

# *Structured Overlays:*




## *Eclipse Attacks on Overlay Networks*

*April 28th, 2006*


*Wyman Park 4<sup>th</sup> Floor Conference Room*

*Presentation by:*

**Dan Liu & Jay Zarfoss**




*“The idea of churn as shelter from route poisoning attacks is an interesting, if simple, idea.”*




*“The ID of a node can’t be tied to actual data like files that would have to be changed at every epoch.”*

“On one hand, for distributed file systems and databases the cost of migrating data across nodes could be high, and induced churn may be inappropriate.”



*“We would want the authors’ defensive scheme to be able to scale to the level of Kazaa and Napster.”*

Structured vs Unstructured overlays is  
not a fair comparison




Thou shall not let nodes pick  
their own identifiers!


Timeserver, timeserver, timeserver...

Low hanging fruit

*“Each node randomly picks a fixed position in the epoch  
and computes everything (ID update, routing table  
removals, etc) related to this.”*

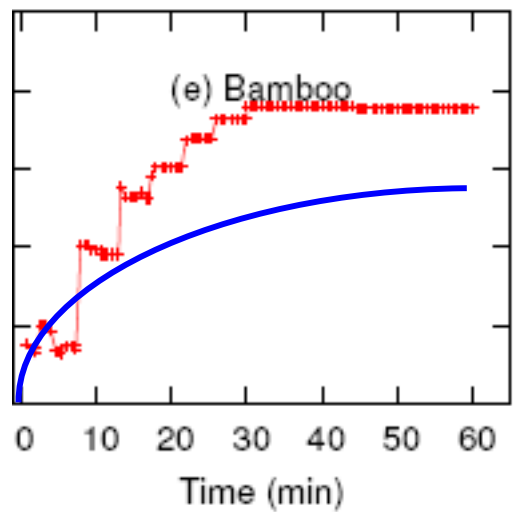
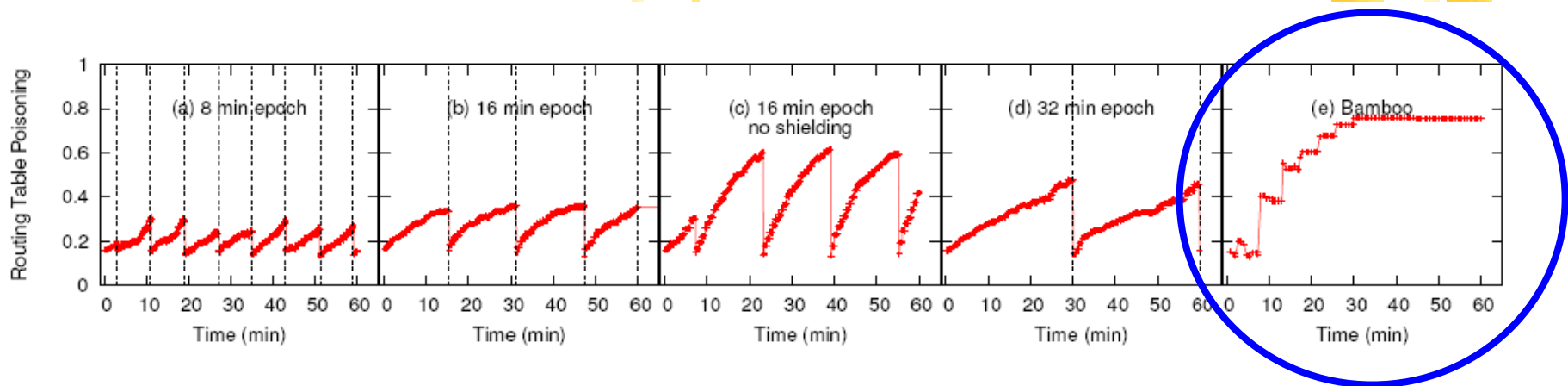


*“My greatest complaint with the analysis is that they evaluate their system exclusively with a very powerful adversary.”*



*“I would have liked to see a more detailed explanation of how the attack on periodic resets + update rate limitation works.”*

*“...the major component of their approach is the rate limiting rather than the actual churn.”*





# Extensions?



*“First, rather than storing only the first hops of queries, we store entire paths”*

*“We would first want to dissect an application for patterns in finding optimized routes.”*

# SimNet?



*“It would have been believable if they had used an established simulator renowned for its real-world network modeling, such as SimNet.”*

# Motivation



- Yesterday we looked at induced churn to defeat routing table poisoning
- Can we defeat poisoning and still support the use of a highly optimized routing table?
- What if we place restrictions on the degree of a node?



# Eclipse Attacks on Overlay Networks: Threats and Defenses

Atul Singh, Tsuen Ngan, Peter Druschel, Dan Wallach  
Rice University  
IEEE Infocom 2006

# Pastry Node Review

- Leaf Set
- Routing Table
- Neighborhood Set
  - Contains node ids and IP addresses of the nodes that are closest to the local node

NodeId 10233102			
Leaf set	SMALLER	LARGER	
10233033	10233021	10233120	10233122
10233001	10233000	10233230	10233232

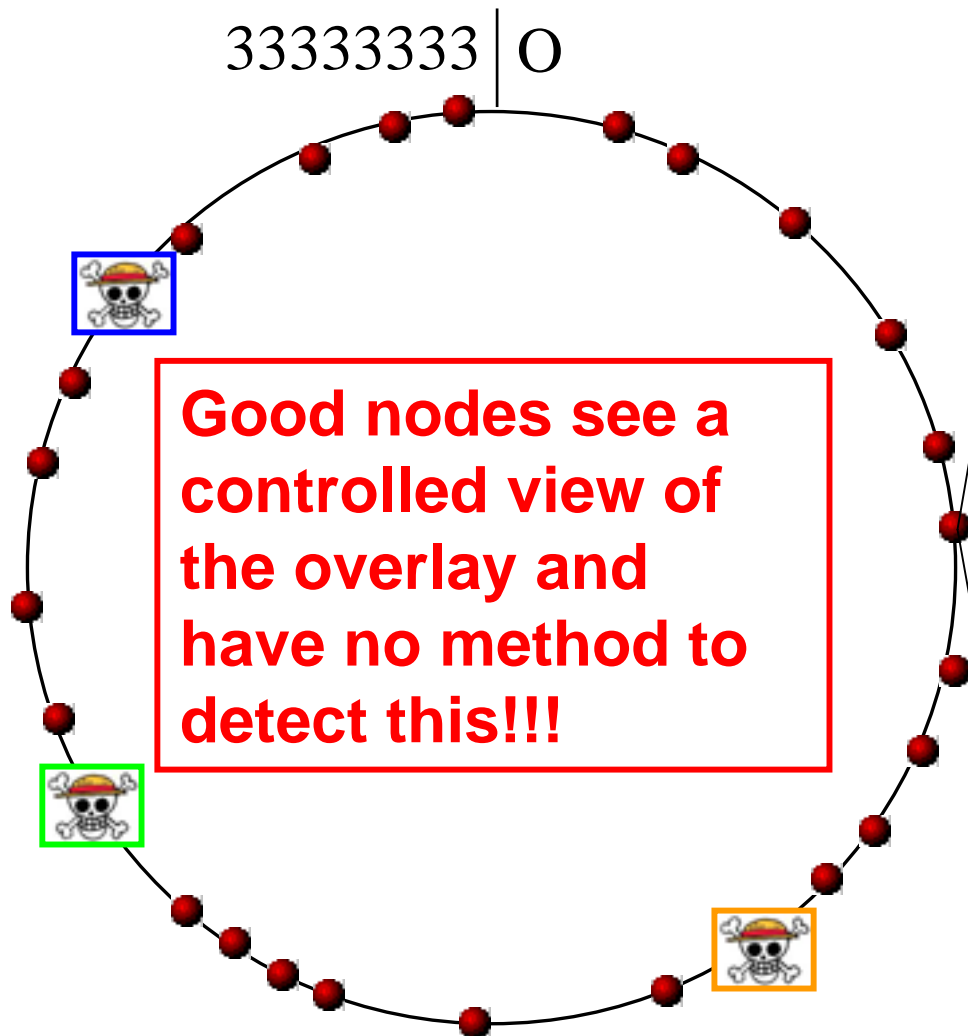
  
















Routing table			
-0-2212102	1	-2-2301203	-3-1203203
0	1-1-301233	1-2-230203	1-3-021022
10-0-31203	10-1-32102	2	10-3-23302
102-0-0230	102-1-1302	102-2-2302	3
1023-0-322	1023-1-000	1023-2-121	3
10233-0-01	1	10233-2-32	
0		102331-2-0	
		2	

