DUO-Onions and Hydra-Onions

Jan Iwanik, Marek Klonowski and Mirek Kutyłowski

Wrocław University of Technology, Poland

Conference on Communications and Multimedia Security
2004

Why do we need anonymity ?

- business to business communication
- consumer protection
- privacy protection
- economic and political security of a country

Possible dangers

- anonymity can be used for good and evil purposes

Applications of Protocols Providing Anonymity

- anonymous communication
- anonymous access to databases
- anonymous browsing
- anonymous file sharing

Target

- messages can be kept secret (easy)
- keep secret who is communicating with whom
  how to hide that two parties are communicating?

Techniques that Provide Anonymity

- MIXes - David Chaum ’81
- DC-networks -David Chuam ’85
- Onions - Rackoff and Simon ’91

Onions

- core of practical systems:
  - BABEL,
  - ONION ROUTING,
  - TOR
- scalable, fully distributed, no a priori infrastructure

Onions

- core of practical systems:
  - BABEL,
  - ONION ROUTING,
  - TOR
- scalable, fully distributed, no a priori infrastructure
- sometimes the same idea is used for evil purposes: hiding a source of an attack

Onions

If A wants send a message m to server B

- A chooses at random λ intermediate nodes $J_1, \ldots, J_\lambda$;
- A creates an onion:

  $O :=$

$$\mathrm{Enc}_B(m)$$

Onions

If A wants send a message m to server B

- A chooses at random λ intermediate nodes $J_1, \ldots, J_\lambda$;
- A creates an onion:

  $O :=$
  $$\mathrm{Enc}_{J_\lambda}(\mathrm{Enc}_B(m), B)$$

Onions

If A wants send a message m to server B

- A chooses at random λ intermediate nodes $J_1, \ldots, J_\lambda$;
- A creates an onion:

  $O :=$

$$\mathrm{Enc}_{J_{\lambda-1}}(\mathrm{Enc}_{J_\lambda}(\mathrm{Enc}_B(m), B), J_\lambda)$$

Onions

If A wants send a message m to server B

- A chooses at random λ intermediate nodes $J_1, \ldots, J_\lambda$;
- A creates an onion:

  $O :=$
  $\mathrm{Enc}_{J_1}(\ldots(\mathrm{Enc}_{J_{\lambda-1}}(\mathrm{Enc}_{J_\lambda}(\mathrm{Enc}_B(m), B), J_\lambda), J_{\lambda-1})\ldots, J_2)$.

Processing an Onion

If A wants send a message m encrypted as O to server B

- A sends onion O to J_1

Processing an Onion

If A wants send a message m encrypted as O to server B

- A sends onion O to J_1
- J_1 decrypts O and obtains some (O', J_2)

Processing an Onion

If A wants send a message m encrypted as O to server B

- A sends onion O to J_1
- J_1 decrypts O and obtains some (O', J_2)
- J_1 sends O' to J_2

Processing an Onion

If A wants send a message m encrypted as O to server B

- A sends onion O to J_1
- J_1 decrypts O and obtains some (O', J_2)
- J_1 sends O' to J_2
- J_2 decrypts ..

Processing an Onion

If A wants send a message m encrypted as O to server B

- A sends onion O to J_1
- J_1 decrypts O and obtains some (O', J_2)
- J_1 sends O' to J_2
- J_2 decrypts ..
- J_2 sends .. to J_3

Processing an Onion

If A wants send a message m encrypted as O to server B

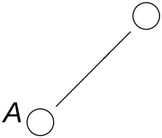
- A sends onion O to J_1
- J_1 decrypts O and obtains some (O', J_2)
- J_1 sends O' to J_2
- J_2 decrypts ..
- J_2 sends .. to J_3
- ...

