We define the *approximate trace-back* problem as finding a candidate attack path for each attacker that contains the true attack path as a suffix. We call this the *valid suffix* of the candidate path.
Send k fragments into network
for $d := 0$ to $maxd$
  for all ordered combinations of fragments at distance $d$
  construct edge $z$
  if $d \neq 0$ then
    $z := z \oplus last$
  if $\text{Hash(EvenBits}(z)) = \text{OddBits}(z)$ then
    insert edge $(z, \text{EvenBits}(z), d)$ into $G$
  $last := \text{EvenBits}(z)$;
<table>
<thead>
<tr>
<th>Method</th>
<th>Management overhead</th>
<th>Network overhead</th>
<th>Router overhead</th>
<th>Distributed capability</th>
<th>Post-mortem capability</th>
<th>Preventative/reactive</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ingress filtering</td>
<td>Moderate</td>
<td>Low</td>
<td>Moderate</td>
<td>N/A</td>
<td>N/A</td>
<td>Preventative</td>
</tr>
<tr>
<td>Link testing</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Input debugging</td>
<td>High</td>
<td>Low</td>
<td>High</td>
<td>Good</td>
<td>Poor</td>
<td>Reactive</td>
</tr>
<tr>
<td>Controlled flooding</td>
<td>Low</td>
<td>High</td>
<td>Low</td>
<td>Poor</td>
<td>Excellent</td>
<td>Reactive</td>
</tr>
<tr>
<td>Logging</td>
<td>High</td>
<td>Low</td>
<td>High</td>
<td>Good</td>
<td>Excellent</td>
<td>Reactive</td>
</tr>
<tr>
<td>ICMP Traceback</td>
<td>Low</td>
<td>Low</td>
<td>Low</td>
<td>Good</td>
<td>Excellent</td>
<td>Reactive</td>
</tr>
<tr>
<td>Marking</td>
<td>Low</td>
<td>Low</td>
<td>Low</td>
<td>Good</td>
<td>Excellent</td>
<td>Reactive</td>
</tr>
</tbody>
</table>

Table 1: Qualitative comparison of existing schemes for combating anonymous attacks and the probabilistic marking approach we propose.
Figure 3. Example of our initial marking scheme. The packet travels from the attacker A to the victim V across the routers R1 to R5. Each router uses the TTL value of the packet to index into the IP identification field to insert its marking. In this example we show a 1-bit marking in a 4-bit field for simplicity.
% dig +dhsec berkeley.edu

69-byte query
Diffuse DDoS: Reflector Attack

Request: \( src = \text{victim} \)
\( dst = \text{reflector} \)

Slave 1 \rightarrow \text{Reflector 1}
Slave 2 \rightarrow \text{Reflector 2}
Slave 3 \rightarrow \text{Reflector 3}
Slave 4 \rightarrow \text{Reflector 4}

Master \rightarrow \text{Slave 1, Slave 2, Slave 3, Slave 4}

Reply: \( src = \text{reflector} \)
\( dst = \text{victim} \)

\text{Reflector 1} \rightarrow \text{Victim}
\text{Reflector 2} \rightarrow \text{Victim}
\text{Reflector 3} \rightarrow \text{Victim}
\text{Reflector 4} \rightarrow \text{Victim}
\text{Reflector 5} \rightarrow \text{Victim}
\text{Reflector 6} \rightarrow \text{Victim}
\text{Reflector 7} \rightarrow \text{Victim}
\text{Reflector 8} \rightarrow \text{Victim}
\text{Reflector 9} \rightarrow \text{Victim}
\text{Reflector 10} \rightarrow \text{Victim}
\text{Reflector 11} \rightarrow \text{Victim}

Control traffic directs slaves at victim & reflectors

Reflectors send streams of non-spoofed but unsolicited traffic to victim