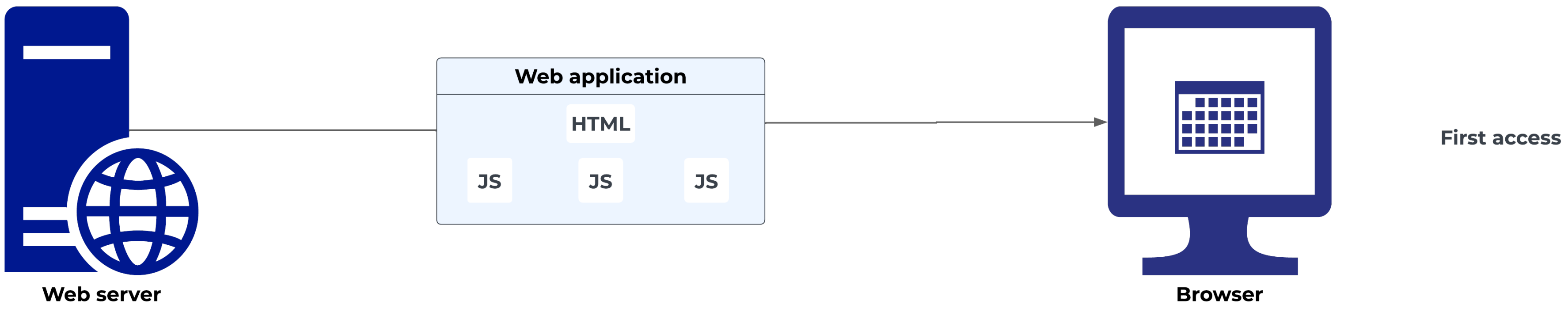
CISPA
HELMHOLTZ CENTER FOR INFORMATION SECURITY

Accountable Javascript Code Delivery

Ilkan Esiyok (CISPA),*
Pascal Berrang (University of Birmingham and Nimiq),
Katriel Cohn-Gordon (Meta),
Robert Kuennemann (CISPA)

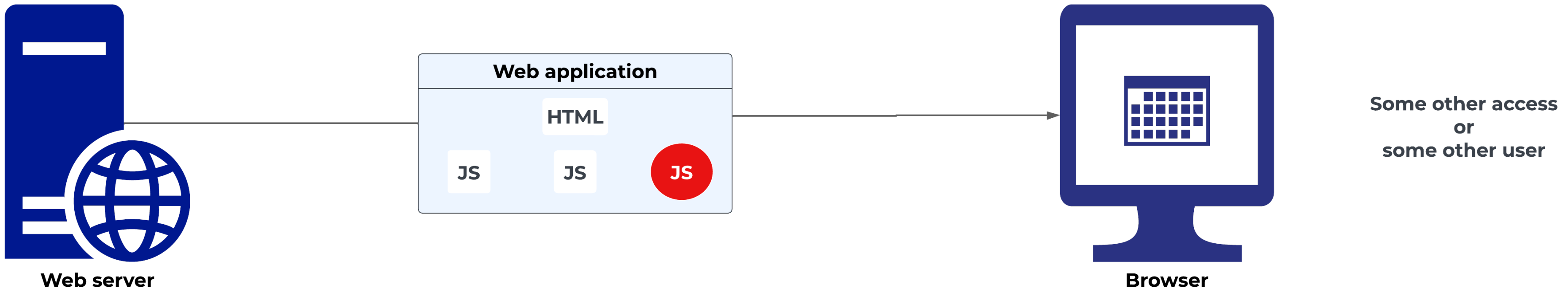
NDSS 2023, San Diego

Motivation



The web is ephemeral

Motivation

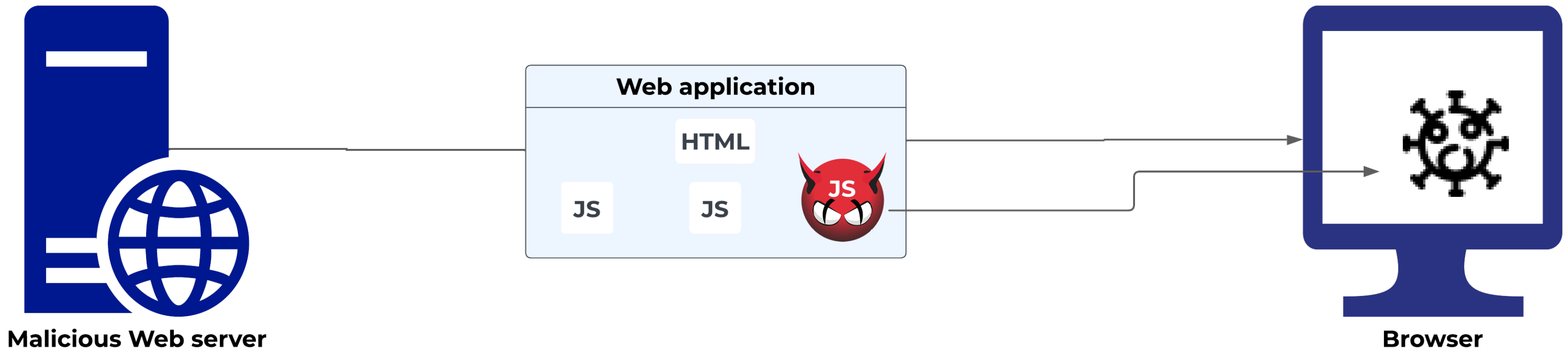


The web page looks the same but the active content has changed

Motivation

A compromised or malicious web server can easily target classes of users:

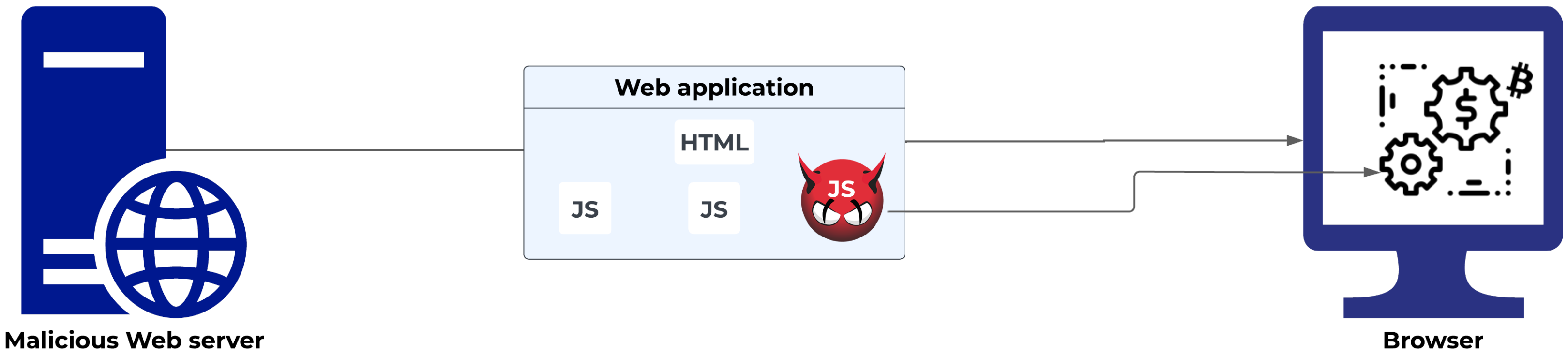
- The web server might insert malware based on browser fingerprint



