Bchain Byzantine Replication with high throughput and embedded reconfiguration

Sisi Duan, Hein Meling, Sean Peisert, and Haibin Zhang

Presented by Ruben Romero
BChain Protocols

• Bchain3:
  – 3f+1 replicas
  – Sub protocols: (1) Chaining, (2) Re-chaining, (3) View Change, (4) Checkpoint and (5) Reconfiguration.

• BChain5:
  - 5f+1 replicas
  - No Reconfiguration protocol
BChain

• Safety:
  – It is hold in any asynchronous environment where messages may be delayed, dropped, or deliver out of order

• Liveness
  – Assure assuming that synchrony holds after some unknown stabilization time
For each replica $p_j$ we define $P(j)$, set predecessor, and $S(p_j)$, set successor, for replicas in the set $A$ as:

- $P(p_j)$: if $j < f+1$ then $P(p_j) = \{p_h, p_1, \ldots, p_{j-1}\}$, else $P(j) = \{p_{j-f-1}, \ldots, p_{j-1}\}$
- $S(p_j)$: if $2f+1 < f < f+1$ then $S(p_j) = \{p_{j+1}, \ldots, p_{2f+1}\}$, else $S(j) = \{p_{j+1}, \ldots, p_{j+f+2}\}$
Chaining Protocol

- Orders clients requests
Chaining Protocol: Step 0

- Client c sends a request <Request, o, T, c> to the head p_h.

-o: state machine operation
-T: Timespan
-c: client id
Chaining Protocol: Step 1

-Head receives <Request, o, T, c> from C
-Head sends <Chain, v, ch, N, m, c, H, R, Λ> to its successor p₁

-v: View number
-ch: Number of rechaining
-c: client id
-H: Hash of its execution history
-R: Hash of reply r to the client containing the execution result
-Λ:: Current chain order
-Replica $p_j$ receives $<\text{Chain}, v, \text{ch}, N, m, c, H, R, \Lambda>$ from his predecessor $p_{j-1}$ that contains valid signatures from $P(p_j)$
-if $p_j \in f+1$ last replicas in $A$ it updates $H$ and $R$
-It appends its signature
-Send $<\text{Chain}, v, \text{ch}, N, m, c, H, R, A>$ to its successor $p_{j+1}$.
-Set a timer $\Delta_1$. Expecting ACK or SUSPECT message
Chaining Protocol: Step 3

- Proxy tail \( p_{2f+1} \) receives \(<\text{Chain, v, ch, N, m, c, H, R, }\Lambda>\) from its predecessor \( p_{2f} \) that contain valid signatures from \( P(p_{2f+1}) \)
- Updates \( H \) and \( R \) and appends its signature.
- Sends reply to client
- Sends \(<\text{ACK,v, ch, N, m, c, H, R, }\Lambda>\) to its \( p_{2f} \)
- Sends \(<\text{Chain, v, ch, N, m, c, H, R, A}>\) to its all replicas in \( B \)
- Client complete the request if it receives reply from proxy that contains the signatures of last $f+1$ replicas in A.
- Otherwise, it retransmit the request to all replicas.
-Replica $p_j$ receives $<\text{ACK}, v, ch, N, m, c, H, R, \Lambda>$ from its successor $p_{j+1}$. (ACK message contains valid signatures from $S(p_j)$). Thus, it commits the request.

-Appends its signature.

-Sends $<\text{ACK}, v, ch, N, m, c, H, R, \Lambda>$ to its predecessor $p_{j-1}$.

-Sends $<\text{Chain, v, ch, N, m, c, H, R, \Lambda}>$ to all replicas in $B$. 

Chaining Protocol: Step 5
Chaining Protocol: Step 6

- Replicas in B collect f+1 CHAIN matching messages
- Then execute and commit the operation
Rechaining Protocol- Failure detector

Algorithm 1 Failure detector at replica $p_i$

1: upon $\langle \text{CHAIN} \rangle$ sent by $p_i$
2: $\text{starttimer}(\Delta_1,p_i)$
3: upon $\langle \text{Timeout, } \Delta_1,p_i \rangle$ \{Accuser $p_i$\}
4: send $\langle \text{SUSPECT, } p_i, m, ch, v \rangle_{p_i}$ to $\check{p}_i$ and $p_h$
5: upon $\langle \text{ACK} \rangle$ from $\check{p}_i$
6: $\text{canceltimer}(\Delta_1,p_i)$
7: upon $\langle \text{SUSPECT, } p_y, m, ch, v \rangle$ from $\check{p}_i$
8: forward $\langle \text{SUSPECT, } p_y, m, ch, v \rangle$ to $\check{p}_i$
9: $\text{canceltimer}(\Delta_1,p_i)$

Head: Handling a suspect message:
- increasing ch
- new $\Lambda$
- sending chain message

Forward also to the head
Algorithm 2 BChain-3 Re-chaining-I (At head, $p_h$)

1: upon $\langle\text{SUSPECT}, p_y, m, ch, v\rangle$ from $p_x$
2: \quad if $p_x \neq p_h$ then \{ $p_x$ is not the head \}
3: \quad $p_z$ is put to the $2^{nd}$ position \{ $p_z = B[1]$ \}
4: \quad $p_x$ is put to the $(2f + 1)^{th}$ position
5: \quad $p_y$ is put to the end

