Proc. 20th Symposium on Reliable Distributed Systems (SRDS'01), New Orleans, USA, p. 194–197, October 28-31, 2001

Consensus with Written Messages Under Link Faults

Bettina Weiss, Ulrich Schmid
Technische Universität Wien
Department of Automation
Treitlstraße 1, A-1040 Vienna
Email: {bw, s}@auto.tuwien.ac.at

Abstract

This paper shows that deterministic consensus with written messages is possible in presence of link faults and compromised signatures. Relying upon a suitable perception-based hybrid fault model that provides different categories for both node and link faults, we prove that the authenticated Byzantine agreement algorithms OMHA and ZA of (Gong, Lincoln & Rushby, 1995) can be made resilient to f_ℓ link faults per node by adding $3f_\ell$ and $2f_\ell$ nodes, respectively. Both algorithms can also cope with compromised signatures if the affected nodes are considered as arbitrary faulty. Authenticated algorithms for consensus are therefore reasonably applicable even in wireless systems, where link faults and intrusions are the dominating source of errors.

Keywords: *Fault-tolerant distributed systems, fault models, link faults, consensus, Byzantine agreement, written messages, authentication.*

1. Motivation

Due to the well-known impossibility of deterministic consensus in presence of link faults [2, 5], most existing work on this problem considers node faults only. Still, in high-speed wireline and, in particular, wireless networks, the dominant cause of errors are message losses/corruptions. Those errors occur on the links and/or in the network interfaces, hence cannot reasonably be ascribed to nodes without quickly exhausting their number. Therefore, in order to reasonably use any distributed algorithm in a wireless network like our W2F fieldbus[1], a fault model incorporating both node and link faults is mandatory.

This problem was solved in [6], where we showed that the impossibility result of Gray [2] can be circumvented by limiting the degree of inconsistency caused by link faults in the broadcasts of a single sender resp. the receptions of a single receiver. Apart from theoretical results like a lower bound on the resilience, we proved that Byzantine agreement algorithms like the *Hybrid Oral Messages* algorithm (OMH) of [4] tolerate a large number of link faults provided that enough nodes are present and one round is added. An analysis of the resulting assumption coverage revealed that our approach allows to solve consensus even in wireless system architectures, where link fault probabilities up to 10^{-2} are quite common.

This paper is a digest of the results of [6] that relate to authenticated consensus in presence of link faults: In Section 2, we briefly review our perception-based hybrid fault model, Section 3 explains authentication issues. In Section 4, we provide definitions and results for the hybrid written messages algorithms OMHA and ZA of [1], Section 5 is devoted to the problem of broken signatures and Section 6 presents our conclusions.

2. Fault Model

Our analysis is based upon a novel perception-based hybrid fault model, which combines a "link fault extension" of the oral messages model of [3] and the hybrid node fault model of [4].

Definition 1 (System Model) *We consider a synchronous distributed system of n nodes interconnected by a fully connected point-to-point network, which has the following properties:*

- *If all nodes $s_i \in \mathcal{S}$ of a set of non-faulty sender nodes send a message containing V^{s_i} to some single receiver node r, at most f_ℓ^r of the values $V_r^{s_i}$ may differ from V^{s_i}.*
- *If a single non-faulty node s broadcasts (= successively sends) a message containing V^s to some set of non-faulty receiver nodes $\mathcal{R}$, at most f_ℓ^s of the values $V_{r_i}^s$ may differ from V^s.*

[1]This research is part of our W2F-project, which targets a wireline/wireless fieldbus based upon spread-spectrum (CDMA) communications, see *http://www.auto.tuwien.ac.at/Projects/W2F/* for details. W2F is supported by the Austrian START programme Y41-MAT.

- *The receiver of a message knows who sent it, and the absence of a message from sender s can be detected at any receiver r, which leads to $V_r^s = E$ for some distinguished value E.*
- *In any execution, there may be at most f_a, f_s and f_c arbitrary, symmetric, and manifest faulty nodes.*

Node faults are classified as follows:

- *A* manifest *faulty node s produces (detectably) missing messages or leads to a value that all non-faulty recipients r can detect as obviously bad; they all deliver $V_r^s = E$ here.*
- *A* symmetric *faulty node s sends the same wrong — but not usually detectably bad— value X^s to every receiver. All non-faulty receivers r deliver $V_r^s = X^s$ —the value* "actually sent"— *in this case.*
- *An* arbitrary (asymmetric) *faulty node may inconsistently send any value to any receiver.*

Note that node and link faults occur independently of each other. Faulty nodes do not change their fault mode, i.e., must be counted in f_a, f_s or f_c according to their most severe behavior. However, each message reception resp. broadcast has its own "budget" f_ℓ^r resp. f_ℓ^s of link faults, and the particular links actually hit by a fault are usually different for each message reception/broadcast.

