Malware al context del programari lliure

Ignaci Colomina Torregrosa

13 de juny de 2012

Resum

Aquest document mostra els resultats d'una recerca basada en cas d'estudi on s'avalua la fortalesa de dos comunitats de programari lliure. L'avaluació d'aquesta fortalesa es du a terme amb una exploració la qual té com a objectiu d'esbrinar si aquestes comunitats acomplixen una serie de procediments els quals les ajuden a protegir-se davant atacs.

Índex

1 Introducció **3**
- 1.1 El problema . . . 3
- 1.2 L'estat de l'art . . . 3
- 1.3 La metodologia de recerca . . . 4

2 Desenvolupament de la recerca **5**
- 2.1 Breu descripció dels projectes . . . 5
- 2.2 Els procediments per a l'estudi . . . 6
  - 2.2.1 Importància dels procediments . . . 6
  - 2.2.2 Definició dels procediments . . . 6
- 2.3 Avaluació dels procediments . . . 8
  - 2.3.1 Criteris per a la cerca i assignació de valoracions . . . 8
  - 2.3.2 Taula resum d'assignacions . . . 9
  - 2.3.3 Qualificació dels procediments . . . 9

3 Anàlisi dels punts febles dels projectes **10**
- 3.1 Projecte Symfony . . . 10
- 3.2 Projecte Chromium . . . 11
- 3.3 Altres projectes que si contemplen els procediments no acomplits . . . 11

4 Conclusions **12**

5 Futures línies de treball **12**

6 Anèxes **14**
- 6.1 Taula detallada d'avaluació de procediments . . . 14
  - 6.1.1 Procediment P.1 . . . 14
  - 6.1.2 Procediment P.2 . . . 14
  - 6.1.3 Procediment P.3 . . . 15
  - 6.1.4 Procediment P.4 . . . 15
  - 6.1.5 Procediment P.5 . . . 16
  - 6.1.6 Procediment P.6 . . . 16

6.1.7 Procediment P.7 17
6.1.8 Procediment P.8 18
6.1.9 Procediment P.9 18
6.1.10 Procediment P.10 19
6.1.11 Procediment P.11 19
6.1.12 Procediment P.12 19

1 Introducció

La introducció d'aquest document es dividirà en tres parts fonamentals. A la primera es realitzarà una descripció del problema a tractar, a la segona s'exposarà quin és l'estat del art en referència al problema plantejat i, en tercer lloc, s'indicarà quina metodologia de recerca es farà servir a la present recerca.

1.1 El problema

El problema que es planteja en aquesta recerca té el seu arrel en la principal característica del programari lliure: *la possibilitat de tercers d'accedir al codi font de programes per a estudiar-lo, corregir-lo i millorar-lo.* La finalitat d'aquesta llibertat és escalar i fer créixer el programari mitjançant l'ajuda de membres de qualsevol part del món, els quals s'organitzen amb estructures anomenades comunitats, i col·laboren amb l'escriptura de codi. Desgraciadament, les intencions de tots aquestos membres no són sempre benèvoles i alguns se'n aprofiten del fet de poder accedir al codi per a introduir programari maliciós i, així, poder trencar la seguretat dels sistemes, violar-ne l'accés i treure'n profit de qualsevol tipus, des de guanyar fama entre la comunitat a obtindre ingressos. Payne C a [8], realitza una reflexió interessant respecte al malware i al programari lliure. En aquesta reflexió fa referència a la robustesa del programari lliure per a fer front al malware ja que, gràcies a l'accés al codi, molts programadors poden cercar, detectar i corregir anomalies. Altrament, també s'anomena que, a causa de l'obertura del codi, les possibilitats d'infeccions són majors ja que l'anàlisi i l'estudi del codi font és possible. Hi han molts casos de sistemes lliures que han sigut atacats, dels quals podem destacar-ne els següents:

- L'aplicació web de codi obert *osCommerce* la qual va ser infectada per un malware anomenat willysy el qual injectava codi maliciós a les pàgines dels comerciants tal i com s'indica a [1].
- El repositori principal del kernel de Linux va ser atacat mitjançant la inserció d'un cavall de troia segons s'explica a [6]. En aquest recurs bibliogràfic, s'esmenta que el cavall de troia va aconseguir accedir als repositoris principals (Hera i Odin) a través de l'ordinador del desenvolupador H Peter Avin.
- De manera semblant al cas anterior, i tal i com es comenta a [7], els repositoris que emmagatzemen les pàgines web del frontend de GNU també van ser l'objectiu d'un atac on es va fer servir la tècnica d'SQL Injection[1] per accedir, en una primera instància, a l'aplicació de gestió de bugs

Per a poder fer front a aquestes situacions, les comunitats de programari lliure han modificat les seues estructures al llarg del temps per a fer-les menys vulnerables davant atacs informàtics i intents d'inserció de malware. Els procediments que es prenen per a millorar aquest tipus de fortalesa seran la base d'aquesta recerca.
A la següent secció, veurem quin és l'estat de l'art de les comunitats de programari lliure a l'actualitat tenint en compte allò que s'acaba de descriure com a problema.

