On Omitting Commits and Committing Omissions: Preventing Git Metadata Tampering That (Re)introduces Vulnerabilities

Santiago Torres-Arias†, Anil Kumar Ammula‡, Reza Curtmola‡, Justin Cappos†
†New York University
‡New Jersey Institute of Technology
USENIX Security ‘16, Austin TX.
Anil Kumar Ammula
New Jersey Institute of Technology

Santiago Torres-Arias
New York University

Reza Curtmola
New Jersey Institute of Technology

Justin Cappos
New York University
The scenario
A central repository and two Devs
Git is a distributed version control system
Git is a distributed version control system
Git is a distributed version control system
Git is a distributed version control system
Git is a distributed version control system
Git is a distributed version control system

Dev

Repo

merge!

Feature

master
Git is a distributed version control system
Git is a distributed version control system
Git is a distributed version control system
Git is a distributed version control system
Git repositories can be compromised
Git repositories can be compromised

Wants to Watch the World burn
While we were having chips and guacamole...

**What the hell?**

Totally crazy. Someone went to extreme lengths, hacking DNS configuration to intercept a single password reset email (I received all other emails except that specific one), to gain authorization to my GitHub account. Why?

I have two best guesses:

1. They wanted access to my company’s private code.
2. They wanted to maliciously modify the Requests codebase (or Certifi, the CA bundle that is shipped with Requests).

Unfortunately, it seems as though #2 is the most likely answer. A crafty entity (like a government, for example), could possibly create a vector into systems running in almost every major tech corporation by adding a special certificate key to the project.

Luckily, the process that we use to generate the bundle is well regulated, highly auditable, and extremely repeatable. Unless they were crafty beyond our imagination, we would have noticed.

But, one can only wonder.
Repository compromises happen

A number of servers belonging to kernel.org were compromised last month in an attack that may have started with a stolen user credential.

**According to a statement** on kernel.org, which hosts the source code for the Linux kernel, the attack is not believed to have affected the source code repositories. While the situation remains under investigation, it is believed the attackers gained access to a server known as ‘Hera.’

“We believe they may have gained this access via a compromised user credential; how they managed to exploit that to root access is currently unknown and is being investigated,” according to kernel.org.
Repository compromises happen

China, GitHub and the man-in-the-middle
Submitted by martin on Wed, Jan 30, 2013

What happened?

At around 8pm, on January 26, reports appeared on Weibo and Twitter that users in China trying to access GitHub.com were getting warning messages about invalid SSL certificates. The evidence, listed further down in this post, indicates that this was caused by a man-in-the-middle attack.

Great piece! Just a minor point: When you say that a CNNIC-signed certificate would allow you to "sign in to Gmail as usual and receive no warning" -- that’s not really true. In Chrome, certain high-value targets (e.g. Google properties) have their certificate fingerprints "pinned". This means Chrome enforces both SSL /and/ the correct certificate trust chain.
See, e.g. [http://www.imperialviolet.org/2011/05/04/pinning.html](http://www.imperialviolet.org/2011/05/04/pinning.html)

Submitted by N.S. on Thu, Jan 31, 2013
Repository compromises happen

RubyGems.org hacked, interrupting Heroku services and putting sites using Rails at risk

JOHN KOETSIER  JANUARY 30, 2013 8:49 PM

TAGS: BLACK HAT, FEATURED, GEMS, HACKER, HEROKU, HEROKU, PASTIE.ORG, RUBY, RUBY GEMS, RUBY ON RAILS, RUBYGEMS, SALESFORCE.COM

Press Releases

Nimble Named Leader in CRM and Sales Intelligence Software and #1 in Overall Customer Satisfaction

CardLinx Announces New Members Including Chevron, Hilton Worldwide, Airbnb, Shop Your Way Rewards, Verifone and Sumitomo Mitsui Card Company

Glowforge Announces $22 Million Series B Investment from Foundry Group and True Ventures to Bring 3D Laser Printers to Mass Production

Ruby package distributor RubyGems.org was hacked today, disrupting web developers globally and causing service shutdowns at popular hosting service Heroku.
Repository compromises happen

Sourceforge Attack: Full Report

As we’ve previously announced, SourceForge.net has been the target of a directed attack. We have completed the first round of analysis, and have a much more solid picture of what happened, the extent of the impact, our plan to reduce future risk of attack. We’re still working hard on fixing things, but we wanted to share what we know with the community.

We discovered the attack on Wednesday, and have been working hard to get things back in order since then. While several boxes were compromised we believe we caught things before the attack escalated beyond its first stages.

Our early assessment of which services and hosts were impacted, and the choice to disable CVS, ishell, file uploads, and project web updates appears to have prevented any further escalation of the attack or any data corruption activities.

We expect to continue work on validating data through the weekend, and begin restoring services early next week. There is a lot of data to be processed.
Repository compromises happen

Attention: Some Fosshub downloads compromised


Some software programs on Fosshub, a free project hosting service, appear to be compromised and serve malware payloads.

Fosshub is a popular file hosting service that software projects such as Classic Shell, qBittorrent, Audacity, MKVToolNix, and others use as their primary file download service.

Basically, what these projects do is link either directly to download files hosted by Fosshub, or link to a download page for their programs on Fosshub.

