Efficient Two-Dimensional Self-Stabilizing Byzantine Clock Synchronization in WALDEN

1st Shaolin Yu  
Tsinghua University  
Beijing, China  
ysl8088@163.com

2nd Jihong Zhu  
Tsinghua University  
Beijing, China  
jhzhu@tsinghua.edu.cn

3rd Jiali Yang  
Tsinghua University  
Beijing, China  
yangjiali-0411@163.com

Abstract—For tolerating Byzantine faults of both the terminal and communication components in self-stabilizing clock synchronization, the two-dimensional self-stabilizing Byzantine-fault-tolerant clock synchronization problem is investigated and solved. By utilizing the time-triggered (TT) stage provided in the underlying networks as TT communication windows, the approximate agreement, hopping procedure, and randomized grandmasters are integrated into the overall solution. It is shown that with partitioning the communication components into 3 arbitrarily connected subnetworks, efficient synchronization can be achieved with one such subnetwork and less than 1/3 terminal components being Byzantine. Meanwhile, the desired stabilization can be reached for the specific networks in one or several seconds with high probabilities. This helps in developing various distributed hard-real-time systems with stringent time, resources, and safety requirements.

Index Terms—self-stabilization, Byzantine faults, clock synchronization, communication faults, network connectivity

I. INTRODUCTION

Reliable synchronization is fundamental in constructing reliable distributed hard-real-time systems. For example, in constructing distributed integrated modular avionics (DIMA) systems under the popular time-triggered (TT) architecture [1][2], the designers should show sufficiently high assumption coverage [3] of the provided fault-tolerance schemes. In this background, to avoid common-mode failures [4] as far as possible, a reliable global synchronization scheme is the basis for developing high-layer fault-tolerant systems. In providing such synchronization schemes for safety-critical hard-real-time systems, self-stabilizing Byzantine-fault-tolerant clock synchronization (SS-BFT-CS) is a promising trend.

Here, the terminology SS-BFT-CS is at the intersection of three widely investigated concepts in the field of distributed systems. Firstly, clock synchronization (CS) is the basis for coordinating the distributed (and often redundant) components of the hard-real-time systems. For such a CS scheme to be reliable, the provided CS algorithms should tolerate various kinds of faults of the components. As it is not easy to show that the failure modes of the faulty components can always be restricted with sufficiently high probabilities, it is safe to assume that the faulty components can fail arbitrarily, i.e., to allow them being Byzantine. Under this assumption, the Byzantine-fault-tolerant clock synchronization (BFT-CS) problem is intensively studied in both the theoretical and the industrial realms. Further, in considering that the system might experience some unforeseen transient disturbances, it is also hard to show there cannot be 1/3 [5] or more Byzantine components in the system under all working conditions. In this context, self-stabilization can take a great role in bringing the system from arbitrary initial states to the desired stabilized states. Following this trend, the widely adopted TT solutions, including the Time-Triggered Protocol (TTP) [1] and the Time-Triggered Ethernet (TTEthernet) [2], are all built upon some practical SS-BFT-CS schemes.

However, although some industrial CS solutions are integrated with both Byzantine-fault-tolerance and self-stabilization, only Byzantine faults of the terminal components can be tolerated in the provided CS algorithms. Meanwhile, the failure-modes of some communication components are still graveley restricted. For example, in TTP, as the number of the central guardians employed in the redundant bus network is typically restricted, these guardians are not allowed to be Byzantine. In TTEthernet, as the network connectivity of typical switched Ethernet is still low, the TT-switches are not allowed to be Byzantine. In this situation, to show sufficiently high assumption coverage of the CS schemes, these core communication components should be designed, manufactured, and verified with great effort. This would restrict the applications of these CS schemes to a broader range.

In this paper, to provide high-reliable CS, we investigate the SS-BFT-CS problem in the presence of Byzantine faults of both the terminal components and the communication components. For a concrete study, we would extend our former work, the basic WALDEN network [6], to the advanced ones with multiple-Ethernet-interface terminal components. Typically, as the network connectivity of switched-Ethernet is not sufficiently high, it is often impossible to locally prevent the propagation of the Byzantine faults generated in the communication components. In preventing the propagation of such faults, we will partition the WALDEN switches (the WS nodes) into several isolated subnetworks, as is shown in Fig.1. Here, each of the isolated subnetworks of the WS nodes
can be viewed as in an independent switch plane. In each such switch plane, we only require that the WS nodes are connected with an arbitrary topology. For tolerating one or more such subnetworks being fail-arbitrarily, we require that each terminal component is connected to several such subnetworks. In the words of [6], such a terminal component is called an advanced end-system (AES). In the whole picture, the isolated subnetworks can also be interconnected for upper-layer applications, as long as the synchronization signals and messages transmitted in these interconnections (sent by faulty nodes) would never be received by the nonfaulty nodes.

We can see, this networking scheme can provide a good tradeoff between performance and fault-tolerance. For performance, firstly, as there can be several switches in each subnetwork, sufficient communication bandwidth and connections can be provided for high-layer communications. Secondly, by designing and optimizing TDMA schedules in each switch plane with arbitrary network topologies, the overall bandwidth resources can be better allocated and utilized. Meanwhile, for fault-tolerance, as the TT communication can be realized in the WALDEN switches, the faults of terminal nodes can be isolated with static TDMA schedules. Also, as the faulty synchronization signals generated in the faulty switches cannot propagate between different subnetworks, each switch plane can be viewed as a fault-containment region (FCR[7], or FTU[1]). With this, the redundant switch planes are orthogonal to the redundant terminal nodes in the view of fault-tolerance.

So, the main problem is to provide an efficient SS-BFT-CS solution for advanced WALDEN networks with the two-dimensional Byzantine resilience, in which \( f_0 \) Byzantine terminal nodes and \( f_1 \) Byzantine switch planes can be tolerated at the same time. For this, the rest of this paper is constructed as follows. The related work and basic definitions are respectively given in Section II and Section III. In Section IV, the synchronization strategies are presented intuitively. With this, the SS-BFT-CS algorithm is provided, analyzed, and discussed in Section V. Lastly, we conclude the paper in Section VI.