Motivation

A compromised or malicious web server can easily target classes of users:

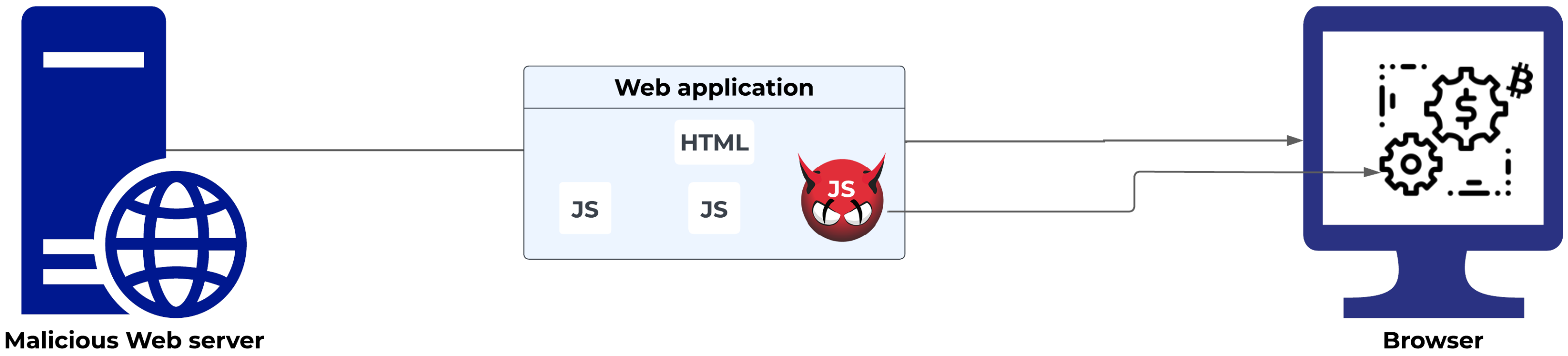
- The web server might insert malware based on browser fingerprint
- The server might use the browser for cryptojacking



Motivation

A compromised or malicious web server can easily target classes of users:

- The web server might insert malware based on browser fingerprint
- The server might use the browser for cryptojacking



There is a lack of trust between developers and users in web infrastructure

Security goals

- Our target audience: websites that want to **establish and maintain trust** to their users
- Examples:

Security goals

- Our target audience: websites that want to **establish and maintain trust** to their users
- Examples:
  - Proton Mail → wants users to trust that they really encrypt their emails

Security goals

- Our target audience: websites that want to **establish and maintain trust** to their users
- Examples:
  - Proton Mail → wants users to trust that they really encrypt their emails
  - NIMIQ → wants users to trust that they don’t have access to users’ funds

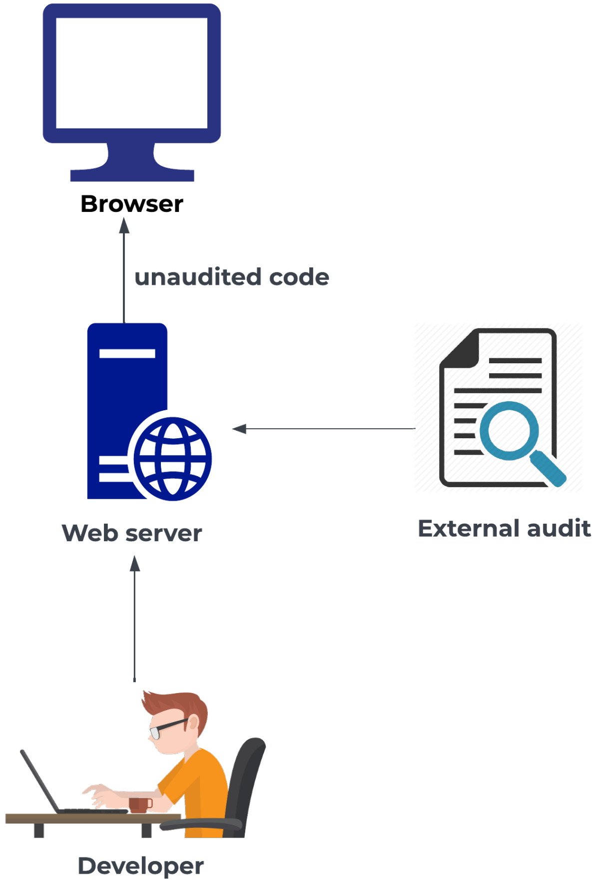
Security goals

- Our target audience: websites that want to **establish and maintain trust** to their users
- Examples:
  - Proton Mail → wants users to trust that they really encrypt their emails
  - NIMIQ → wants users to trust that they don’t have access to users’ funds
  - WhatsApp → Code Verify : allow users to trust that the web client keeps their messages secret