Neighborhood set			
13021022	10200230	11301233	31301233
02212102	22301203	31203203	33213321

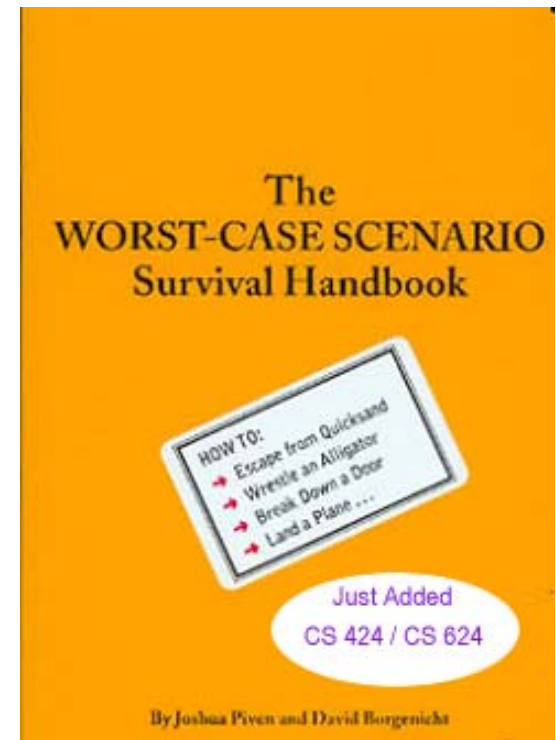
# Notion of Eclipse Attack



Nodeid 10233102				
Leaf set		SMALLER	LARGER	
10233033		 1	1 	10233122
		10233000	10233230	10233232
Routing table				
-  2	1	-  3	-  3	
0	1-1-301233	1-2-230203	1-3-021012	
10-0-31203	1 	2	1 	
102-0-0230	102-1-1-302	102-2-2302	3	
1  2	10 	1022-2-121	3	
10233-0-01	1	1 		
0		102331-2-0		
		2		
Neighborhood set				
13021022	1 	11301233	3 	
 2	22301203	3 	33213321	

# Worst Case Scenario

- Bootstrapping Process
- Continually Spreading Over Time
- Complete Control of Overlay
  - Arbitrary Denial of Service
  - Censorship Attack
- Our threat model is to prevent this global attack on **every** neighbor set/routing table



# Eclipse Defenses



- Centralized Membership Service
- Stronger Structural Constraints
- Proximity Constraints
- Induced Churn
- Enforcing Degree Bounds
- Anonymous Auditing

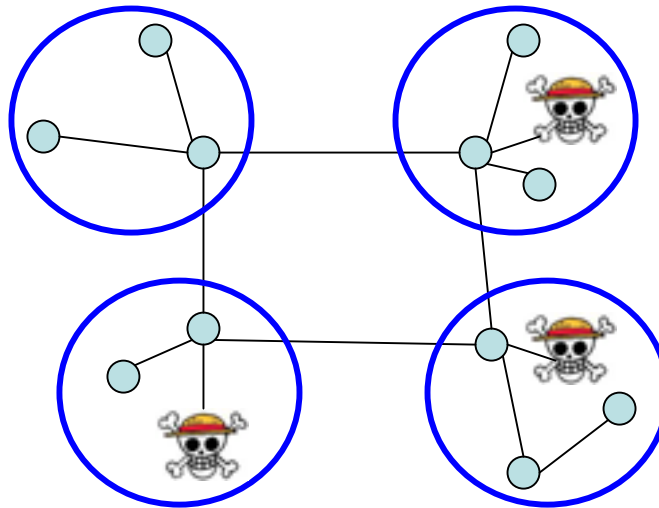




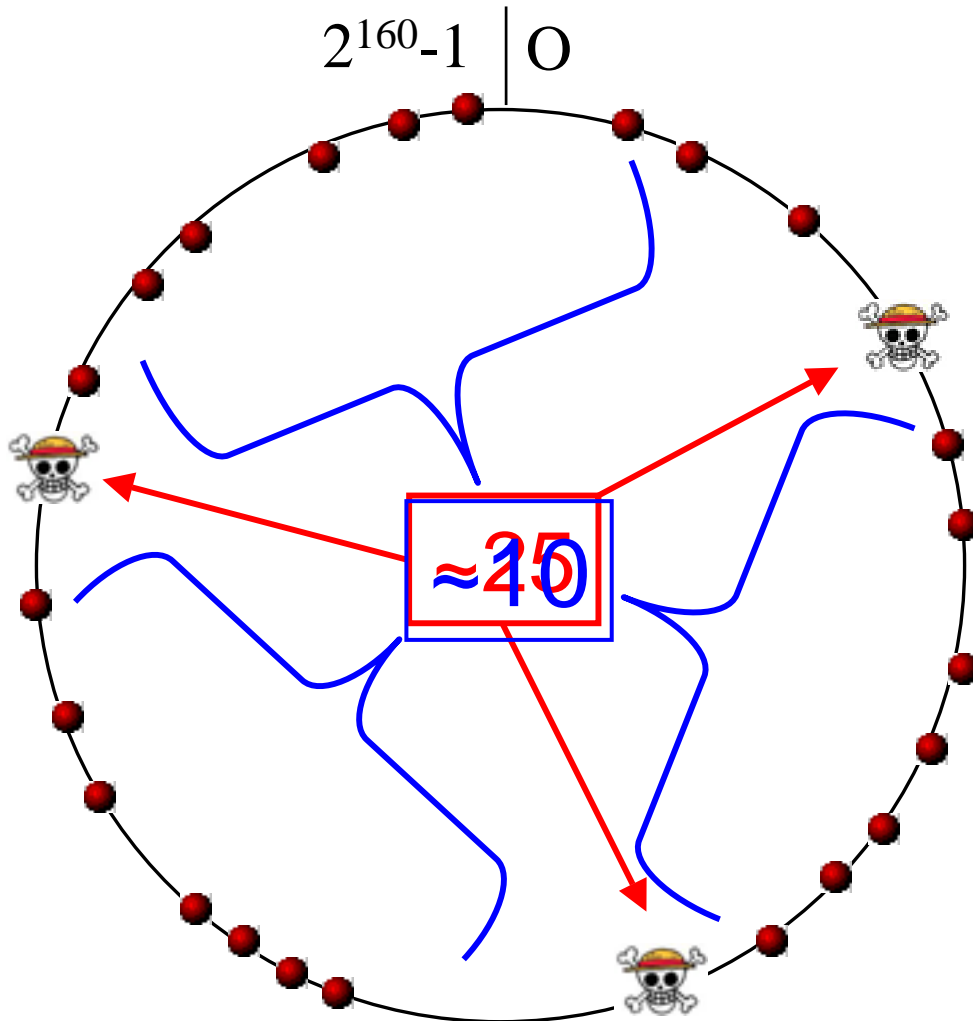
# More on Proximity Constraints

- This defense assumes a small number of malicious nodes cannot be within a low network delay of all nodes (PNS Defense)