II. RELATED WORK

In the literature, several SS-BFT-CS solutions have been provided in both the deterministic and probabilistic approaches, but most of them can only tolerate Byzantine faults of the terminal nodes. In deterministic approaches, linear stabilization time solutions [8, 9] are provided by employing some self-stabilizing Byzantine agreement protocols as building blocks [10]. However, as is discussed in [11], these BA-based SS-BFT-CS solutions are often with poor performance in considering the required message and computation complexities. In [12], by iteratively constructing the higher-layer SS-BFT-CS algorithms with the lower-layer resynchronization algorithms and consensus routines as building blocks, it is shown that the BA-based SS-BFT-CS solutions can be almost as easy as consensus. However, the message complexity and network connectivity required in the consensus routines still gravely restrict the wide application of these solutions in real-world sparsely connected networks.

In probabilistic approaches, [13] first provides a randomized SS-BFT-CS solution with intuitive hopping procedures with expected hyper-exponential stabilization time. Later in [11], the expectation of the stabilization time is lowered to \( O(n) \), where \( n \) is the number of the nodes in the system. Another advantage of [11] is the very little requirement on computation resources. However, to apply [11], the communication network is required to be fully connected. Meanwhile, the frequency of the synchronization signals is high, since the solution is mainly for synchronizing some tiny components deployed in VLSI circuits within the range of several centimeters.

In the industrial realm, TTEthernet [2] provides standard and high-reliable SS-BFT-CS solutions. However, Byzantine faults of the TT-switches should be locally guarded and filtered with special hardware schemes like the monitor-pairs [2]. In [6], a self-stabilizing synchronization solution is provided by employing COTS Ethernet components as building blocks, but only Byzantine terminal nodes can be tolerated. Further, for tolerating Byzantine communication components, although we may provide some multi-path transmission schemes with the multiple switch planes, the stabilization time of SS-BFT-CS would be restricted by the number of the Byzantine terminal nodes. With our limited knowledge, no efficient SS-BFT-CS solution tolerating Byzantine faults of both the terminal and communication components exists with the stabilization time being independent of the number of the Byzantine terminal nodes.

III. SYSTEM MODEL AND THE PROBLEM

A. Basic assumptions

Following [2], the WALDEN network \( G(V,E) \) is comprised of three kinds of nodes: the master end-system (MES) nodes \( V_A \), the WALDEN switch (WS) nodes \( V_B \), and the client end-system (CES) nodes \( V_C \). In discussing the core SS-BFT-CS problem, the CES nodes are ignored. In considering fault-tolerance, the faulty nodes in \( V \) are denoted as \( F \), with which the nonfaulty nodes are denoted as \( Q = V \setminus F \). For convenience, we also denote \( Q_x = Q \cap V_x \) and \( F_x = F \cap V_x \) for \( x \in \{A, B\} \). To our goal, all nodes in \( F \) are allowed to fail arbitrarily. Moreover, in considering self-stabilization, every node in \( Q \) runs the given algorithms since some instant \( t_0 \) with an arbitrary initial state.

Since \( t_0 \), in running the synchronization algorithms given in this paper, we make the following assumptions. Firstly, every node \( i \in Q_A \) can send and receive standard Ethernet messages (the standard messages). Every node \( j \in Q_B \) can send, receive, and deliver standard messages. Besides, every node
Meanwhile, each advanced MES node is denoted as $H$ and $\rho$ received, and processed) within preempted, it can be propagated (including sent, delivered, and receiving) of the standard messages. For simplicity, the preempted standard messages would be discarded in the nonfaulty WS nodes. In each nonfaulty subnetwork (including the terminal nodes), if a standard message is not preempted, it can be propagated (including sent, delivered, received, and processed) within $d_{max}$ time.

Secondly, for approximately measure the time (here the time can be taken as the Newtonian time for simplicity), every node $i \in Q$ is equipped with a hardware clock $H_i$ that can count the periodically generated ticking events with a tick-counter. For accuracy, the ticking cycles of $H_i$ are always bounded in $[(1 - \rho)T_{Hi}(1 + \rho)T_{Hi}]$, where $T_{Hi}$ is the nominal ticking cycle, and $\rho$ is the maximal drift-rate of the hardware clocks. At any given instant $t$, the current value of the tick-counter in $H_i$ is denoted as $H_i(t)$. In real-world systems, we assume $H_i(t)$ can only take values in $[[\tau_{max}]]$, where $[x] = \{0, 1, \ldots, x - 1\}$ is the set of the first $x$ non-negative integers and $\tau_{max}$ is sufficiently large. In considering self-stabilization, $H_i(t_0)$ can be arbitrarily valued in $[[\tau_{max}]]$ in every $i \in Q$. Denoting $t_1$ and $t_2$ as two adjacent ticking instants of $H_i$ with $t_2 > t_1 > t_0$, we always have $H_i(t_2) = (H_i(t_1) + 1) \mod \tau_{max}$. As $H_i$ is often used as an adjustable timing source (for implementing various timers), we define the adjustable local clock $C_i$ in each node $i \in Q$. By maintaining a local variable $\text{offset}_i$ in node $i$, the value of $C_i$ at $t$ can be represented as $C_i(t) = (H_i(t) + \text{offset}_i(t)) \mod \tau_{max}$, with which the accuracy of $H_i$ is also shared in $C_i$. For convenience, as the clocks all take values in $[[\tau_{max}]]$, we denote $\tau_1 \oplus \tau_2 = (\tau_1 + \tau_2) \mod \tau_{max}$, $\tau_1 \ominus \tau_2 = (\tau_1 - \tau_2) \mod \tau_{max}$, and $d(\tau_1, \tau_2) = \min\{\tau_1 \ominus \tau_2, \tau_2 \ominus \tau_1\}$.

B. The advanced WALDEN network and the problem

As is shown in Fig. 1, we assume that the WALDEN network $G = (V_A \cup V_B, E_A \cup E_B)$ is composed of $n_0 = |V_A|$ MES nodes and $n_1$ switch subnetworks. The $s$th ($s \in S = \{1, 2, \ldots, n_1\}$) switch subnetwork can be represented as a connected graph $G_B^{(s)} = (V_B^{(s)}, E_B^{(s)})$ where $V_B^{(s)}$ and $E_B^{(s)}$ are respectively the WS nodes and edges on the $s$th switch plane. Then, every $G_B^{(s)}$ is connected to the MES nodes $V_A$ with the edges $E_A^{(s)}$. For clarity here, we have $\bigcup_{s=1}^{n_1} V_B^{(s)} = V_B$, $\bigcup_{s=1}^{n_1} E_B^{(s)} = E_B$, $\bigcup_{s=1}^{n_1} E_A^{(s)} = E_A$, $V_B^{(s)} \cap V_B^{(s)} = \emptyset$, $E_B^{(s)} \cap E_B^{(s)} = \emptyset$, and $E_A^{(s)} \cap E_A^{(s)} = \emptyset$ for all $s, s_1, s_2 \in S$. With this, each subnetwork $G(V_A \cup V_B^{(s)}, E_A^{(s)} \cup E_B^{(s)})$ can be viewed as a basic WALDEN network in the $s$th network plane.