Risk mitigation strategies

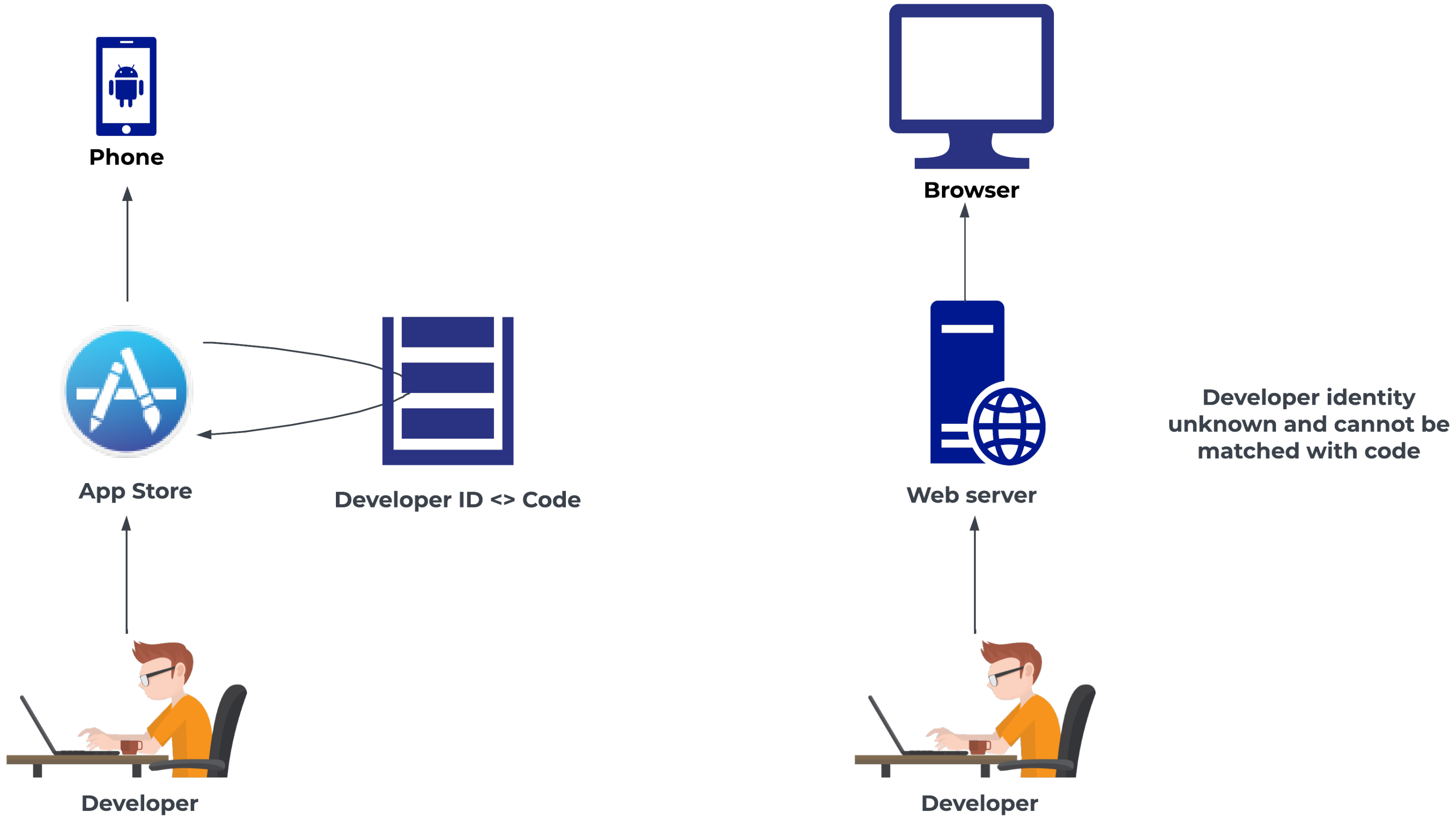
Risk mitigation strategies

- Auditing
  - ✅ Works for App stores
  - ❌ Malicious server can choose to load unaudited code in runtime



Risk mitigation strategies

- Accountability
  - ✅ Works for App stores (Developers can be held accountable for malicious code)
  - ❌ No public record of the code and the developer's identity



Accountable JS

Provide accountable delivery of active content, using :

Accountable JS

Provide accountable delivery of active content, using :

Efficient and easy code signing

Proof of origin

Accountable JS

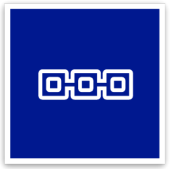
Provide accountable delivery of active content, using :

Efficient and easy code signing

Proof of origin

+

Public transparency Logs



Proof that everyone receives the same code in a timeframe

Accountable JS

Provide accountable delivery of active content, using :