These routing tables will tend to have more good entries



# Simple Observation



- Eclipse attackers will have a high in-degree in the overlay
- Every other node has an average in degree

# Effect of Enforcing Degree Bounds

$f$  = fraction of malicious nodes

$O_{exp}$  = expected out degree of good nodes

$N(1 - f)O_{exp}$  = total out degree of good nodes

$I_{max}$  is bound on in-degree for all nodes

$I_{max} = tO_{exp}$  for some  $t \geq 1$

$NfI_{max}$  = total in degree of malicious nodes

$f'$  = fraction of out degree of good nodes consumed by malicious nodes

---

$$f' N(1 - f)O_{exp} \leq Nf t O_{exp}$$

$$f' \leq \frac{ft}{(1-f)}, \text{ bound } t = 1$$

$$f' \leq \frac{f}{(1-f)}$$

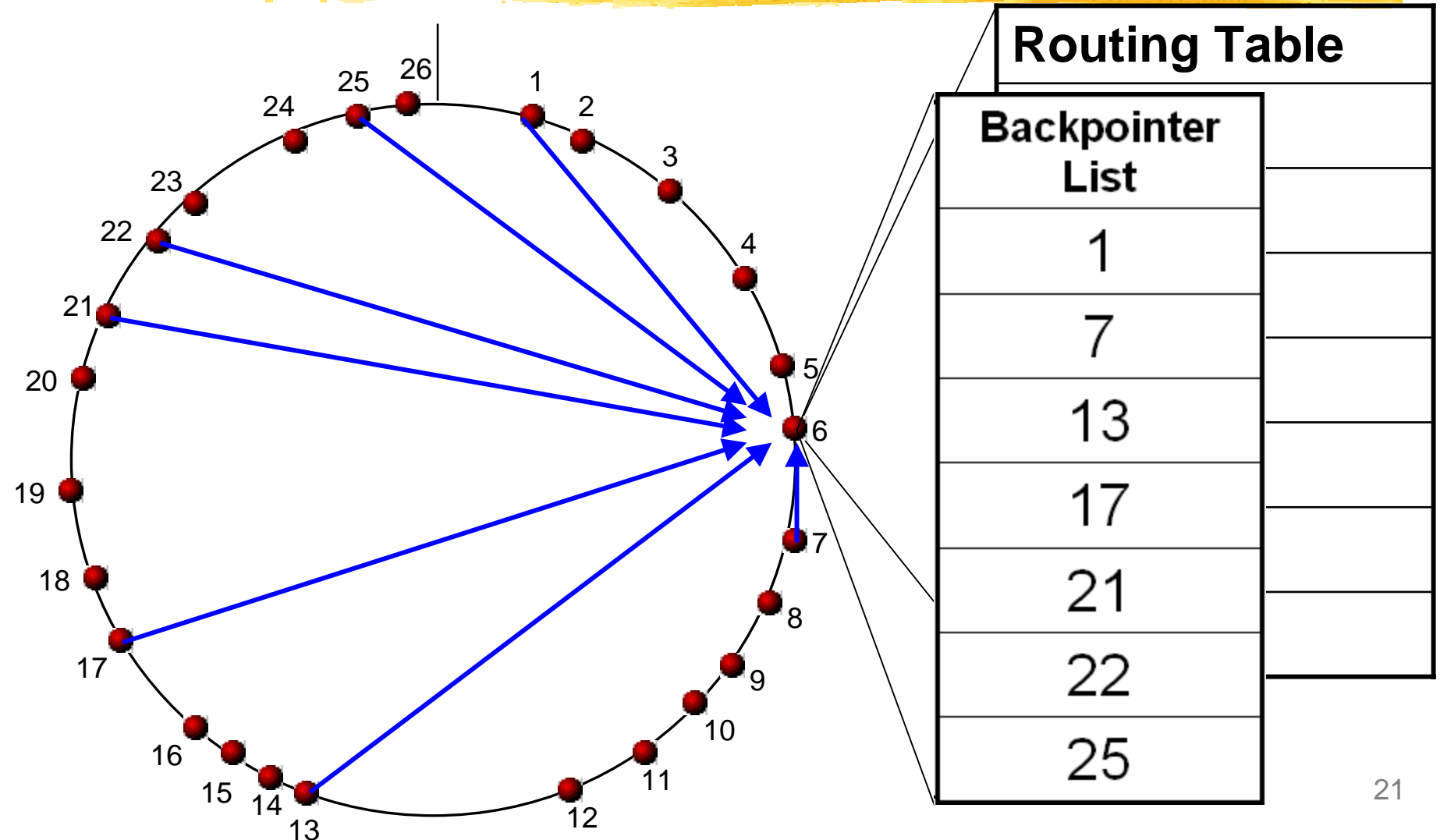
**How Do We Enforce  
Bounds in the Overlay?**

# Enforcing Degree Bounds



- Could use a centralized membership service
  - Dedicated service keeps track of each overlay member's degree
  - Single point of failure, availability, and scalability issues
- Can we come up with a distributed mechanism where everyone checks each other's **back**?

# Every Node Maintains a Backpointer List



# Checking Backpointer Lists



- Periodically, a node  $x$  challenges each of its neighbors for its backpointer list
- If the list is too large or does not contain  $x$ , the audit **fails** and the node is removed
- Periodically, a node  $x$  also checks its backpointer list to make sure each node on the list has a **correct** neighbor set/routing table size

# Fresh and Authentic Replies

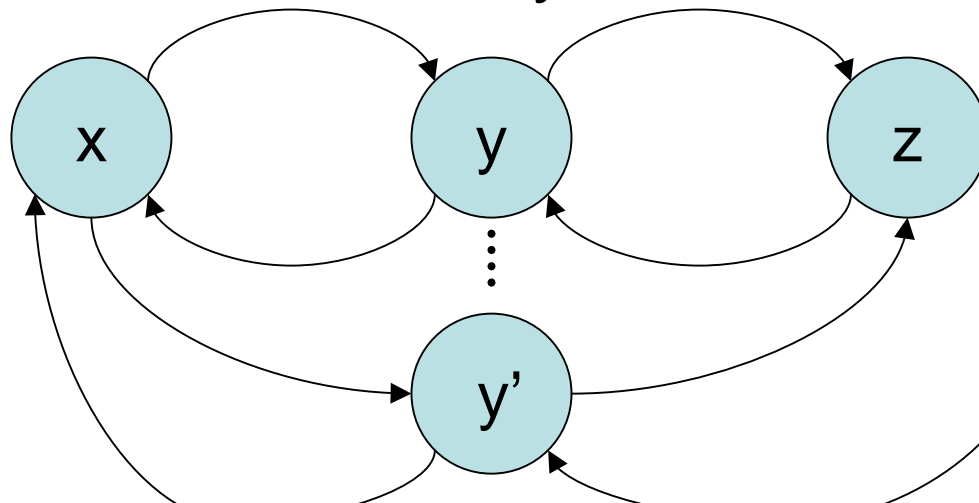


- Every node maintains a **certificate** that binds the node id to a public key
- Node x includes a **nonce** in the challenge
- The auditee sends back the nonce and digitally **signs** the response
- Node x checks the signature and the nonce before accepting the reply

**How Can We Do This Anonymously?**

# Use an Anonymizer Node

- Good node x wants to audit node z via y
  - Case 1: z is malicious, y is correct
  - Case 2: z is malicious, y is malicious
  - Case 3: z is correct, y is correct
  - Case 4: z is correct, y is malicious



