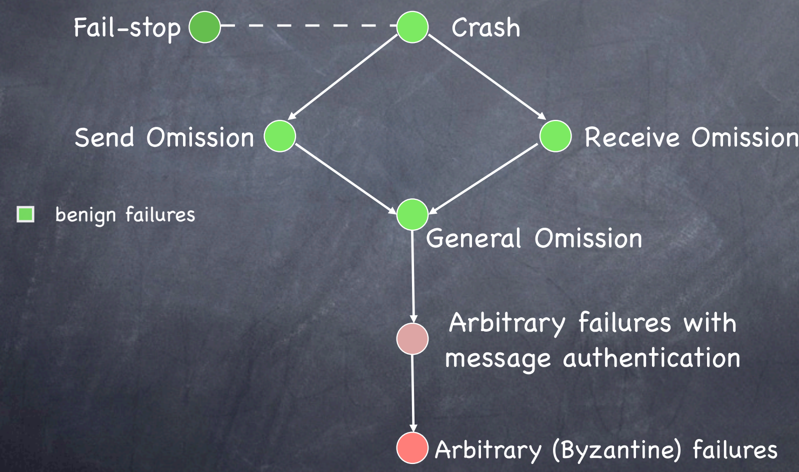
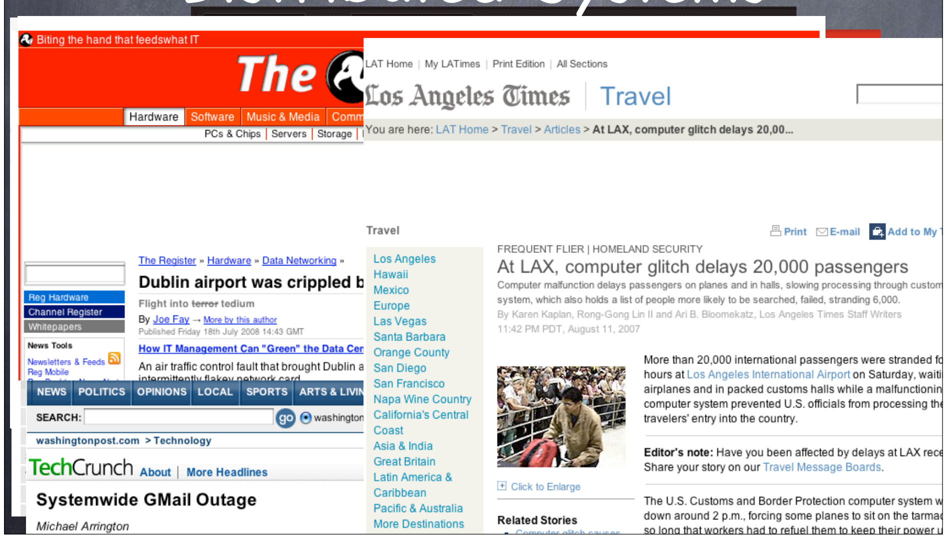
Meet BFT

A hierarchy of failure models



Weird Things Happen in Distributed Systems

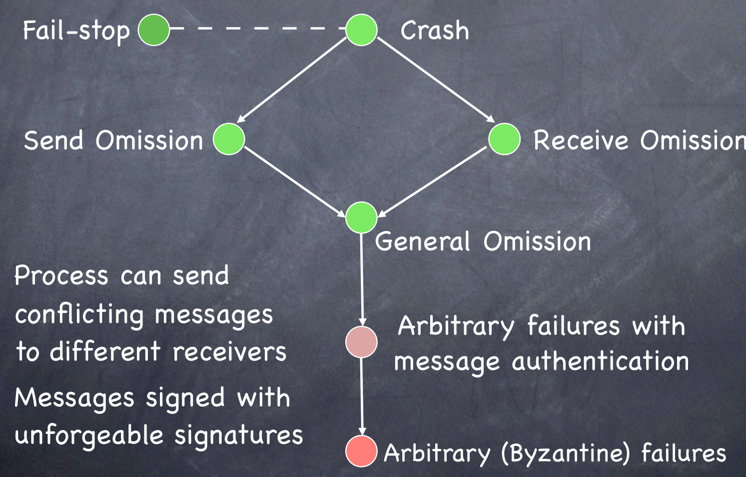
Weird Things Happen in Distributed Systems



Terminating Reliable Broadcast

Validity	If the sender is correct and broadcasts a message m, then all correct processes eventually deliver m
Agreement	If a correct process delivers a message m, then all correct processes eventually deliver m
Integrity	Every correct process delivers at most one message, and if it delivers $m \neq$ SF, then some process must have broadcast m
Termination	Every correct process eventually delivers some message

Arbitrary failures with message authentication



- Process can send conflicting messages to different receivers
- Messages signed with unforgeable signatures

Valid messages

A valid message m has the following form:

in round 1:

$m : s_{id}$ (m is signed by the sender)

in round r > 1, if received by p from q :

$m : p_1 : p_2 : \ldots : p_r$ where

- p_1 = sender; $p_r = q$
- $p_1, \ldots, p_r$ are distinct from each other and from p
- message has not been tampered with

AFMA: The Idea

- A correct process p discards all non-valid messages it receives
- If a message is valid,
  - it "extracts" the value from the message
  - it relays the message, with its own signature appended
- At round $f+1$:
  - if it extracted exactly one message, p delivers it
  - otherwise, p delivers SF

AFMA: The Protocol

Initialization for process p:
  if p = sender and p wishes to broadcast m then
    extracted := relay := $\{m\}$

Process p in round $k, 1 \leq k \leq f+1$
  for each $s \in$ relay
    send $s : p$ to all
  receive round k messages from all processes
  relay := $\emptyset$
  for each valid message received $s = m : p_1 : p_2 : \ldots : p_k$
    if $m \notin$ extracted then
      extracted := extracted $\cup \{m\}$
      relay := relay $\cup \{s\}$

At the end of round $f+1$
  if $\exists m$ such that extracted = $\{m\}$ then
    deliver m
  else deliver SF

Termination

Initialization for process p:
  if p = sender and p wishes to broadcast m then
    extracted := relay := $\{m\}$

Process p in round $k, 1 \leq k \leq f+1$
  for each $s \in$ relay
    send $s : p$ to all
  receive round k messages from all processes
  relay := $\emptyset$
  for each valid message received $s = m : p_1 : p_2 : \ldots : p_k$
    if $m \notin$ extracted then
      extracted := extracted $\cup \{m\}$
      relay := relay $\cup \{s\}$

At the end of round $f+1$
  if $\exists m$ such that extracted = $\{m\}$ then
    deliver m
  else deliver SF

In round $f+1$, every correct process delivers either m or SF and then halts

Agreement

Initialization for process p:
  if p = sender and p wishes to broadcast m then
    extracted := relay := $\{m\}$

Process p in round $k, 1 \leq k \leq f+1$
  for each $s \in$ relay
    send $s : p$ to all
  receive round k messages from all processes
  relay := $\emptyset$
  for each valid message received $s = m : p_1 : p_2 : \ldots : p_k$
    if $m \notin$ extracted then
      extracted := extracted $\cup \{m\}$
      relay := relay $\cup \{s\}$

At the end of round $f+1$
  if $\exists m$ such that extracted = $\{m\}$ then
    deliver m
  else deliver SF

Lemma. If a correct process extracts m, then every correct process eventually extracts m

Proof

Let r be the earliest round in which some correct process extracts m. Let that process be p.

- if p is the sender, then in round 1 p sends a valid message to all.

All correct processes extract that message in round 1

- If $r \leq f, p$ will send a valid message

$$m : p_1 : p_2 : \ldots : p_r : p$$

  in round $r+1 \leq f+1$ and every correct process will extract it in round $r+1 \leq f+1$

- If $r = f+1$, p has received in round $f+1$ a message

$$m : p_1 : p_2 : \ldots : p_{f+1}$$

- Each $p_j, 1 \leq j \leq f+1$ has signed and relayed a message in round $j-1 < f+1$
- At most f faulty processes - one p_j is correct and has extracted m before p

CONTRADICTION

Agreement follows directly, since all correct process extract the same set of messages

Validity

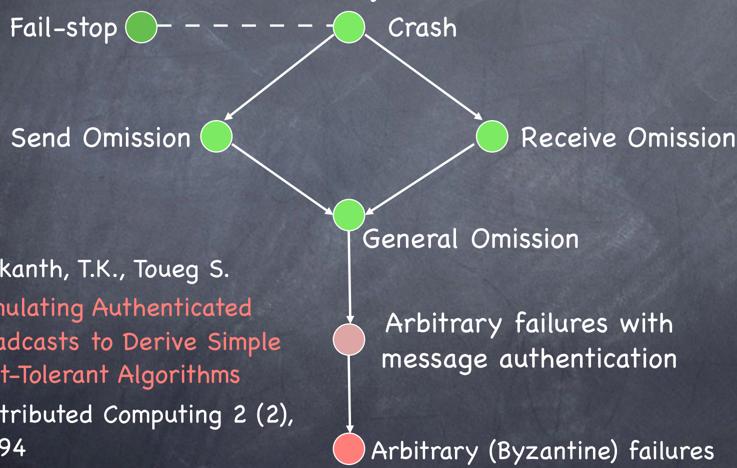
Initialization for process p:
  if p = sender and p wishes to broadcast m then
    extracted := relay := $\{m\}$

Process p in round $k, 1 \leq k \leq f+1$
  for each $s \in$ relay
    send $s : p$ to all
  receive round k messages from all processes
  relay := $\emptyset$
  for each valid message received $s = m : p_1 : p_2 : \ldots : p_k$
    if $m \notin$ extracted then
      extracted := extracted $\cup \{m\}$
      relay := relay $\cup \{s\}$

At the end of round $f+1$
  if $\exists m$ such that extracted = $\{m\}$ then
    deliver m
  else deliver SF

From Agreement and the observation that the sender, if correct, delivers its own message.