Efficient and easy code signing

Proof of origin

+

Public transparency Logs

Proof that everyone receives the same code in a timeframe

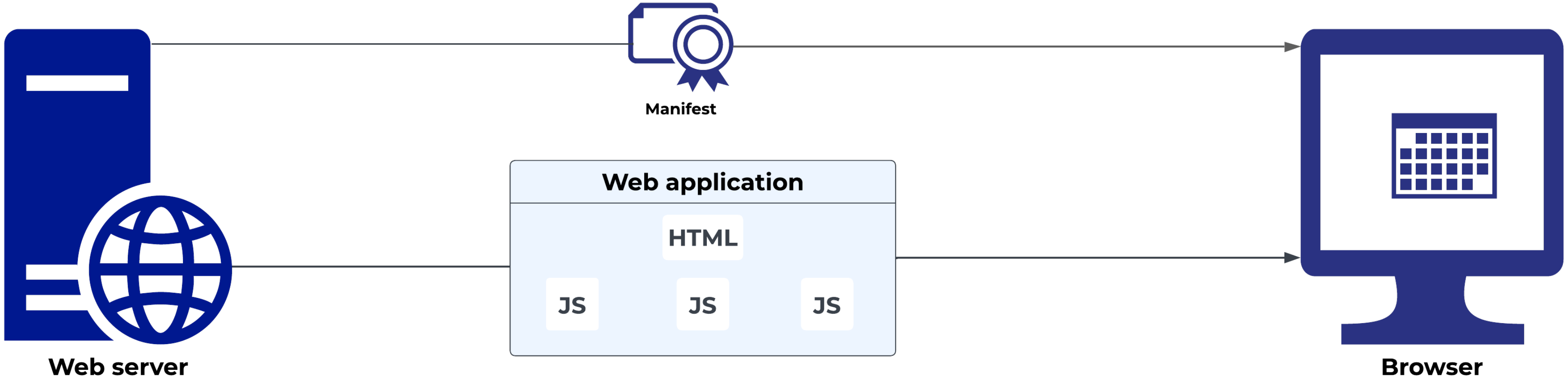
+

Sub-protocol for non-repudiable

Proof that content was received by individual user

ACCOUNTABLE JS

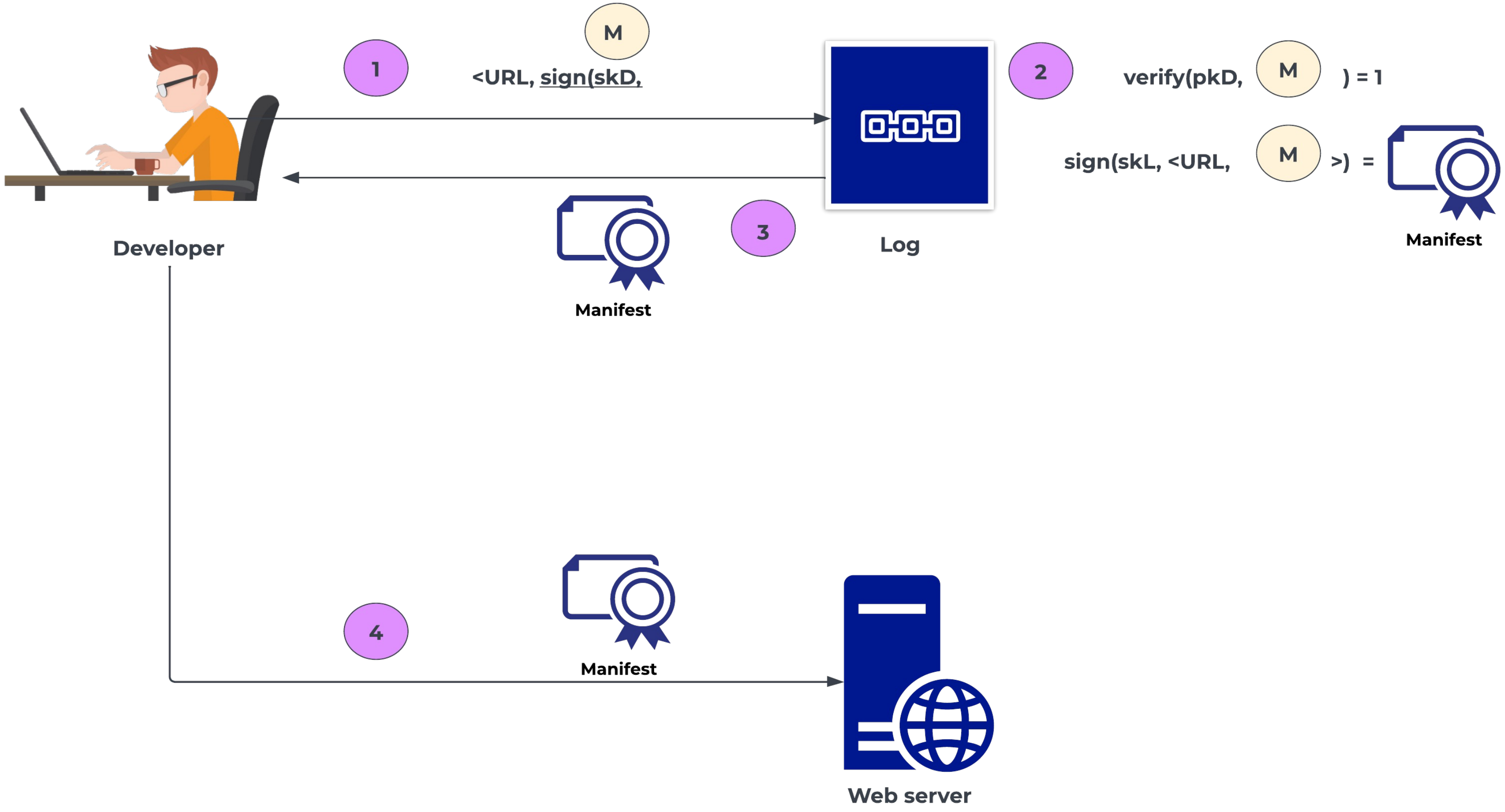
Accountable JS



- Provide a signed manifest enumerating all the active content
- Browser extension
  - Measures the delivered active content and compares with the manifest

Accountable JS

- Separate the developer and the web server
- Use public transparency logs



Manifest file

- Simple text file in JSON format
- List of metadata for each active content in the web page

manifest.json

```json
{
  "url": "https://helloworld.com/index.html",
  "name": "Hello world application",
  "manifest_version": "v1.0",
  "description": "",
  "contents": [
    {
      "seq": 0,
      "type": "inline",
      "load": "sync",
      "hash": "sha256-XT0yF1DRjbn5ymbnsasJnag4+53huda0TZ3bRPIcrAA=",
      "dynamic": false,
      "trust": "assert"
    }
  ]
}
```

Manifest file

- Each active content metadata must have a trust declaration
- The compliance check method is decided based on trust value

Manifest file

- Trust values : assert
    - The developer provides hash of active content and asserts that it behaves as intended

Manifest file

- Trust values : assert
  - The developer provides hash of active content and asserts that it behaves as intended
  - Measurement – compliance check :
    - Compare the delivered hash against the hash in the manifest

Manifest file

- Trust values : assert
  - The developer provides hash of active content and asserts that it behaves as intended
  - Measurement – compliance check :
    - Compare the delivered hash against the hash in the manifest
  - Case studies :

Manifest file

- Trust values : assert
  - The developer provides hash of active content and asserts that it behaves as intended
  - Measurement – compliance check :
    - Compare the delivered hash against the hash in the manifest
  - Case studies :
    - Self contained applications : WhatsApp web

Manifest file

- Trust values : assert
    - The developer provides hash of active content and asserts that it behaves as intended
    - Measurement – compliance check :
        - Compare the delivered hash against the hash in the manifest
    - Case studies :
        - Self contained applications : WhatsApp web
            - Developer vouches for their own content

Manifest file

- Trust values : assert
    - The developer provides hash of active content and asserts that it behaves as intended
    - Measurement – compliance check :
        - Compare the delivered hash against the hash in the manifest
    - Case studies :
        - Self contained applications : WhatsApp web
            - Developer vouches for their own content
        - Trusted third party code : JQuery

Manifest file

- Trust values : assert
  - The developer provides hash of active content and asserts that it behaves as intended
  - Measurement – compliance check :
    - Compare the delivered hash against the hash in the manifest
  - Case studies :
    - Self contained applications : WhatsApp web
      - Developer vouches for their own content
    - Trusted third party code : JQuery
      - Developer pins the third-party code to a precise version that was audited

Manifest file

- Trust values : delegate
    - The developer refers the trust to the third-party that provides the element

Manifest file

- Trust values : delegate
    - The developer refers the trust to the third-party that provides the element
    - Now the third party is taking the responsibility for this code