**How do we know if z should pass or fail the audit?**



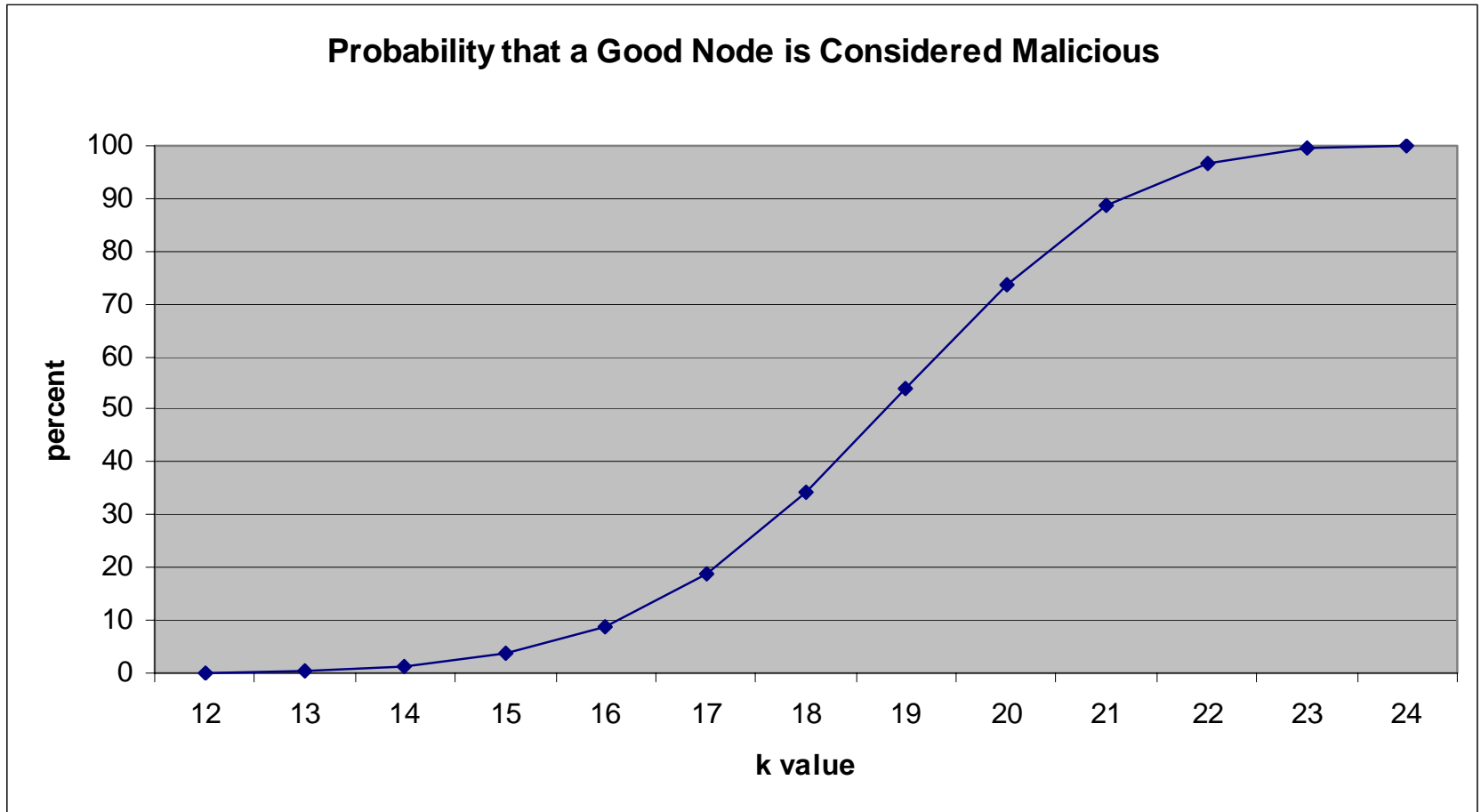
# Dissing a Good Node

- Probability that a good node is considered malicious (Binomial Distribution)
- Node considered malicious if it answers fewer than  $k$  out of  $n$  challenges correctly:

$$\sum_{i=0}^{k-1} \binom{n}{i} (1-f)^i f^{n-i}$$

- Example, assume  $f = .2$ ,  $n = 24$ ,  $k = 12$
- Probability is less than 0.02%

# What If We Vary k?



# Malicious Node Passing an Audit

- $r$  is the overload ratio
- $c$  is the probability a malicious node answers
- For each challenge, four cases
  - With probability  $f$ , the anonymizer is colluding and the malicious node passes
  - $(1-f)c/r$ , random response includes auditor and malicious node passes
  - $(1-f)c(1-1/r)$ , random response does not include auditor and malicious node fails
  - $(1-f)(1-c)$ , malicious node does not respond

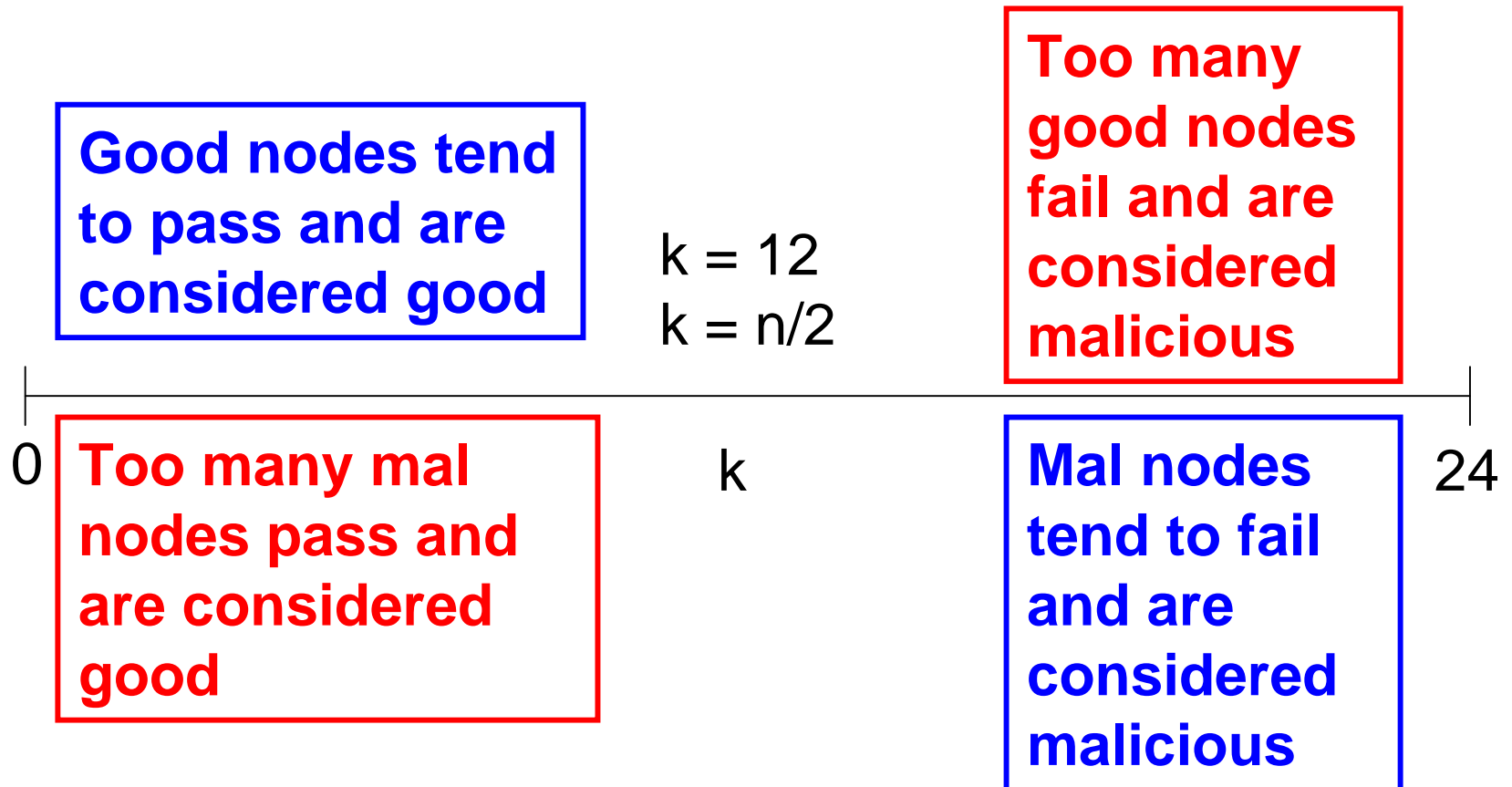
$$\sum_{i=k}^n \binom{n}{i} [f + (1-f)c/r]^i [(1-f)(1-c)]^{n-i}$$

