Project Plan - Local Network Access

Document revision	Description	Author/Reviewers	Date
0.1	Initial Revision	Valentin Gosu	
		Kershaw Jang	2025-01-10
		Gregory Hess	2025-02-03
		Daniel Veditz	2025-04-09
		Randell Jesup	Mar 10, 2025
		Sunil Mayya	
0.2	Modified the following • Overview • Scope • MileStones • Assumptions	Sunil Mayya	2025-01-23
0.3	Modified the following • Added more details to the scope of Rollout • Aligned Milestones and Scope • Added Detailed Schedule	Sunil Mayya	2025-02-04
0.4	Rescoped the feature based on Chrome's latest update to de-prioratize PNA and opt for an alternative approach Local Network Access	Sunil Mayya	2025-02-27
0.5	Addressed Randell's comments	Sunil Mayya	2025-03-14
	Review	Daniel Veditz	2025-03-31
0.6	Addressed Dan's comments	Sunil Mayya	2025-04-02
	Review	Daniel Veditz	2025-04-09
1.0	Addressed Dan's comments	Sunil Mayya	2025-04-11

1.1	Added stricter Mixed Content Checks	Person	Date

Overview

This project is an effort to enhance our defence against attacks where the browser's vicinity to the user's local network is exploited. The class of attack involves a malicious website sending requests to devices/services located in the user's machine or local network.
Our initial approach to the project was to implement the Private Network Access (PNA) proposal. However, after 4 years of trial (and lots of reported site breakages) the chrome platform has decided to pursue this problem with a newer CORS-less solution - Local-Network-Access (LNA). Given this latest update from chromium, we would like to pursue a lighter version of the LNA and ship a basic defence for protecting users from such attacks.
Our long-term strategy should be to jointly collaborate with the wider web community and implement a standard way of handling this problem. Chrome's LNA would take time to be standardized, hence having a light-weight solution would ensure we deploy this feature sooner rather than catching up with Chrome.
Hence, we propose to implement a leaner version of LNA. LNA is aligned with Chrome's local network access approach for the most common use case. This strategy mitigates any potential web compatibility issues that might arise once the proposed solution is standardized.
LNA will be designed to be customizable by the users. It will incorporate settings to customize hosts, domains and wildcard matching that would make a particular origin local or public regardless of the IP address.

The following table presents use cases, along with the respective approaches taken by Chrome and Firefox.

Use Case	Chrome's Solution	Firefox's Approach
1. Blocking direct access from Public websites to device/services on user's host and local network	Categorise the IP addresses into *loopback, local, and public* IP AddressSpaces. If the request is crossing boundaries to a more private address space, prompt the user for action. Introduce permission prompt and let the user grant/deny the access	Follow Chrome's approach except have two separate permissions - one for accessing local host and another for accessing local network[1]
2. Allow public front-end websites to communicate with devices on user's	Skip Mixed Content Checks if:	Skip **Passive** Mixed Content Checks if:

[1] We will propose this approach to the specification as well.

network to avoid mixed content blocking	- The hostname is a private IP address literal (per RFC1918 etc.) OR - The hostname is a .local domain OR - The fetch() call is annotated withtargetAddressSpace: "local" option	- The hostname is a private IP address literal (per RFC1918 etc.) OR - The hostname is a .local domain OR - Origin explicitly configured by user in the about:config /configuration settings / enterprise settings

We've decided not to use the same approach for Use Case 2 right now, which means any changes requiring modifications to the web servers (modifications to the fetch API) are being postponed. Instead, we will adopt Chrome's solution for Use Case 2 after Chrome's LNA proposal has become more established or standardized. Our reasoning is to first address the most impactful use case and the part of the specification that is least likely to change.

Why Implement LNA Now?
We propose implementing this feature now to improve our public perception of a secure browser and fix a long standing security issue.

Goals

- Prevent attacks on users' local and private networks using Firefox as a tool.

Success Criteria and Metrics

- Access to the user's local network and localhost is blocked and not progressed until the user grants the permission.
- Collect telemetry data to compare access to user's local and private devices before and after implementing the feature.

Risks

Risk	Mitigation

This feature introduces a new permission prompt that the user might not understand and/or accidentally grant permission	Make the permission prompt very user-friendly and intuitive. As usual there will be visual indicators of the granted permission in the URL bar so the user can easily undo a mistake, as well as a management dialog grouped with other permissions in about:settings.
Users might be in an environment subjected to a wide variety of local network access leading to a lot of prompts. This could annoy users.	Leverage existing permission features like camera notifications to change the default settings from Ask to Always Allow or Always block. Additonally, provide users an option to disable this feature through configuration settings and prefs.
Websites could launch a DoS/eviltrap attack on a user by spamming them with local network access attempts	Block additional attempts to access local resources from a page after the first denial. Leverage existing UI template to remember the choice to permanently deny or allow local accesses.