3. Authentication

Consensus with written messages [3] assumes that no node can make undetectable modifications to messages and that the originator of a message is always known. It is generally agreed that electronic signatures can be used to achieve these goals, although there are some pitfalls [1]. We must also ensure that a node can detect whether a message belongs to the current execution run. The assumptions placed on the authentication scheme are:

(A1) A node cannot change the contents of a message.

(A2) A node cannot forge a signature.

(A3) A valid signature cannot be mistaken for an invalid one (i.e., the signature does not introduce new errors).

Generally, every node p uses its signature σ_p to sign a message v, thereby generating the signed message $\sigma_p(v)$. In the Byzantine agreement algorithms of Section 4, the value v of a dedicated transmitter must be consistently disseminated to all remaining nodes in the system. This is accomplished by forwarding messages via paths of distinct processors in every round, such that a value v that is sent from node p_1 along the chain of nodes $p_2 \dots p_k$ arrives as a message $M = \sigma_{p_k} \cdots \sigma_{p_1}(v)$. This allows a node to recognize several manifest faults upon message reception:

(M1) If a message arrives in round $m - k$, then it must either bear $k + 1$ signatures, the first of which is from the original transmitter, or it must contain the value E. All other messages are manifest faulty.

(M2) The message arrives on the link from node p_i but has not been signed by p_i last.

(M3) The message contains a signature at least twice (i.e., one node has signed the message twice).

(M4) Two messages bear the same signature chain and contain different values.

Of course, for (M1) all nodes must know who the initial transmitter is. The reaction of the node to these manifest faults depends on its own fault status. We assume the following:

• A non-faulty node recognizes (M1)-(M4) and discards the message received in (M1)-(M3), reporting E instead. In (M4), it may either forward the first message and discard the second one, or it may discard both messages and report E instead.

• A *manifest* faulty node produces a manifest fault on all receivers regardless of what it receives.

• A *symmetric* faulty node may ignore (M1)-(M3) and send the manifest faulty messages. If the signatures are secure, then it may also ignore (M4) and send both messages it has received. However, if we assume that signatures are broken, then we must assume that a symmetric faulty node recognizes (M4) and does not send two different messages along, cf. Lemma 1.

• An *arbitrary* faulty node ignores (M1)-(M4) and sends along whatever it likes. In particular, it may send a message it should have recognized as manifest faulty.

Using signatures has a beneficial effect upon both node and link faults, because faulty nodes and links cannot introduce new values into the system. More specifically, due to (M1)–(M4), faulty links can either produce a detectably faulty message or replace the original one by some other valid message sent by the same node. In case of an outgoing link from a not arbitrary faulty node, replacing the message has no effect since all valid messages are the same; in case of an arbitrary faulty node, the value sent is not important anyway. So authentication definitely has a positive effect upon the severity of link faults.

4. Algorithms and Analysis

The two algorithms OMHA and ZA [1] considered here have the same recursive structure. For space reasons, we will provide only an informal description; please refer to [6] for the exact definition. Note that the algorithms are solutions for the Byzantine agreement problem, where one dedicated transmitter disseminates its value to all receivers. For consensus, every node has its own initial value. Consensus can be achieved by executing the Byzantine agreement algorithm on every node and voting on the values delivered for the nodes.

In **ALG(0)**, the transmitter signs and sends its value to every receiver. Each receiver p delivers the value v_p received from the transmitter, or the value E if a missing or manifestly erroneous value was received.

In **ALG(m)**, $m > 0$, the transmitter signs and sends its value to every receiver. Upon reception, each receiver p acts as the transmitter in ALG($m-1$) to communicate this value w_p (the value $R(w_p)$ in the case of OMHA) to the other receivers (including itself). For each p and q, let w_p^q be the value receiver p delivers as the result of ALG($m-1$) initiated by receiver q, or else E if no w_p^q or a manifestly bad value was delivered. Each receiver p calculates the hybrid-majority value among all values w_p^q (and applies R^{-1} to that value in case of OMHA). The result v_p is delivered as the transmitter's value.

The *hybrid-majority* of a set $\mathcal{V}$ of values provides the majority of all non-E values in $\mathcal{V}$; if no such majority exists, some arbitrary, but functionally determined, value is returned.