# Malicious Node Passing an Audit

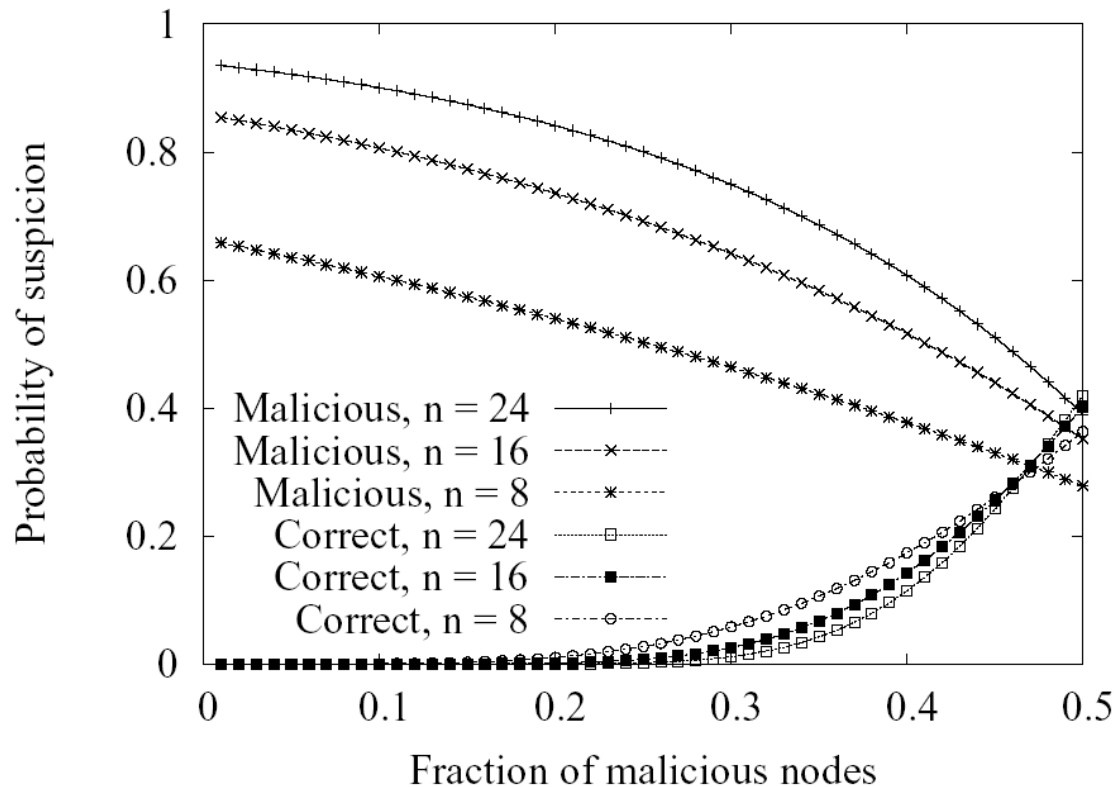


- A malicious node passes an audit with probability 0.034  
 $f = .20, n = 24, k = 12, r = 1.2$
- A malicious node fails an audit with probability 0.966
- A good node passes an audit with probability .9998 (as we previously saw)

# Choosing the k Value



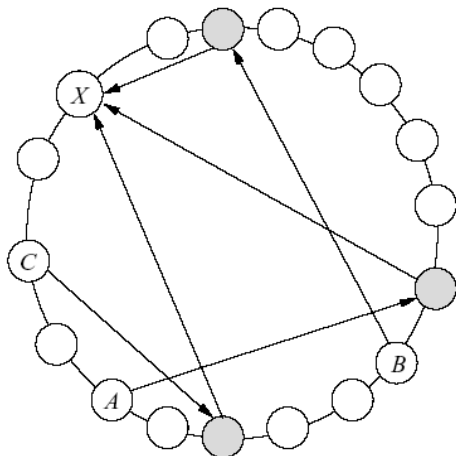
# Marking Malicious/Correct Suspicious



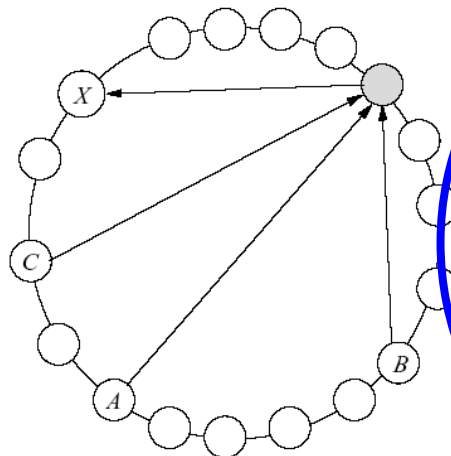
- More malicious nodes make it harder to detect them
- Correct nodes will also be marked as malicious
- Parameters
  - $k = n/2$
  - $r = 1.2$

# Picking the Anonymizer Node

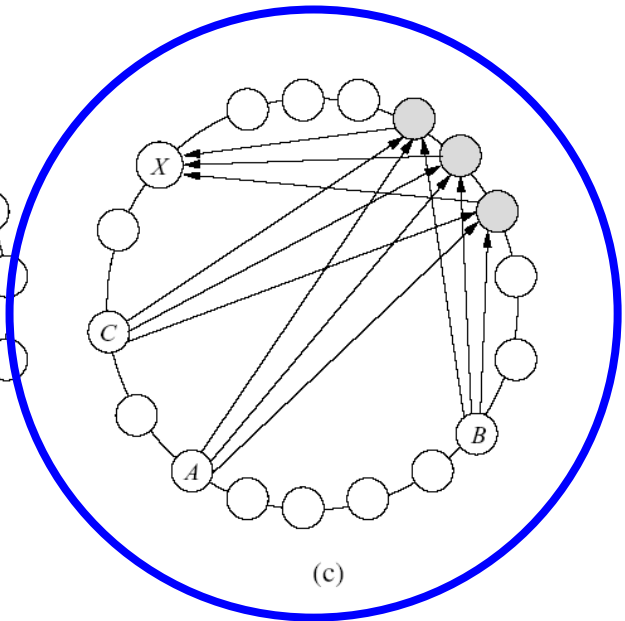
- (a) Randomly
- (b) Node Closest to  $H(x)$
- (c) Random Node Among the  $L$  Closest to  $H(x)$



(a)



(b)



(c)

# Evaluation Questions



- How serious is the eclipse attack on structured overlays?
- How effective is the PNS defense?
- Is degree bounding a more effective defense?
- How does it effect PNS performance?
- Is distributed auditing effective and efficient at bounding node degrees?




