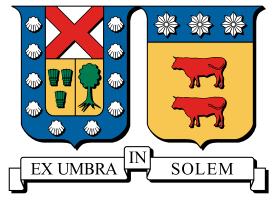
Current Investigation
Toward a Security Reference Architecture of Web Browser

Paulina Silva Ghio

pasilva@alumnos.inf.utfsm.cl

22-03-2016/23-03-2016.



UNIVERSIDAD TECNICA
FEDERICO SANTA MARIA

Index

Context

- Web Browser's war in the nineties.
- Built and fix.
- Web Browser: a tool used daily.
- Common user uses its services.
- Many type of implementations.
- Web 2.0 y 3.0: AJAX (Asynchronous Javascript and XML).

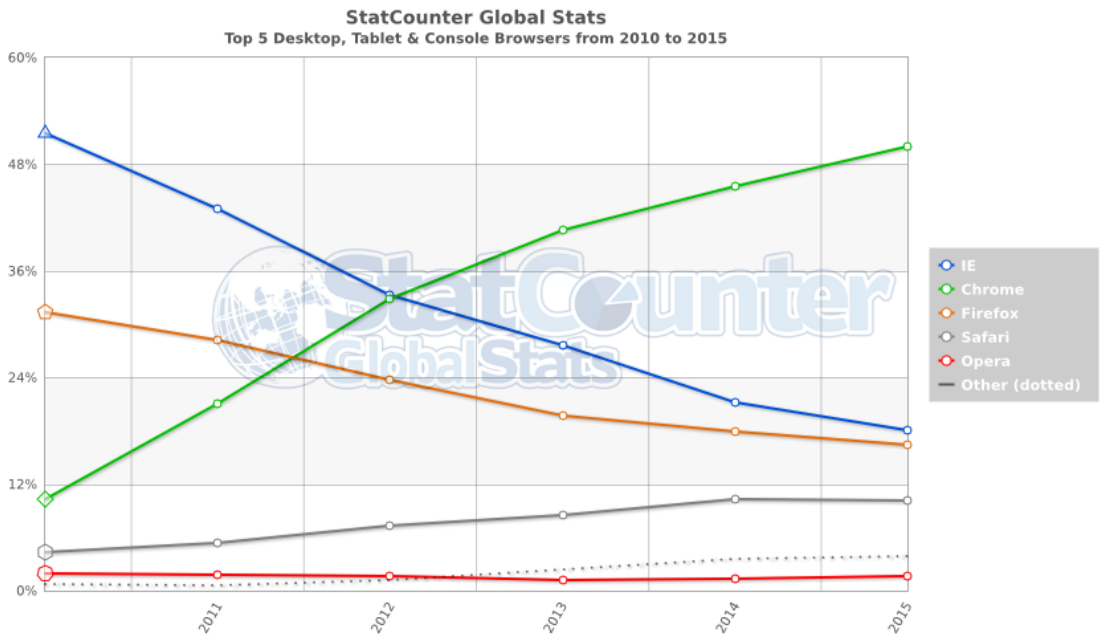


Figura: Market Percentaje of each Browser. Cite: [1]

Context

- Web Browser's war in the nineties.
- Built and fix.
- Web Browser: a tool used daily.
- Common user uses its services.
- Many type of implementations.
- Web 2.0 y 3.0: AJAX (Asynchronous Javascript and XML).

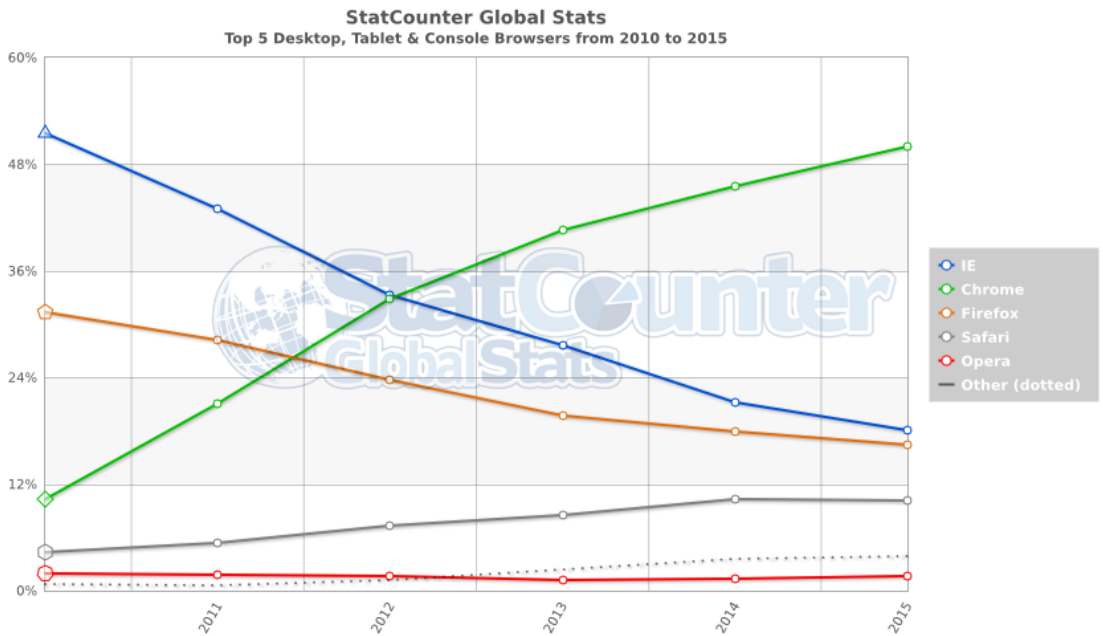


Figura: Market Percentaje of each Browser. Cite: [1]