TRB for arbitrary failures



Srikanth, T.K., Toueg S.
Simulating Authenticated Broadcasts to Derive Simple Fault-Tolerant Algorithms
Distributed Computing 2 (2), 80-94

AF: The Idea

- Identify the essential properties of message authentication that made AFMA work
- Implement these properties without using message authentication

AF: The Approach

- Introduce two primitives

  broadcast(p, m, i) (executed by p in round i)
  accept(p, m, i) (executed by q in round $j \geq i$)

- Give axiomatic definitions of broadcast and accept
- Derive an algorithm that solves TRB for AF using these primitives
- Show an implementation of these primitives that does not use message authentication

Properties of broadcast and accept

- Correctness   If a correct process p executes broadcast(p, m, i) in round i, then all correct processes will execute accept(p, m, i) in round i
- Unforgeability   If a correct process q executes accept(p, m, i) in round $j \geq i$, and p is correct, then p did in fact execute broadcast(p, m, i) in round i
- Relay   If a correct process q executes accept(p, m, i) in round $j \geq i$, then all correct processes will execute accept(p, m, i) by round $j+1$

AF: The Protocol - 1

sender s in round 0:
0: extract m

sender s in round 1:
1: broadcast$(s, m, 1)$
Process p in round $k, 1 \leq k \leq f+1$
2: if p extracted m in round $k-1$ and $p \neq$ sender then
4: broadcast(p, m, k)
5: if p has executed at least k accept(q_i, m, j_i) $1 \leq i \leq k$ in rounds 1 through k
(where (i) q_i distinct from each other and from p, (ii) one q_i is s, and
(iii) $1 \leq j_i \leq k$) and p has not previously extracted m then
6: extract m
7: if $k = f+1$ then
8: if in the entire execution p has extracted exactly one m then
9: deliver m
10: else deliver SF
11: halt

Termination

sender s in round 0:
0: extract m
sender s in round 1:
1: broadcast$(s, m, 1)$
Process p in round $k, 1 \leq k \leq f+1$
2: if p extracted m in round $k-1$ and $p \neq$ sender then
4: broadcast(p, m, k)
5: if p has executed at least k accept(q_i, m, j_i) $1 \leq i \leq k$ in rounds 1 through k
(where (i) q_i distinct from each other and from p, (ii) one q_i is s, and (iii) $1 \leq j_i \leq k$)
and p has not previously extracted m then
6: extract m
7: if $k = f+1$ then
8: if in the entire execution p has extracted exactly one m then
9: deliver m
10: else deliver SF
11: halt

In round $f+1$, every correct process delivers either m or SF and then halts

Validity

sender s in round 0:
0: extract m
sender s in round 1:
1: broadcast$(s, m, 1)$
Process p in round $k, 1 \leq k \leq f+1$
2: if p extracted m in round $k-1$ and $p \neq$ sender then
4: broadcast(p, m, k)
5: if p has executed at least k accept(q_i, m, j_i) $1 \leq i \leq k$ in rounds 1 through k
(where (i) q_i distinct from each other and from p, (ii) one q_i is s, and (iii) $1 \leq j_i \leq k$)
and p has not previously extracted m then
6: extract m
7: if $k = f+1$ then
8: if in the entire execution p has extracted exactly one m then
9: deliver m
10: else deliver SF
11: halt

- A correct sender executes broadcast$(s, m, 1)$ in round 1
- By CORRECTNESS, all correct processes execute accept$(s, m, 1)$ in round 1 and extract m
- In order to extract a different message m', a process must execute accept$(s, m', 1)$ in some round $i \leq f+1$
- By UNFORGEABILITY, and because s is correct, no correct process can extract $m' \neq m$
- All correct processes will deliver m

Agreement - 1

sender s in round 0:
0: extract m
sender s in round 1:
1: broadcast$(s, m, 1)$
Process p in round $k, 1 \leq k \leq f+1$
2: if p extracted m in round $k-1$ and $p \neq$ sender then
4: broadcast(p, m, k)
5: if p has executed at least k accept(q_i, m, j_i) $1 \leq i \leq k$ in rounds 1 through k
(where (i) q_i distinct from each other and from p, (ii) one q_i is s, and (iii) $1 \leq j_i \leq k$)
and p has not previously extracted m then
6: extract m
7: if $k = f+1$ then
8: if in the entire execution p has extracted exactly one m then
9: deliver m
10: else deliver SF
11: halt

Lemma
If a correct process extracts m, then every correct process eventually extracts m