Route of an Onion

single onion

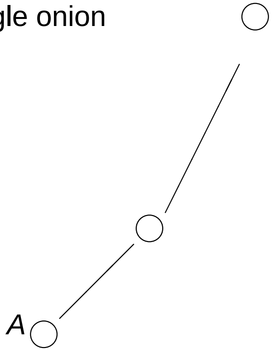
Route of an Onion

single onion



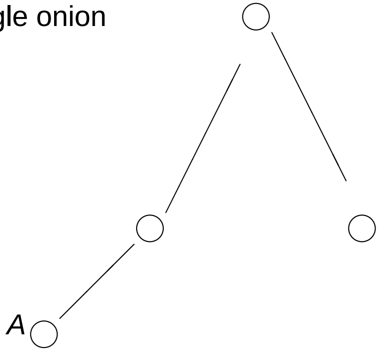
Route of an Onion

single onion



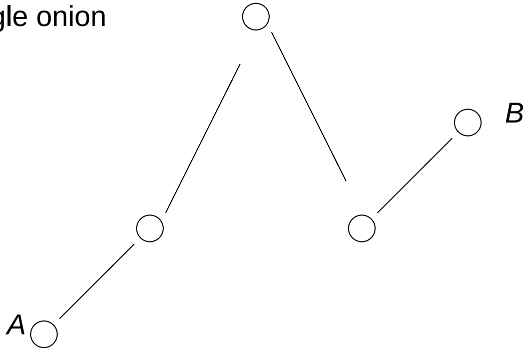
Route of an Onion

single onion



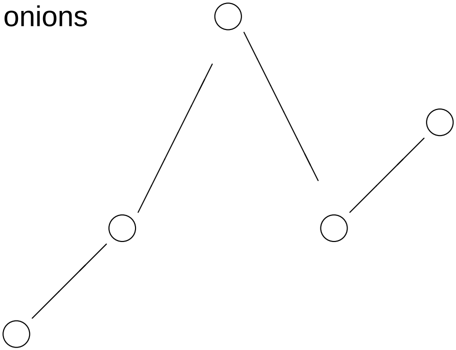
Route of an Onion

single onion



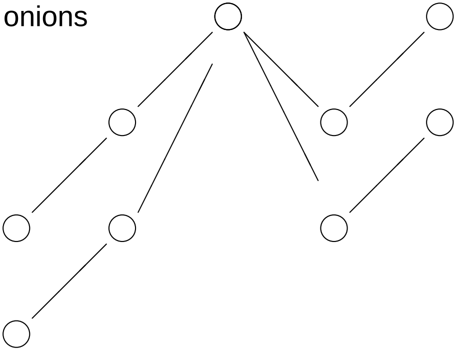
Onions at Work

many onions



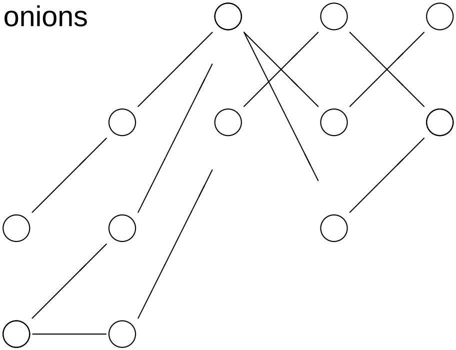
Onions at Work

many onions



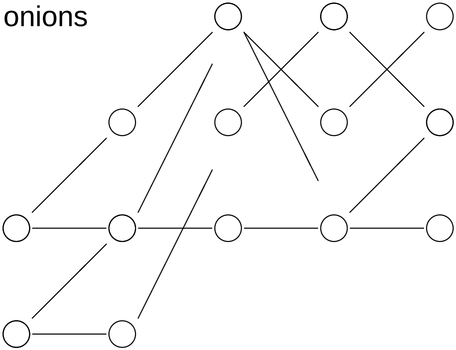
Onions at Work

many onions



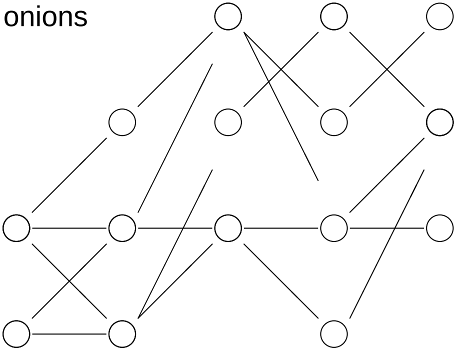
Onions at Work

many onions



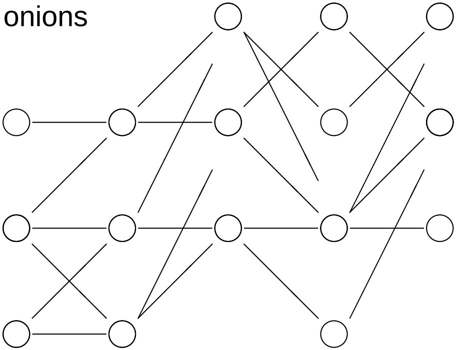
Onions at Work

many onions



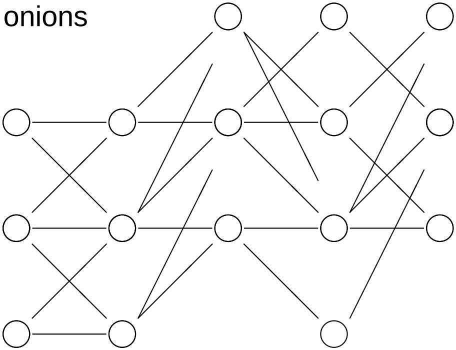
Onions at Work

many onions



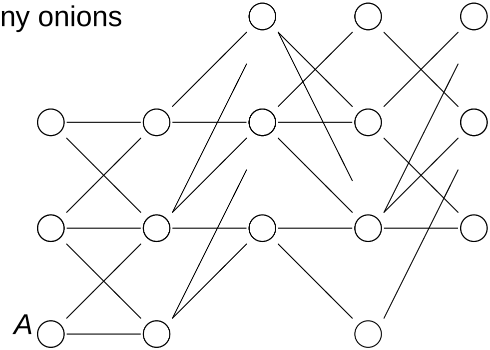
Onions at Work

many onions



Onions at Work

many onions



destination of the message starting at *A*?

Viewpoint of an External Observer

- no relationship can be derived between messages entering a node and leaving a node at the same time
  (probabilistic encryption, padding, ... have to be used)

Viewpoint of an External Observer

- no relationship can be derived between messages entering a node and leaving a node at the same time
  (probabilistic encryption, padding, ... have to be used)
- but: transmitting a message from a node to another node can be detected

Security of Onions - Problem Areas

1. breaking anonymity by eavsdropping and traffic analysis

Security of Onions - Problem Areas

1. breaking anonymity by eavsdropping and traffic analysis
2. breaking anonymity – as before + inserting, deleting, modifying, delaying ... messages