Motivation

Browser is an indispensable tool, this lets:

- New ways of interacting.
- Lower building costs for a client program.
- Already implemented a lot of robust security features in the browser.
- Reuse.

Principal Concerns

- Systems which are called by users using a browser.
- Stakeholders involved: browser's user, host's user and the external service used.

Motivation

Browser is an indispensable tool, this lets:

- New ways of interacting.
- Lower building costs for a client program.
- Already implemented a lot of robust security features in the browser.
- Reuse.

Principal Concerns

- Systems which are called by users using a browser.
- Stakeholders involved: browser's user, host's user and the external service used.

Motivation

Browser is an indispensable tool, this lets:

- New ways of interacting.
- Lower building costs for a client program.
- Already implemented a lot of robust security features in the browser.
- Reuse.

Principal Concerns

- Systems which are called by users using a browser.
- Stakeholders involved: browser's user, host's user and the external service used.

Motivation

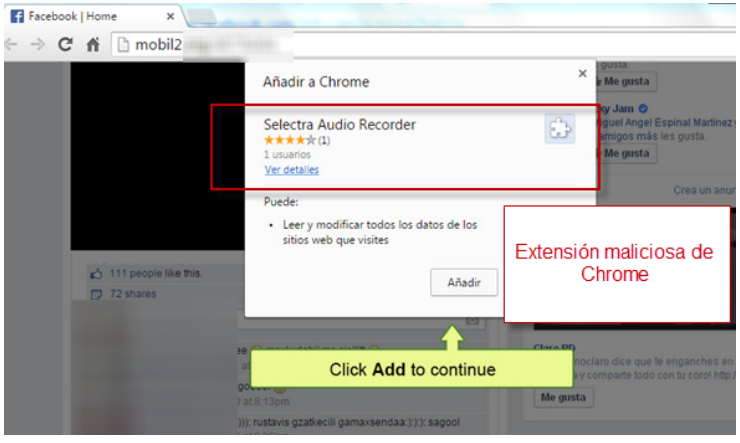
Browser is an indispensable tool, this lets:

- New ways of interacting.
- Lower building costs for a client program.
- Already implemented a lot of robust security features in the browser.
- Reuse.