Typically, as each switch subnetwork $G_B^{(s)}$ has a relatively low network connectivity, each $G_B^{(s)}$ is viewed as an FCR, i.e., $G_B^{(s)}$ (and the corresponding $s$th switch plane) is regarded as nonfaulty iff all nodes and edges in $G_B^{(s)}$ are nonfaulty. Meanwhile, each advanced MES node $i \in V_A$ is viewed as an FCR. To our goal, only the nodes in the nonfaulty FCRs are required to be synchronized. For convenience, we also use $V_P = S$ to denote the set of all switch planes and use $Q_P \subseteq V_P$ to denote the nonfaulty switch planes. And the set of the considered nonfaulty nodes is denoted as $Q_{AP} = Q_A \cup \bigcup_{s \in Q_P} V_B^{(s)}$. With this, we say the system $S$ is ($\epsilon_0, \varrho, \Delta$, synchronized in $[t_1, t_2]$ iff

$$d(C_i(t), C_j(t)) \leq \epsilon_0$$  \hspace{1cm} (1)

$$\left|\left(\bigcup_{i,j} C_i(t) \ominus C_j(t) - (t' - t)\right) \right| \leq \varrho(t' - t) + \epsilon_0$$  \hspace{1cm} (2)

hold for all $i, j \in Q_{AP}$ and all $t', t \in [t_1, t_2]$ with $0 < t' - t \leq \Delta$. For simplicity, we can set $\Delta$ as approximately the synchronization cycle and thus the core problem is to make $S$ being ($\epsilon_0, \varrho$)-synchronized in $[t_1 + \infty)$. Concretely, in the two dimensional SS-BFT-CS problem, given that all nodes in $Q$ are locally correct, $S$ should be ($\epsilon_0, \varrho$)-synchronized in $[t, +\infty)$ with some $t \leq t_c + \Delta_{stb}$, where the stabilization time $\Delta_{stb}$ is expected to be sufficiently small.

At the same time, to prevent $S$ from being ($\epsilon_0, \varrho$)-synchronized, the adversary knows all the provided algorithms and can arbitrarily choose $f_0$ MES nodes and an arbitrary number of WS nodes in $f_1$ switch planes being Byzantine. Also, the adversary can arbitrarily set the state of $S$ at $t_0$. Since $t_0$, all chosen Byzantine nodes are under the full control of the adversary who knows everything about the executions of $S$. Moreover, the adversary can arbitrarily choose the delays and clock drifts in their bounded ranges in the executions of $S$. Now to play the game with such a static adversary, we assume $n_0 > 3f_0$ and $n_1 > 2f_1$.

IV. Strategies

As is shown in Fig. 1 every advanced MES node in the advanced WALDEN network $G = (V_A \cup V_B, E_A \cup E_B)$ is connected to $n_1$ switch subnetworks to tolerate up-to $f_1$ faulty switch subnetworks. This is the main difference from the basic WALDEN network proposed in [6]. Now, in solving the two-dimensional SS-BFT-CS problem with $G$, although the advanced MES nodes now gain more resources to tolerate the faults generated in the WS nodes, the faulty MES nodes can gain more power than that of the nonfaulty MES nodes, as the adversary can know more about the nonfaulty nodes and can collusively control the delays, clock drifts, and the Byzantine nodes. In such a situation, we take the following strategies.

A. The decoupled synchronization strategies

Firstly, to restrict the power of faulty MES nodes, every node $i \in Q_A$ is allowed to send standard messages but not allowed to send SIG. Secondly, to be decoupled with the basic synchronization strategies and algorithms provided in [6], every advanced MES node would act as $n_1$ independent CES nodes that parallel run the basic self-stabilizing synchronization algorithm (referred to as the SSS algorithm) provided in [6]. Namely, for each switch plane $p \in V_P$, every node $i \in V_A$ can be viewed as a virtual CES node (denoted as $CES(i, p)$) in running the SSS algorithm. Thirdly, as each switch plane $p \in V_P$ is viewed as a single FCR, we can choose a specific WS node $s \in V_B^{(p)}$, denoted as $WS(p)$, to be the master-WS (MWS) node of $p$ and only allow the MWS node $s$ to generate new SIG signals in $p$. And when $p$
is nonfaulty, all nodes in $V_B^{(p)}$ can propagate the SIGs. With this, for every $p \in Q_P$, the basic synchronization round [6] can be periodically initiated in $p$ by the SIG generated in $\text{MWS}(p)$.

Now to synchronize the $C$ clocks of all nodes in $Q_{AP}$, we want to use the $C$ clocks of the MWS nodes $V_S = \{\text{MWS}(p) \mid p \in V_P\}$ as the master $C$ clocks to synchronize the $C$ clocks (referred to as the client $C$ clocks) of all the other nodes in $Q_{AP}$. Namely, denoting all MWS nodes in the nonfaulty switch planes as $Q_S = \{\text{MWS}(p) \mid p \in Q_P\}$, if the $C$ clocks of all nodes in $Q_S$ are synchronized, the $C$ clocks of all WS nodes in $Q_{AP}$ can be trivially synchronized as the client $C$ clocks. Also, the $C$ clock of every node $i \in Q_A$ can also be synchronized by taking the median of the $n_1$ client $C$ clocks of the parallel virtual CES nodes $\text{CES}(i, p)$ for all $p \in V_P$.