Manifest file

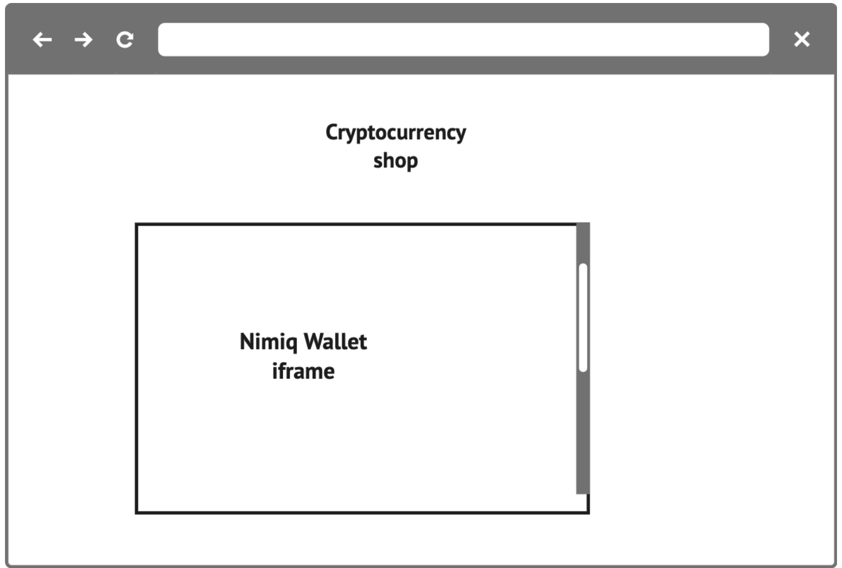
- Trust values : delegate
    - The developer refers the trust to the third-party that provides the element
    - Now the third party is taking the responsibility for this code
    - The developer doesn’t want to vouch for the third-party

Manifest file

- Trust values : delegate
    - The developer refers the trust to the third-party that provides the element
    - Now the third party is taking the responsibility for this code
    - The developer doesn’t want to vouch for the third-party
    - Or she always wants to use the latest version

Manifest file

- Trust values : delegate
  - The developer refers the trust to the third-party that provides the element
  - Now the third party is taking the responsibility for this code
  - The developer doesn't want to vouch for the third-party
  - Or she always wants to use the latest version
  - Case study :
    - The third-party willing to vouch for their code : Nimiq Wallet



Manifest file

- Trust values : blind-trust
    - The developer blindly trusts the active content without identifying the code

Manifest file

- Trust values : blind-trust
  - The developer blindly trusts the active content without identifying the code
  - The developer is responsible to properly sandbox that code

Manifest file

- Trust values : blind-trust
    - The developer blindly trusts the active content without identifying the code
    - The developer is responsible to properly sandbox that code
    - Measurement – compliance check :
        - Compare delivered sandbox against the sandbox in manifest

Manifest file

- Trust values : blind-trust
  - The developer blindly trusts the active content without identifying the code
  - The developer is responsible to properly sandbox that code
  - Measurement – compliance check :
    - Compare delivered sandbox against the sandbox in manifest
  - Case study :
    - The third-party code through Adbidding blind-trusted :
      - Adsense (blind-trust + sandbox) + Nimiq Wallet (delegate)

Manifest file

- Trust values : blind-trust
    - The developer blindly trusts the active content without identifying the code
    - The developer is responsible to properly sandbox that code
    - Measurement – compliance check :
        - Compare delivered sandbox against the sandbox in manifest
    - Case study :
        - The third-party code through Adbidding blind-trusted :
          ❌ Adsense (blind-trust + sandbox) + Nimiq Wallet (delegate)

Manifest file

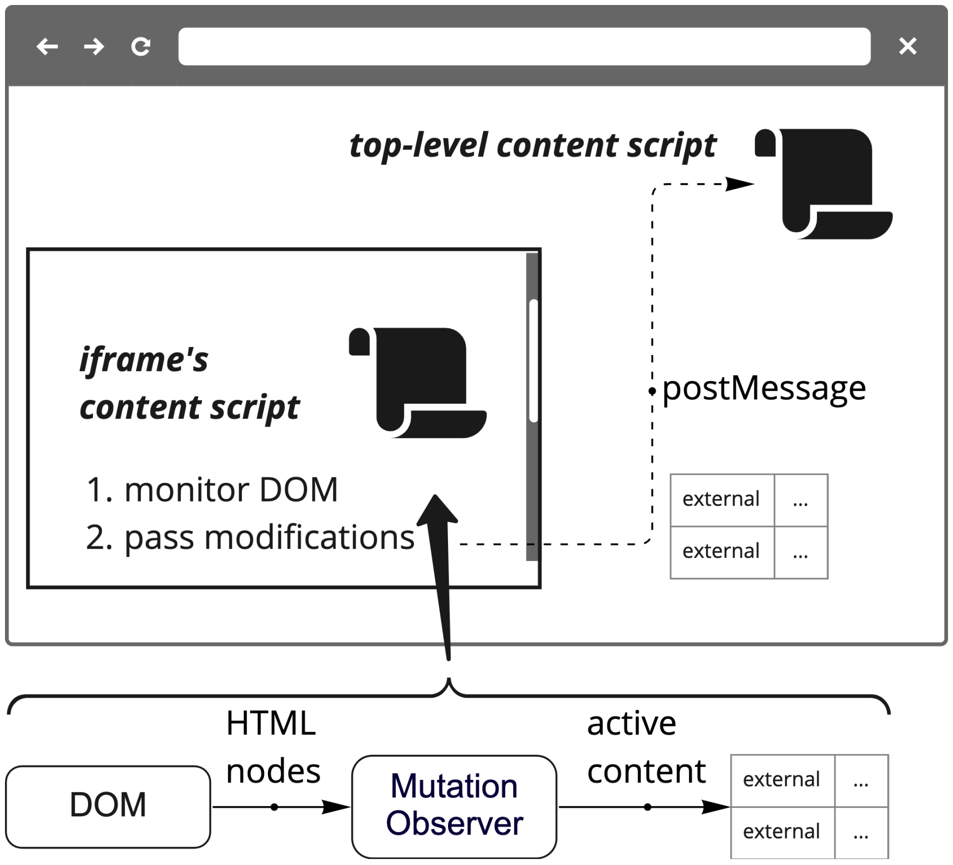
- Trust values : blind-trust
    - The developer blindly trusts the active content without identifying the code
    - The developer is responsible to properly sandbox that code
    - Measurement – compliance check :
        - Compare delivered sandbox against the sandbox in manifest
    - Case study :
        - The third-party code through Adbidding blind-trusted :

          ❌ Adsense (blind-trust + sandbox) + Nimiq Wallet (delegate)

          ✅ Adsense (blind-trust) + Nimiq Wallet (delegate + sandbox)