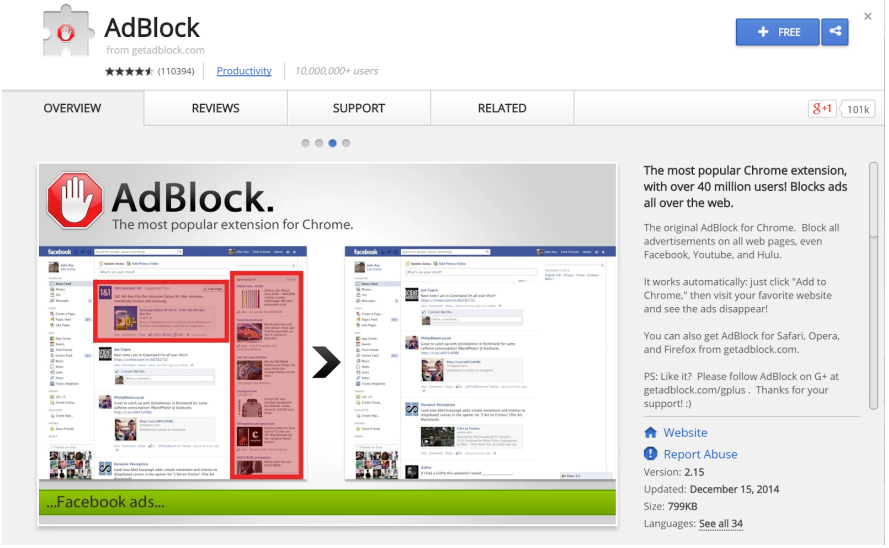
Principal Concerns

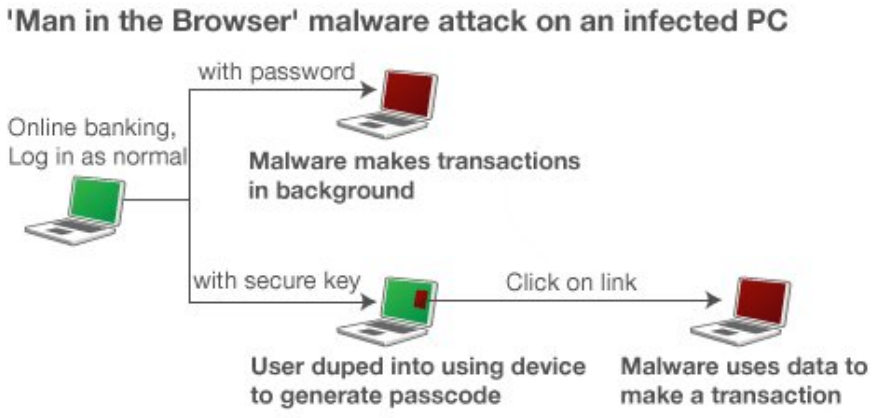
- Systems which are called by users using a browser.
- Stakeholders involved: browser's user, host's user and the external service used.

Threats and Vulnerabilities



1. Installation of Malware and malicious Extensions.
2. Benign-but-buggy Extensions.
3. Man in the Browser.
4. Code Injection.





Problems

- Lack of knowledge in browser's security aspects, could affect directly the development of applications and stakeholders.
- Scarce documentation and none unification of concepts. No formal descriptions for browser related concepts.

Reference Architecture (RA) for the browser

- Specifies the decomposition of the systems into subsystems, interactions between these parts and functionality distribution between them.
- Captures the essence of the architecture through a collection of similar systems, using architectonic reuse.
- Currently, there is no consensus in how to define an RA, what should have and how should be built. We use architectural patterns.
- Describes concerns and quality attributes needed.
- Helps: developers, in general stakeholders.
- Compares design decisions.
- Holistic view of the system.

Reference Architecture (RA) for the browser

- Specifies the decomposition of the systems into subsystems, interactions between these parts and functionality distribution between them.
- Captures the essence of the architecture through a collection of similar systems, using architectonic reuse.
- Currently, there is no consensus in how to define an RA, what should have and how should be built. We use architectural patterns.
- Describes concerns and quality attributes needed.
- Helps: developers, in general stakeholders.
- Compares design decisions.
- Holistic view of the system.

Reference Architecture (RA) for the browser

- Specifies the decomposition of the systems into subsystems, interactions between these parts and functionality distribution between them.
- Captures the essence of the architecture through a collection of similar systems, using architectonic reuse.
- Currently, there is no consensus in how to define an RA, what should have and how should be built. We use architectural patterns.
- Describes concerns and quality attributes needed.
- Helps: developers, in general stakeholders.
- Compares design decisions.
- Holistic view of the system.

Security Reference Architecture (SRA)

- Holistic view of security, taking into account the threats.
- Abstraction of defense mechanism described as security patterns.
- Unifies terminology.
- Can help to evaluate a system and its security requirements.

Security Reference Architecture (SRA)

- Holistic view of security, taking into account the threats.
- Abstraction of defense mechanism described as security patterns.
- Unifies terminology.
- Can help to evaluate a system and its security requirements.

Securing a RA

- We start from a list of use cases which describe the typical cloud uses and their associated roles. Lists of cloud use cases are shown in [Bad12], [nis11], and [Has13c].
- We analyze each use case looking for vulnerabilities and threats as in [Bra08]. This implies checking each activity in the activity diagram of the use cases to see how it can be attacked. This approach results in a systematic enumeration of threats.
- We use the list of threats from [Has13a] to confirm these threats and to find possible further vulnerabilities and threats.
- These threats are expressed in the form of misuse patterns. We developed some misuse patterns for Cloud Computing in [Has13b], we show more later.
- We apply policies to handle the threats and we identify security patterns to realize the policies. There are also regulatory policies which are realized as security patterns.
- SRA can be used as a guideline to specify what the customer wants, and cmply to SLA of Cloud services

Securing a RA

- We start from a list of use cases which describe the typical cloud uses and their associated roles. Lists of cloud use cases are shown in [Bad12], [nis11], and [Has13c].
- We analyze each use case looking for vulnerabilities and threats as in [Bra08]. This implies checking each activity in the activity diagram of the use cases to see how it can be attacked. This approach results in a systematic enumeration of threats.
- We use the list of threats from [Has13a] to confirm these threats and to find possible further vulnerabilities and threats.
- These threats are expressed in the form of misuse patterns. We developed some misuse patterns for Cloud Computing in [Has13b], we show more later.
- We apply policies to handle the threats and we identify security patterns to realize the policies. There are also regulatory policies which are realized as security patterns.
- SRA can be used as a guideline to specify what the customer wants, and cmply to SLA of Cloud services

Securing a RA

- We start from a list of use cases which describe the typical cloud uses and their associated roles. Lists of cloud use cases are shown in [Bad12], [nis11], and [Has13c].
- We analyze each use case looking for vulnerabilities and threats as in [Bra08]. This implies checking each activity in the activity diagram of the use cases to see how it can be attacked. This approach results in a systematic enumeration of threats.
- We use the list of threats from [Has13a] to confirm these threats and to find possible further vulnerabilities and threats.
- These threats are expressed in the form of misuse patterns. We developed some misuse patterns for Cloud Computing in [Has13b], we show more later.
- We apply policies to handle the threats and we identify security patterns to realize the policies. There are also regulatory policies which are realized as security patterns.
- SRA can be used as a guideline to specify what the customer wants, and cmply to SLA of Cloud services