The *wrapper function* $R(v)$ used by OMHA encodes a statement "I am reporting v" as a unique value. Reporting is undone by means of the inverse function $R^{-1}(v)$, which must guarantee $R^{-1}(R(v)) = v$.

To solve Byzantine agreement, an algorithm has to satisfy the following properties:

Agreement: If nodes p and q are non-faulty, then both deliver the same value $v_p = v_q$.

Validity: If node p is non-faulty, then p delivers the value

- the value sent by the transmitter, if the transmitter is non-faulty or symmetrically faulty,
- E, if the transmitter is manifest faulty,
- unspecified, if the transmitter is arbitrary faulty.

As argued in Section 3, faulty nodes cannot inject new values into the system, so the only values that do occur in OMHA are those originally sent by the transmitter, E, and various $R(E)$'s. In the hybrid fault model of [4], the fact that a faulty node can inject an $R(E)$ value is enough to make the performance of OMHA no better than that of OMH. But how does authentication affect link faults in OMHA? According to the previous section, link faults cannot introduce any new values if authentication is used either. However, the original version of OMHA does not sign messages in OMHA(0), and at this stage, a link fault could insert a bogus $R(E)$ value. Therefore, we must assume that messages are signed even in OMHA(0).

The algorithm ZA has been derived from the flawed algorithm Z of [7] and provides a much better fault-tolerance degree than OMHA. However, its correctness depends critically upon Assumptions (A1)–(A3) and upon the fact that the transmitter is known. The strength of the signed algorithm lies in the fact that the only values that can occur are the values sent by the initial transmitter and E. So if the transmitter is not arbitrary faulty, then every node can only receive the original value and E.

The analysis of the two algorithms reveals the following requirements for validity and agreement (we added the results of OMH for comparison). Please consult [6] for details and proofs.

Alg.	**# nodes**
OMH	$2f_\ell^s + f_\ell^r + f_\ell^{ra} + 2(f_a + f_s) + f_c + m + 1$
OMHA	$2f_\ell^s + f_\ell^r + 2(f_a + f_s) + f_c + m + 1$
ZA	$f_\ell^s + f_\ell^r + f_a + f_s + f_c + 2$

All algorithms need $m \geq min\{1, f_\ell^s\}$ to achieve validity and require $m \geq f_a + min\{1, f_\ell^s\}$ to achieve agreement.

Remarks:

1. In the formulas for OMH, $f_\ell^{ra} \leq f_\ell^r$ is the number of receive link faults that cause (unrecognized) value faults. Due to the signatures, the authenticated algorithms recognize such value link faults, effectively causing $f_\ell^{ra} = 0$.
2. It is apparent that ZA is much better than OMHA. In fact, ZA has optimal link fault resilience since it matches the lower bound established in [6, Thm. 5].
3. We mentioned that OMHA must sign messages even in OMHA(0) to circumvent the injection of bogus $R(E)$ values. Since ZA does not use the wrapper R, we do not require signatures for ZA(0) if $m \geq 1$, i.e., if the message has been signed at least once.

5. Broken Signatures

Let us now assume that the signatures of all arbitrary faulty nodes as well as the signatures of at most f_b not arbitrary faulty nodes are common knowledge. Knowing a signature allows an arbitrary faulty node to generate a message on behalf of the other node, although this does not imply that it is able to eventually generate a valid chain of signatures:

Lemma 1 (Signature Chains) *At the end of the execution run, there are no two messages $M = \sigma_{p_k} \ldots \sigma_{p_1}(v)$ and*

$M' = \sigma_{p_k} \ldots \sigma_{p_1}(v')$ with $v \neq v'$, if at least one signature σ_{p_i} in M, M' is from a not arbitrary faulty node p_i with an unbroken signature.

Proof: Let us assume that two such messages M and M' exist. If we consider any node p_i that has signed both messages, then this node must have received both $\sigma_{p_{i-1}} \ldots \sigma_{p_1}(v)$ and $\sigma_{p_{i-1}} \ldots \sigma_{p_1}(v')$. But according to (M4) in Section 3, every not arbitrary faulty node at most signs the first message, but not the second one. So if p_i has signed both messages, it must be arbitrary faulty. If the node has not signed the messages, then someone else must have done so on its behalf, so its signature must have been broken. □

Recalling the operation of the consensus algorithms of Section 4, it is apparent that in round 1 of the algorithm, that is, in OMHA(1) or ZA(1), every non-faulty node uses the hybrid-majority on an input set whose chain of signatures only differs in the last signature. The chains may have different lengths, though, but if a chain has less than $m+1$ signatures, then it must contain the value E. For all chains with length $m+1$, we can deduce from Lemma 1 that each node will act upon the same input set if there is at least one not arbitrary faulty node in the chain whose signature has not been broken. So we can solve the problem of broken signatures by treating nodes with compromised signatures like arbitrary faulty nodes.

