AVMON: Consistent and Scalable Availability Monitoring Overlay

Ramsés Morales and Indranil Gupta

Dept. of Computer Science
University of Illinois at Urbana-Champaign
Motivation

- Large scale distributed applications must deal with churn.
- Difficult to predict availability variations across nodes and time cause churn.
  - Availability: fraction of time a node is online.
Motivation (cont.)

av(a) = 0.45
av(b) = 0.95
av(c) = 0.99

gossip multicast $m$:  
$\times f(\text{availability}(a))$
$\checkmark f(\text{availability}(b))$
$\checkmark f(\text{availability}(c))$

- Multicast reception reliability $\propto$ availability
  [Pongthawornkamol 2006]
Motivation (cont.)

Generic availability monitoring (overlay) problem has not been addressed.

Motivation (cont.)

Generic availability monitoring (overlay) problem has *not* been addressed.

Selfish

My availability is 1.0
Motivation (cont.)

Generic availability monitoring (overlay) problem has not been addressed.

Colluding

Its availability is 1.0

Selfish

My availability is 1.0
Availability Monitoring Problem

I. Availability Monitoring Overlay [our focus]
II. Availability History Maintenance
   - [Bhagwan 2004][Mickens 2006]
Availability Monitoring Overlay
Problem

- For each node $x$, select and discover a small subset of nodes to monitor $x$. 
Availability Monitoring Overlay Problem

- For each node $x$, select and discover a small subset of nodes to monitor $x$.

$PS(x)$

$x$'s Pinging Set
Availability Monitoring Overlay Problem

- For each node $x$, select and discover a small subset of nodes to monitor $x$.  

$$PS(x)$$  
$x$'s Pinging Set

$$TS(x)$$  
$x$'s Target Set

$$PS = TS^{-1}$$
Outline

- Motivation
- Availability Monitoring Problem
- System Model
- Design Goals
- AVMON System
- Experimental Results
System Model

My availability is 1.0

Selfish

Colluding

Its availability is 1.0

My availability is 1.0
System Model (cont.)

Joins

Leaves

Crashes
System Model (cont.)

• Stable system size, $N$.
  
  – The number of **online** nodes in the system varies within a constant factor of $N$.

• Valid for:
  
  – P2P systems [Bhagwan 2003]
  – PlanetLab based overlays
  – Grid systems
Outline

- Motivation
- Availability Monitoring Problem
- System Model
- Design Goals
- AVMON System
- Experimental Results
Design Goal: Consistency

- $y \in PS(x)$, must be consistent, i.e.,
  - won't change regardless of system changes.

$PS(x)$: x's Pinging Set
Design Goal: **Consistency** (cont.)

- $y \in PS(x)$ must be consistent.

- Useful for
  - long term availability history maintenance,
  - no history transferral under churn,
  - avoid pollution with colluders.

$x$'s Pinging Set $PS(x)$
Design Goal: Verifiability

- Consistency, Verifiability, ________, ________, ________, ________.
Design Goal: **Verifiability** (cont.)

- Consistency, Verifiability, _______, _______, _______, _______.

\[ x' \in PS(x) \]

\[ x' \notin PS(x) \text{ false!} \]
Design Goal: Randomness

- Consistency, Verifiability, Randomness, __________, __________, __________.

$PS(x)$

$PS(x)$ populated uniformly at random, - identically distributed fashion, and - no correlation among nodes.
Design Goal: Randomness (cont.)

- Consistency, Verifiability, Randomness, __________, __________, __________.

Correlated ping sets

x

Consistency, Verifiability, Randomness, __________, __________, __________.
Design Goal: Randomness (cont.)

- Helps with
  - scalability,
  - load balancing.

- Consistency, Verifiability, Randomness, __________, __________, __________.
Design Goal: Discoverability

- Consistency, Verifiability, Randomness, Discoverability, ________, __________.