Securing a RA

- We start from a list of use cases which describe the typical cloud uses and their associated roles. Lists of cloud use cases are shown in [Bad12], [nis11], and [Has13c].
- We analyze each use case looking for vulnerabilities and threats as in [Bra08]. This implies checking each activity in the activity diagram of the use cases to see how it can be attacked. This approach results in a systematic enumeration of threats.
- We use the list of threats from [Has13a] to confirm these threats and to find possible further vulnerabilities and threats.
- These threats are expressed in the form of misuse patterns. We developed some misuse patterns for Cloud Computing in [Has13b], we show more later.
- We apply policies to handle the threats and we identify security patterns to realize the policies. There are also regulatory policies which are realized as security patterns.
- SRA can be used as a guideline to specify what the customer wants, and cmply to SLA of Cloud services

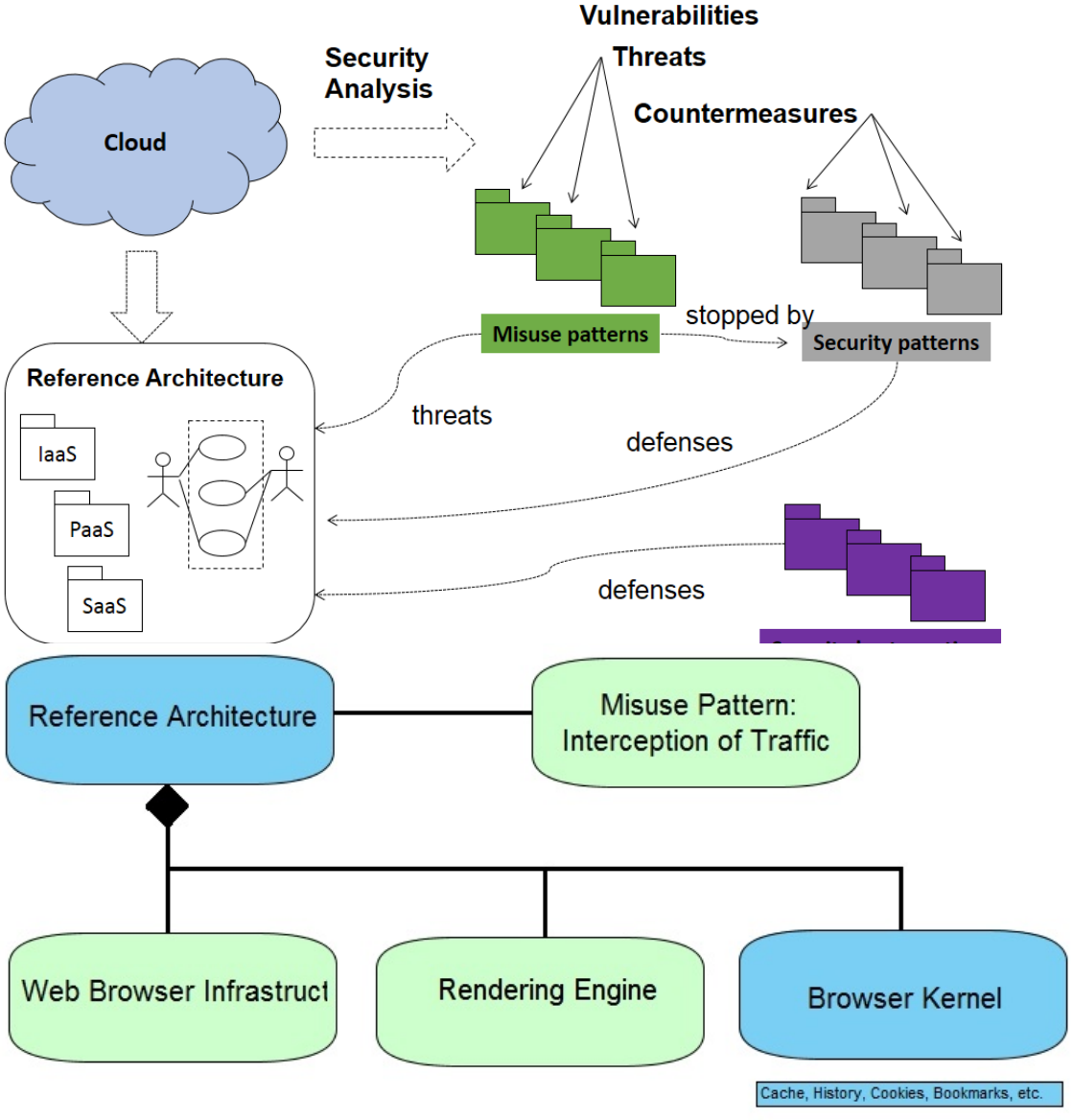
Securing a RA

- We start from a list of use cases which describe the typical cloud uses and their associated roles. Lists of cloud use cases are shown in [Bad12], [nis11], and [Has13c].
- We analyze each use case looking for vulnerabilities and threats as in [Bra08]. This implies checking each activity in the activity diagram of the use cases to see how it can be attacked. This approach results in a systematic enumeration of threats.
- We use the list of threats from [Has13a] to confirm these threats and to find possible further vulnerabilities and threats.
- These threats are expressed in the form of misuse patterns. We developed some misuse patterns for Cloud Computing in [Has13b], we show more later.
- We apply policies to handle the threats and we identify security patterns to realize the policies. There are also regulatory policies which are realized as security patterns.
- SRA can be used as a guideline to specify what the customer wants, and cmply to SLA of Cloud services

Securing a RA

- We start from a list of use cases which describe the typical cloud uses and their associated roles. Lists of cloud use cases are shown in [Bad12], [nis11], and [Has13c].
- We analyze each use case looking for vulnerabilities and threats as in [Bra08]. This implies checking each activity in the activity diagram of the use cases to see how it can be attacked. This approach results in a systematic enumeration of threats.
- We use the list of threats from [Has13a] to confirm these threats and to find possible further vulnerabilities and threats.
- These threats are expressed in the form of misuse patterns. We developed some misuse patterns for Cloud Computing in [Has13b], we show more later.
- We apply policies to handle the threats and we identify security patterns to realize the policies. There are also regulatory policies which are realized as security patterns.
- SRA can be used as a guideline to specify what the customer wants, and cmply to SLA of Cloud services

RA and SRA


Vulnerabilities
Threats
Countermeasures
Security Analysis
Cloud
Misuse patterns
stopped by
Security patterns
Reference Architecture
IaaS
PaaS
SaaS
threats
defenses
defenses
Reference Architecture
Misuse Pattern: Interception of Traffic
Web Browser Infrastruct
Rendering Engine
Browser Kernel
Cache, History, Cookies, Bookmarks, etc.

RA and SRA

- RAs and SRAs are not implementable, they are abstract artifacts, their evaluation is based on completeness (they cover all concrete RAs as special cases), precision (UML modeling), understandability, extensibility (produced by the use of patterns).
- Value:
  - can be used to produce new concrete architectures
  - can be used to analyze the effect of threats
