Red Rover

A collaborative approach to content filtering

Tommy Pauly & Richard Barnes
August 2022

Some network operators use content filtering, via DNS filtering and other network-based mechanisms, as a way to prevent their networks from being used to access content that violates their policies. For example, these networks will filter out malware or objectionable content.

Simultaneously, client devices and applications want to protect users from networks collecting sensitive and private information about their activity. Techniques here include encrypting TLS metadata, encrypting DNS traffic and selecting trusted DNS resolvers, and employing proxies or VPNs.

Although the mechanisms employed for user privacy and security can interfere with the mechanisms traditionally used for network-based content filtering, the objectives of the networks and clients are not fundamentally in conflict. Client devices and applications likely don't want their users exposed to bad content, users likely don't want to violate any network terms and conditions, and network operators usually don't want to collect private information to track users.

In this paper, we suggest an approach to designing collaborative solutions for network-informed content filtering.

A Collaborative Approach

If endpoints and network infrastructure work together, then the network's blocking objectives can be accomplished without wholesale exposure of user information. The endpoint needs to trust the network to provide a block list, and the network needs to trust the endpoint to enforce it.

This approach is already widely deployed in the form of "safe browsing" services, Google's Safe Browsing[1] being the flagship example. These services allow security providers to expose blocklists that are too large to be downloaded by a client, while still leaking minimal information about the client's interests. The novelty here is to allow the endpoint to dynamically discover a service that encodes the network operator's preferred block lists.

In order to build a solution for collaborative filtering, we would need to define two standardized mechanisms:

1. A discovery mechanism, by which the network can tell cooperating clients that a block-list service is offered, provide information on how to interact with the service. This mechanism also should include a way for the network to learn that a client is using the service.
2. A block-list service that operates in a similar fashion to Safe Browsing, that allows clients to learn about specific domains or URLs to filter based on network policy, without revealing what content the client is accessing.

[1] https://developers.google.com/safe-browsing/v4

Collaborative solutions are intended for cases where client devices and applications are cooperating in order to benefit the user — to make it easier for the user to avoid malicious content, and avoid the user accidentally violating terms of use on the network. This is likely a good fit for public networks that need to enforce terms of use, for example. Such networks likely are today only performing filtering at relatively basic levels — blocking specific DNS names and known bad IP addresses.

Networks that require deeper interception and more guarantees for enforcement, such as those that rely on TLS-decrypting firewalls, are out of scope for this kind of solution.

Solution Ideas

As described above, the solution space needs to include both a discovery mechanism to allow the endpoint and network to coordinate, and a service that provides blocking information to clients.

In order to implement discovery, the network would likely need to provide some URI to endpoints, which would indicate the safe browsing or similar service. There are several options that could leverage IETF standards, including:

- Add the service URI to the Captive Portal API[2] defined in the IETF CAPPORT working group, which allows networks to provide additional network information, bootstrapped using a DHCP or RA options.
- Add the service URI to Provisioning Domain Additional Information[3], which can also be advertised via RA options.

Then, when the client connects to the network, it will learn the URI of a service that can provide the network's block list.

Assuming this service uses something like the existing safe browsing protocol, the client would download an initial list of partial URL or domain hashes that should be considered for filtering. The act of downloading this initial list would allow the network to know that this client is aware of the filtering service, and intends to apply the filtering.

When an application on the client endpoint attempts to connect to a domain name or URL, the endpoint's safe browsing logic would first vet the domain name our URL against the network's safe browsing list, and only allow the application to connect if it is allowed by the safe browsing service.

A standardized version of a safe browsing service for this purpose would require further analysis to ensure that it can preserve user privacy as much as possible. For example, there have been proposals[4] for ways to improve safe browsing by using Private Information Retrieval.

[2] https://www.rfc-editor.org/rfc/rfc8908

[3] https://www.rfc-editor.org/rfc/rfc8801.html#name-provisioning-domain-additio

[4] https://eprint.iacr.org/2021/345.pdf

Circumvention and Legacy Endpoints

As with many blocking approaches[5], this approach can be circumvented by a malicious endpoint. This system arguably provides a harder assurance than approaches like DNS-based or SNI-based blocking, since the endpoint operating system has complete control over applications' access to the network. But if the endpoint operating system itself chooses not to enforce the block list, then it will not be applied. This includes legacy endpoints that lack the logic to apply blocking.

As a result, networks will need to selectively enable this approach for endpoints, based on some signal that the endpoint is ready and willing to participate. For example, the network might have a default posture of blocking proxies and public DoH endpoints, but allow connections from clients that have agreed to apply the network's blocking.

In an ideal world, the network would have a very strong assurance of the endpoint's willingness, for example, a remote attestation[6] rooted in secure hardware that the endpoint was running an operating system that applies the network's controls as expected. However, these technologies are not yet widely deployed, and involve tricky trust establishment problems. The network infrastructure would need to know how to verify attestations, and since the endpoint would be exposing fairly sensitive information to the network, it would likely require that the network authenticate itself more strongly than simply an ability to send a DHCP message.

Solutions such as the above, however, still provide the network with some usable signals. Continuing the example above, the network could deactivate DoH blocking for an endpoint when the endpoint connects to the safe browsing service. While this would still enable a malicious endpoint to unblock itself by taking this step and then not applying the blocklist, this is not a behavior that endpoint vendors would be likely to enable as a default. So circumvention of this type would be of a similar level of difficulty to, say, using Tor Browser or a VPN — but again, arguably harder because action needs to be taken at the operating system level.

Conclusion

User privacy and support for blocking access to unwanted content on networks do not fundamentally conflict, and can be addressed by collaborative solutions. These solutions are appropriate when the intentions of networks, endpoints, and users align.

More work needs to be done to standardize such approaches, specifically in defining privacy-preserving services to provide block lists. Taking on such work in the IETF would help address an area of conflict between client and network policies.

[5] https://www.rfc-editor.org/rfc/rfc7754

[6] https://datatracker.ietf.org/wg/rats/about/