---

Rechaining Protocol

1 
2 
3 
4

\text{head} \quad \text{proxy tail} \quad \text{tail}

\text{timeout!}

\text{\langle SUSPECT \rangle}

\text{\textbf{Algorithm 2 BChain-3 Re-chaining-I (At head, } p_h \text{)}}

1: \text{upon } \langle \text{SUSPECT}, p_y, m, ch, v \rangle \text{ from } p_x
2: \quad \text{if } p_x \neq p_h \text{ then} \quad \{ p_x \text{ is not the head} \}
3: \quad p_z \text{ is put to the } 2^{nd} \text{ position} \quad \{ p_z = B[1] \}
4: \quad p_x \text{ is put to the } (2f + 1)^{th} \text{ position}
5: \quad p_y \text{ is put to the end}
Rechaining Protocol

(a) $p_3$ generates a $\langle$SUSPECT$\rangle$ message to maliciously accuse $p_4$

(b) $p_{2f+1}$ generates a $\langle$SUSPECT$\rangle$ message to accuse $p_3$

(c) $p_3$ is moved to the tail and reconfigured
Rechaining Protocol

Algorithm 3 BChain-3 Re-chaining-II

1: upon \langle SUSPECT, p_y, m, ch, v \rangle from p_x
2: if \( p_x \neq p_h \) then \{\( p_x \) is not the head\}
3: \( p_x \) is put to the \((3f)^{th}\) position
4: \( p_y \) is put to the end
Time setup and preventing performance attacks

Time setup:

\[ \Delta_{1,i} \] for each replica \( i = F(\Delta_{1,i,l_i}) \) such that if \( i = 0 \), \( l_h = 1 \) and \( \Delta_{1,h} = F(\Delta_{1,1}) = \Delta_1 \),

If \( i = 2f+1 \), \( l_p = 2f+1 \) and \( \Delta_{1,2f+1} = F(\Delta_{1,2f+1}) = 0 \)

Performance threshold

\[ \Delta'_{1,pi} < \Delta_{1,pi} \]

If average time answer is higher than \( \Delta'_{1,pi} \). Replica starts suspect procedure
View Change protocol

1- Select a new head when the current one is deemed faulty

2- Adjust timers to ensure eventual progress
View Change protocol

1- Select a new head when the current one is deemed faulty:

A correct replica votes for VIEWCHANGE if:

1- It suspects the head to be faulty.
2- It receives f+1 <VIEWCHANGE> messages.
View Change protocol

1- Select a new head when the current one is deemed faulty:

If a replica votes for a VIEWCHANGE:
   - Move to a new view
   - Send <VIEWCHANGE,......> to all replicas
   - Stop receiving messages except:
     <CHECKPOINT>, <NEWVIEW>, and <VIEWCHANGE>
View Change protocol

1- Select a new head when the current one is deemed faulty:

When new head collect 2f+1 <VIEWCHANGES>:
   -Send <NEWVIEW,....... newΛ, set valid viewChange messages, set of CHAIN messages> to all replica
   -In the new Λ, the previous head was moved to the end of the chain
View Change protocol

2- Adjust timers to ensure eventual progress:

$\Delta 1 =$ Timer for rechaining
$\Delta 2 =$ Timer for current view when replica is waiting for a request to be committed
$\Delta 3 =$ Timer for new view
View Change protocol

2- Adjust timers to ensure eventual progress:

Algorithm 4 View Change Handling and Timers at $p_i$

1: $\Delta_2 \leftarrow init_{\Delta_2}$; $\Delta_3 \leftarrow init_{\Delta_3}$
2: $\text{voted} \leftarrow \text{false}$
3: upon $\langle$Timeout, $\Delta_2$ $\rangle$
4: send $\langle$VIEWCHANGE$\rangle$
5: $\text{voted} \leftarrow \text{true}$
6: upon $f + 1 \langle$VIEWCHANGE$\rangle \land \neg \text{voted}$
7: send $\langle$VIEWCHANGE$\rangle$
8: $\text{voted} \leftarrow \text{true}$
9: canceltimer($\Delta_2$)
10: upon $2f + 1 \langle$VIEWCHANGE$\rangle$
11: starttimer($\Delta_3$)
12: upon $\langle$Timeout, $\Delta_3$ $\rangle$
13: $\Delta_3 \leftarrow g_3(\Delta_3)$
14: send new $\langle$VIEWCHANGE$\rangle$
15: upon $\langle$NEWVIEW$\rangle$
16: canceltimer($\Delta_3$)
17: $\Delta_1 \leftarrow g_1(\Delta_1)$
18: $\Delta_2 \leftarrow g_2(\Delta_2)$
Reconfiguration protocol

It is a general technique, often abstract as stopping the current state machine and restarting with a new set of replicas.

BChain reconfiguration concerns with re-chaining to replaces faulty replicas with new ones.
Checkpoint Protocol

• Similar to the PBFT
• It is used to bound the growth of message log and reduce the cost of view changes
Questions