambda l}$), S_W(H1..l)
M3	T_{Bi}‘	S_W(H1..l), T_{Bi}´, $S_{Bi}(S_W$(H1..l), T_{Bi}´)	->	
		Publication on board i		Assembling the signature

Table 11: Protocol of variant 3.

Each board is contacted by the writer and returns its last value H_λ. The writer creates a hash for each board H_i with the corresponding H_λ and sends the message with all hashes and a signature over all hashes to all l boards. These boards publish the message and create a threshold signature. The receipt is valid only if the writer receives the signatures of at least k + 1 boards.

The variant has the following disadvantages: firstly on highly frequented boards it is unlikely, that without locking the boards 1... l the values Hλ of k + 1 boards do not change between message M0i and message M2. Secondly a single board can although publish the message and issue the partial signature, even if the total signature fail because the value H_λ of another board has changed in the meantime and this board therefore rejects the message. This leads to inconsistencies. Similar to the synchronous model, a locking mechanism could provide remedy - with the well-known disadvantages:

- Loss of performance by locking.
- Malignant deadlocks by writer or proxy board.
- If a two-phase commit procedure is used as in the synchronous model the system still scales, since in contrast to the synchronous procedure only l out of n boards must be addressed with $l \geq k + 1$ and $2 k \leq n$, but communication overhead tends to be higher, as the writer needs to get H_λ not just from a single but from all l boards.

3.4.4. Variant 4: Writer publishes to multiple independent boards

Table 12 shows the protocol if the writer independently publishes his message on l boards.

No.	Time	Board 1...l		Writer
$M0_1$			<-	Read 1
$M1_1$	T_{B1}	$H_{\lambda 1}$, T_B, $S_{B1}(H_{\lambda 1}, T_B)$	->	
$M2_1$	T_1		<-	m, T, W, H1(m, T, W, $H_{\lambda 1}$), S_W(H1)
	$T_{B1'}$	Publication on board 1		
$M3_1$		S_W(H1), T_{B1}', $S_{B1}(S_W(H1), T_B')$	->	
$M0_2$			<-	Read 2
$M1_2$	T_{B2}	$H_{\lambda 2}$, T_B, $S_{B2}(H_{\lambda 2}, T_B)$	->	
$M2_2$	T_2		<-	m, T, W, H2(m, T, W, $H_{\lambda 2}$), S_W(H2)
	$T_{B2'}$	Publication on board 2		
$M3_2$		S_W(H2), T_{B2}', $S_{B2}(S_W(H2), T_B')$	->	
...				
$M0_l$			<-	Read l
$M1_l$	T_{Bl}	$H_{\lambda l}$, T_B, $S_{Bl}(H_{\lambda l}, T_B)$	->	
$M2_l$	T_l		<-	m, T, W, Hl(m, T, W, $H_{\lambda l}$), S_W(Hl)
	$T_{Bl'}$	Publication on board l		
$M3_l$		S_W(Hl), T_{Bl}', $S_{Bl}(S_W(Hl), T_B')$	->	

Table 12: Protocol of variant 4.

In this variant the writer again selects l boards, but publishes the message individually, i.e. he performs the protocol for the single WBB with each board. Individually means that communication with the individual boards is independent from each other. The individual boards can still be contacted simultaneously.

There is no threshold signature; the writer will get a receipt from each respective board individually. Here lies the problem: while in the variants with threshold signature the writer only has a valid receipt when at least k + 1 boards have signed ($l \geq k + 1$), in this variant, it is possible that the writer gets a valid receipt of less than k + 1 boards. The condition that a message is published at least to k + 1 boards is not anymore part of the system, but is only dependent on the writer (even if it is in his interest that the message is shared across at least k + 1 boards). If the writer publishes the message on too few boards causes that the message at the end may be disregarded and that the voice will not feed into the accumulation.

3.4.5. Discussion of variants 1 to 4

The proposed solutions of Heather and Lundin (synchronous and asynchronous) are based on a distributed signature. The general problem with this procedure is that a single WBB do not know whether the distributed signature at the end will emerge or whether only at most k of the involved WBBs publish and sign the message and thus the distributed signature fail, since the threshold is at k + 1. If this erratic behavior is not caught it results in inconsistencies in the individual histories. Entries will be published on individual WBBs although no valid receipt to the writer is issued.