Security of Onions - Problem Areas

1. breaking anonymity by eavsdropping and traffic analysis
2. breaking anonymity – as before + inserting, deleting, modifying, delaying ... messages
3. **random transmission faults**

Security of Onions - Problem Areas

1. breaking anonymity by eavsdropping and traffic analysis
2. breaking anonymity – as before + inserting, deleting, modifying, delaying ... messages
3. **random transmission faults**
4. **transmission faults by an adversary**

Security of Onions - Problem Areas

1. breaking anonymity by eavsdropping and traffic analysis
2. breaking anonymity – as before + inserting, deleting, modifying, delaying ... messages
3. **random transmission faults**
4. **transmission faults by an adversary**

- problems 1,2 - some results and techniques are known
- **not concerned so far, this paper**

Adversaries

Adversary wants to **determine any <u>nontrivial</u> relation between the senders and receivers** and/or **break the traffic**
Different models of an adversary:

passive adversary :

- an adversary can monitor the whole traffic, or
- only a fraction of connections may be traced at each moment

Adversaries

Adversary wants to **determine any nontrivial relation between the senders and receivers** and/or **break the traffic**

Different models of an adversary:

passive adversary :

- an adversary can monitor the whole traffic, or
- only a fraction of connections may be traced at each moment

active adversary : may influence the traffic

- non-adaptive (an attack cannot be adapted to the traffic observed), or
- adaptive

Adversary Model is Important!

Required path length in different models. Let n be a number of messages.