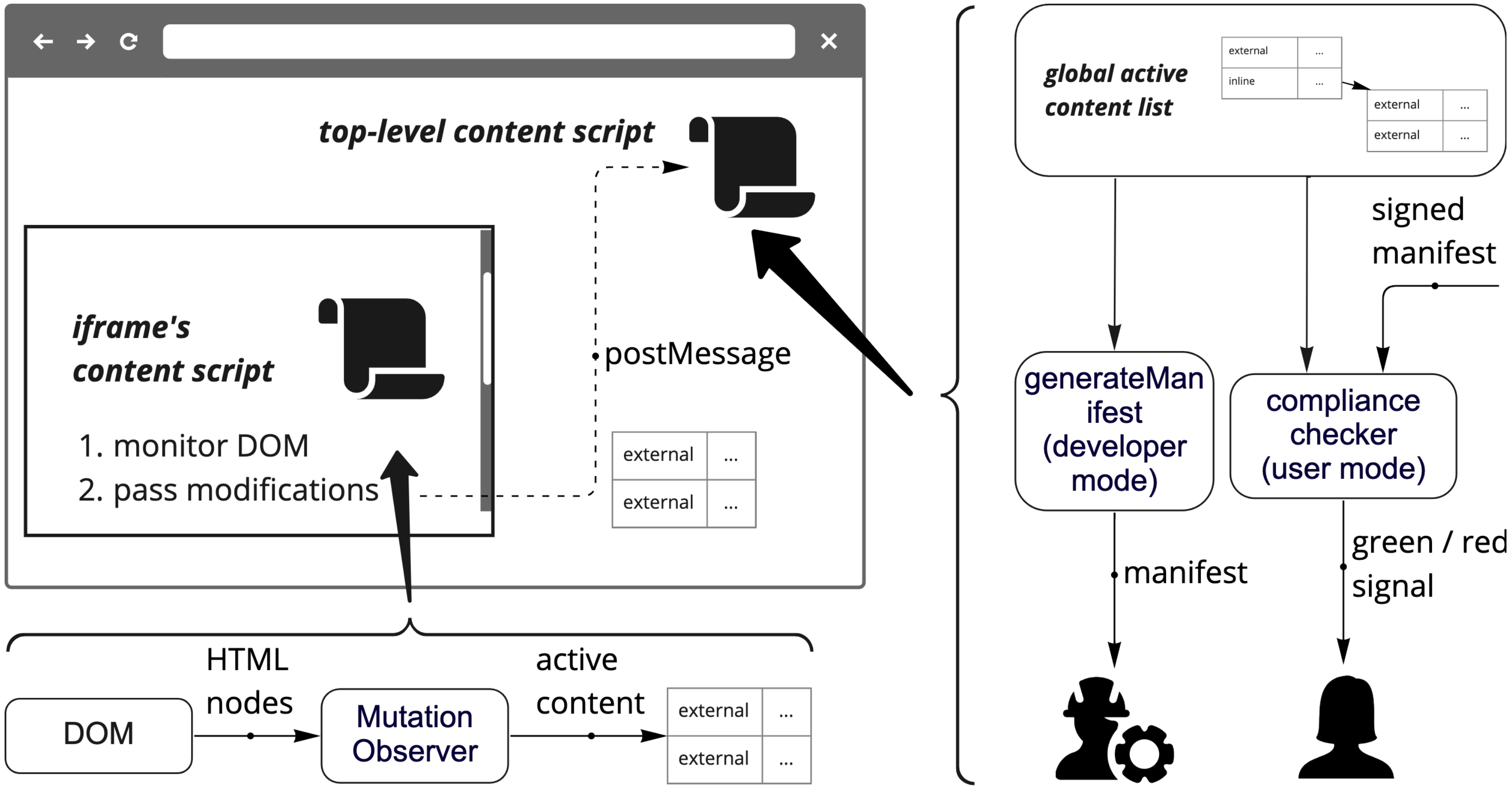
Measurement procedure

- Content scripts collect active content metadata



Measurement procedure

- Compliance check : measures the active content and compares w/ manifest



Evaluation

- Compatibility and performance analysis on the case studies
- How much does Accountable JS extension affect page load time?
  - Lighthouse metrics :
    - Time until browser paints the first pixel,
    - Total blocking time

Evaluation results

Case study	First pixel		Total blocking time	
	Baseline	Accountable JS	Baseline	Accountable JS
Trusted third-party (JQuery)	462	+21	0	+0
Delegate trust (Nimiq Wallet)	262	- 10	172	+87
Untrusted third-party (Adsense + Nimiq Wallet)	747	+91	159	+77

	Total page load		
	Baseline	Code Verify	Accountable JS
WhatsApp Web	204	+16	+40

- Baseline is all extensions disabled
- All numbers are in milliseconds
- Change below 100 ms is considered imperceptible

Related work

- Content Security Policy (CSP)
  - ✅ List of valid sources

Related work

- Content Security Policy (CSP)
  - ✅ List of valid sources
  - ✅ Unknown resources denied

Related work

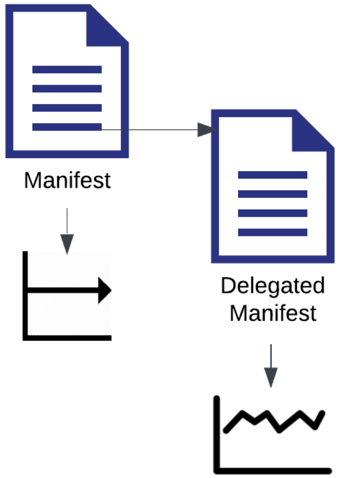
- Content Security Policy (CSP)
  - ✅ List of valid sources
  - ✅ Unknown resources denied
  - ❌ No accountability

Related work

- Content Security Policy (CSP)
  - ✅ List of valid sources
  - ✅ Unknown resources denied
  - ❌ No accountability
  - ❌ Not designed to know the order of resources in the webpage
    - Resource A loaded before B might mean something different from B then A
    - This can be used for microtargeting