The only way to caught this erratic behavior is a transaction oriented locking procedure, i.e. all involved WBBs must first agree to publish the message before it is effectively published. Otherwise no WBB may publish the message. However, on the one hand, this presupposes a "Master WBB" which coordinates the locking procedure. On the other this procedure opens the floodgates to a malicious WBB to paralyze the entire system.

Is a distributed signature waived, it depends on the writer, whether he fulfills the protocol properly or not. The system itself no longer ensures the fulfillment of the condition that each message is replicated to at least k + 1 boards.

A system uses a locking procedure without a master WBB, having a distributed signature and ensures that a message is replicated to at least k + 1 boards is defined in Chapter 3.5.

3.4.6. Independence of the boards

One thing was not yet examined in previous discussions, is the independence of the operators of the various boards. If all boards are operated by the same interest group (such as the government) and this stakeholder does not agree with the foreseeable result, it can destroy the boards and invalidating the vote. Therefore, it is best if all the boards are operated by different parties. But there is only a limited number of stakeholders say p where p is normally in the single-digit range. However since many writers should be able to publish messages in parallel, the number of boards should be as large as possible, so that many threshold sets of size l can be made at the same time. In order that different boards within the threshold sets can be used, the boards should be evenly distributed on the performing parties. Figure 2 shows an ideal distribution in a five-party system with 40 boards. As a threshold set should consist not only of boards from a single party we place k equal to the number of boards of the party with the most boards. Uniform distribution of the boards means k = n/p.

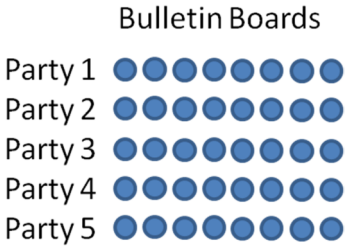

Figure 2: Ideal allocation of 40 WBB on five Parties (n=40, p=5, k=8).

We will assume, for the worst case, that when the writer randomly chooses his threshold set, he selects all boards from the same party. Therefore l, the number of boards per threshold set must be greater than k, the maximum number of boards, operated by a single party. This corresponds to the original definition of l, however, results that k is independent of n, i.e. the entire system no longer scales with n but only with p. This means that the maximum number of parallel write operations is limited upward by the number of parties regardless of how many boards are operated by the individual parties. Figure 3 shows this fact graphically.

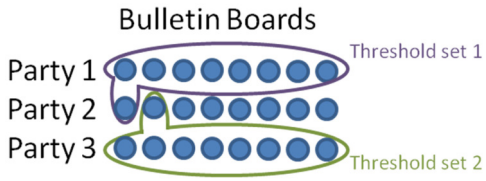

Figure 3: Worst, but possible combination of two threshold sets (n=24, p=3, k=8).

Result:

- All WBBs operated by the same party are not independent, because they could fail at the same time by sabotage by the operator. Therefore they provide together just as much reliability as a single independent WBB.
- To get the WBB scaling in the number of the boards used, k must remain constant, regardless of the scaling.

These problems can be addressed as follows: Since several WBBs of an operator provide just as much security as a single one and it only scales in the number of the parties but not in the number of used boards, each party runs only a single WBB: n = p. However, now a single WBB can consist of *multiple histories*. This causes that the writer can arrange his threshold set still randomly. Because each WBB is independent from others, it is always ensured that all WBB threshold sets are independent. The WBB itself can process multiple parallel write requests by locking only individual histories for write operations and set new write requests to other, not locked histories. Figure 4 shows the appropriate behavior.

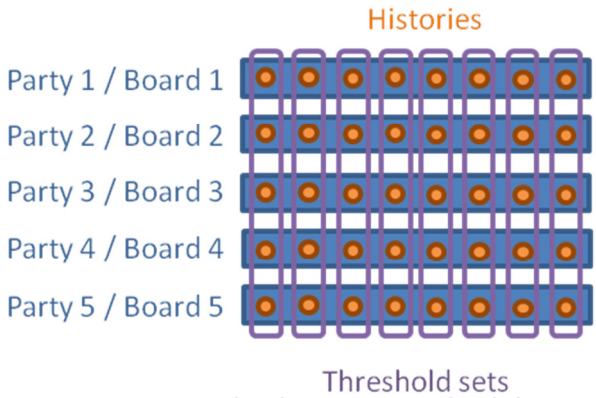

Figure 4: Five parties, each with an own WBB with eight histories and eight possible threshold sets.