1.2 L'estat de l'art

Des de l'aparició del programari lliure i de tot el fenomen que l'acompanya, la manera d'organitzar les comunitats s'ha perfeccionat força. El problema definit a l'apartat anterior ha sigut, sens dubte, una motivació fonamental per a millorar les comunitats i fer-les més segures, encara que s'ha de tenir en compte que altres fenòmens, com ara la utilització del programari lliure com a part d'un model de negoci empresarial, també han contribuït a millorar-les. Així doncs, s'ha arribat a un punt on les comunitats han passat de ser grups de persones que simplement col·laboren en la producció de programari per a convertir-se en estructures complexes que engloben altres elements com el següents:

[1] Mètode d'inserció de codi maliciós el qual aprofita la debilitat de les consultes SQL embegudes dins d'un altre programari (per exemple un script web) per afegir codi SQL de l'atacant

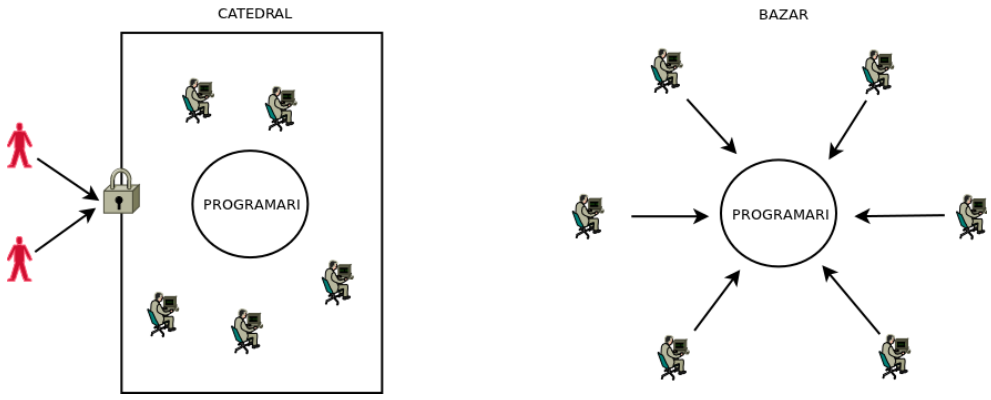

Figura 1: La catedral i el basar

- Papers (Rols): Els membres de les comunitats solen posseir papers definits que identifiquen la seua funció
- Seguretat: El codi del programari lliure sol emmagatzemar-se en repositoris distribuïts l'accés dels quals està protegit amb credencials obligatòries.
- Protocols i promoció: Sovint, s'inclouen protocols ben definits per a la promoció (accés a rols de major importància) dels membres. La figura que apareix a [2] a la pàgina 59 (*The onion model*), representa la responsabilitat dels membres d'una comunitat on els membres amb més responsabilitats ocupen els llocs més propers al nucli de la ceba, mentre que els de menys responsabilitats ocupen els llocs més allunyats del nucli. Ara per ara, aquest model s'implementa en bona part de les comunitats existents de programari lliure.
- Autoritat: La presa de decisions es deixa, cada vegada més, en mans d'un nombre determinat de persones més properes al projecte i que assumixen els rols amb més grau de responsabilitat. Com es pot deduir, aquesta característica, esta completament relacionada amb la seguretat.

A [9], Eric Raymond fa referència a la catedral i al basar com a dos models completament contraris. A la figura 1 podem veure com la part que representa la catedral té l'accés bloquejat per a programadors externs, en canvi, a la part que representa el basar, totes aquelles persones que volgueren col·laborar tindrien les portes obertes. Els papers que representen els membres del basar no estan definits de la mateixa manera (ni molt menys) que a la catedral. Per a veure-ho des d'un punt de vista pràctic, una catedral podria representar el model de l'equip de treball intern d'una empresa (analistes, programadors, cap de projecte, testers etc) i un basar posseiria una estructura molt més descentralitzada on qualsevol programador podria realitzar qualsevol tasca, encara que, sol haver-hi una persona que actua com a *líder*.
La transformació, que han experimentat les comunitats de programari lliure que s'ha comentat al principi d'aquest punt aprofita, doncs, les característiques de seguretat que proporciona la catedral amb els avantatges de col·laboració i intercanvi de coneixements propis del basar. La figura 2 en mostra un exemple d'aquesta última afirmació.

1.3 La metodologia de recerca

Com que el tema fonamental d'aquesta recerca són els procediments utilitzats pels projectes de programari lliure per a gestionar les seues comunitats, l'estratègia de la qual en podem traure més profit és el cas d'estudi. La realització d'un o més casos d'estudi poden servir per a conéixer a fons els procediments o protocols que utilitza una comunitat per a realitzar la seua gestió interna i, aleshores, analitzar la "fortalesa" amb la qual és capaç de fer front a usuaris malèvols.
Les característiques principals que definiran aquestos casos d'estudi són les següents:

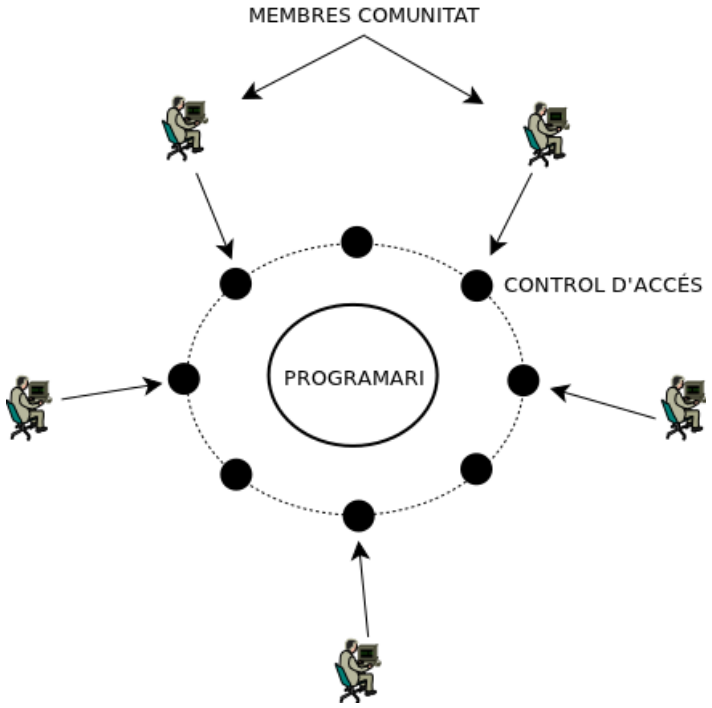

Figura 2: Comunitat restringuida

- Els casos d'estudi seran exploratius. Es realitzara l'anàlisi de la fortalesa de les comunitats prenent com a punt de partida la implementació de una serie de procediments que, hipotèticament poden ajudar a millorar la qualitat del programari desenvolupat i protegir-se davant la inserció de programari maliciós.
- La instància del tema de la recerca són les comunitats de programari lliure les quals són cas d'estudi
- Els procediments base serviran com a *framework conceptual* per a generalitzar els casos d'estudi d'aquesta recerca a altres casos.
- Internet serà la base d'observació de les comunitats d'estudi, és a dir, s'aprofitarà al màxim la informació de les pàgines web de les comunitats i els fòrums de discussió d'aquestes.

2 Desenvolupament de la recerca

En aquesta secció, en primer lloc, es descriuran breument els projectes que seran cas d'estudi i després s'enumeraran tots aquells procediments amb els quals s'avaluaran els projectes. Després es realitzarà l'avaluació dels procediments per a cada projecte i, al cas de no acomplir algun dels procediments, es comentarà com un atacant se'n podria aprofitar. Per a finalitzar, es qualificaran els projectes en funció del grau de compliment dels procediments.

2.1 Breu descripció dels projectes

A continuació mostra una breu descripció dels projectes que seran cas d'estudi:

- **Symfony**: Symfony és un framework per al desenvolupament d'aplicacions web amb el llenguatge PHP sota el paradigma MVC (Model-vista-controlador). Symfony proporciona una estructura de classes per a la implementació ràpida i senzilla d'aquest tipus de desenvolupament.

  - Llicència de distribució: MIT
  - Lloc web del projecte: `http://www.symfony.com/`

- **Chromium**: Chromium és un navegador d'Internet de codi obert l'objectiu del qual és la rapidesa, la seguretat i la estabilitat. El codi de Chromium es la base de *Google Chrome*.
  - Llicència de distribució: La part desenvolupada per Google s'allibera sota llicència BSD mentres que altres parts de Chromium s'alliberen sota diferents llicències, com ara: MIT, LGPL, Ms PL o la trillicència MPL/GPL/LGPL
  - Lloc web del projecte: `http://www.chromium.org/Home`

2.2 Els procediments per a l'estudi

2.2.1 Importància dels procediments

Abans de començar amb la definició dels procediments, cal definir els termes d'importància que s'assignaran a cada procediment.

- **Alta**: El no compliment d'un procediment d'alta importància pot deixar-hi buits de fàcil accés per als hackers i, aleshores, per al malware
- **Normal**: El no compliment d'un procediment de normal importància hi pot deixar buits, però amb menys nivell que els anteriors. Aquestos procediments doten de més robustesa als d'alta importància.

Com a símil, podríem comparar la importància alta i normal amb l'escala numèrica de riscos que es realitza a un dels exemples del l'estàndard *MAGERIT* v2 (ISO1779, Metodologia d'Anàlisi y Gestió de Riscos als Sistemes d'Informació) el qual es pot consultar a [15]. Aquesta escala consta d'un interval del 0 al 5 on el risc augmenta a mesura que augmenta el número. En aquest cas, una importància normal equivaldria a un risc entre 0 i 3, i una importància alta equivaldria a un risc de 4 o 5.

2.2.2 Definició dels procediments

A continuació s'enumeren els procediments de base a partir dels quals es realitzarà la exploració de les comunitats que seran cas d'estudi amb la fi de poder catalogar-les. Aquestos procediments s'agruparan fent servir els elements de *salvaguarda* definits l'estàndard *MAGERIT* v2 els quals es poden consultar a [15]. Pel que respecta a aquest document, es faran servir les salvaguardes: *Control d'accés*, *Gestió de privilegis*, *Organització de la seguretat* i *Registre d'actuacions* .
Abans d'agrupar els procediments, es definirà breument quin es el context d'actuació de les salvaguardes utilitzades en aquest document.

- **Control d'accés** : Requerix identificació de la persona aspirant a obtindre accés al sistema
- **Gestió de privilegis**: Es coordina amb el control d'accés per a gestionar i controlar els privilegis que posseïx el aspirant que ha obtingut accés al sistema
- **Organització de la seguretat**: Definix l'organització del sistema en termes de seguretat com ara rols etc.
- **Registre d'actuacions**: Controla l'activitat que es realitza sobre el sistema per part d'usuaris amb accés i privilegis

Una vegada definides les salvaguardes, es llisten els procediments sense agrupar. Després es tornaran a llistar agrupats fent servir aquestes salvaguardes.