An adversary can monitor the whole traffic:

- no security proof for the original protocol
- modified version of the protocol (routing in growing groups)
  Rackoff, Simon, FOCS'91, for $\lambda \approx \log^{11} n$,
  Czumaj, Kutyłowski, SODA'98, for $\lambda = O(\log^2 n)$

Only a fraction of connections may be traced

- Berman, Fiat, Ta-Shma, FC'2004, for $\lambda = O(\log^4 n)$
- Gomułkiewicz, Klonowski, Kutyłowski, ISC'2004, for $\lambda = \Theta(\log n)$

Server Failures

- a long path makes failure of delivery more probable,
- no detours can be applied to avoid failure nodes — at least for the original onions

$\Rightarrow$ anonymity at a price of service quality

How to Cope with Servers Failures ?

problem case: If $n/\log n$ out of n servers are down and the length of the paths is $\lambda = \log n$, then each packet gets lost with a constant probability.

a simple solution: Send the same message many times via independant paths.
disadvantage: communication overhead

DUO-Onions Protocol

Situation: An adversary destroys some number of servers (of his choice) to break communication with onions.

Countermeasure:

- at each step two servers can be used as the next hop server on the path,
  but each server sends a message **to exactly one of them**,
- encoding of an onion is modified.

Result: (Exponentially) better probability of delivery than through sending the same message through many paths.

DUO-Onions -Construction Details

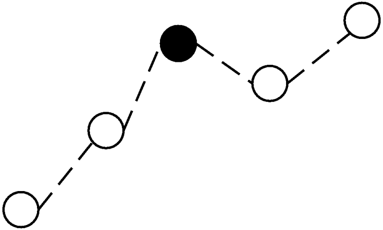
- For each step i two servers $J_{i,1} \neq J_{i,2}$ are chosen.
- encoding:

$$\begin{aligned}
\mathcal{DO}_\lambda = &\ \left(\mathsf{Enc}_B(k_{\lambda+1}), \mathsf{SEn}_{k_{\lambda+1}}(m, r_{\lambda+1})\right)\ , \\
\mathcal{DO}_i = &\ \left(\mathsf{Enc}_{J_{i,1}}(k_{i+1}, 1), \mathsf{Enc}_{J_{i,2}}(k_{i+1}, 2), \right. \\
&\qquad \left. \mathsf{SEn}_{k_{i+1}}(J_{i+1,1}, J_{i+1,2}, \mathcal{DO}_{i+1}, r_{i+1})\right) \quad \text{for } i < \lambda\ , \\
\mathcal{DO} = &\ \mathcal{DO}_1\ .
\end{aligned}$$

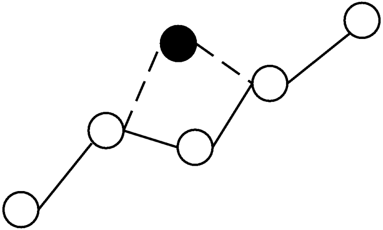
where SEn - symetric encryption scheme.

Instead of 2 servers we can choose K alternative servers at each step.

DUO-Onions at Work



DUO-Onions at Work



DUO-Onions versus Regular Onions Sent Many Times

advantages
- much higher probability of delivery,
- faster reaction to faults, faster delivery,

disadvantages size of an onion increases

HYDRA-Onions

- Adaptive adversary wants to block delivering a particular message. The adversary controls a constant fraction of servers and links between them at each moment.
- LET the countermeasures be also dynamic!

HYDRA-Onions - Idea

- We send a stream of k paths with messages encoding the same m.

HYDRA-Onions - Idea

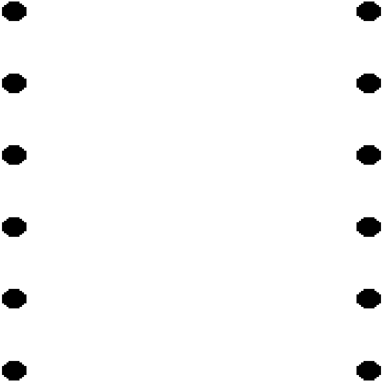
- We send a stream of k paths with messages encoding the same m.
- At each moment we should have k subonions encoding m.