Now the system scales in the number of the established histories: the more histories are used, the more parallel write operations are possible. The messages on the boards must be combined for the evaluation of the vote. First, all histories of a WBB are combined. There should be no duplicate messages because per board each message can only be written in a single history. Then the accumulated messages of at least n-k + 1 boards are combined.

The question arises, whether the number of the histories of the board must be predefined or whether boards can open on-the-fly new histories, if they are busy. The problem can be reduced to the consideration of the first entry in a new history: A malicious WBB should derive an advantage from opening a new history for a message rather than that it appends the message to an existing history. This usually doesn't matter because a new history has the same conditions as an already existing one. However, the case must be considered when a WBB opens a new history, publishes a message, issues the receipt for this message and then again destroys the history. Using the receipt, the writer can prove that the history has existed and the message was published. For this purpose the history must be extended with a global ID, which is included in the signature of the board.

It should still not be allowed to open arbitrarily many histories but only up to a priori defined maximum. This maximum is the expected percentage of voters who cast their vote at the same time at peak times. If the number of the histories is not limited, a WBB could open as many histories as it receives messages which would degenerate the WBB to a list. The disadvantage is that if fewer messages in a history are stored, fewer writers are involved in this history and less frequently the history is verified for correctness (building and comparing the hash chain). But verifying the hash chain is fundamentally important for safety. This maximum must be defined on the basis of experienced data. If this experience is missing I suggest estimating the expected maximum based on the number of incoming persons in a polling station at peak times (e.g. on voting day) in conventional votes.

3.5. Definition of a distributed Web Bulletin Board

Based on the definition of single Web Bulletin Board by Heather and Lundin and the knowledge gained in the last chapter a proposal for the implementation of a distributed Web Bulletin Board for e-voting applications is made now.

3.5.1. Infrastructure

For the definition an existing infrastructure as shown in Figure 1 is assumed. Both the voting Server (or voting machines depending on the e-voting technology used) and the Web bulletin boards were equipped with key pairs. The public keys of all agents are known to the other agents in the form of X.509 certificates. Further, each WBB has a sub key for the threshold signature, where the threshold lies at k + 1. Communication between the WBBs as well as communication from and to the e-voting servers must be guaranteed. The remaining infrastructure meets the requirements of the used e-voting technology.

3.5.2. Technical implementation

The Web Bulletin Board is implemented as a Web application as shown in Figure 5. The application consists of a database for persistent storage of the histories, a Web interface to respond to read requests and a Web service interface for write requests and exchange of messages between the WBBs. The individual histories are stored in a common database and processed for read queries as XML content. This database is ideally locally replicated. A SOAP Web service is available for distributing the requests to individual handlers. Each handler (Thread) looks then for a free history to process this write request.

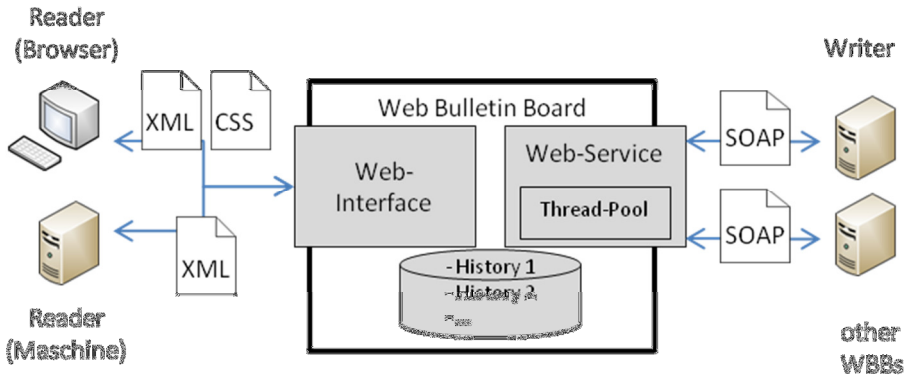

Figure 5: Sketch of a Web Bulletin Board with multiple histories and the appropriate interfaces.