- P.1.- S'utilitza un repositori de codi
  - P.1.1.- El repositori és centralitzat (Alta)
  - P.2.2.- El repositori és distribuït, però s'utilitza un màster únic sobre el qual tots els usuaris realitzen les operacions PULL/PUSH (Alta)
- P.2.- Existixen certes persones que controlen qui accedix al repositori i perquè (Alta)
- P.3.- Existix un procés de sol·licitud i accés al repositori regulat pel Core Team o autoritat central (Alta)
- P.4.- Es seguixen convenis de codi que tots els programadors han d'acomplir (Normal)
- P.5.- S'utilitzen procediments de control de la qualitat del programari (Alta)
- P.6.- S'utilitza un sistema de tickets per a controlar quins canvis, correccions o millores s'estan realitzant (Alta)
- P.7.- S'utilitza un procediment o protocol d'actuació mitjançant el qual es realitzen revisions del codi resultant de les noves funcionalitats o correccions realitzades (Alta)
  - P.7.1.- El procediment inclou una revisió per part dels membres del Core-Team (Alta)
  - P.7.2.- El procediment inclou una separació del codi per revisar en una branca diferent a la principal (Alta)
  - P.7.3.- El procediment inclou una relació directa entre el sistema de tickets i el procediment de revisió (Normal)
- P.8.- La jerarquia de la comunitat seguix un model de ceba (Alta)
- P.9.- El Core-team decidix quins canvis s'afegixen a una nova release (Alta)
- P.10.- S'utilitza un procediment per arribar a una release (Alta)
- P.11.- S'utilitza un sistema per a nombrar versions que identifique l'objectiu d'aquestes (A, A.B, A.B.C etc) (Normal)
- P.12.- La descàrrega del codi per part dels col·laboradors es realitza tenint en compte aspectes de seguretat que puguen evitar fakes maliciosos del projecte (Alta)

A continuació, es mostren els procediments agrupats:

- **Control d'accés i gestió de privilegis**:
  - P.1.- S'utilitza un repositori de codi
    - P.1.1.- El repositori és centralitzat (Alta)
    - P.2.2.- El repositori és distribuït, però s'utilitza un màster únic sobre el qual tots els usuaris realitzen les operacions PULL/PUSH (Alta)
  - P.2.- Existixen certes persones que controlen qui accedix al repositori i perquè (Alta)
  - P.3.- Existix un procés de sol·licitud i accés al repositori regulat pel Core Team o autoritat central (Alta)
  - P.9.- El Core-team decidix quins canvis s'afegixen a un nova release (Alta)
  - P.12.- La descàrrega del codi per part dels col·laboradors es realitza tenint en compte aspectes de seguretat que puguen evitar fakes maliciosos del projecte (Alta)

- **Registre d'actuacions**:
  - P.4.- Es seguixen convenis de codi que tots els programadors han d'acomplir (Normal)
  - P.5.- S'utilitzen procediments de control de la qualitat del programari (Alta)
  - P.6.- S'utilitza un sistema de tickets per a controlar quins canvis, correccions o millores s'estan realitzant (Alta)
  - P.7.- S'utilitza un procediment o protocol d'actuació mitjançant el qual es realitzen revisions del codi resultant de les noves funcionalitats o correccions realitzades (Alta)
    * P.7.1.- El procediment inclou una revisió per part dels membres del Core-Team (Alta)
    * P.7.2.- El procediment inclou una separació del codi per revisar en una branca diferent a la principal (Alta)
    * P.7.3.- El procediment inclou una relació directa entre el sistema de tickets i el procediment de revisió (Normal)
  - P.10.- S'utilitza un procediment per arribar a una release (Alta)
  - P.11.- S'utilitza un sistema per a nombrar versions que identifique l'objectiu d'aquestes (A, A.B, A.B.C etc) (Normal)
- **Organització de la seguretat. Rols**
  - P.8.- La jerarquia de la comunitat seguix un model de ceba (Alta)

2.3 Avaluació dels procediments

En aquesta secció ens centrarem en descriure quin ha sigut el protocol que s'ha utilitzat per a avaluar els projectes i assignar les valoracions duals: *projecte-procediment* així com en mostrar una taula resum amb les assignacions realitzades a cada parell *projecte-procediment*.

2.3.1 Criteris per a la cerca i assignació de valoracions

Abans de poder assignar valoracions a cada parell, s'haurà de cercar al lloc web de la comunitat d'ambdós projectes la zona concreta on s'especifica que s'acomplixen els requeriments establerts a cadascun dels procediments.
Per a poder segmentar la cerca d'informació tenint en compte que l'objectiu és d'esbrinar si la comunitat acomplix amb els procediments, s'intentarà segmentar la cerca en funció del tipus de procediment que s'està tractant, és a dir, si volem cercar informació sobre procediments que guarden relació amb el repositori de codi o amb les contribucions de codi en concret, s'haurà de segmentar la cerca mitjançant paraules clau del tipus: *How to be involved, Contributing code, Submmiting a patch* etc. En aquest punt, s'haurà de tindre en compte que paraules les quals poden ajudar-nos a trobar informació sobre com fer una contribució de codi també poden ajudar-nos a trobar informació sobre el tipus de repositori que utilitza la comunitat ja que, potser que la secció on s'haja arribat amb eixa paraula clau ens explique com fer una contribució a nivell de codi i com actualitzar el repositori amb la nova contribució. Com a exemple, si la secció on es trobem ens diguera que hem d'executar una ordre del tipus *git merge branchname*, aleshores ja sabrien que el repositori que la comunitat utilitza és distribuït perquè GIT és un repositori distribuït.
Una vegada realitzada la cerca per a tots els projectes i procediments, les assignacions es realitzaran fent servir els següents criteris:

- En cas de no trobar-se el procediment durant la cerca o especificar-se que aquest no s'acomplix, s'assignarà una valoració: *No acomplix* al parell *projecte-procediment*

- En cas de trobar-se que el procediment s'acomplix, però que hi han alguns punts que no es duen a terme, s'assignarà una valoració: *Parcialment*
- Per últim, si es troba que el procediment s'acomplix completament, s'assignarà una valoració: *Acomplix*

2.3.2 Taula resum d'assignacions

A continuació, es mostra una taula resum amb les assignacions realitzades a cadascun dels projectes. Als annexes d'aquest article, es pot trobar una taula amb informació més detallada i de possible interés per al lector.

Procediment	**Symfony**	**Chromium**
P.1.1	No acomplix	Acomplix
P.1.2	Acomplix	Acomplix
P.2	Acomplix	Acomplix
P.3	Parcialment	Acomplix
P.4	Acomplix	Acomplix
P.5	Parcialment	Acomplix
P.6	Acomplix	Acomplix
P.7.1	Acomplix	Acomplix
P.7.2	Acomplix	Acomplix
P.7.3	Acomplix	Acomplix
P.8	Acomplix	Acomplix
P.9	Acomplix	Acomplix
P.10	Parcialment	Acomplix
P.11	Acomplix	Acomplix
P.12	Parcialment	Parcialment

2.3.3 Qualificació dels procediments

La qualificació dels procediments es realitzarà tenint en compte les assignacions de l'apartat anterior per a cadascun dels projectes junt amb els criteris que es mostren a continuació:

- Primer es calcularà la puntuació de cada projectes fent servir les segents regles:
  - Els procediments acomplits sumaran 1 punt
  - Els procediments acomplits parcialment sumaran 0.5 punts
  - Els procediments no acomplits restaran 1 punt
  - Els procediments 1.1 i 1.2 seran mútuament excloents, és a dir, com que l'objectiu d'ambdós és de tenir un repositori de codi centralitzat, amb el compliment d'un d'ells és suficient. Així doncs, si un dels projectes els acomplix ambdós, com és el cas de Chromium, només és sumarà la puntuació corresponent a un d'ells.
- Després, a partir de les puntuacions obtingudes es qualificarà el projecte fent servir els següents criteris:
  - Fortalesa òptima: 14 punts
  - Fortalesa alta: (11 - 13(punts
  - Fortalesa mitja:)8 - 11(punts
  - Fortalesa baixa: (6 - 8(punts

  - Fortalesa crítica: (0 - 6) punts

- Com a nota important, cal ressaltar que la suma o resta de puntuacions és completament independent de l'agrupació dels procediments descrita a l'apartat 2.2.2.

Fent servir, doncs, aquestos criteris i les dades de la taula anterior, el nombre final de procediments acomplits per cada projecte i importància i, aleshores, les seues qualificacions són els que es resumixen a continuació:

- **Per a Symfony**
  - Puntuació per procediments parcials: 4 – 2 punts
  - Puntuació per procediments acomplits: 10 – 10 punts
  - Penalització per procediments no acomplits: No penalitza perquè es tracta del no compliment del procediment 1.2. Com que 1.1 i 1.2 són mútuament excloents, simplement es sumarà la puntuació del procediment 1.1.
  - Qualificació: 12 punts – Fortalesa alta
- **Per a Chromium**
  - Puntuació per procediments parcials: 1 – 0.5 punts
  - Puntuació per procediments acomplits: 14 – 13 punts. Restem un punt perquè acomplix tant el procediment 1.1 com el 1.2, però, com s'ha dit, són mútuament excloents.
  - Penalització per procediments no acomplits: 0 – 0 punts
  - Qualificació: 13.5 punts – Fortalesa alta

3 Anàlisi dels punts febles dels projectes

Després d'haver avaluat el compliments dels procediments en relació als projectes d'estudi, aquest punt es centrarà en com un atacant en podria treure profit de la debilitat presentada a causa del no compliment d'un procediment.

3.1 Projecte Symfony

- **Procediment P.3 (S'acomplix parcialment)**: Tindre un procediment que actue com a via per arribar a convertir-se en *commiter* en una eina necessària per tancar-hi una porta més al programari maliciós. La raó de la importància d'acomplir aquest procediment és que, encara que el Core Team revise el codi del commit d'un membre abans de publicar-lo com a vàlid, podria donar-se el cas on el propi Core Team no detectara la inserció de codi maliciós per part d'un *commiter* que, a la fi, ha resultat ser una persona amb intencions malèvoles. Així doncs, un procediment que analitze a fons el comportament i compromís d'un membre amb el projecte pot resultar de gran ajuda per a evitar aquest tipus de situacions.
- **Procediment P.5 (S'acomplix parcialment)**: Encara que es demane a tots els programadors que col·laboren que afegisquen tests unitaris i funcionals en tots els desenvolupaments que realitzen, un *commiter* malèvol podria afegir-hi programari maliciós, tant unitari com funcional, i utilitzar el propi test per a camuflar el malware. Amb eines de testing pròpies, és a dir, no preparades per els col·laboradors sinó desenvolupades per l'equip intern i només configurables, es podrien estalviar situacions d'aquest tipus.