So, the core problem is to synchronize the $C$ clocks of the nodes in $Q_S$. For this, firstly, as the basic synchronization round (round for short) can be periodically initiated by the MWS node $s = \text{MWS}(p)$ in every nonfaulty switch plane $p$, we want to use the $T_T$ stage in every round of $p$ to record the $C$ clock of $\text{MWS}(p)$ in every node $i \in Q_A$ and then relay this information to other nodes in $Q_S$. As is introduced in [6], the $T_T$ stage can be used to provide statically scheduled $TT$ communication in each nonfaulty switch plane. Namely, by running the SSS algorithms in each plane, the ticks in the $T_T$ stage can be partitioned into $T_T$-slots with fixed $T_T$-schedules.

By utilizing the $T_T$ stages to exchange the $C$ clocks and other information, the upper-layer synchronization strategies can be well-decoupled with the basic ones provided in [6]. As is shown in Fig. 2 denoting the $T_T$-slot of sending the clock information in the MWS node as $T_T^{end}$, $s$ can send a $T_T$-message containing the clock information of $s$ to all nodes in $V_A$ during $T_T^{end}$ of $s$. Then, during the same round, every node $i \in Q_A$ can receive and record the $T_T$-message of $s$ during the $T_T$-slot $T_T^{end}$ of $i$. Thus, as $s$ can initiate a new round within a bounded time, some historical messages of every node in $Q_S$ can be recorded in every node $i \in Q_A$ with timestamps. So, after a fixed duration, when $s$ initiates a new round, every node $i \in Q_A$ can send an $n_1$ dimensional vector of the records of the MWS nodes to $s$ during the $T_T$-slot $T_T^{end}$ of $i$. Then, during the same round, $s$ can receive an $n_1 \times n_0$ matrix of the records during the $T_T$-slot $T_T^{rec}$ of $s$. With this, $s$ can compute a new value of its $C$ clock and broadcast it in the $T_T$-message sent during $T_T^{end}$. Then, $s$ can adjust its $C$ clock at the end of every $T_T$ stage or later.

**Fig. 2: A basic synchronization round in a switch plane.**

Denoting the scheduled beginning and end ticks of a $T_T$-slot $T_T^k$ as respectively begin($T_T^k$) and end($T_T^k$), the scheduled $T_T$-slots between begin($T_T^{rec}$) and end($T_T^{end}$) are referred to as the $T_T$-exchanging window. With the $T_T$-exchanging windows, every $s \in Q_S$ can distribute its newly computed $C$ clock value to $V_A$ and collect such values relayed by $V_A$ in an $n_1 \times n_0$ matrix $M^{(s)}$. Also, every $i \in Q_A$ can estimate and send the current $C$ clock values of every node in $V_S$ and then every $s$ can collect such values in another $n_1 \times n_0$ matrix $C^{(s)}$.

With this, the main problem is to compute the new value of $C_s$ in every $s \in Q_S$ with the received TT-messages in every $T_T$-exchanging window. For SS-BFT-CS, two subproblems should be considered. One is to establish the initial synchronization with the arbitrary state of the system. The other one is to maintain the synchronized state of the system.

B. The randomized grandmasters

To establish the initial synchronization with an arbitrary system state since $t_0$, firstly, as the rounds in different switch planes are initially asynchronous, the items in $M^{(s)}$ and $C^{(s)}$ may be outdated. Also, as the $C$ clocks of the MWS nodes may be symmetrically distributed in $[T_{max}]$ at $t_0$, some asymmetric operations should be taken to break this possible symmetry. On the one hand, in breaking the symmetry, we want to take the $C$ clock of some specific MWS node to coordinate the $C$ clocks of other nodes, just like the grandmaster employed in PTP [14]. On the other hand, as the adversary knows everything about the provided algorithms and the executions, this grandmaster node cannot be fixed in the algorithms. In this situation, one possible way is to elect a unique grandmaster. However, as the elected grandmaster might be in the faulty switch plane and thus be deliberately controlled by the adversary, the stabilization of the system cannot be reached in a deterministic way. Meanwhile, the temporally synchronized state of the system cannot be deterministically maintained if the elected grandmaster is controlled by the adversary. Also, the election of the grandmaster would be hard, if it is not impossible, in the network $G$ where the communication rounds in different switch planes are asynchronous and nearly one-half of these switch planes can fail arbitrarily.

Instead, the grandmaster is randomly chosen. Concretely, to be a grandmaster, every node $s \in Q_S$ would toss a biased coin $b^{(s)}_{\text{coin}} \in \{0, 1\}$ before $s$ computing its new $C$ clock value during every round. And if $b^{(s)}_{\text{coin}} = 1$, $s$ would regard itself as the grandmaster during the current and the next several rounds. Otherwise, if $b^{(s)}_{\text{coin}} = 0$ during the current and several previous rounds, $s$ would not regard itself as the grandmaster.

Obviously, in this way, the number of the grandmasters in $Q_S$ is random. Nevertheless, there is a probability that a unique grandmaster exists in $Q_S$ during a sufficiently long duration. For a specific example, if we partition the WS nodes in the communication system into 3 subsets and employ the nodes in every such subset to form a switch subnetwork in $G$, there would be $n_1 = 3$ MWS nodes, in which one MWS can be in a faulty switch subnetwork. In this case, only two MWS nodes need to be synchronized in establishing the initial synchronization. Namely, if only two nodes in $Q_S$ are synchronized, the required minimal synchrony of the system is established.
C. The filters and the guarded conditions

Meanwhile, to deterministically maintain the synchronized state of the system, when a node $s \in Q_S$ cannot observe any evidence of an unsynchronized state, $s$ should try to maintain the possibly synchronized state of the system. So, whenever $s$ finds that the collected records can be generated by $n_1 - f_1$ or more synchronized nodes in $Q_S$, $s$ would assume that the system is synchronized and try to participate in some synchronization-maintaining routine.

However, when $s$ observes that the matrix $C(s)$ might be generated in the synchronized system, as $f_1$ rows in $C(s)$ can be from the faulty switch planes, there might be only $n_1 - 2f_1$ nodes in $Q_S$ are synchronized. As we only require $n_1 > 2f_1$, there might be only one row in $C(s)$ that can be used as reference. In this situation, two different nodes $s,s' \in Q_S$ can get very different references from $C(s)$ and $C(s')$ in the presence of the $f_1$ Byzantine switch planes. Nevertheless, as there are $n_0 > 3f_0$ MES nodes, we can leverage the nonfaulty nodes in $Q_A$ to filter the rows of $C(s)$ for every $p \in V_P$.