- $x$ should learn quickly its $\text{PS}(x)$ and $\text{TS}(x)$

$x$'s Pinging Set

$\text{PS}(x)$

$x$'s Target Set

$\text{TS}(x)$
Design Goal: Discoverability (cont.)

- $x$ should learn quickly its $PS(x)$ and $TS(x)$

$PS(x)$: x's Pinging Set

$TS(x)$: x's Target Set

Consistency, Verifiability, Randomness, Discoverability, ________, ________.
Design Goal: Load Balancing

- Discovery overhead should be uniformly distributed.
  - Messages.
  - Memory.
  - Computation.
Design Goal: Scalability

- Discovery overhead should be **low and scalable**.
  - Messages.
  - Memory.
  - Computation.

• Consistency, Verifiability, Randomness, Discoverability, Load Balancing, Scalability.
Design Goals

- Consistency
- Verifiability
- Randomness
- Discoverability
- Load Balancing
- Scalability

} \[ \text{Selection of PS(.) & TS(.)} \]
Outline

- Motivation
- Availability Monitoring Problem
- System Model
- Design Goals
- AVMON System
  - Selection and Discovery
- Experimental Results
AVMON's Monitor Selection

PS(x) \xrightarrow{x} TS(x)
AVMON's Monitor Selection

\[ H(y, x) \leq K / N \]
AVMON's Monitor Selection

\[ H(y, x) \leq \frac{K}{N} \]
AVMON's Monitor Selection

\( y \in PS(x), x \in TS(y) \)

\( H(y, x) \leq K / N \)

\((\text{MD5, SHA1})\) [0, 1]
AVMON's Monitor Selection

\[ H(y, x) \leq \frac{K}{N} \]

\[ y \in PS(x), \ x \in TS(y) \]

Approx. system size
AVMON's Monitor Selection

$y \leftarrow \langle \text{ip, port}\rangle \rightarrow H(y, x) \leq K / N \rightarrow \langle \text{ip, port}\rangle \rightarrow x$

Small, system-wide, fixed integer, usually $\log(N)$

$y \in PS(x), x \in TS(y)$
AVMON's Monitor Selection

- $|PS(x)|$ is $\Theta(K)$ on expectation
- Consistent, Random, and Verifiable

$H(y, x) \leq K / N$

$y \in PS(x) \iff H(y, x) \leq K / N$
AVMON's Monitor Discovery

- For any monitor selection scheme that is consistent and verifiable
AVMON's Monitor Discovery (cont.)

- Coarse View at $x$
  - Random subset of nodes

- Managed by
  - join subprotocol, and
  - maintenance and discovery subprotocol.
AVMON's Monitor Discovery:
CV Join Subprotocol

Random targets taken from CV(x)

- $p$ will be included in a total of cvs number of coarse views.
AVMON's Monitor Discovery: CV Maintenance and Discovery

- Periodically ($T_{cv}$):

<table>
<thead>
<tr>
<th>CV(x)</th>
</tr>
</thead>
<tbody>
<tr>
<td>a: &lt;ip, port&gt;</td>
</tr>
<tr>
<td>...</td>
</tr>
<tr>
<td>y: &lt;ip, port&gt;</td>
</tr>
<tr>
<td>...</td>
</tr>
<tr>
<td>z: &lt;ip, port&gt;</td>
</tr>
</tbody>
</table>

$cvs$ (coarse view size)
AVMON's Monitor Discovery: CV Maintenance and Discovery

- Periodically ($T_{cv}$):

  \[
  CV(x) = \begin{array}{l}
  a: \langle ip, \text{port} \rangle \\
  \quad \ldots \\
  y: \langle ip, \text{port} \rangle \\
  \quad \ldots \\
  z: \langle ip, \text{port} \rangle \\
  \end{array}
  \]

  \[\text{cvs (coarse view size)}\]
AVMON's Monitor Discovery: CV Maintenance and Discovery

- Periodically ($T_{cv}$):