# Experimental Setup



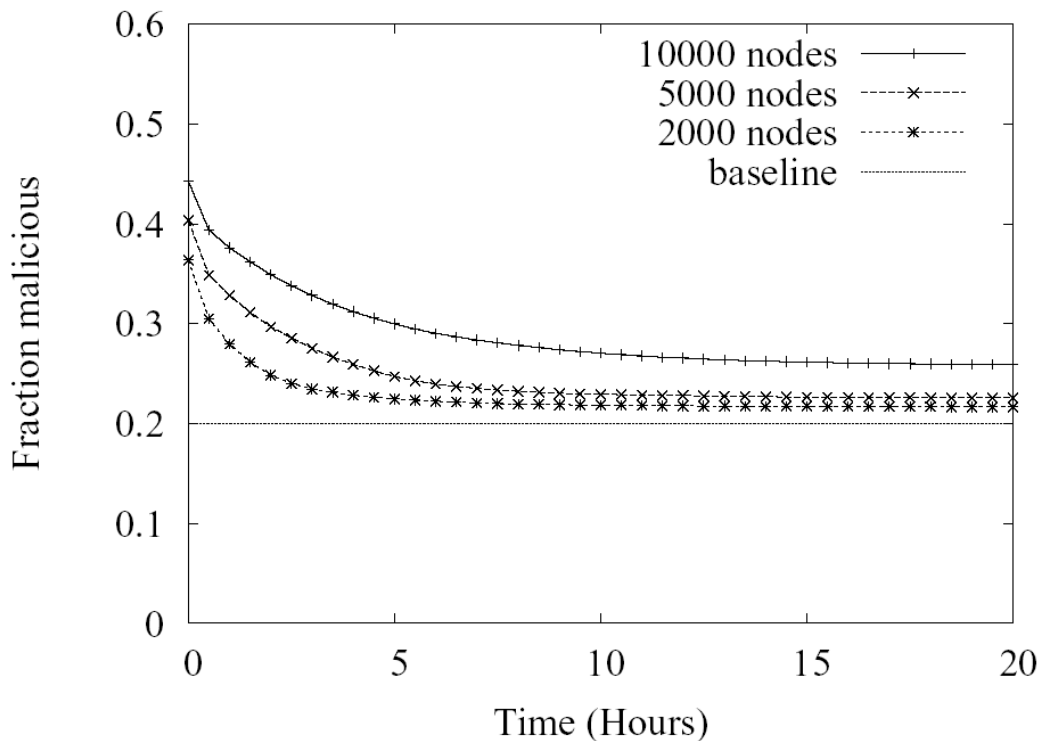
- MSPastry
  - GT-ITM transit stub network topology
  - GT-ITM topology has a good separation of nodes in the delay space
  - Pair wise latency values for up to 10,000 real Internet nodes obtained with King tool
  - Pastry settings:  $b = 4$ ,  $l = 16$ ,  $f=0.2$

# Know Your Enemy



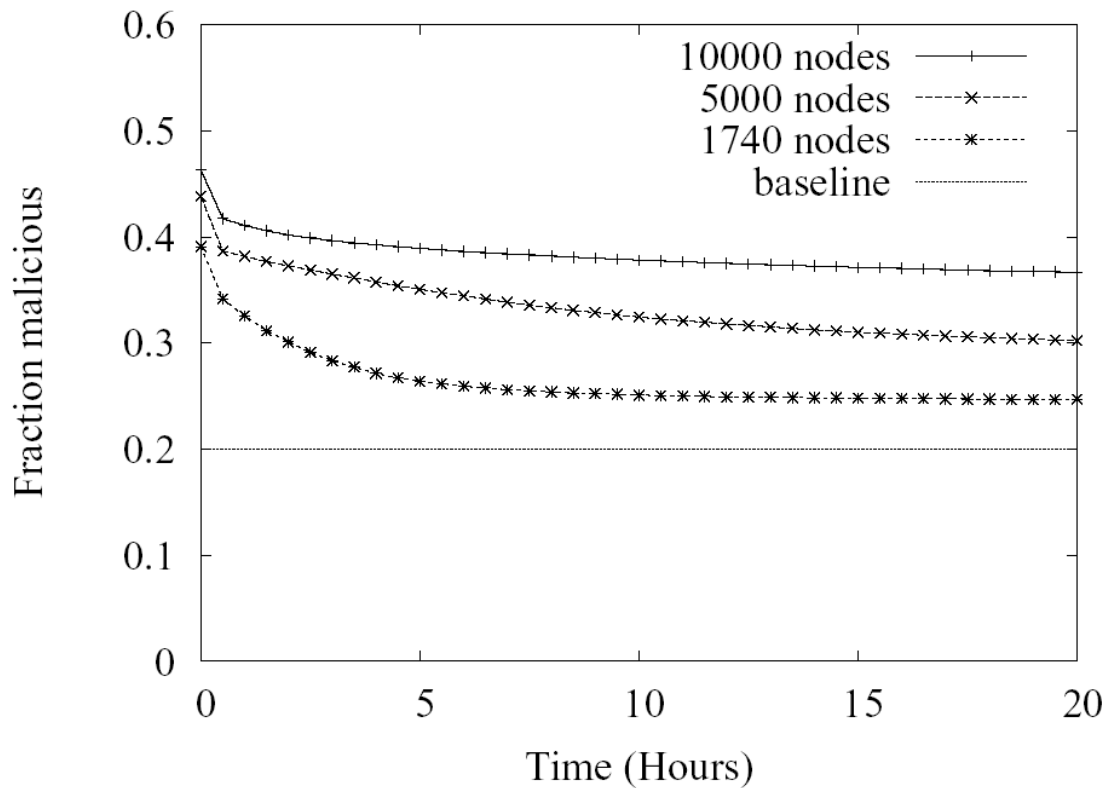
- Fraction of malicious nodes = .2
- Collude to maximize the number of router table entries referring to malicious nodes
- Malicious nodes misroute join messages of correct nodes to each other
- Malicious nodes set their routing tables to refer to good nodes whenever possible

# Effectiveness of PNS Defense

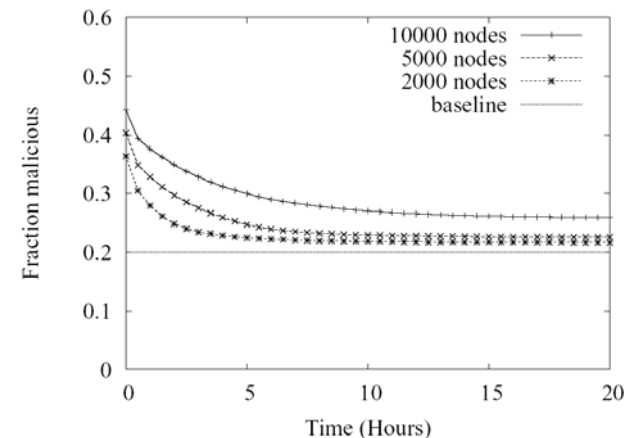


- Malicious fraction in top row drops from 78% to 41% for a 10,000 node overlay
- PNS not as effective in large overlays