HYDRA-Onions - Idea

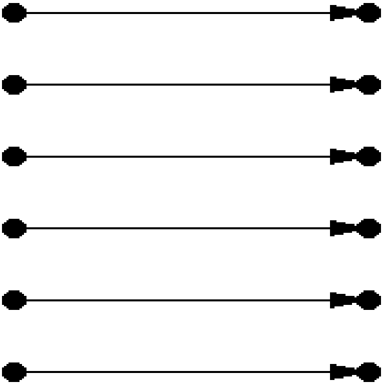
- We send a stream of k paths with messages encoding the same m.
- At each moment we should have k subonions encoding m.
- If an adversary kills some of the subonions, a mechanism of HYDRA-Onions enables the stream to regenerate quickly:

each intermediate server sends the message to the next server on its path
and
to another server on a randomly chosen path of the same stream

HYDRA-Onions - Idea

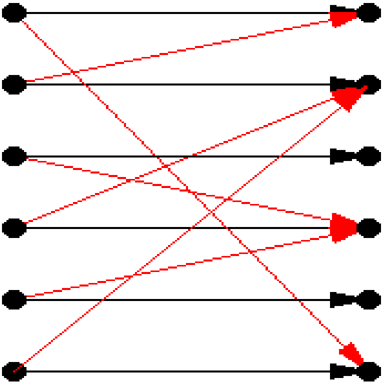


HYDRA-Onions - Idea



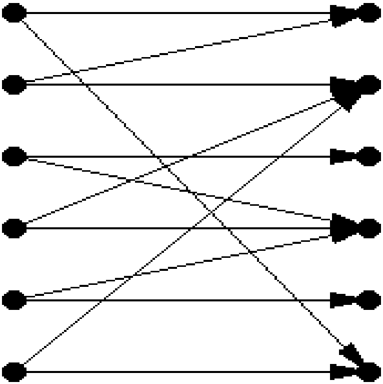
basic links

HYDRA-Onions - Idea

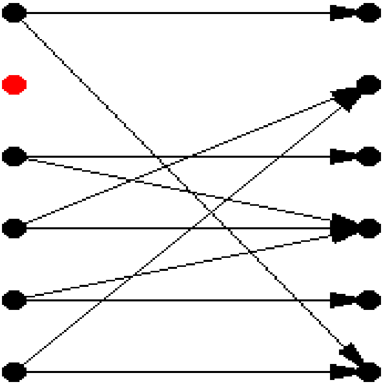


additional links

HYDRA-Onions - Idea

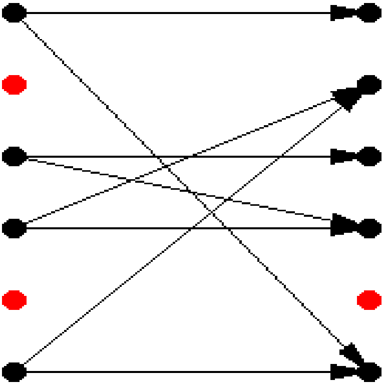


HYDRA-Onions - Idea



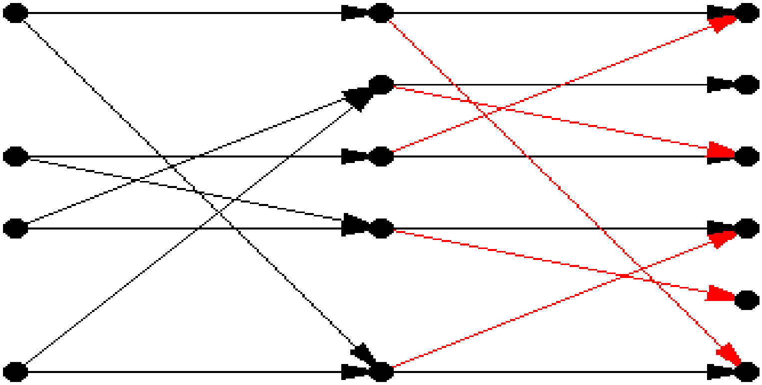
one server is blocked

HYDRA-Onions - Idea



two servers are blocked

HYDRA-Onions - Idea



stream regeneration during two steps

HYDRA-Onions -Idea

- an adversary has to “catch” all k streams simultaneously
- otherwise the strem regenerates fast and again k paths exist

reason: random graphs are expanders with high probability

HYDRA-Onions -Construction for $k = 3$

$$\mathcal{RO}_\lambda = \big(\mathsf{Enc}_B(k_{\lambda+1}), \mathsf{SEn}_{k_{\lambda+1}}(m, r_{\lambda+1})\big)$$