  - $x$
  - $y$
  - $<CV?>$
  - $CV(y)$
  - $a: <$ip, port$>$
  - $y: <$ip, port$>$
  - CVS (coarse view size)
AVMON's Monitor Discovery: CV Maintenance and Discovery

\[
\begin{align*}
\text{CV}(x) \\
a: & <\text{ip, port}> \\
y: & <\text{ip, port}>
\end{align*}
\quad\quad
\begin{align*}
\text{CV}(y) \\
b: & <\text{ip, port}> \\
v: & <\text{ip, port}>
\end{align*}
\]

\[
\text{compute } H(., .) \leq K / N \text{ using } \{a, y\} \times \{b, v\}, \\
\{b, v\} \times \{a, y\}
\]
AVMON's Monitor Discovery: CV Maintenance and Discovery

- \( H(a, v) \leq K / N \)

\[ H(a, v) \leq K / N, \quad v \rightarrow TS(a) \]

\[ H(a, v) \leq K / N, \quad a \rightarrow PS(v) \]
AVMON's Monitor Discovery: CV Maintenance and Discovery

<table>
<thead>
<tr>
<th>CV(x)</th>
<th>CV(y)</th>
</tr>
</thead>
<tbody>
<tr>
<td>a: &lt;ip, port&gt;</td>
<td>b: &lt;ip, port&gt;</td>
</tr>
<tr>
<td>y: &lt;ip, port&gt;</td>
<td>v: &lt;ip, port&gt;</td>
</tr>
</tbody>
</table>

Random shuffle cvs elements

<table>
<thead>
<tr>
<th>CV(x)</th>
</tr>
</thead>
<tbody>
<tr>
<td>a: &lt;ip, port&gt;</td>
</tr>
<tr>
<td>v: &lt;ip, port&gt;</td>
</tr>
</tbody>
</table>
Outline

- Motivation
- Availability Monitoring Problem
- System Model
- Design Goals
- AVMON System
  - Analysis
- Experimental Results
Optimal Memory, Discovery, Computation

- Minimize: \( f(cvs) = M + C + E[D] \)
- \( f(cvs) = cvs + cvs^2 + \frac{N}{cvs^2} \)
- \( \frac{d(f(cvs))}{d(cvs)} = 1 + 2cvs - \frac{2N}{cvs^3} = 0 \)
- \( cvs \) for Optimal MDC \( \approx N^{1/4} \), e.g.,
  - \( N = 1 \) Million
  - \( cvs = 32 \)
  - \( CV(.) \) is 192 Bytes
  - \( K = \log_2(N) = 20 \)
  - Computation overhead \( (cvs^2) \), 0.384ms (per min. on a PC)
  - Discover 1 monitor every 20 rounds on average
Collusion Resilience

- \( \mathbf{x} \) colludes with \( C \) nodes
  - \( C = o(N / \log(N)) \)

- \( K = \Theta(\log(N)) \)

- Probability no such colluders appear in \( PS(\mathbf{x}) \)
  - \( (1 - K / N)^C \approx (1 - CK / N) \quad \rightarrow \quad 1 \quad \text{as} \quad N \rightarrow \infty \)
Outline

• Motivation
• Availability Monitoring Problem
• System Model
• Design Goals
• AVMON System
  → Using AVMON
• Experimental Results
Using **AVMON**

**Querying Availability**

x's av?
request (policy)
L monitors out of K

PS(x)
<table>
<thead>
<tr>
<th>a</th>
</tr>
</thead>
<tbody>
<tr>
<td>v</td>
</tr>
<tr>
<td>...</td>
</tr>
</tbody>
</table>

<x, L monitors?>

<a, v>
Using **AVMON**

**Querying Availability**

✓ $H(a, x) \leq K / N$
✓ $H(v, x) \leq K / N$

$x$'s availability?