Theorem 1 (ZA with Broken Signatures) *For any $m \geq f_a + f_b + min\{1, f_\ell^s\}$, algorithm ZA($m$) satisfies agreement and validity if there are strictly more than $f_\ell^s + f_\ell^r + f_a + f_b + f_s + f_c + 1$ participating nodes.*

Proof: For validity, it is easy to see that if the transmitter is not arbitrary faulty and does not have a broken signature, then the only values that a non-faulty node considers are the value v sent by the transmitter and E. Therefore, validity still holds.

For agreement, we use enough rounds to ensure that every message received in ZA(0) either contains E or has been signed by at least one not arbitrary faulty node with unbroken signature. Therefore, Lemma 1 guarantees that no bogus messages can occur, so the algorithm is the same as without broken signatures. □

Algorithm OMHA could be made tolerant to broken signatures in the same fashion as ZA. However, for OMHA, it is probably cheaper with respect to the required number of nodes to simply let it degrade to OMH.

6. Conclusions

We provided a digest of the analysis of two different authenticated Byzantine agreement algorithms under a novel hybrid perception-based fault model, which captures both node and link faults. For the *Authenticated Hybrid Oral Messages* algorithm OMHA(m) of [1], we showed that $n > 2f_\ell^s + f_\ell^r + 2(f_a + f_s) + f_c + m$ nodes are needed for tolerating at most f_ℓ^s broadcast and f_ℓ^r receive link faults per node and round, and $f_a \leq m-1$, f_s, f_c arbitrary, symmetric, and manifest node faults. A considerably better (in fact, optimal) resilience was established for the simple authenticated algorithm ZA(m) of [1], which needs only $n > f_\ell^s + f_\ell^r + f_a + f_s + f_c + 1$ nodes for coping with the same number of faults.

Both algorithms can withstand intrusions to some extent: In case of broken signatures, OMHA degrades to OMH and hence requires an additional f_ℓ^{ra} in the lower bound for n. For ZA, a node with a compromised signature must be considered as arbitrary faulty and therefore counted in f_a. As far as link faults are concerned, authentication serves to identify and tolerate link value faults: Any algorithm that requires $f_\ell^r + f_\ell^{ra}$ nodes to tolerate link faults will only require f_ℓ^r nodes in the authenticated version.

We can hence conclude that written messages consensus algorithms are definitely more powerful than their oral messages counterparts. They can reasonably be employed even in wireless system architectures, where link faults and intrusions are the dominating source of errors. However, signing a message is a time-consuming operation, so algorithms that require few messages would be preferable. Part of our further research will address this problem.

References

[1] L. Gong, P. Lincoln, and J. Rushby. Byzantine agreement with authentication: Observations and applications in tolerating hybrid and link faults. In *Proceedings Dependable Computing for Critical Applications-5*, pages 139–157, Champaign, IL, Sept. 1995.

[2] J. Gray. Notes on data base operating systems. In G. S. R. Bayer, R.M. Graham, editor, *Operating Systems: An Advanced Course*, volume 60 of *Lecture Notes in Computer Science*, chapter 3.F, page 465. Springer, New York, 1978.

[3] L. Lamport, R. Shostak, and M. Pease. The byzantine generals problem. *ACM Transactions on Programming Languages and Systems*, 4(3):382–401, July 1982.

[4] P. Lincoln and J. Rushby. A formally verified algorithm for interactive consistency under a hybrid fault model. In *Proceedings Fault Tolerant Computing Symposium 23*, pages 402–411, Toulouse, France, June 1993.

[5] N. Lynch. *Distributed Algorithms*. Morgan Kaufman, 1996.

[6] U. Schmid and B. Weiss. Consensus with oral/written messages: Link faults revisited. Technical Report 183/1-110, Department of Automation, TU Vienna, Feb. 2001.

[7] P. M. Thambidurai and Y. K. Park. Interactive consistency with multiple failure modes. In *Proceedings 7th Reliable Distributed Systems Symposium*, Oct. 1988.