HYDRA-Onions -Construction for $k = 3$

$$
\begin{aligned}
\mathcal{RO}_\lambda = &\ \big(\mathrm{Enc}_B(k_{\lambda+1}), \mathrm{SEn}_{k_{\lambda+1}}(m, r_{\lambda+1})\big) \\
\mathcal{RO}_i = &\ \Big(\mathrm{Enc}_{J_{i,1}}(k_{i+1,1}, r_{i+1,1}), \\
&\quad \mathrm{Enc}_{J_{i,2}}(k_{i+1,2}, r_{i+1,2}), \\
&\quad \mathrm{Enc}_{J_{i,3}}(k_{i+1,3}, r_{i+1,3}),
\end{aligned}
$$

HYDRA-Onions -Construction for $k = 3$

$$\mathcal{RO}_\lambda = \left(\mathrm{Enc}_B(k_{\lambda+1}), \mathrm{SEn}_{k_{\lambda+1}}(m, r_{\lambda+1})\right)$$

$$\begin{aligned}\mathcal{RO}_i = \Big(&\mathrm{Enc}_{J_{i,1}}(k_{i+1,1}, r_{i+1,1}),\\ &\mathrm{Enc}_{J_{i,2}}(k_{i+1,2}, r_{i+1,2}),\\ &\mathrm{Enc}_{J_{i,3}}(k_{i+1,3}, r_{i+1,3}),\\ &\mathrm{SEn}_{k_{i+1,1}}(J_{i+1,1}, J_{i+1,a(1)}, k'_{i+1}),\\ &\mathrm{SEn}_{k_{i+1,2}}(J_{i+1,2}, J_{i+1,a(2)}, k'_{i+1}),\\ &\mathrm{SEn}_{k_{i+1,3}}(J_{i+1,3}, J_{i+1,a(3)}, k'_{i+1}),\end{aligned}$$

HYDRA-Onions -Construction for $k = 3$

$$
\begin{aligned}
\mathcal{RO}_\lambda = \quad & \big(\mathsf{Enc}_B(k_{\lambda+1}), \mathsf{SEn}_{k_{\lambda+1}}(m, r_{\lambda+1})\big) \\
\mathcal{RO}_i = \quad & \Big(\mathsf{Enc}_{J_{i,1}}(k_{i+1,1}, r_{i+1,1}), \\
& \quad \mathsf{Enc}_{J_{i,2}}(k_{i+1,2}, r_{i+1,2}), \\
& \quad \mathsf{Enc}_{J_{i,3}}(k_{i+1,3}, r_{i+1,3}), \\
& \quad \mathsf{SEn}_{k_{i+1,1}}(J_{i+1,1}, J_{i+1,a(1)}, k'_{i+1}), \\
& \quad \mathsf{SEn}_{k_{i+1,2}}(J_{i+1,2}, J_{i+1,a(2)}, k'_{i+1}), \\
& \quad \mathsf{SEn}_{k_{i+1,3}}(J_{i+1,3}, J_{i+1,a(3)}, k'_{i+1}), \\
& \quad \mathsf{SEn}_{k'_{i+1}}(\mathcal{RO}_{i+1})\Big) \quad \text{for } i < \lambda \\
\mathcal{RO} = \quad & \mathcal{RO}_1
\end{aligned}
$$

Major Problem

- streams of messages encoding *m*:
  may the additional links betray the structure of a path and **reveal where to attack**??