- **Procediment P.10 (S'acomplix parcialment)**: És important mantindre un calendari específic de llançament de noves versions. Si no és així, podria ser fàcil realitzar un fake maliciós amb un nom de release fals i fer que els usuaris es baixaren eixa release infectada.

- **Procediment P.12 (S'acomplix parcialment)**: Iniciatives com GITHUB són ben valorades ja que permeten allotjar projectes de codi obert i controlar-ne les versions i els canvis que es realitzen, dit d'un altra manera, permeten centralitzar el control de versions en una infraestructura comuna i afegir seguretat per als projectes que s'allotgen. Tanmateix, hi ha que tindre en compte que si el propi Github fos l'objectiu d'un atac i aquest resultara satisfactori, se'n podrien veure afectats tots els projectes que l'utilitzen per al control de versions. A [3] s'exposa un cas on Egor Homakov va aconseguir accés d'administrador a Github i, en conseqüència, a molts dels projectes allotjats, com ara *Ruby on Rails*, *Linux* i molts altres.
  D'altra banda, si el projecte Symfony es descarrega via un paquet *zip* o *tar.gz* s'hauria d'incloure una verificació del paquet mitjançant un codi hash, md5 etc, de manera que s'evitara un fake maliciós aprofitant aquesta vulnerabilitat.

3.2 Projecte Chromium

- **Procediment P.12. (S'acomplix parcialment)**: De la mateixa manera que Symfony, la descarrega de *Chromium via paquet no implica la la verificació d'aquest mitjançant cap tipus de mecanisme la qual cosa, i de la mateixa manera que s'ha comentat per a Symfony, es podria realitzar un fake maliciós.*

A continuaci es mostra el tipus d'atac normalitzat segons l'estàndard *MAGERIT* v2 on s'englobaria cadascun dels atacs que es podrien dur a terme per possibles usuaris malèvols els quals s'acaben de descriure.

- Procediment P.3 – Abús de privilegis d'accés
- Procediment P.5 – Difusió de programari malèvol
- Procediment P.10 – Difusió de programari malèvol
- Procediment P.12 – Abús de privilegis d'accés i Difusió de programari malèvol

3.3 Altres projectes que si contemplen els procediments no acomplits

Com que, tant Chromium com Symfony només acomplixen parcialment el procediment P.12, resulta d'interès anomenar un projecte de codi obert que si acomplira aquest procediment a més grau que els projectes que són cas d'estudi.

- **Apache HTTP Server**: Per a poder instal·lar el servidor web Apache, si aquest s'ha descarregat via un paquet des de la web oficial, abans de la instal·lació s'ha de verificar la integritat del paquet mitjançant l'us de verificació MD5 o PGP. A [4] es pot trobar més informació i aquells passos que s'han de dur a terme (mitjançant comandaments) per a verificar-ne la integritat.

- **MySQL Database Server**: El DBMS (Database Management System), MySQL, tamb permet comprovar la integritat del paquet descarregat mitjançant l'us de sumes sume MD5 i firmes GnuPG tal i com s'indica a [5]

- **gSOAP Toolkit Software Download Instructions**: La llibreria gSOAP per a l'accés i consum de serveis web tipus SOAP i també per a serveis no estàndars (intercanvi XML) permet la comprovació mitjançant una suma de comprovació MD5.

- **SquirrelMail**: SquirrelMail, el gestor de web mail lliure i escrit en PHP, permet la comprovació fent us de MD5, GPG i SHA1, tal i com es comprova a [11]
- **Apache Ant**: Apache Ant, la llibreria Java per a facilitar la construcció i desplegament d'aplicacions realitzades amb aquest mateix llenguatge, permet verificació via MD5, SHA1, SHA512 i GPG segons s'indica a [12]
- **Cups**: Cups, el sistema que utilitzen moltes distribucions de Linux com a sistema d'impressió, permet la comprovació a MD5 com s'indica a [13]
- **Puppet Labs**: Puppet, un programari per a l'administració de sistemes *nix el qual proporciona funcionalitats con la d'afegir usuaris, actualitzar configuracions, instal·lar paquets etc, també permet la comprovació amb GPG segons les indicacions de [14]

4 Conclusions

Al llarg de tot aquest document, s'ha comprovat que les comunitats són la base a partir de les quals es construïx un projecte de programari lliure. Així doncs, la conclusió principal que surt de l'anàlisi d'aquesta recerca és que la fortalesa que acompanya a un projecte de programari lliure està fortament lligada a la manera d'organitzar la comunitat i a la qualitat de la seguretat que aquesta proporciona en tots els àmbits, des de la gestió dels seus membres a la seguretat, protecció i emmagatzemament del codi.

5 Futures línies de treball

A continuació, es descriuran dues possibles futures línies de treball associades a aquesta la recerca d'aquest document:

- A partir dels casos d'estudi realitzats en aquesta recerca, dur a terme el disseny d'uns procediments estandaritzats que puguen servir com a guia per a la protecció i enfortiment de noves comunitats i projectes de programari lliure.
- Realitzar un anàlisi de les pèrdues econòmiques i de viabilitat que pot presentar un projecte de programari lliure el qual no acomplira els procediments establerts a la primera proposta de futures línies de treball.

