



Securing Peer-to-Peer Content Sharing Service from Poisoning Attacks

R. Chen, E.K. Lua, J. Crowcroft

W. Guo, L. Tang, Z. Chen



北京大学



Background

- P2P content sharing service has grown in significance on the Internet, both in terms of the number of participating users and the traffic volume.
- Self-organization and self-maintenance.
- Poisoning attacks in the P2P content sharing service have become a serious security problem.



北京大学



Poisoning Attack

- **Attack process:**
 - **Corrupt the target file without changing the metadata**
 - **Inject a large number of poisoned files into the P2P content sharing system**
 - **Unsuspecting P2P users download the poisoned file into their own shared folders**
 - **Other P2P users may download later without knowing that the file has been poisoned**



北京大學



Signature Scheme

- **Why not simple digital signature?**
 - **Due to the lack of centralized trusted content distribution authorities in the P2P content sharing systems, the applicability of the signature scheme is questionable.**



北京大学



Related Work

- Related work:
 - **Reputation model:** penalized by the lack of reliable user cooperation.
 - **Others:** micropayment, repeat and compare, etc.
- Our consideration: file providers are the only sources to accurately distinguish poisoned files and verify the integrity of the requested files.



北京大学



Poisoning Defense

- **Main operations:**
 - Publish
 - Lookup and download





Publish (1/2)

- Underlying structure: DHT-based overlay
- Two kinds of identifiers:
 - **User identifier: chosen by hashing the participant's IP address.**
 - **File identifier: produced by hashing the filename.**



北京大学



Publish (2/2)

- A provider maps the information of file F onto the associated *maintainer* M .
- Redundancy mechanism:
 - $M_i = \text{successor}(\text{filename} \mid i), 1 \leq i \leq m$.



北京大學



Lookup and downloading (1/5)

- **Step 1 — Query:** A requestor U , who wants to acquire a specific file F , issues a query (i.e., F 's file identifier) towards the m associated maintainers in the system.
- **Step 2 — Response:** These m maintainers respond with the corresponding information of the requested file F .



北京大学



Lookup and downloading (2/5)

- **Step 3 — Filtering:** On harvesting these responses, the requestor mixes them and generates a list L consisting of $\langle VI, PI \rangle$ pairs, where the VI and PI denote the version and provider identifiers associated with the requested file F .
 - **Method 1:**
 - Percentage of poisoners does not exceed θ_1 ($0 \leq \theta_1 < 0.5$).
 - Each $\langle VI, PI \rangle$ pair existing in the generated list L for at least $|m \times (\theta_1 - 1)|$ times is authentic with high probability.
 - **Method 2:**
 - If a provider publishes more than θ_2 ($\theta_2 \geq 1$) versions of the requested file F , we filter the list L through removing all these $\langle VI, PI \rangle$ pairs associated with the provider.



北京大學



Lookup and downloading (3/5)

- **Step 4 — Selection:** the requestor's choice is biased towards versions with more providers — the probability of selecting a version is proportional to the number of the associated providers.



北京大学



Lookup and downloading (4/5)

- **Step 5—Downloading and verification:** The requestor starts downloading the data blocks of the selected version V from these associated providers in parallel.

Digest checking: verify the integrity of the selected version V while downloading.

Probabilistic verification scheme: To reduce the verification overhead, we verify randomly selected blocks instead of all blocks.

$$FPR = \begin{cases} \frac{C_{b-r}^v}{C_b^v} = \frac{(b-r)! \times (b-v)!}{(b-r-v)! \times b!} & \text{if } r + v \leq b \\ 0 & \text{if } r + v > b \end{cases}$$
$$\leq EFPR$$



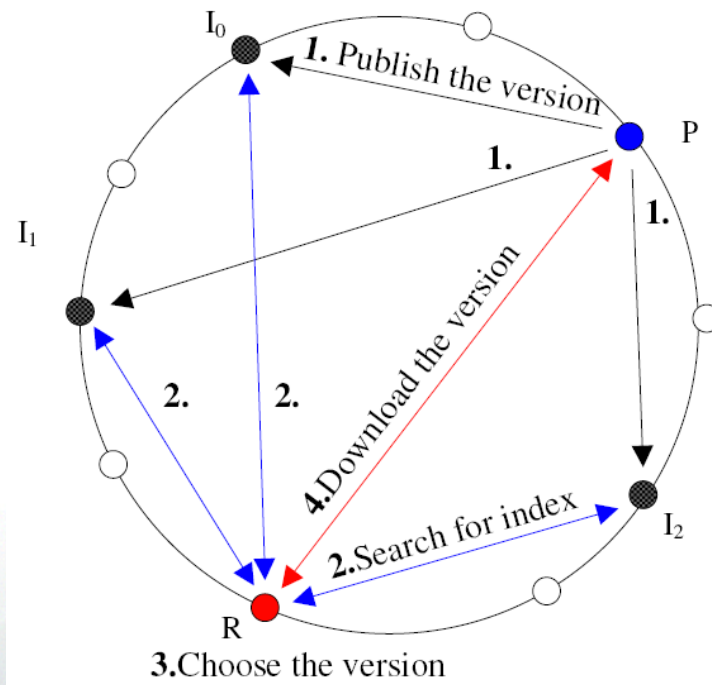
北京大學



Lookup and downloading (5/5)

- **Step 6 — Build:** After the downloading and verification, the requestor can build the complete requested file from these data blocks.

- **Summary:**



北京大学



Evaluation (1/2)

- **Simulation setup:**
 - Network model:
 - Overlay weaver toolkit
 - User model:
 - 2000 users including genuine users and poisoners
 - Content model:
 - 100 unique files
 - 500 different versions
 - Execution model:
 - 20 simulation cycles
 - 2000 lookup cycles



北京大學



Evaluation (2/2)

- The requestor generally needs to merely verify 30% of all blocks to achieve a low false positive rate of probabilistic verification.
- Only a minority of all the downloads will end up with downloading a poisoned version of the requested file.
- Our security framework can work well even in a highly malicious environment with 40% of all users being poisoners.
- The convergence time of our framework is very short, and it is able to efficiently defend against content poisoning in various scenarios.



北京大學



End

Q & A