  - can be used to define monitoring points
  - can be used as the basis of Service Contracts
  - can be used to define use cases.
- Browser: Until now, there is no clear and precise model of the actual requirements and components. Only old models.

RA and SRA

- RAs and SRAs are not implementable, they are abstract artifacts, their evaluation is based on completeness (they cover all concrete RAs as special cases), precision (UML modeling), understandability, extensibility (produced by the use of patterns).
- Value:
  - can be used to produce new concrete architectures
  - can be used to analyze the effect of threats
  - can be used to define monitoring points
  - can be used as the basis of Service Contracts
  - can be used to define use cases.
- Browser: Until now, there is no clear and precise model of the actual requirements and components. Only old models.

RA and SRA

- RAs and SRAs are not implementable, they are abstract artifacts, their evaluation is based on completeness (they cover all concrete RAs as special cases), precision (UML modeling), understandability, extensibility (produced by the use of patterns).
- Value:
  - can be used to produce new concrete architectures
  - can be used to analyze the effect of threats
  - can be used to define monitoring points
  - can be used as the basis of Service Contracts
  - can be used to define use cases.
- Browser: Until now, there is no clear and precise model of the actual requirements and components. Only old models.

RA and SRA

- RAs and SRAs are not implementable, they are abstract artifacts, their evaluation is based on completeness (they cover all concrete RAs as special cases), precision (UML modeling), understandability, extensibility (produced by the use of patterns).
- Value:
  - can be used to produce new concrete architectures
  - can be used to analyze the effect of threats
  - can be used to define monitoring points
  - can be used as the basis of Service Contracts
  - can be used to define use cases.
- Browser: Until now, there is no clear and precise model of the actual requirements and components. Only old models.

Misuse Patterns

- Describes from the point of view of the attacker, how an attack can be done (which units uses and how), analizes the ways of stopping the attack by enumerating the possible security patterns that could be used, and describes how to trace back the attack once it has occured (recolection of forensic data).
- Let teach and communicate possible ways how a system can be misused.

Misuse Patterns

- Describes from the point of view of the attacker, how an attack can be done (which units uses and how), analizes the ways of stopping the attack by enumerating the possible security patterns that could be used, and describes how to trace back the attack once it has occured (recolection of forensic data).
- Let teach and communicate possible ways how a system can be misused.

State of Art

- Have not found updated works about Reference Architectures of the Browser. The work of Grosskurth et al. [2] is close, but the technique is different.
- Works:
  - Larrondo et al. [3]: analyzes the browser and obtains a Domain Model, an Object Model, a Feature Tree which describes structure and functionality of the browser.
  - Grosskurth et al. [4, 2]: Reverse engineering tool to obtain a high-level architecture of open-source browsers: Mozilla y Konqueror.
  - Godfrey et al. [5]: similar to the above but only for Mozilla, and also obtained architectural views of the system.
  - Lwin [6]: proposes a browser called Anfel SOFT, with AI to improve user's experience.
- Scarse and poor documentation. None unification of concepts.
- As for a SRA, thre was no work found.