Referències

[1] "Is your website accused of phising? Phishing sites on the rise due to web application vulnerabilities." [Online]. Disponible en: http://info.brandprotect.com/Blog/bid/64544 Anunciat en: 2011/10/03

[2] Mark Aberdour. "Achieving Quality in Open Source Software, " en *IEEE Software* Volum: 24 No: 1, Pàgines: 58-64, Publicació: Gener del 2007

[3] "Github hacked." [Online]. Disponible en: http://it.slashdot.org/story/12/03/05/1243235/github-hacked Anunciat en: 2012/03/05

[4] "Verify the integrity of the files" [Online] Disponible en: http://httpd.apache.org/download.cgi#verify Darrera consulta: 2012/06/13

[5] "MySQL Community Server 5.5.24" [Online] Disponible en: http://dev.mysql.com/downloads/mysql/ Darrera consulta: 2012/06/13

[6] "Kernel.org Linux repository rooted in hack attack." [Online]. Disponible en: http://www.theregister.co.uk/2011/08/31/linux_kernel_security_breach/, anunciat en Enterprise Security - data: 2011/08/31

[7] "Free Software Foundation's software repository hacked" [Online]. Disponible en: http://www.infoworld.com/d/open-source/free-software-foundations-software-repository-hacked-345, anunciat en Desembre del 2010

[8] Payne C "On the security of Open Source Software," en *INFORMATION SYSTEMS JOURNAL*, Volum 12 No: 1, pàgines 61-78, Publicació: Gener del 2002

[9] Eric S. Raymond. "The Cathedral and the bazaar. " Editorial: O'Reilly, N Pàgines: 247, Pàgines: 41-54, Publicació: Gener del 2001

[10] "gSOAP Download Package MD5 Checksums" [Online] Disponible en: http://www.cs.fsu.edu/~engelen/soap.html Darrera consulta: 2012/06/13

[11] "SquirrelMail Downloads" [Online] Disponible en: http://squirrelmail.org/download.php Darrera consulta: 2012/06/13

[12] "Apache Ant Binary Distributions" [Online] Disponible en: http://ant.apache.org/bindownload.cgi Darrera actualització: 06/13/2012 19:55:50

[13] "CUPS Download" [Online] Disponible en: http://www.cups.org/software.php Darrera consulta: 2012/06/13

[14] "Puppet Labs download options" [Online] Disponible en: http://puppetlabs.com/misc/download-options/ Darrera consulta: 2012/06/13

[15] "EAR / PILAR - Entorno de Análisis de Riesgos" [Online] Disponible en: https://www.ccn-cert.cni.es/index.php?option=com_wrapper&view=wrapper&Itemid=187&lang=es Darrera consulta: 2012/06/13

6 Anèxes

6.1 Taula detallada d'avaluació de procediments

6.1.1 Procediment P.1

SUBPROCEDIMENT P.1.1		
PROJECTE	VALORACIO	COMENTARIS
Symfony	No acomplix	El repositori GIT utilitzat amb el projecte Symfony es distribuït
Chromium	Acomplix	Chromium permet utilitzar el repositori Subversion per a la descàrrega del codi
Referències	Symfony	`http://symfony.com/doc/current/contributing/code/patches.html`
	Chromium	`http://www.chromium.org/developers/how-tos/get-the-code`
SUBPROCEDIMENT P.1.2		
PROJECTE	VALORACIO	COMENTARIS
Symfony	Acomplix	El codi de Symfony es descarrega des de Github
Chromium	Acomplix	També permet l'us de GIT com a màster únic
Referències	Symfony	`http://symfony.com/doc/current/contributing/code/patches.html`
	Chromium	`http://code.google.com/p/chromium/wiki/UsingNewGit`

6.1.2 Procediment P.2

PROJECTE	VALORACIO	COMENTARIS
Symfony	Acomplix	EL Core Team revisa els moviments al repositori i publica els canvis d'altres membres al màster branch després de revisar-ho
Chromium	Acomplix	Els canvis que es realitzen han de ser revisats per un "reviewer" assignat pel Core Team
Referències	Symfony	`http://symfony.com/doc/current/contributing/code/patches.html`
	Chromium	`http://www.chromium.org/developers/contributing-code` – Secció *Request review*

6.1.3 Procediment P.3

PROJECTE	VALORACIO	COMENTARIS
Symfony	Parcialment	Symfony 2 no establix un procés descrit com ho fa Chromium, però els canvis que efectuen els membres els revisa i publica el Core Team
Chromium	Acomplix	Un futur *committer* ha d'acomplir certes exigències per a poder arribar-hi. A més, un altre *committer* ha de nominar-hi al nou mitjançant un correu de confirmació a *committers@chromium.org*
Referències	Symfony	`http://symfony.com/doc/current/contributing/code/patches.html`
	Chromium	`http://www.chromium.org/getting-involved/become-a-committer` – Secció *How do I become a committer?*

6.1.4 Procediment P.4

PROJECTE	VALORACIO	COMENTARIS
Symfony	Acomplix	No hi han comentaris addicionals
Chromium	Acomplix	No hi han comentaris addicionals
Referències	Symfony	`http://symfony.com/doc/current/contributing/index.html` – Seccions *Coding Standards, Code Conventions*
	Chromium	`http://www.chromium.org/developers/coding-style`