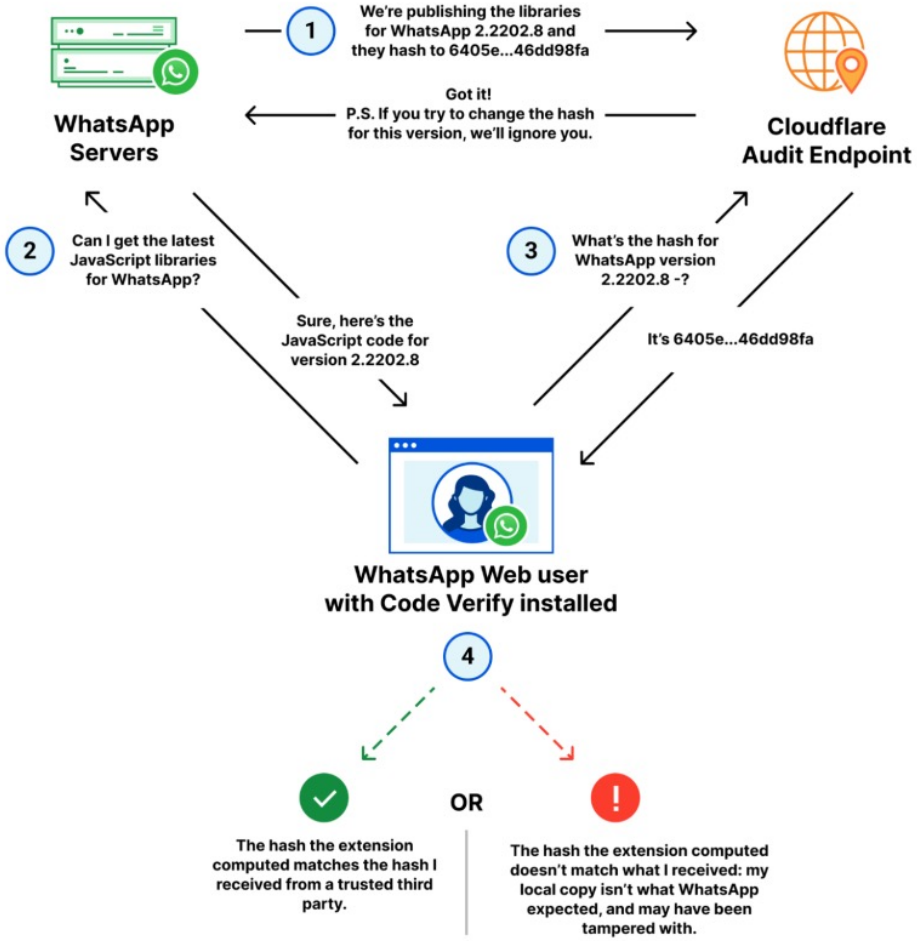
Related work

- Content Security Policy (CSP)
  - ✅ List of valid sources
  - ✅ Unknown resources denied
  - ❌ No accountability
  - ❌ Not designed to know the order of resources in the webpage
    - Resource A loaded before B might mean something different than B then A
    - This can be used for microtargeting
  - ❌ No delegation support



Related work

- Code Verify from Meta
  - ✅ Likewise implementing accountability for active content



Related work

- Code Verify from Meta
  - ✅ Likewise implementing accountability for active content
  - ❌ Manifest is hashed not signed -> no accountability

Related work

- Code Verify from Meta
  - ✅ Likewise implementing accountability for active content
  - ❌ Manifest is hashed not signed -> no accountability
  - ❌ No history of versions -> no transparency
    - Public cannot know how often the versions change

Conclusion

- Accountable JS

Conclusion

- Accountable JS
  - ✅ improve the trust on security-critical websites

Conclusion

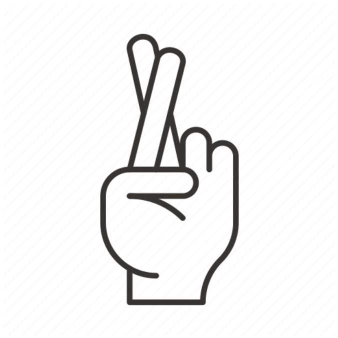
- Accountable JS
  - ✅ improve the trust on security-critical websites
  - ✅ enhance security by deterrence

Conclusion

- Accountable JS
  - ✅ improve the trust on security-critical websites
  - ✅ enhance security by deterrence
  - ✅ increase transparency
    - public can see how their data is used

Conclusion

- Accountable JS
  - ✅ improve the trust on security-critical websites
  - ✅ enhance security by deterrence
  - ✅ increase transparency
    - public can see how their data is used
  - ✅ become part of the browsers some day



Conclusion

- What will you find in the paper?

Conclusion

- What will you find in the paper?
  - Details about Accountable JS protocol flow

Conclusion

- What will you find in the paper?
  - Details about Accountable JS protocol flow
  - Case studies and evaluations on CSP and Code Verify

Conclusion

- What will you find in the paper?
  - Details about Accountable JS protocol flow
  - Case studies and evaluations on CSP and Code Verify
  - Threat model and assumptions

Conclusion

- What will you find in the paper?
  - Details about Accountable JS protocol flow
  - Case studies and evaluations on CSP and Code Verify
  - Threat model and assumptions
  - Protocol verification details
    - Automated protocol verification : Tamarin and SAPIC