Concretely, with the recorded $C$ clocks of the MWS nodes, every MES node $i$ can check the basic accuracy condition $a_{p,i}^{cur}\tau$ for every switch plane $p \in V_P$. Denoting the instants of receiving the current and the previous records from $p \in V_P$ in $i$ as $\tau_{cur}$ and $\tau_{pre}$, $a_{p,i}^{cur}$ can be computed as

$$d(m_{p,i}(\tau_{cur}), m_{p,i}(\tau_{pre}) \oplus T) \leq 2\epsilon_0 \land d(H_i(\tau_{cur}), H_i(\tau_{pre}) \oplus T) \leq (2\epsilon_0 + 2\rho T + d_{max})/(1 - \rho)^2$$ (3)

where $m_{p,i}(t)$ is the received value from $p$ in node $i$ at $t$.

Then, $i$ can maintain an accuracy-counter $a_{p,i}$ to count the rounds in which $a_{p,i}^{cur}$ is always true. To our aim, the value of $a_{p,i}$ can be upper-bounded by $a_0 = 3$. And once $a_{p,i}^{cur}$ is not true in the current round, $a_{p,i}$ would be reset to 0. With this, besides sending the estimated values of the $C$ clocks of the MWS nodes, every node $i \in Q_A$ is also scheduled to send $a_{p,i}$ during the TT-slot $T_{tag}\tau_{end}$ of $i$. Then, in every node $s \in Q_S$, by collecting these $a_{p,i}$ into the $n_1 \times n_0$ matrix $A(s)$, if the $p$th row of $A(s)$ contains $n_0 - f_0$ or more $a_0$, $p$ would pass the accuracy filter and would be put in $P_{acc}^{s}$. Otherwise, $p$ would be filtered out and would not be put in $P_{acc}$.

Besides, when the system is synchronized, as the recorded messages of every $p \in Q_P$ in all nonfaulty MES nodes would be the same value sent by $\mathsf{MWS}(p)$, there would be at least $n_0 - f_0$ items in the $p$th row of $M(s)$ with the same value. So we can define

$$P_{\text{maj}}^{s} = \{ p \in V_P \mid \exists V' \subseteq V_A : |V'| \geq n_0 - f_0 \land \forall i \in V': m_i^{s} = m_{p,i}^{s} \}$$ (4)

where $m_{p,i}^{s} = \text{med}(m_{p,1}^{s}, \ldots, m_{p,n_0}^{s})$ is the majority value in the $p$th row of $M(s)$ with $\text{med}$ being the median function and $m_{p,i}^{s}$ being the $p$th row $i$th column element of $M(s)$.

So, by collecting the row numbers in $P_{\text{maj}}^{s} = P_{acc}^{s} \cap P_{\text{maj}}^{s}$, only the $P_{\text{acc}}^{s}$ rows of $C(s)$ need to be considered in deciding if $C(s)$ might be generated in the synchronized system. With this, by configuring a sufficiently large $\epsilon_1 = O(\epsilon_0 + \rho T + \epsilon_{max})$, the stabilization condition $E_{stb}^{s}$ can be guarded in $s$ as

$$\exists P' \subseteq P_{\text{acc}}^{s}, V' \subseteq V_A : |P'| \geq n_1 - f_1 \land |V'| \geq n_0 - f_0 \land \forall p_1, p_2 \in P', i_1, i_2 \in V' : d(c_{p_1,i_1}^{s}, c_{p_2,i_2}^{s}) \leq \epsilon_1$$ (5)

with $c_{p,i}^{s}$ being the $p$th row $i$th column element of $C(s)$. However, as there can be $f_0$ faulty MES nodes, $E_{stb}$ might be inconsistently observed among the nodes in $Q_S$. So $s$ guards another condition $E_{\text{weak}}^{s}$ such that if $E_{\text{weak}}^{s}$ is true in some $s' \in Q_S$, $E_{\text{weak}}^{s}$ would be true in some desired synchronization procedures. For this, by configuring a sufficiently large $\epsilon_2 = O(\epsilon_1)$, $E_{\text{weak}}^{s}$ can be guarded as

$$\exists P' \subseteq V_P, V' \subseteq V_A : |P'| \geq n_1 - f_1 \land |V'| \geq n_0 - f_0 \land \exists E_{\text{weak}}^{s} \subseteq \{ s : |s| = \tau_{max} \} : \forall p \in P', i \in V' : d(c_{p,i}^{s}, c_{\text{weak}}^{s}) \leq \epsilon_2/2$$ (6)

where $E_{\text{weak}}^{s}$ is referred to as a weak reference.

D. Synchronization rules and fault-tolerant functions

Now, to integrate the strategies of establishing the initial synchronization state and maintaining the synchronized state of the system, we give the following synchronization rules.

In each round of a node $s \in Q_S$, firstly, if $s$ does not regard itself as the grandmaster, $s$ would try to be synchronized by some grandmaster. For this, as $s$ does not know which node is currently a grandmaster or if there is a grandmaster, $s$ would call a randomized fault-tolerant function (denoted as RFT) to compute the new $C$ clock value $\hat{C}_{\text{new}}^{s}$, providing that $\hat{C}_{\text{new}}^{s}$ is not true in $s$. For $n_1 = 3$, we can define

$$\text{RFT}(C^{s}, c_{s}^{pre}) = \text{rd}_{p_0}(\text{FTA}(C^{s}), \text{rd}(c_1, c_2, c_3, c_{s}^{pre}))$$ (7)

with $c_{p} = \text{med}(c_{p,1}^{s}, \ldots, c_{p,n_0}^{s})$ and $c_{s}^{pre}$ being some previous clock value of $s$. Here, the $\text{rd}_{p_0}(x_0, x_1)$ function randomly chooses $x_0$ with the probability $p_0$ (and thus chooses $x_1$ with the probability $1 - p_0$). The $\text{rd}(X)$ function randomly chooses an element of $X$ with the uniform probability. The FTA function is the deterministic fault-tolerant averaging function provided in [15], which is defined as

$$\text{FTA}(C^{s}) = \text{mean}(\text{select}_{f_0}(\text{reduce}_{f_0}(c_1, \ldots, c_{n_0})))$$ (8)

with $c_i = \text{med}(c_{i,1}^{s}, \ldots, c_{i,n_0}^{s})$ and the mean, select, and reduce functions being all from [15]. With this, if $E_{stb}^{s}$ is true, as $s$ should also try to maintain the synchronized state of the system, $s$ would directly call the FTA function (instead of the RFT function) to compute $\hat{C}_{\text{new}}^{s}$. In both cases, $s$ would send the newly computed $\hat{C}_{\text{new}}^{s}$ during the TT-slot $T_{C}\tau_{end}$ and use $\hat{C}_{\text{new}}^{s}$ to adjust $C_s$ at the end of the current round.