6.1.5 Procediment P.5

PROJECTE	VALORACIO	COMENTARIS
Symfony	Parcialment	Symfony 2 demana als usuaris que treballen en un bug o una nova funcionalitat que afigen tests unitaris per assegurar-se que el codi funciona correctament
Chromium	Acomplix	Chromium obliga a tots els col·laboradors a configurar i executar un test automàtic amb les eines de *Chromium BuildBots*. Si hi ha qualsevol incidència, aquesta es pot reportar via IRC o email. A més també s'han de realitzar test unitaris per part dels col·laboradors
Referències	Symfony	`http://symfony.com/doc/current/contributing/code/patches.html` – Secció *Working on a Patch, línia 16*
	Chromium	`http://www.chromium.org/developers/testing` i `http://www.chromium.org/developers/testing/chromium-build-infrastructure` i `http://www.chromium.org/developers/contributing-code` – Secció *Get your code ready*, punt 2

6.1.6 Procediment P.6

PROJECTE	VALORACIO	COMENTARIS
Symfony	Acomplix	Symfony 2 utilitza el sistema de tickets de Github
Chromium	Acomplix	Chromium utilitza el sistema de tickets de Google Code
Referències	Symfony	`https://github.com/symfony/symfony-docs/issues/`
	Chromium	`http://code.google.com/p/chromium/issues/list`

6.1.7 Procediment P.7

SUBPROCEDIMENT P.7.1		
PROJECTE	VALORACIO	COMENTARIS
Symfony	Acomplix	Sense comentaris
Chromium	Acomplix	Sense comentaris
Referències	Symfony	`http://symfony.com/doc/current/contributing/code/patches.html` – al paràgraf final
	Chromium	`http://www.chromium.org/developers/contributing-code` – Secció *The review process*
SUBPROCEDIMENT P.7.2		
PROJECTE	VALORACIO	COMENTARIS
Symfony	Acomplix	Sense comentaris
Chromium	Acomplix	S'utilitza l'eina Google Rietvield per als procediments de revisió. Aquesta inclou la separació en branques
Referències	Symfony	`http://symfony.com/doc/current/contributing/code/patches.html` – al paràgraf final
	Chromium	`http://code.google.com/p/rietveld/`
SUBPROCEDIMENT P.7.3		
PROJECTE	VALORACIO	COMENTARIS
Symfony	Acomplix	Les revisions es realitzen en relació als tickets de GITHUB
Chromium	Acomplix	El sistema de tickets i Rietveld pertanyen a Google Code
Referències	Symfony	`http://symfony.com/doc/current/contributing/code/patches.html` – al paràgraf final
	Chromium	`http://code.google.com/p/rietveld/`

6.1.8 Procediment P.8

PROJECTE	VALORACIO	COMENTARIS
Symfony	Acomplix	Els membres del Core Team són les persones més properes al projecte, al seu creador i a la empresa Sesio Labs
Chromium	Acomplix	Els membres del Core-Team i els commiters són els encarregats de les revisions del codi, les quals són les persones més properes al projecte
Referències	Symfony	`https://connect.sensiolabs.com/login`
	Chromium	`http://www.chromium.org/getting-involved/become-a-committer`

6.1.9 Procediment P.9

PROJECTE	VALORACIO	COMENTARIS
Symfony	Acomplix	Existix un calendari de llançament de noves releases proporcionat per l'autoritat del projecte
Chromium	Acomplix	Existix un calendari de llançament de noves releases proporcionat per l'autoritat del projecte
Referències	Symfony	`http://symfony.com/blog/category/releases`
	Chromium	`http://www.chromium.org/developers/calendar`

6.1.10 Procediment P.10

PROJECTE	VALORACIO	COMENTARIS
Symfony	Parcialment	No s'especifica un procediment concret, però al changelog es pot veure com per cada canvi s'agrupen una serie d'incidències resoltes que conformen la nova release
Chromium	Acomplix	Existix un calendari de llançament de noves releases proporcionat per l'autoritat del projecte
Referències	Symfony	`https://github.com/symfony/symfony/blob/2.0/CHANGELOG-2.0.md`
	Chromium	`http://www.chromium.org/developers/calendar`

6.1.11 Procediment P.11

PROJECTE	VALORACIO	COMENTARIS
Symfony	Acomplix	De nou, no hi ha una explicació concreta, però pel que es pot llegir a l'enllaç, el llançament d'una versió estable va des de la creació d'una o diverses *previous release* a una o diverses *release candidate RC* passant per la creació d'una o diverses versions *beta*, abans d'arribar a la versió estable
Chromium	Acomplix	S'utilitzen versions *dev*, *beta* i *stable*
Referències	Symfony	`http://symfony.com/blog/symfony-2-0`
	Chromium	`http://www.chromium.org/developers/tech-talk-videos/release-process`

6.1.12 Procediment P.12

PROJECTE	VALORACIO	COMENTARIS
Symfony	Parcialment	Es seguixen les mesures de seguretat pròpies de *GitHub* però si aquest lloc sofrira un atac, el codi de tercers allotjat es podria veure afectat
Chromium	Parcialment	No s'especifica cap codi tipus hash o md5 per al tarball encara que, al fer el checkout del SVN si que hi ha que acceptar el cerficicat SSL
Referències	Symfony	`http://symfony.com/doc/current/contributing/code/patches.html` – Secció *Initial Setup*
	Chromium	`http://dev.chromium.org/developers/how-tos/get-the-code`