State of Art

- Have not found updated works about Reference Architectures of the Browser. The work of Grosskurth et al. [2] is close, but the technique is different.
- Works:
  - Larrondo et al. [3]: analyzes the browser and obtains a Domain Model, an Object Model, a Feature Tree which describes structure and functionality of the browser.
  - Grosskurth et al. [4, 2]: Reverse engineering tool to obtain a high-level architecture of open-source browsers: Mozilla y Konqueror.
  - Godfrey et al. [5]: similar to the above but only for Mozilla, and also obtained architectural views of the system.
  - Lwin [6]: proposes a browser called Anfel SOFT, with AI to improve user's experience.
- Scarse and poor documentation. None unification of concepts.
- As for a SRA, thre was no work found.

State of Art

- Have not found updated works about Reference Architectures of the Browser. The work of Grosskurth et al. [2] is close, but the technique is different.
- Works:
  - Larrondo et al. [3]: analyzes the browser and obtains a Domain Model, an Object Model, a Feature Tree which describes structure and functionality of the browser.
  - Grosskurth et al. [4, 2]: Reverse engineering tool to obtain a high-level architecture of open-source browsers: Mozilla y Konqueror.
  - Godfrey et al. [5]: similar to the above but only for Mozilla, and also obtained architectural views of the system.
  - Lwin [6]: proposes a browser called Anfel SOFT, with AI to improve user's experience.
- Scarse and poor documentation. None unification of concepts.
- As for a SRA, thre was no work found.

State of Art

- Have not found updated works about Reference Architectures of the Browser. The work of Grosskurth et al. [2] is close, but the technique is different.
- Works:
  - Larrondo et al. [3]: analyzes the browser and obtains a Domain Model, an Object Model, a Feature Tree which describes structure and functionality of the browser.
  - Grosskurth et al. [4, 2]: Reverse engineering tool to obtain a high-level architecture of open-source browsers: Mozilla y Konqueror.
  - Godfrey et al. [5]: similar to the above but only for Mozilla, and also obtained architectural views of the system.
  - Lwin [6]: proposes a browser called Anfel SOFT, with AI to improve user's experience.
- Scarse and poor documentation. None unification of concepts.
- As for a SRA, thre was no work found.

State of Art

- Have not found updated works about Reference Architectures of the Browser. The work of Grosskurth et al. [2] is close, but the technique is different.
- Works:
  - Larrondo et al. [3]: analyzes the browser and obtains a Domain Model, an Object Model, a Feature Tree which describes structure and functionality of the browser.
  - Grosskurth et al. [4, 2]: Reverse engineering tool to obtain a high-level architecture of open-source browsers: Mozilla y Konqueror.
  - Godfrey et al. [5]: similar to the above but only for Mozilla, and also obtained architectural views of the system.
  - Lwin [6]: proposes a browser called Anfel SOFT, with AI to improve user's experience.
- Scarse and poor documentation. None unification of concepts.
- As for a SRA, thre was no work found.

General and specific objectives/goals

General objective

- Build an organized body of information about the Web Browser and its security.
- Systematize, organize and classify adquiered knowledge in a document, with a semi-formal format.
- Better comprehension of browser's security.

Specific objectives

- A guide to comunicate relevant concepts.
- Improve our Reference Architecture and continue our misuse pattern catalog.
- Build a conceptual model of browser's security, a Security Reference Architecture.
- Get to know how social engineering can affect the browser.
- Use Experimental Software Engineering techniques.

General and specific objectives/goals

General objective

- Build an organized body of information about the Web Browser and its security.
- Systematize, organize and classify adquiered knowledge in a document, with a semi-formal format.
- Better comprehension of browser's security.

Specific objectives

- A guide to comunicate relevant concepts.
- Improve our Reference Architecture and continue our misuse pattern catalog.
- Build a conceptual model of browser's security, a Security Reference Architecture.
- Get to know how social engineering can affect the browser.
- Use Experimental Software Engineering techniques.

General and specific objectives/goals

General objective

- Build an organized body of information about the Web Browser and its security.
- Systematize, organize and classify adquiered knowledge in a document, with a semi-formal format.
- Better comprehension of browser's security.

Specific objectives

- A guide to comunicate relevant concepts.
- Improve our Reference Architecture and continue our misuse pattern catalog.
- Build a conceptual model of browser's security, a Security Reference Architecture.
- Get to know how social engineering can affect the browser.
- Use Experimental Software Engineering techniques.

Hypotesis

Main Hypotesis

H1: The definition of a Security Reference Architecture for the Web Browser allows to abstract and capture main structural aspects, its behavior and security related requirements.