Otherwise, if $s \in Q_S$ regards itself as the grandmaster, $s$ would try to remain its $C$ clock being unadjusted and expect that all other nodes in $Q_S$ would choose $s$ as the grandmaster. However, as every $s' \in Q_S \setminus \{s\}$ should check the $E_{stb}$ condition to decide which function should be called in computing $\hat{C}_{\text{new}}^{s}$, the adversary can prevent $s'$ from calling the RFT function by making $E_{stb}^{s}$ being always true in $s'$. Also,
when \( s \) and \( s' \) are synchronized or coarsely synchronized, \( s \) should participate in the desired approximate agreement with a high probability. So, when \( b_{\text{coin}}^{(s)} = 0 \) or \( E_{\text{sta}}^{(s)} \) is true, \( s \) would also call the FTA function. Otherwise, when \( E_{\text{weak}}^{(s)} \) is true, \( s \) would use a weak reference \( c_{\text{weak}}^{(s)} \) as \( c_{\text{new}}^{(s)} \).

E. The desired self-stabilization procedures

To put it together, we expect that some desired synchronization procedures (inspired by the hopping procedures presented in [13]) would be performed with sufficiently high probabilities. In Fig. 3, we present two such procedures for the specific case \( n_1 = 3 \) and \( f_1 = 1 \).

**Fig. 3:** The desired synchronization procedures.

By referring to the corresponding time interval of the TT-exchanging window as the exchanging period, we expect that the exchanging periods of all nodes in \( Q_S \) can be eventually aligned in time. For this, firstly, we expect that a node \( s \in Q_S \) can regard itself as the grandmaster during some exchanging periods. For example, in the leftmost exchanging period of \( s \) shown in Fig. 3a, we expect that the currently tossed coin \( b_{\text{coin}}^{(s)} \) would show 1 at some instant \( t_1 \). Meanwhile, we expect that every node \( s' \in Q_S \setminus \{ s \} \) would not be the grandmaster during the first exchanging period of \( s' \) since \( t_1 \) (with the begin(TT\text{rece}) of \( s \)) would be no earlier than \( t_1 \). With this, as is shown in Fig. 3a, if \( E_{\text{sta}}^{(s')} \) is not true during this exchanging period of \( s' \), we expect that \( s' \) can be synchronized by \( s \) with calling the RFT function in \( s' \). Otherwise, if \( E_{\text{sta}}^{(s')} \) is true, as \( s' \) would call the FTA function, we require that \( s \) can find \( E_{\text{weak}}^{(s')} \) being true and thus can be coarsely synchronized by \( s' \), as shown in Fig. 3b. A variant of the procedure shown in Fig. 3b is that when \( s \) finds \( E_{\text{weak}}^{(s')} \) being true and thus adjusts \( C_s \) with calling the FTA function, the adversary can make \( E_{\text{sta}}^{(s')} \) being false in \( s' \). In this situation, for the specific case \( n_1 = 3 \), we expect that \( s' \) chooses a value near that of \( s \). For the general case \( n_1 > 2f_1 \), the RFT function can also be extended to allow \( c_{\text{new}}^{(s')} \) and \( c_{\text{new}}^{(s)} \) to be sufficiently near.

Then, in all desired procedures, after the first exchanging period of \( s' \) since \( t_1 \), we expect \( C_s \) and \( C_s' \) to be at least coarsely synchronized. With this, during the following several exchanging periods, we expect that \( s \) and \( s' \) would all use the result of the FTA function, and thus the synchronous approximate agreement [15] would be simulated in at least \( n_1 - f_1 \) MWS nodes. So we expect that every node in \( Q_S \) would find \( E_{\text{sta}} \) being always true at the end of the desired procedure, and thus the system would be stabilized since then.

V. Algorithm and Analysis

A. The SS-BFT-CS algorithm

With these strategies, we provide the algorithm SSBCS in Fig. 4 in solving the two-dimensional SS-BFT-CS problem.

**Fig. 4:** Algorithm SSBCS.
switch plane, they are ignored in the core algorithm. With the static TT-schedules, we can configure the time parameters as $\delta_{00} = end(T_{C,rev}) - begin(T_{C, recv})$, $\delta_{01} = end(T_{C, rev}) - begin(T_{C,recv})$, $\delta_{10} = end(T_{C,recv}) - begin(T_{C,rev})$, and $\delta_{11} = end(T_{C,recv}) - end(T_{C,recv})$. Also, the nominal SIG cycle $T$ of an MWS node can be configured as the maximal round cycle $T_0$ (which is bounded by the time parameter $\tau_b^{ES}$ in (6)) plus the bounded error $\epsilon_2$. For randomization, the coin tossed in executing line 13 should get the head up with the fixed probability $q_0$. And the parameter $g_0$ can be configured as $g_0 + k_0$ with $k_0 = \lfloor \log_{22}(2q_0 + 8p(1 + \rho)T)/\epsilon_0 \rfloor$, $\epsilon_0 = [(n_0 - 2f_0 - 1)/f_0] + 1$ (see [15]). For simplicity, here we assume the lines in the algorithm are atomically executed, and the nonfaulty MES and MWS nodes can adjust their $C$ clocks at the end of the TT-slot $T_{C,recv}$. In practice, the $C$ clocks can also be adjusted at some other scheduled ticks after $end(T_{C,rev})$ with extra static and dynamic compensations.

Practically, although the RFT function can be generally extended, the SSBES algorithm is best for the specific case $n_l = 3$ and $f_l = 1$, since $n_1$ should be sufficiently small in considering fast stabilization. For the larger $n_l$ and $f_l$, multilayer solutions can be built by employing the 3-switch-plane solution as a basic building block like that in [12]. As is limited here, we only provide the basic 3-switch-plane solution.

**B. Analysis**

Initially, we investigate the desired resynchronization point.