Assumptions and Dependencies

1. PNA wont be implemented by google in the near future
2. The web standard for mitigating the aforementioned attacks is on similar lines as the current proposal by Chrome
3. We have dependency on the UI team for implementing some of the features described below.

If Assumption 1 changes this would mean additional work in terms of CORS handling. If assumption 2 changes then we might end-up behaving in a non-webcompat way

Existing local network defences across various applications:

- Brave
- MacOS

Scope of Work

In this section we first describe the feature's functionality in detail and later list the work breakdown.

LNA in Details

Concepts

IPAddressSpace Concept

This feature introduces a concept of IPAddressSpace. Each IP address can belong to one of the following categories - public, local, loopback, with "loopback" being the most private. The PNA specification describes in detail on how to categorize IP addresses into the aforementioned categories.

What is Local Network Access?

A request is a Local Network Request if the request's URL points to an IPAddressSpace that belongs to a more private network range than the IPAdressSpace of the document making the request. The goal of this project is to block local network access until the user has granted permission to it.

LNA in Action

If a request is detected as a local network access and the origin of the request was not previously granted permission by Firefox we will block the request. If the request is being made from a secure context Firefox will prompt the user for permission to allow access to the local network. If the user grants permission we will allow the request and remember this decision for subsequent requests from the origin. We will remember this decision for the lifetime of the document by default. If the user checks the remember box, we will remember this decision permanently.
However if the user denies permission we block the request.
Status of any subsequent requests from the origin to local host/local networks should be displayed in a non-intrusive way to the user. This nudges the user to re-think their decision if they had accidentally granted/denied the permission. This would require implementation support from the UI team.

LNA in the Context of Mixed Content

Mixed content specification blocks secure contexts making requests over HTTP except for localhost connections (by IP literals and host name) embedded in the subresources. The mixed content specification also attempts to upgrade some kinds of resources from HTTP to HTTPS, but will not do so for literal IP addresses.
Mixed content checks are made before the IP address of the target is known and hence this could block the request to local addresses.

Mixed-content checks prevent access to local devices running on HTTP.

To avoid this, the Google's latest proposal posits using the following options:

Option 1. A new option in fetch API (targetAddressSpace: local) to skip mixed content checks for such requests and will prompt the user for permission.

Option 2. Hostname is private IP address literal or .local domain or .internal[2]

We don't plan to implement Option 1 or Option 2 as these are not standard yet. Not implementing this would not make things worse than the status quo. We will consider implementing this once Google's proposal is standardized.

However, this would also mean there is no way for legitimate secure public servers to communicate with local devices over http. Use cases of such requests include configuration requests for devices on the local network. We will provide an option to do this via configuration settings. We choose configuration settings over permission prompts as the former is a more deliberate decision for the user and chances of accidental access grants are minimized.

Detecting Local Network Access in Necko

While most of the features outlined are relatively simple to implement, determining whether a network request qualifies as local network access requires careful consideration.

In light of this requirement, we evaluated some design options and finally agreed to the following approach.

The transaction layer in Necko will error when we make public to local access. Specifically, during the activation of the transaction, we will have DNS resolved and information about the IP address. We will check if the request is crossing the local network boundary and prompt the user for permission.

Based on our feature description and design discussion, we have come up with broadly 4 categories of work for this feature:

- Implementation of LNA detection in Necko
- Permissions for LNA feature and UI permission prompts
- Rollout

We discuss the sub-work-items for each of the above mentioned categories.

1. **LNA Detection in Necko**
    a. Necko helper functions for detecting private network access

[2] Internal is not yet specified in the standard. This was introduced recently by ICANN. We should propose this to the standard.

b. Add triggering address space to loadInfo and add IP address space to browsingContext https://wicg.github.io/private-network-access/#integration-html
c. **Block access from non-https public requests to private requests Local Network Access ("secure context" required) - Not implemented in firefox**
d. We need an API or prefs to allow configuration of specific IP addresses/hostname as public, local or loopback. We need to also support wildcards/suffixes for domains and address ranges/masks for IP addresses.
    i. An enterprise policy will also likely be necessary.
e. Unit test infra for LNA testing
f. Implement Local Network Access detection mechanism in Necko
g. Add telemetry to access the number and type of private network access. This is necessary for planning our rollout.

2. **Permissions for LNA feature and Permission and UI Prompts**
    a. Permission prompts LNA.
    b. Permission for Local Network Access (both host and network)
    c. Permission prompt for navigations - fail navigation on LNA failures
    d. Permission Policy support for iframe delegation
    e. State icon for the permission settings section of the URL bar to show when access has been granted or blocked for that site.