# PNS with King Latencies

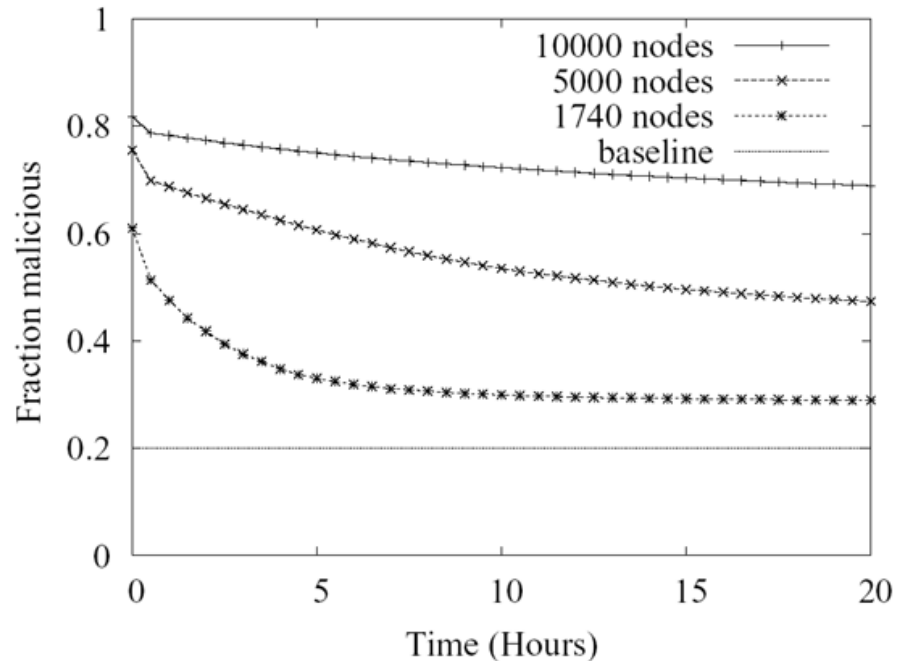
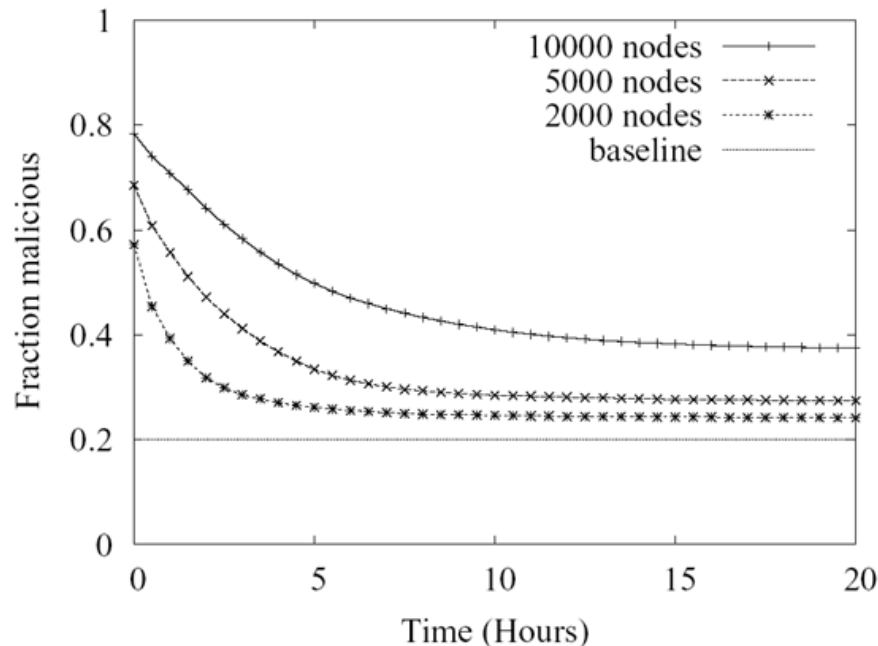


PNS less effective because a large amount of nodes are in the same delay band



# Top Row Comparison

- Fraction of malicious nodes in the top row of a correct node's routing table
  - GT-ITM (Left), King Latencies(Right)



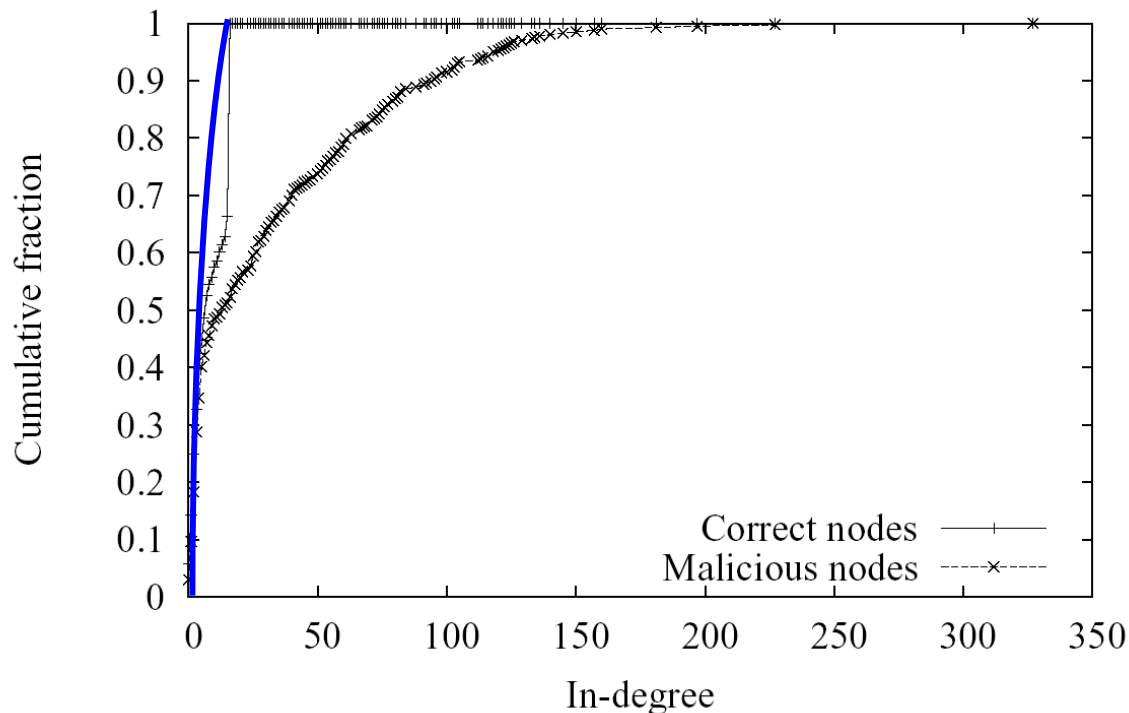
# Auditing Parameters



- Neighbor nodes randomly audited every 2 minutes (staggered)
- It takes 24 challenges to audit a node
- 2000 node simulation
- Churn: 0%, 5%, 10%, 15% per hour
- Target environment is low to moderately high churn

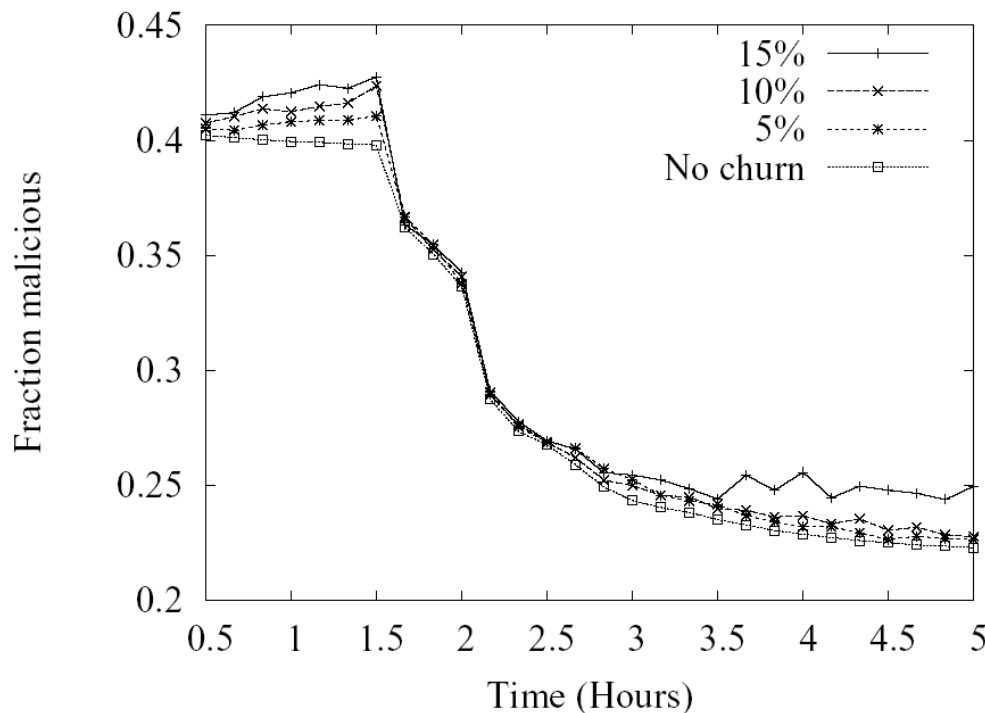
# In-Degree Distribution

- Before auditing has started, malicious nodes are able to obtain high in-degrees
- After 10 hours of operating with auditing...(Blue curve)