**Definition 1:** $t$ is a resynchronization point iff line 13 is executed in some $s \in Q_S$ at $t$ with $b_{col}(s) = 1$ and there is some $s' \in Q_S \setminus \{s\}$ with grand_fi = 0 during the first exchanging period since $t$.

**Lemma 1:** During any time interval $[t_1, t_1 + T_{max}]$ with $t_1 \geq t_e$ and $T_{max} = 2T(1 + \rho)$, with at least a probability $2q_0(1 - q_0)^g_0$ that there is a resynchronization point $t \in [t_1, t_1 + T_{max}]$.

**Proof:** (Sketch) Firstly, for every node $s \in Q_S$, as the condition checked in line 8 cannot always be false in every $T + T_0$ ticks, at least one $SIG$ would be sent in $s$ in every $T + T_0$ ticks. So line 13 would be executed during every time interval $[t_1, t_1 + (1 + \rho)(T + T_0)]$. By tossing the coin, with at least a probability $q_0$ that $b_{col}(s) = 1$ when line 13 is executed at some $t \in [t_1, t_1 + T_{max}]$. And for $t$ being a resynchronization point, it only requires that some $s \in Q_S \setminus \{s\}$ would not get $b_{col}(s') = 1$ during the first exchanging period since $t$ and the previous $g_0 - 1$ exchanging periods of $s'$. So as $|Q_S| \geq 2$, the overall probability is no less than $2q_0(1 - q_0)^g_0$.

With this, we show the following property of $S$. As is limited here, the proof is sketched with relaxed parameter values. A full discussion of the exact bounds of these values is out of the range of the paper.

**Theorem 1:** For any $t_l \geq t_e$, with a probability $q_1 \geq g_0(1 - q_0)^2g_0\rho_0^p(1 - p_0)/2$ that $S$ would be $(\epsilon_0, \rho)$-synchronized since some $t_2 \in [t_1, t_1 + (g_0 + 1)T_{max}]$ with $\epsilon_0 = 3(1 + \rho)\delta_{max}$.

**Proof:** (Sketch) Assume $t_1$ is a resynchronization point. As $b_{col}(t_1) = 1$, $s$ would remain its $C$ clock being unadjusted unless $E^{(s)}_{weak}$ is satisfied. If $E^{(s)}_{weak}$ is satisfied, as $n_1 = 3$, only two cases need to be considered. In the first case, $s$ is in $P'$ of (6) and thus the adjustment of $C_s$ is no more than $\epsilon_2$. In the second case, $s$ is not in $P'$ and thus all nodes in $Q_S \setminus \{s\}$ are in $P'$. In this case, $C_s$ would be adjusted to a value with no more than $\epsilon_2$ difference to that of all other nodes in $Q_S$. Now for another node $s' \in Q_S \setminus \{s\}$, as $E^{(s)}_{weak}$ is satisfied in $s$, there are also two cases. In the first case, $E^{(s')}_{weak}$ is true and thus the difference between $c^{(s)}_{weak}$ and $c^{(s')}_{new}$ would be no more than $\epsilon_2 + 2\rho T_{max}$ when $s'$ adjusts $C_{s'}$ at $t_2$ during its first exchanging period since $t_1$. With this we have $d(C_s(t_2), C_{s'}(t_2)) < 2\rho + 4\rho T_{max}$. In the second case, $E^{(s')}_{weak}$ is not true and thus RFT would be called in $s'$. So if $s$ is not in $P'$, with a probability $(1 - p_0)/4$ $s'$ would choose a value in $(c^{(s'), c^{(s')}_{pre}})$ with no more than $\epsilon_2 + 2\rho T_{max}$ difference to $c^{(s')}_{new}$ and thus $d(C_s(t_2), C_{s'}(t_2)) < 2\rho + 4\rho T_{max}$ holds. And if $s$ is in $P'$, there is also a probability $(1 - p_0)/4$ that $s'$ would choose the value of $s$ and thus $d(C_s(t_2), C_{s'}(t_2)) < 2\rho + 4\rho T_{max}$ would be true.

Otherwise, if $E^{(s')}_{weak}$ is not satisfied in $s$, as the instant corresponding to $begin(T_{recv})$ of $s'$ is no earlier than $t_1$, both $E^{(s')}_{weak}$ and $E^{(s')}_{weak}$ would be false. So $s$ would remain its $C$ clock being unadjusted during the current exchanging period, and there is also a probability $(1 - p_0)/4$ that $s'$ would choose the value of $s$ in executing the RFT function.

So, at the end of the first exchanging period of $s'$ since $t$, there is at least a probability $(1 - p_0)/4$ that $d(C_s(t_2), C_{s'}(t_2)) < 2\rho + 4\rho T_{max}$. Then, during the following $g_0 - a_0$ rounds, with a probability $((1 - q_0)p_0)^{g_0 - a_0}$ the approximate agreement would be simulated in $s$ and $s'$. So $d(C_s(t), C_{s'}(t)) < \epsilon_0$ holds at the end of the $g_0 - a_0$ rounds. Similarly, with another probability $(1 - q_0)p_0)^{a_0}$ that $E^{(s')}_{weak}$ would be true in every node $s'' \in Q_S$ at the end of the $g_0$ rounds and thus the system is $(\epsilon_0, \rho)$-synchronized since then. And with Lemma 1, the overall probability of this is no less than $2q_0(1 - q_0)^g_0((1 - q_0)p_0)^{a_0}(1 - p_0)/4$.

**C. Discussion**

With the proof of Theorem 1, the expectation of $\Delta_{stb}^{exp}$ (denoted as $\Delta_{stb}^{(exp)}$) can be approximately $T_{max}/q_1 + g_0 T_{max}$. To maximize $q_1$, we can set $g_0 = 1/(2g_0 + 1)$ and $p_0 = 1 - 1/(2g_0 + 1)$ and get $q_1 > 1/(2(2q_0 + 1))/2g_0 + 1) + 1$, we can increase $n_0$. Namely, $\Delta_{stb}^{(exp)}$ can be lowered by deploying more MES nodes in the network. And when $n_0$ is sufficiently large, we would get $g_0 = a_0 + 1$, which would make $\Delta_{stb}^{(exp)}$ being independent of $g_0$. Then, as $T_{max} \approx 2T$, we can set $a_0 \approx 3$. With these, we can approximately get $\Delta_{stb}^{(exp)} \leq (2/q_1 + 4)/T$ with $q_1 > 1/(90e^2)$. As the nominal SIG cycle $T$ can be at the order of milliseconds or even sub-millisecond, $\Delta_{stb}^{(exp)}$ can be at the order of a second or better. Also, as $(1 - q_1)^{g_0^2} < 10^{-7}$ with $q_1 > 1/(90e^2)$, the probability that $S$ is not $(\epsilon_0, \rho)$-synchronized before $(10^3 + 4)T_{max}$ is less than $3 \times 10^{-7}$. So by taking $T_{max}$ as approximately 1 millisecond, $S$ can
be synchronized in less than 10 seconds with a very high probability.