Validation

- Reference Architecture and Security Reference Architecture are not implemented. They are abstract models.
- Experts opinions.
- *PLoP conferences and “shepherding”process. Conferences: AsianPLoP y EuroPLoP.
- Experimental Software Engineering to validate or reject hypotesis.

Validation

- Reference Architecture and Security Reference Architecture are not implemented. They are abstract models.
- Experts opinions.
- *PLoP conferences and “shepherding”process. Conferences: AsianPLoP y EuroPLoP.
- Experimental Software Engineering to validate or reject hypotesis.

Work already done

- RA and misuse patterns.
  - **Browser Infrastructure Pattern**. A pattern which abstracts web browser's architecture implementations and request/response communication (mostly, since implementations may differ).
  - **Web Content Renderer Pattern**. A pattern which abstracts th interior of the web browser's Renderer Engine.
  - **Browser Kernel Pattern** (not ready yet). A pattern which abstracts the principal component of the browser, like cache, cookies, bookmarks, network communication, etc.
  - **Misuse Pattern: Interception of Traffic**. A pattern which abstract a misuse that could occur in the web browser when an attacker beforehand achieves a social engineering attack. When the SE attack is succesful, the misuse is capable of intercepting the messages between Browser Kernel and Web Content Renderer.
- AsianPLoP and EuroPLoP.

Work already done

- RA and misuse patterns.
  - **Browser Infrastructure Pattern**. A pattern which abstracts web browser's architecture implementations and request/response communication (mostly, since implementations may differ).
  - **Web Content Renderer Pattern**. A pattern which abstracts th interior of the web browser's Renderer Engine.
  - **Browser Kernel Pattern** (not ready yet). A pattern which abstracts the principal component of the browser, like cache, cookies, bookmarks, network communication, etc.
  - **Misuse Pattern: Interception of Traffic**. A pattern which abstract a misuse that could occur in the web browser when an attacker beforehand achieves a social engineering attack. When the SE attack is succesful, the misuse is capable of intercepting the messages between Browser Kernel and Web Content Renderer.
- AsianPLoP and EuroPLoP.

Work already done

- RA and misuse patterns.
  - **Browser Infrastructure Pattern**. A pattern which abstracts web browser's architecture implementations and request/response communication (mostly, since implementations may differ).
  - **Web Content Renderer Pattern**. A pattern which abstracts th interior of the web browser's Renderer Engine.
  - **Browser Kernel Pattern** (not ready yet). A pattern which abstracts the principal component of the browser, like cache, cookies, bookmarks, network communication, etc.
  - **Misuse Pattern: Interception of Traffic**. A pattern which abstract a misuse that could occur in the web browser when an attacker beforehand achieves a social engineering attack. When the SE attack is succesful, the misuse is capable of intercepting the messages between Browser Kernel and Web Content Renderer.
- AsianPLoP and EuroPLoP.

Work already done

- RA and misuse patterns.
  - **Browser Infrastructure Pattern**. A pattern which abstracts web browser's architecture implementations and request/response communication (mostly, since implementations may differ).
  - **Web Content Renderer Pattern**. A pattern which abstracts th interior of the web browser's Renderer Engine.
  - **Browser Kernel Pattern** (not ready yet). A pattern which abstracts the principal component of the browser, like cache, cookies, bookmarks, network communication, etc.
  - **Misuse Pattern: Interception of Traffic**. A pattern which abstract a misuse that could occur in the web browser when an attacker beforehand achieves a social engineering attack. When the SE attack is succesful, the misuse is capable of intercepting the messages between Browser Kernel and Web Content Renderer.
- AsianPLoP and EuroPLoP.

Side work

- Fondecyt Project: Software Development Initiatives to Identify and Mitigate Security Threats – A Systematic Mapping.
  - A systematic mapping has been conducted to cover the existent technologies for identification and mitigation of security threats.
  - A total of 10 different techniques covering threats identification and 8 covering the mitigation of threats were found.
  - All the initiatives were integrated to at least one activity of the Software Development Lifecycle (SDLC), while 7 show signs of being adopted in the industry.
  - The mapping found only 15 studies that covered 11 different iniatiatives.
  - Only two techniques presented scientific evidence of its results through controlled experiments, while others selected studies presented informal case studies or examples.
- Fondecyt Project: Research
  - A Theoretical Framework for quality assessment of security metrics.
  - How to evaluate security in software development

Side work

- Fondecyt Project: Software Development Initiatives to Identify and Mitigate Security Threats – A Systematic Mapping.
  - A systematic mapping has been conducted to cover the existent technologies for identification and mitigation of security threats.
  - A total of 10 different techniques covering threats identification and 8 covering the mitigation of threats were found.
  - All the initiatives were integrated to at least one activity of the Software Development Lifecycle (SDLC), while 7 show signs of being adopted in the industry.
  - The mapping found only 15 studies that covered 11 different iniatiatives.
  - Only two techniques presented scientific evidence of its results through controlled experiments, while others selected studies presented informal case studies or examples.
- Fondecyt Project: Research
  - A Theoretical Framework for quality assessment of security metrics.
  - How to evaluate security in software development