Reading messages

Reading messages is analogous to the single Web Bulletin Board of Heather and Lundin. Each WBB returns its history as a HTTP response with XML content. Processing history data as XML has the advantage that the history can be both prepared for browser applications as a Web page using CSS files and read by machines. The presentation as a web page gives voter the possibility to verify on the browser if her voice on the WBB exists and therefore will be feed at the end of the vote in the overall result. For checking the correctness of the histories and totaling the votes a machine-readable form of the history is important. XML offers both and is therefore suitable. In practice the WBB may not be able to differentiate, whether a machine or a browser history will read it, therefore it would be incorrect to implement two different interfaces.

```
<?xml version="1.0" encoding="utf-8"?>
<Board Id="">                       <!--Identification of the board-->
 <History Id="">                    <!--Identification of the history-->
  <Header>
   <StatusHash Value=""/>           <!--Hλ -->
   <Timestamp Value=""/>            <!--TB-->
   <HistorySignature Value=""/> <!--Signature of the board over the values
                                    Hλ, TB and history id-->
  </Header>
  <Body>
   <Message Id="">                  <!--Global identification of the message-->
    <DistributedOn>                 <!--Identification of all of the boards where this
                                    message has been distributed to-->
     <BoardId value=""/>
     <BoardId value=""/>
     <BoardId value=""/>
    </DistributedOn>
    <Content>
                                    <!--E-Voting message body-->
    </Content>
    <CreationTime Value=""/>        <!--Time of creation by the writer-->
    <Writer Id=""/>                 <!--Identification of the writer-->
    <Hash Value=""/>                <!--Hash value over the values content, date of
                                    creation and writer-->
    <PublicationTime Value=""/> <!--Time of publication by the board-->
    <Signatures>
     <Writer Value=""/>             <!--Signature of the writer over the hash-->
     <Board Value=""/>              <!--Signature of the board over the writer's signature,
                                    date of publication and history id-->
    </Signatures>
   </Message>
   <Message>...</Message>
   ...
  </Body>
 </History>
 <History>...</History>
 ...
</Board>
```

Source code 1: XML draft for providing a WBB history.

Publishing messages

Writing messages to the distributed WBB takes place via a SOAP interface. If it is requested by the e-voting application, this communication-channel can also use SSL and be hardened by using client and server certificates. For the security of the WBB this is not necessary because the authentication is performed by the digital signature of the messages.

The protocol for the write operation is an extension of variant 4: the writer again performs the single protocol with l boards up to and including message M2. As usual $l \geq k + 1$, but here it is good if $l > k+1$: If up to a maximum of l-k+1 of the respective boards are not responding, the protocol must not be replayed again. Table 13 shows the corresponding protocol.

No.	Time	Board j, j:1...l, j≠i		Board i, i:1...l		Writer
$M0_i$					<-	Read i
$M1_i$	T_{Bi}			ID_H, $H_{\lambda i}$, T_B, $S_{Bi}(H_{\lambda i}, T_{Bi})$	->	
$M2_i$	T_i				<-	m, ID_M, T, W, ID_H, $H_i(m, ID_M, T, W, ID_H, H_{\lambda i})$, $S_W(H_i)$, <1,...,l>
	T_{Bi}'			Lock(Timeout)		
M4			<-	ID_M, $S_{Bi}(H(m, ID_M, T, W))$		
		Publication on all boards that received k signatures from other boards. Unlock of all boards.				
$M3_i$		$S_W(H_i)$, T_B', $S_{Bi}(S_W(H_i), T_B')$, $S_{Threshold}(H(m, ID_M, T, W))$			->	

Table 13: Protocol for posting messages on the distributed WBB.

After receiving the message M2 the respective board i locks and sends the message M4 to all other l-1 boards, which states that the board i will publish the message when it receives the message M4 from k other boards. A receipt is valid, if k + 1 of these l boards signed it using the private key of the board as well as the threshold signature key where the threshold lies at k + 1. In contrast to variant 4, now the boards agree before the publication, whether the message can be published on enough boards. Are at least k + 1 boards ready to publish the message, it is published and the partial signatures of the receipt are issued.