<table>
<thead>
<tr>
<th>$PS(x)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$a$</td>
</tr>
<tr>
<td>$v$</td>
</tr>
<tr>
<td>...</td>
</tr>
</tbody>
</table>

$x$
Using **AVMON**

**Monitoring Availability**

- Periodically ($T_{mon}$):

```
<AV_PING>  
\[ x \quad <AV_PONG> \quad TS(x) \quad \cdots \]  
\[ t_1 \quad t_k \]
```
Monitoring Optimization: Forgetful Pinging

- Periodically ($T_{mon}$):

- $t_{unresponsive} > \tau$

- Ping with probability ($c \cdot t_{lastUptime})/(t_{lastUptime} + t_{unresponsive})$
Outline

- Motivation
- Availability Monitoring Problem
- System Model
- Design Goals
- AVMON System
- Experimental Results
Experimental Design

- Synthetic churn models:
  - STAT: no churn
  - SYNTH: 20% leave/join per hour
  - SYNTH-BD: 20% birth/death per day

- Availability Traces:
  - OV: Overnet system
  - PL: Planetlab system
### Experimental Design (cont.)

<table>
<thead>
<tr>
<th>Maintenance</th>
<th>N</th>
<th>Total Nodes</th>
<th>Coarse View Size</th>
<th>$K = \log_2(N)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>STAT</td>
<td>2000</td>
<td>2200</td>
<td>27</td>
<td>11</td>
</tr>
<tr>
<td>SYNTH</td>
<td>2000</td>
<td>2200</td>
<td>27</td>
<td>11</td>
</tr>
<tr>
<td>SYNTH-BD</td>
<td>2000</td>
<td>2809</td>
<td>27</td>
<td>11</td>
</tr>
<tr>
<td>OV</td>
<td>550</td>
<td>1319</td>
<td>19</td>
<td>9</td>
</tr>
<tr>
<td>PL</td>
<td>239</td>
<td>239</td>
<td>16</td>
<td>8</td>
</tr>
</tbody>
</table>

$K = \log_2(N)$
First Monitor Discovery

- Discovery of first monitor (for an arbitrary node) happens within one protocol period.
First Monitor Discovery

- BW: 6.81 Bps
- Mem: 52 Bytes
- Comp: 0.57ms (0.0000095% CPU) on a PC
Further Monitor Discovery

- Discovery time stays low, regardless of birth, death, join, rejoin.
Bandwidth

- BW is \textit{uniform}, and \textit{low}

10 Bytes per second
Discovery time vs. Coarse View size

- coarse view size = 68
- 9375.79 hashes
  - 3.52ms (per min)
- $K = 11$
- memory = 720 B
- BW = 632 Bps

- coarse view size $= 4^N^{1/4}, 6^N^{1/4}, 8^N^{1/4}, 10^N^{1/4}$
Discovery time vs. Coarse View size

- coarse view size = 68
- 9375.79 hashes
  - 3.52ms (per min)
- $K = 11$
- memory = 720 B
- BW = 632 Bps

- coarse view size = $4^*N^{1/4}$, $6^*N^{1/4}$, $8^*N^{1/4}$, $10^*N^{1/4}$
Forgetful Pinging

- Average error rate, <5%
- BW reduced by half
- Monitoring ping every 1 minute
- Start optimization after 2 minutes unresponsive
Overreporting Attack

- Random nodes selected as malicious
- Malicious nodes always report 100% availability for its $TS(.)$
- Node “hurt” if its reported availability is off by at least 20% from real
Overreporting Attack

- Worst case only 3.5% nodes are “hurt”
Conclusions

- **AVMON** is the first system to address **Availability Monitoring Overlay** problem with goals:
  - Consistency, Verifiability, Randomness, Discoverability, Load Balancing, Scalability

- Efficient discovery and selection

  For $N = 2000$:
  - Discovery within 30s (half a protocol period)
  - Low BW: 6.81 Bps
  - Low Mem: 52 Bytes
  - Low CPU overhead: 0.57ms per min.
Thank you!

Questions?

AVMON's source code will soon be posted at:
http://kepler.cs.uiuc.edu/~rvmorale/avmon