Major Problem

- streams of messages encoding *m*:
  may the additional links betray the structure of a path and **reveal where to attack**??
- certainly – the number of additional links should be kept as small as possible (less links, less information for an adversary)
  well, 1 additional link is enough for expansion features

Chances of an Adversary

- if an adversary chooses a constant fraction of links at random and blocks them, then with probability ... a stream dies — *easy calculations*

Chances of an Adversary

- if an adversary chooses a constant fraction of links at random and blocks them, then with probability ... a stream dies — *easy calculations*
- **can an adversary design a clever strategy to improve his chances?**

Chances of an Adversary

- if an adversary chooses a constant fraction of links at random and blocks them, then with probability ... a stream dies — *easy calculations*
- **can an adversary design a clever strategy to improve his chances?**
- **a strategy can be focused not only on killing a stream but also on detecting it and killing at the next move**

Noga Alon's Lemma [2001]

there are limitations on clever strategies

Lemma

For every fixed $\varepsilon > 0$*, and every fixed integer* $t > 0$*, and for any graph* G *with* n *vertices and at least* εn^2 *edges, the number of subgraphs of* G *isomorphic to* $K_{t,t}$ *(bipartite complete graph with* t *vertices on each side) is at least:*

$$\frac{1}{2}\binom{n}{t}\binom{n}{t}(2\varepsilon)^{t^2}$$

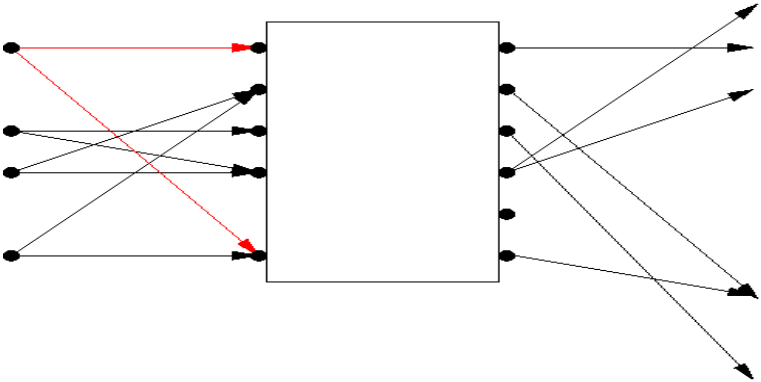
Consequences of Alon's Lemma

For every fixed $\varepsilon > 0$, and every fixed integer $t > 0$, and for any graph G with n vertices and at least εn^2 edges, the number of subgraphs of G isomorphic to $K_{t,t}$ (bipartite complete graph with t vertices on each side) is at least:

$$\frac{1}{2}\binom{n}{t}\binom{n}{t}(2\varepsilon)^{t^2}$$

- graph G – links not monitored by an adversary
- $K_{t,t}$ in G – a subset of nodes within which an adversary has NO information – so called **crossover structure**
- the lemma says: no matter how clever is the adversary in determining G, a large number of crossover structures emerge

Why a Crossover is Bad for an Adversary?



which link to disrupt?

Consequences

- for the servers $J_{t,1}, J_{t,2}, J_{t,3}$ holding m at step t and servers $J_{t+1,1}, J_{t+2,2}, J_{t+3,3}$ holding m at step $t+1$:

a crossover of size 2 occurs with a constant probability

Consequences

- for the servers $J_{t,1}, J_{t,2}, J_{t,3}$ holding m at step t and servers $J_{t+1,1}, J_{t+2,2}, J_{t+3,3}$ holding m at step $t+1$:

a crossover of size 2 occurs with a constant probability

- if such a crossover occurs, then the links between $J_{t,1}, J_{t,2}, J_{t,3}$ and $J_{t+1,1}, J_{t+2,2}, J_{t+3,3}$ seen by the adversary do not form a connected graph

the adversary does not know that they belong together!

Consequences

- for the servers $J_{t,1}, J_{t,2}, J_{t,3}$ holding m at step t and servers $J_{t+1,1}, J_{t+2,2}, J_{t+3,3}$ holding m at step $t+1$:

a crossover of size 2 occurs with a constant probability

- if such a crossover occurs, then the links between $J_{t,1}, J_{t,2}, J_{t,3}$ and $J_{t+1,1}, J_{t+2,2}, J_{t+3,3}$ seen by the adversary do not form a connected graph

the adversary does not know that they belong together!

- for $K > 3$ the chances that the links seen by the adversary to see a disconnected graph from links belonging to the same stream grow substantially

Future work

- a deeper analysis of graph-theoretic aspects,
- security proofs regarding traffic analysis.

Thanks for your attention!