3. **Feature Rollout**
    a. Investigate and add any additional mechanism for early detection of site-breakages
    b. QA support for the feature
    c. Investigate the probes landed in Milestone 1 and plan the size for the rollout
    d. Write a connect post and blogpost for the feature to seek early feedback
    e. Send intent to ship email
4. **Advanced Blocking**
    a. Add ip-address-space to HTTP cache entry metadata
    b. Handle configuration changes of cached resources: https://wicg.github.io/private-network-access/#http-cache-main-resources

5. **Improving User experience**
    a. Configuration of whitelisted origins for mixed-content
    b. Configuration of whitelisted targets for skipping LNA checks
    c. Handling LNA Checks for Captive portals
    d. Adding more info in the URL of the prompts
    e. LNA checks with extensions

Milestones

Meta bug [1481298 - (private-network-access) meta (Local) Private Network Access]

Milestone 1: Telemetry and Implementation of Core LNA Framework in Necko

Goal: The goal of this milestone is to ensure we detect LNA requests and measure

Work Items	Estimate (days)	Status
1.a Networking: Helper functions for local network access	1	Landed (140)
1.b Add address space to loadInfo and browsingContext	1	Landed (140)
~~1.d Add ip-address-space to HTTP cache entry metadata~~	1	Deferred not needed fo MS1
~~1.e Handle configuration changes of cached resources (incl. Enterprise policy)~~	2	Deferred not needed fo MS1
1.c Unit test infra for LNA testing	8	In Review (planned 141)
1.d Implement LNA detection mechanism in transaction/connection management layer	8	Landed 140
1.e Add telemetry probes to determine the number of LNA accesses and categorize them. This will help us accessing the risk and plan rollout	1	Landed 140
Total	24	

Milestone 2: Block Tracking Scripts from Local Network Access **142**

Goal: The goal of this milestone is to stop tracking scripts from making local host and local network connections

Work Items	Estimate (days)	Status
2.a Fail Local network Access Transaction	5	Landed
2.b Automatically block LNA access coming from tracking scripts	3	Landed (Enabled only in nightly and early beta). Will enable it next week.
2.c Add a flag in LoadInfo to indicate whether a request is from a tracker	5	Landed

Milestone 3: Permissions for LNA feature and user Permission UI Prompts (**143**)

Goal: The goal of this milestone is to implement permissions and UI prompts around LNA feature and handling these permissions inside Necko

Work Items	Estimate (days)	Status
3.a UI Changes for LNA Permissions - Desktop	8	Done
3.b UI Changes for LNA Permissions - Fenix	5	Done
3.c UX and Content Design Changes for LNA Permissions	5	Done
3.d Integrating user permission with Necko code and handling transaction retries	5	Done
3.f State icon for the permission settings section of the URL	3	Done

bar to show when access has been granted or blocked for that site.		
Total	25	

Milestone 4: Advanced Blocking

This milestone ensures protections for cached resources and navigational access

Work Items	Estimate (days)	Status
4.a UI Changes for configuration settings of the LNA	5	Done
4.b LNA blocking and permission prompt for Navigation	5	Done
4.d Add ip-address-space to HTTP cache entry metadata	2	Done
4.e Handle configuration changes of cached resources (incl. Enterprise policy)	2	Done
4.f Permission Policy support for iframe delegation	2	Ongoing
Total	16	

Milestone 5: Improving User Experience

This milestone focuses on enhancing the user's interaction with the LNA feature and addressing edge cases.

Work Items	Estimate (days)	Status
6.a Configuration of whitelisted origins for mixed-content	2	Moved out of scope
6.b Configuration of whitelisted targets for skipping LNA checks	2	Will be implemented after rollout

Work Items	Estimate (days)	Status
6.c Handling LNA Checks for Captive portals	2	Ongoing
6.d Adding more info in the URL of the prompts	1	Done
6.e LNA checks with extensions	1	ongoing
Total		

Milestone 6: Rollout (145)

Goal: The goal of this milestone is to ship this feature to release.

Work Items	Estimate (days)	Status
5.a Investigate and add any additional mechanism for early detection of site-breakages	2	
5.b LNA QA Testing	2	Ongoing
5.c Investigate the probes landed in Milestone 1 and writing experiment brief and feature rollout and intent to ship email	1.5	Ongoing
5.d Write a connect post and blogpost for the feature to seek early feedback	0.5	Not started
Total	6	

Schedule

Milestone	Start	End	Release Target	Status	Notes
Telemetry and Implementation of Core LNA Framework in Necko	4th May	30th May	140/141	Compl…	Telemetry has landed for 140 Changes for core LNA

					framework is ready and will be merged by 30th May
Block Tracking Scripts from Local Network Access	25th May		143	Compl…	
Permissions for LNA feature and user Permission UI Prompts	27th May	15th June	**143**	Compl…	
Advanced Blocking			143	Compl…	
Improving User Experience			145	On Track	
Rollout			144	Not St…	

References

- https://wicg.github.io/private-network-access/
- Firefox_PNA Project Plan
- Google's Local Network Access
- PNA standards position - Standards position: positive
- PNA Web-Platform Tests

Current Status

LNA Status

Open Points:

- Pi Hole Mitigations