Side work

- Fondecyt Project: Software Development Initiatives to Identify and Mitigate Security Threats – A Systematic Mapping.
  - A systematic mapping has been conducted to cover the existent technologies for identification and mitigation of security threats.
  - A total of 10 different techniques covering threats identification and 8 covering the mitigation of threats were found.
  - All the initiatives were integrated to at least one activity of the Software Development Lifecycle (SDLC), while 7 show signs of being adopted in the industry.
  - The mapping found only 15 studies that covered 11 different iniatiatives.
  - Only two techniques presented scientific evidence of its results through controlled experiments, while others selected studies presented informal case studies or examples.
- Fondecyt Project: Research
  - A Theoretical Framework for quality assessment of security metrics.
  - How to evaluate security in software development

Side work

- Fondecyt Project: Software Development Initiatives to Identify and Mitigate Security Threats – A Systematic Mapping.
  - A systematic mapping has been conducted to cover the existent technologies for identification and mitigation of security threats.
  - A total of 10 different techniques covering threats identification and 8 covering the mitigation of threats were found.
  - All the initiatives were integrated to at least one activity of the Software Development Lifecycle (SDLC), while 7 show signs of being adopted in the industry.
  - The mapping found only 15 studies that covered 11 different iniatiatives.
  - Only two techniques presented scientific evidence of its results through controlled experiments, while others selected studies presented informal case studies or examples.
- Fondecyt Project: Research
  - A Theoretical Framework for quality assessment of security metrics.
  - How to evaluate security in software development

Side work

- Fondecyt Project: Software Development Initiatives to Identify and Mitigate Security Threats – A Systematic Mapping.
  - A systematic mapping has been conducted to cover the existent technologies for identification and mitigation of security threats.
  - A total of 10 different techniques covering threats identification and 8 covering the mitigation of threats were found.
  - All the initiatives were integrated to at least one activity of the Software Development Lifecycle (SDLC), while 7 show signs of being adopted in the industry.
  - The mapping found only 15 studies that covered 11 different iniatiatives.
  - Only two techniques presented scientific evidence of its results through controlled experiments, while others selected studies presented informal case studies or examples.
- Fondecyt Project: Research
  - A Theoretical Framework for quality assessment of security metrics.
  - How to evaluate security in software development

Side work

- Fondecyt Project: Software Development Initiatives to Identify and Mitigate Security Threats – A Systematic Mapping.
  - A systematic mapping has been conducted to cover the existent technologies for identification and mitigation of security threats.
  - A total of 10 different techniques covering threats identification and 8 covering the mitigation of threats were found.
  - All the initiatives were integrated to at least one activity of the Software Development Lifecycle (SDLC), while 7 show signs of being adopted in the industry.
  - The mapping found only 15 studies that covered 11 different iniatiatives.
  - Only two techniques presented scientific evidence of its results through controlled experiments, while others selected studies presented informal case studies or examples.
- Fondecyt Project: Research
  - A Theoretical Framework for quality assessment of security metrics.
  - How to evaluate security in software development

Side work

- Fondecyt Project: Software Development Initiatives to Identify and Mitigate Security Threats – A Systematic Mapping.
  - A systematic mapping has been conducted to cover the existent technologies for identification and mitigation of security threats.
  - A total of 10 different techniques covering threats identification and 8 covering the mitigation of threats were found.
  - All the initiatives were integrated to at least one activity of the Software Development Lifecycle (SDLC), while 7 show signs of being adopted in the industry.
  - The mapping found only 15 studies that covered 11 different iniatiatives.
  - Only two techniques presented scientific evidence of its results through controlled experiments, while others selected studies presented informal case studies or examples.
- Fondecyt Project: Research
  - A Theoretical Framework for quality assessment of security metrics.
  - How to evaluate security in software development

Questions?

¡Muchas Gracias!, Thank you!, Arigatou Gozaimashita!, Grazie!


G. S. StatCounter, "Top 5 desktop, tablet and console browsers," 2015. [Online]. Available: http://gs.statcounter.com/


A. Grosskurth and M. W. Godfrey, "Architecture and evolution of the modern web browser," uRL: `http://grosskurth.ca/papers.html#browser-archevol`. Note: submitted for publication.


M. Larrondo-Petrie, K. Nair, and G. Raghavan, "A domain analysis of web browser architectures, languages and features," in *Southcon/96. Conference Record*, Jun 1996, pp. 168–174.


A. Grosskurth and M. W. Godfrey, "A reference architecture for web browsers," 2005, pp. 661–664, uRL: `http://grosskurth.ca/papers.html#browser-refarch`.


M. W. Godfrey and E. H. S. Lee, "Secrets from the Monster: Extracting Mozilla's Software Architecture," *In Proc. of 2000 Intl. Symposium on Constructing software engineering tools (CoSET 2000*, pp. 15–23, 2000. [Online]. Available: http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.42.9211


A. Systems and N. Lwin, "Agent Based Web Browser," *2009 Fifth International Conference on Autonomic and Autonomous Systems*, 2009.