In comparing with existing solutions, firstly, the solution provided here is for tolerating Byzantine faults in both the terminal components and the communication components. So in considering Byzantine-resilience, this solution is inherently better than TTP, TTEthernet, and the basic WALDEN solution where the failure-modes of the communication components are strictly restricted. Also, as only \( n_1 > 2f_1 \) is required, this solution is better than that of synchronizing the switches and then distributing their clocks to the terminals, since that would require \( n_1 > 3f_1 \). In considering the stabilization time, as we do not rely on any BA routine nor fully connected network, the stabilization time is independent of \( f_0 \), and thus the solution provides better scalability of \( n_0 \) than that of the classical ones \([9, 11, 12]\). Also, as the communication and computation required in the SSBCS algorithm can be optimized by static TDMA schedules, the required system resources can be minimized in each switch plane. With this, the SS-BFT-CS solution provided in the advanced WALDEN networks can be more efficient than the traditional SS-BFT-CS solutions where the communication might be asynchronous when the system is not synchronized.

VI. Conclusion

In this paper, we have investigated the two-dimensional SS-BFT-CS problem and solved this problem with the WALDEN networks. The WALDEN networks are specially investigated as they are composed of common COTS Ethernet components. Now with the provided SS-BFT-CS solution, more reliable distributed CS systems can be built with common COTS Ethernet components. This would help in developing distributed hard-real-time systems with stringent time, resources, and safety requirements.

Firstly, to make the SS-BFT-CS solution decoupled with the underlying networks, we utilize the TT stages provided by the underlying networks to establish upper-layer TT-exchanging windows. As we do not rely on any concrete realization of the TT-exchanging windows, the multi-plane SS-BFT-CS solution is independent of the realization of the underlying basic CS solution. Then, to align the TT stages in all nonfaulty switch planes, we have developed a randomized synchronization scheme to synchronize the \( C \) clocks of the nonfaulty MES and MWS nodes. We have shown that the provided SS-BFT-CS algorithm can reach stabilization in an expected small duration independent of \( f_0 \) with \( n_1 = 3, f_1 = 1 \), and a sufficiently large \( n_0 \). In applying to WALDEN networks, the system can be synchronized in one or several seconds with high probabilities.

Despite the merits, the provided solutions can be improved in several ways. Firstly, to tolerate more faulty switch planes, multi-layer solutions can be further explored. Secondly, for deterministic stabilization, the basic method can be further integrated with BA-based SS-BFT-CS solutions. Also, for better synchronization precision, the basic method can be integrated with high precision protocols like PTP [14].

References

[1] H. Kopetz and G. Grunsteidl, “TTP - a time-triggered protocol for fault-tolerant real-time systems,” in FTCS-23 The Twenty-Third International Symposium on Fault-Tolerant Computing, 1993, Conference Proceedings, pp. 524–533.
[2] AS6802, “Time-triggered ethernet,” SAE International, 2016.
[3] D. Powell, “Failure mode assumptions and assumption coverage,” in The Second-International Symposium on Fault-Tolerant Computing, 1992, pp. 386–395.
[4] J. Lala and R. Harper, “Architectural principles for safety-critical real-time applications,” Proceedings of the IEEE, vol. 82, no. 1, pp. 25–40, 1994.
[5] D. Dolev, J. Y. Halpern, and H. R. Strong, “On the possibility and impossibility of achieving clock synchronization,” Journal of Computer and System Sciences, vol. 32, no. 2, pp. 230–250, 1986.
[6] S. Yu, J. Zhu, and J. Yang, “Reaching self-stabilising distributed synchronisation with cots ethernet components: the walden approach,” Real-Time Systems, vol. 57, no. 4, pp. 347–386, 2021.
[7] H. Kopetz, “The fault hypothesis for the time-triggered architecture,” Building the Information Society, vol. 156, pp. 221–233, 2004.
[8] A. Daliot, D. Dolev, and H. Parnas, “Linear time byzantine self-stabilizing clock synchronization,” 2006, also updated in http://arxiv.org/abs/cs.DC/0608096.
[9] D. Dolev and E. N. Hoch, “Byzantine self-stabilizing pulse in a bounded-delay model,” in Stabilization, Safety, and Security of Distributed Systems, T. Masuzawa and S. Tixeuil, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2007, pp. 234–252.
[10] A. Daliot and D. Dolev, “Self-stabilizing byzantine agreement,” in Proceedings of the Twenty-Fifth Annual ACM Symposium on Principles of Distributed Computing, ser. PODC ’06. New York, NY, USA: Association for Computing Machinery, 2006, p. 143–152.
[11] D. Dolev, M. Fugger, U. Schmid, and C. Lenzien, “Fault-tolerant algorithms for tick-generation in asynchronous logic: Robust pulse generation,” Journal of the Acm, vol. 61, no. 5, 2014.
[12] C. Lenzien and J. Rybicki, “Self-stabilising byzantine clock synchronisation is almost as easy as consensus,” Journal of the Acm, vol. 66, no. 5, 2019.
[13] S. Dolev and J. L. Welch, “Self-stabilizing clock synchronization in the presence of byzantine faults,” Journal of the Acm, vol. 51, no. 5, p. 780–799, Sep. 2004.
[14] IEEE1588, “Standard for a precision clock synchronization protocol for networked measurement and control systems,” IEEE Standard 1588-2008, July 2008.
[15] D. Dolev, N. A. Lynch, S. S. Pinter, E. W. Stark, and W. E. Weihl, “Reaching approximate agreement in the presence of faults,” Journal of the Acm, vol. 33, no. 3, pp. 499–516, 1986.