End

- Thank you very much

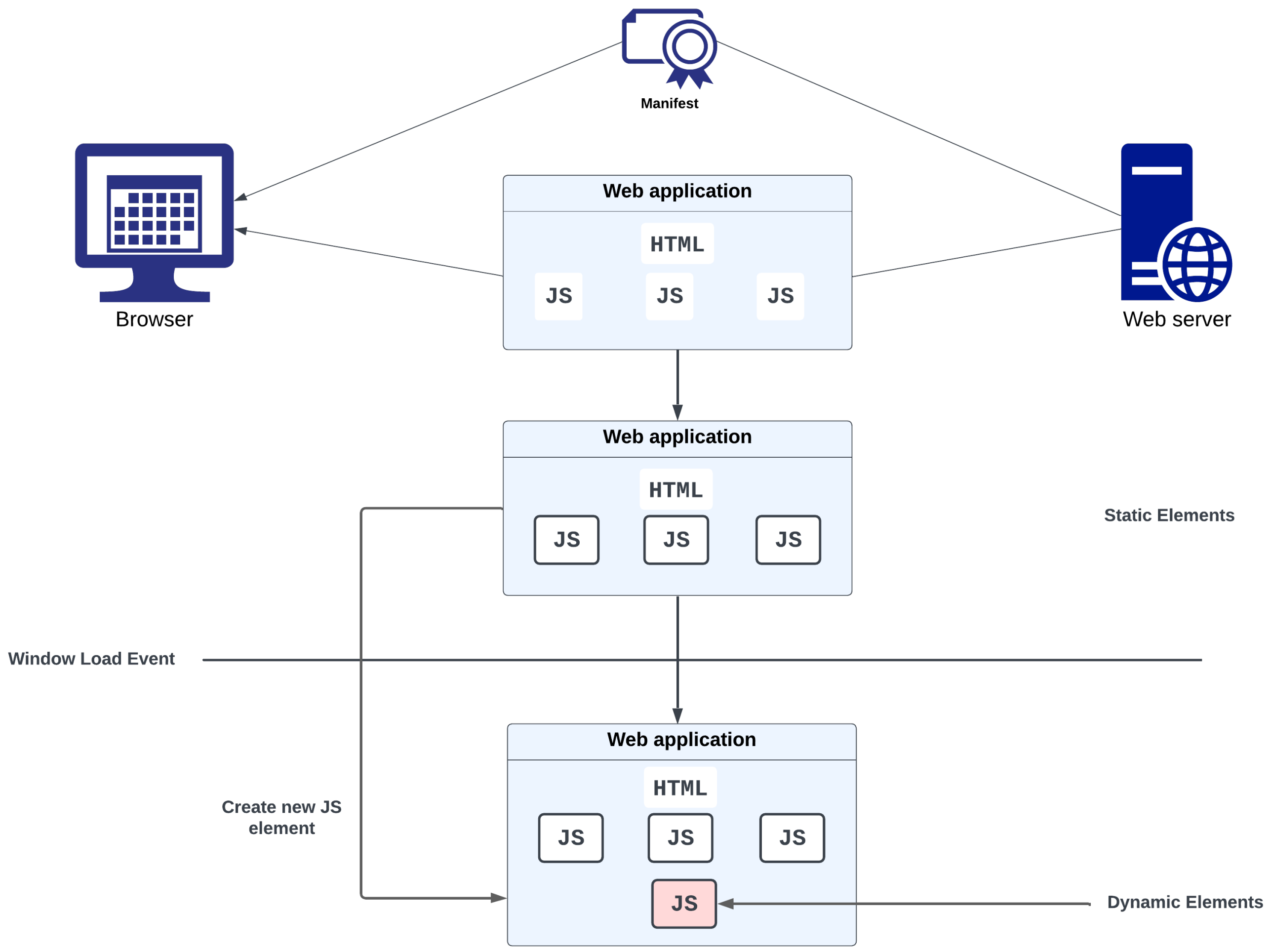
Manifest file

- Active content types

inline javascript without src attribute	**event_handler** html element that includes e.g. onclick
external the scripts that are outsourced	**iframe** might have its own manifest

Manifest file

- Execution order and static-dynamic content



Protocol verification

- Security protocol
  - Establish security guarantees → formal methods

Protocol verification

- Security protocol
  - Establish security guarantees → formal methods
- Analysed with Tamarin Prover + SAPIC

Security properties

Security properties

Accountable JS	Code Verify
• Authentication of origin	• Authentication of origin
• Transparency	
• Accountability	• Non-accountability
• End-to-end guarantee	• End-to-end guarantee

Authentication of origin : The client executes active content only if the corresponding manifest was generated by the honest developer unless the developer is corrupted (or Cloudflare in CV),
Transparency : If the client executes code then its manifest is present in a transparency log,
Accountability : When the public accepts a claim, then even if the client was corrupted, the code must exist in the logs and the server must have sent that data
Non-accountability : The data provided to the client is not sufficient to prove they received certain content from the web server, even if web server and Cloudflare are honest.
End-to-end guarantee : Only by corrupting the developer it is possible to distribute malicious code.

Security properties

- Accountability and authentication of origin
  - A client executes the code only if it was made public by the developer

Security properties

- Accountability and authentication of origin
  - A client executes the code only if it was made public by the developer
- Non-repudiation of reception
  - A client wants to present false claim about the executed code

Security properties

- Accountability and authentication of origin
  - A client executes the code only if it was made public by the developer
- Non-repudiation of reception
  - A client wants to present false claim about the executed code
- Accountability of latest version
  - A client wants to ensure he is delivered the latest code

Security properties

- Accountability and authentication of origin
  - A client executes the code only if it was made public by the developer
- Non-repudiation of reception
  - A client wants to present false claim about the executed code
- Accountability of latest version
  - A client wants to ensure he is delivered the latest code

Security properties of the Code Verify are discussed in the paper

Transparency logs

- Clients can verify they received the latest and the same version of the code as any other user
- Public append-only log:
  - Trusted, efficient, available
  - Provides non-equivocation
  - Third-party auditors and monitors keep it honest
- Trillian : allows to prove append operations efficiently
  - Misbehaviour can be detected by trusted public auditors or by honest logs distributing such proofs (with gossiping)

Transparency logs – availability, scalability

- Use load balancing, avoid single point of failure
- Stapling method decreases the number of requests to the log
- Websites that frequently update active content:

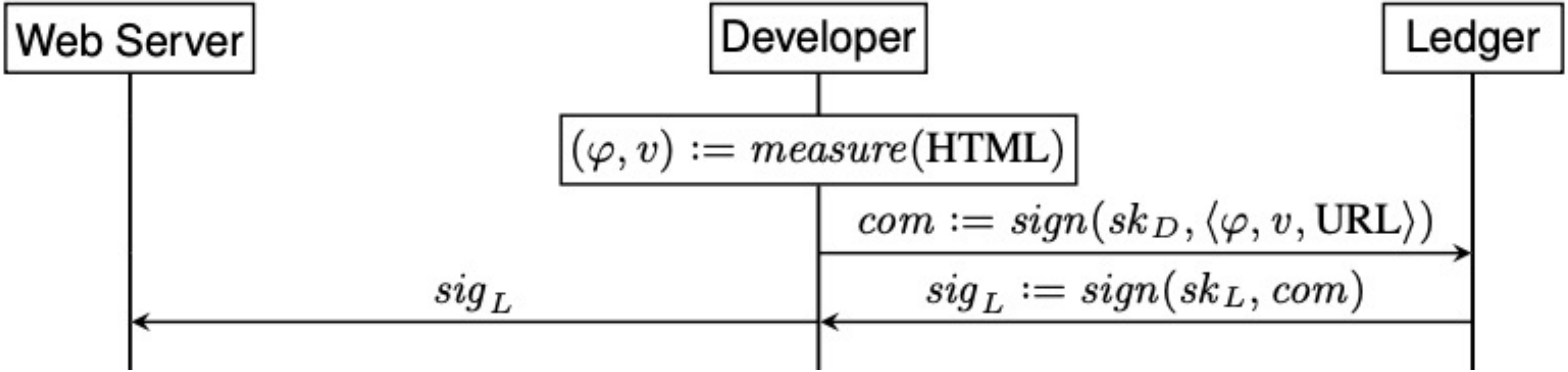
Websites that frequently update their active contents can create significant burden on the log size. We calculate approximately how many times each log can be updated for a limited time and space. We assume a non-leaf node overhead is approximately 100 bytes and for the leaf nodes it is 700 bytes(signature 600 bytes + 100 bytes). If a log provider has 100 TB of space for 5 years, it can contain 137 billion signatures in total. To make sense of this number, take the following example. We start with a log of 10M URLs with eight updates per month on average. The number of URLs also increases exponentially at a rate of 1% with each update (i.e. also eight times per month). [5] This number would be well below 137 billion signatures.

Limitations

- Active content injected by other browser extensions
- Data – only attacks
  - e.g. modified button labels or redirect form URLs, change recipient’s wallet address during payment transaction

- **Protocol flow**

- Code stapling



- **Protocol flow**

- Code delivery