There is a problem to solve on the transport layer: A board may only publish if it has received the message 4 from k boards and sent successfully the message 4 to these k boards ("at least once" scheme). Figure 6 shows the needed communication channels for l = 3.

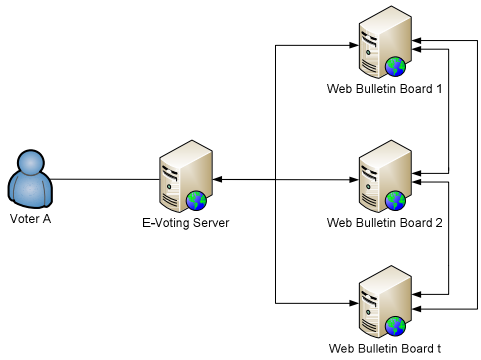

Figure 6: Communication channels needed during writing of messages on the distributed WBB.

One can see that with the agreement of the boards communication complexity is increasing significantly. Expecting for each message M4 two messages on the transport layer (transmission of the message and acknowledgement of receipt of the message), one can do following complexity calculation:

M0	Read	l
M1	H_λ	l
M2	Message from writer to t boards	l
M4	OK-Messages	l * (l-1) * 2
M3	Receipt to writer	l

Table 14: Communication complexity calculations for the variant 5

The communication complexity is therefore $O(l^2)$, $l \geq k+1$. This needs to be taken into account when estimating k and the maximum number of history per board.

However, the communication can be done concurrently. In contrast to the variant 3 the writer does not wait with sending message M2 until it has received all messages M1, but sends to each board i the message $M2_i$ immediately after receiving the message $M1_i$. This minimizes the possibility that another write operation takes place between message M0 and M2. Nevertheless if this is the case the board can be omitted as long as at least still k+1 other boards were successfully contacted.

The protocol must be expanded with the identification numbers $ID_{M,}$ ID_H and a list of <1,..., l>. The former identifies system-wide uniquely the message M2. This can be a combination of writer identification W and an incrementing number or a UUID as defined in RFC 4122 (Leach, et al., 2005). ID_H identifies the history of the WBB according to Chapter 3.4.6. <1,...,l> is a list of identification numbers the l boards, chosen by the writer to publish the message. This is necessary because each board must know with which other boards it needs to communicate to verify the message.

For the timeout at T_{Bi}' the following limit applies: The timeout must be selected larger than the sum of

- the time between T_{BFirst}' of the first locked and T_{BLast}' of the last locked board and
- two times the longest expected latency between board i and j, i≠j.

3.5.3. Discussion

The shown implementation has the following characteristics:

- The requirements identified by Heather and Lundin have been met.
- The system ensures that a message is either not or at least on k + 1 boards published.
- The writer receives only a valid receipt if the message is published at least on k + 1 boards.
- Faulty, dumb or malicious boards can be tolerated up to a number k.
- At k + 1 or more failed boards the correctness of the distributed WBB is no longer guaranteed.
- The system scale in proportion to the number of the existing histories.

4. Summary

Heather and Lundin have shown a way how to implement a bulletin board for e-voting applications. On this basis different protocol variants were discussed for a distributed implementation of such a system. Finally an own distributed Web Bulletin Board was defined, based on the findings, which meets the requirements of cryptographically voting systems of any scale.

Bibliography

Adida, Ben. 2006. Advances in Cryptographic Voting Systems. [Online] August 2006. http://groups.csail.mit.edu/cis/theses/adida-phd.pdf.

Heather, James und Lundin, David. 2008. *The Append-Only Web Bulletin Board.* Guildford, Surrey, UK : University of Surrey, 2008.

Leach, P., Mealling, M. und Salz, R. 2005. *A Universally Unique IDentifier (UUID) URN Namespace.* [Online] July 2005. http://tools.ietf.org/html/rfc4122.

Shoup, Victor. 2000. *Practical Threshold Signatures.* Rüschlikon, Switzerland : IBM Zürich Research Lab, 2000.