# Reducing Fraction of Malicious Nodes

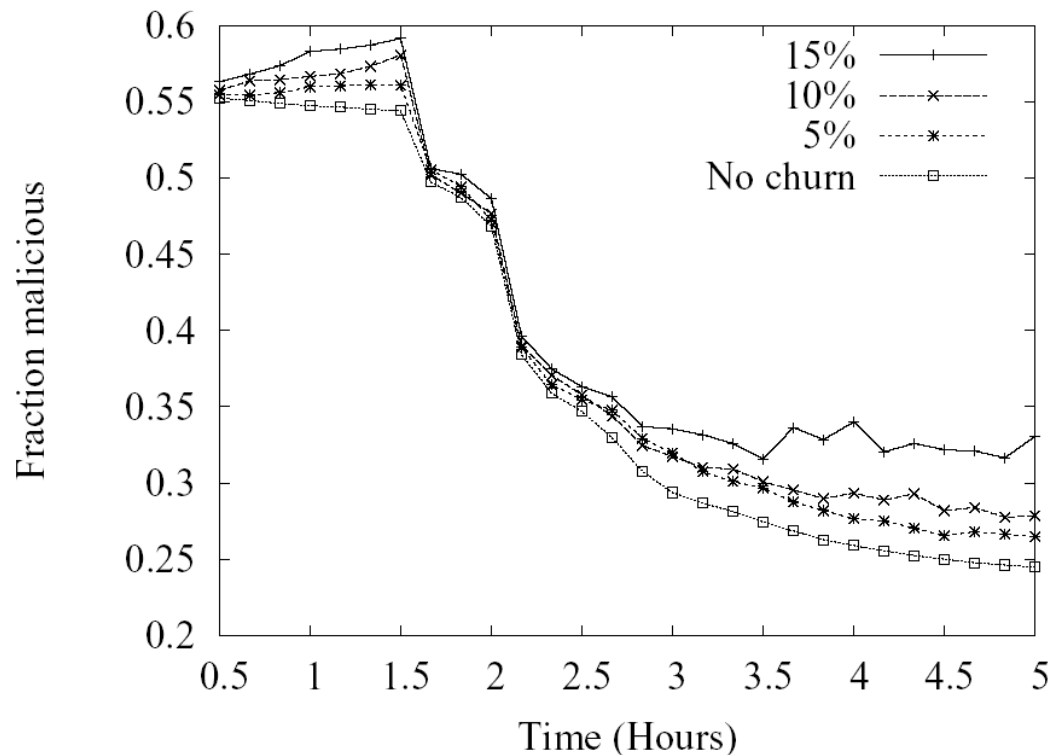
- Auditing starts 1.5 hours into simulation
- Correct nodes always enforce in-degree bound of 16 per row



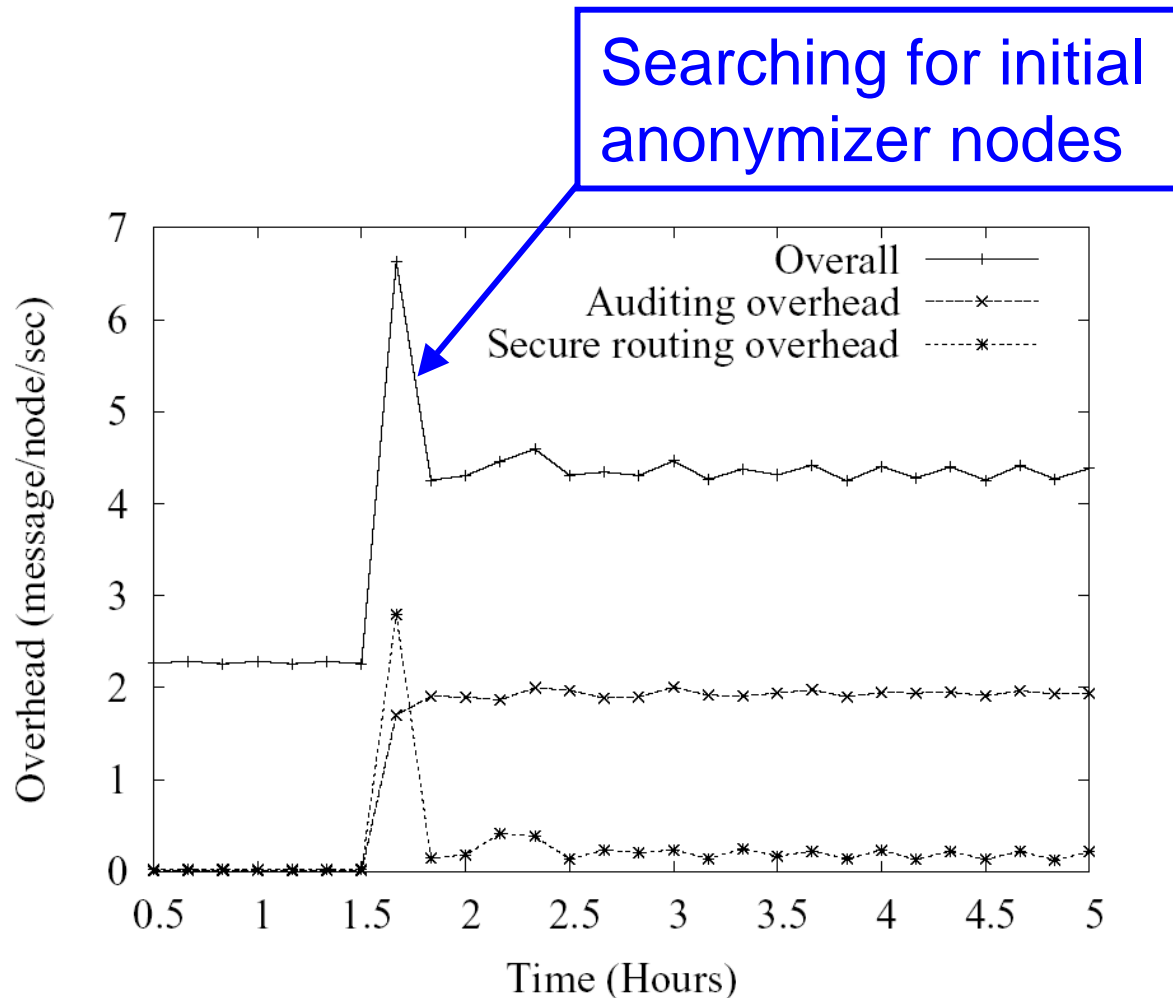


# Reducing Fraction of Malicious Nodes

- Top Row Analysis
  - Higher churn requires more auditing



# Communication Overhead of Auditing



# How Did They Do?



- How serious is the eclipse attack on structured overlays? **Very Bad!**
- How effective is the PNS defense? **Poor!**
- Is degree bounding a more effective defense? **Depends...but looks good**
- How does it effect PNS performance? **Depends**
- Is distributed auditing effective and efficient at bounding node degrees? **Yes!**

# Further Issues...



- Limitations of auditing
- Adversary response strategy
- How to incorporate into overlays based on supernode structures?
- Auditing in unstructured overlays?
- **Still vulnerable to localized attack!**