A thread started on August 2 on the Classic Shell forum by a new user indicated that the user’s computer would not boot Windows anymore after installing the application.
Repository compromises happen

Adobe source code breach; it’s bad, real bad

Barb Darrow

nised

be compromised and serve

Shell, qBittorrent, Audacity,

by Fosshub, or link to a

that the user’s computer

in our community  Sign in
Repository compromises happen

Adobe source code breach; it’s bad, real bad

Barb Darrow

Major Open Source code repository hacked for months, says FSF

By Aug. 14, 2003 11:01 am

If you've downloaded any free, Open Source software since March of this year you might've downloaded more than you bargained for. It seems that back in March of 2003 someone compromised the root FTP servers that function as the code repository for thousands of Open Source software projects. The compromise was severe enough that the attacker could have inserted trojaned
Repository compromises happen

gigaom.com

Adobe source co

Barb Darrow

Major Op for montl

By Aug. 14, 2003 11:01 am

Open-source ProFTPD hacked, backdoor planted in source code

The open-source ProFTPD project has been hacked by unknown attackers who planted a backdoor in the source code.

By Ryan Naraine for Zero Day | December 3, 2010 -- 01:46 GMT (09:46 GMT+08:00) | Topic: Security

The open-source ProFTPD project has been hacked by unknown attackers who planted a backdoor in the source code.

As a result of the hack, the project's main FTP server, as well as all of the mirror servers, have carried compromised versions of the ProFTPD1.3.3c source code, from the November 28 2010 to December 2 2010.

ProFTPD, which positions itself as a secure FTP server for Linux and Unix based operating system, urged all users who run versions of ProFTPD which were downloaded and compiled in this time window to check their systems for security vulnerabilities.
Repository compromises happen

Open-source ProFTPD hacked, backdoor planted in source code

Adobe source code

‘GOOGLE’ HACKERS HAD ABILITY TO ALTER SOURCE CODE

The open-source ProFTPD project has been hacked by unknown actors. A backdoor has been planted in the source code.

-- 01:46 GMT (09:46 GMT+08:00) | Topic: Security

server, as compromised versions of
server for Linux and Unix based
ones of ProFTPD which were
crack their systems for security.

RELATED STORIES

Security
Google: Unwanted software is way aggressive than others

Security
This hush-hush has been quietly done

Security
Google wants you on Android - with manager

Security
Senator Xenoph...
Repository compromises happen

Open-source ProFTPD hacked, backdoor planted in source code

Adobe source code repositories were cracked

Red Hat's Ceph and Inktank code repositories were cracked

Red Hat reports that the Ceph community project and Inktank download sites were hacked last week and it's possible that some code was corrupted.

By Steven J. Vaughan-Nichols for Linux and Open Source | September 17, 2015 -- 19:52 GMT (03:52 GMT+08:00) | Topic: Security
Luckily, we have git’s security features
Luckily, we have git’s security features

- Hash chaining
Luckily, we have git’s security features

- Hash chaining

- Git commit and tag signatures
Luckily, we have git’s security features

- Hash chaining
- Git commit and tag signatures
- Push certificates (more on them later).
Luckily, we have git’s security features

- Hash chaining
- Git commit and tag signatures
- Push certificates (more on them later).

- What could go wrong?
Example
What happened here?

```bash
santiago at ~ ✓: pip install -e git+https://github.com/santiagotorres/django/@1.9.3#egg=django
Obtaining django from git+https://github.com/santiagotorres/django/@1.9.3#egg=django
[...]
Successfully installed django
santiago at ~ ✓: django-admin.py --version
1.4.11
```

I want to install django 1.9.3
What happened here?

santiago at ~ ✓: pip install -e git+https://github.com/santiagotorres/django/@1.9.3#egg=django
Obtaining django from git+https://github.com/santiagotorres/django/@1.9.3#egg=django
[...]
Successfully installed django
santiago at ~ ✓: django-admin.py --version
1.4.11

But I get django 1.4.11
What happened here?

I try to verify the tag...

santiago at ~/django  x  git verify-tag 1.9.3
warning: Duplicated ref: refs/tags/1.5.11
gpg: Signature made Wed 03 Sep 2014 01:10:58 AM EDT using RSA key ID 2D9266A6808FE067
gpg: Good signature from "James Bennett <james@b-list.org>" [full]
Primary key fingerprint: BD47 7E2E 05F7 EF63 71B6  E8EE 2D92 66A6 808F E067
What happened here?

pgp verification passes...

santiago at ~/django  git verify-tag 1.9.3
warning: Duplicated ref: refs/tags/1.5.11
gpg: Signature made Wed 03 Sep 2014 01:10:58 AM EDT using RSA key ID 2D9266A6808FE067
gpg: Good signature from "James Bennett <james@b-list.org>" [full]
Primary key fingerprint: BD47 7E2E 05F7 EF63 71B6 E8EE 2D92 66A6 808F E067
What happened here?

I ask for more detail...

santiago at ~/django ✔ git verify-tag --verbose 1.9.3
object [...]
tagger James Bennett <james@b-list.org> 1409721058 -0500

[...]
Tag 1.4.11
gpg: Signature made Wed 03 Sep 2014 01:10:58 AM EDT using RSA key ID 2D9266A6808FE067
gpg: Good signature from "James Bennett <james@b-list.org>" [full]
Primary key fingerprint: BD47 7E2E 05F7 EF63 71B6 E8EE 2D92 66A6 808F E067
What happened here?

It’s the wrong tag!

santiago at ~/django ✔ git verify-tag --verbose 1.9.3
object [...]
tagger James Bennett <james@b-list.org> 1409721058 -0500
[...]
Tag 1.4.11

gpg: Signature made Wed 03 Sep 2014 01:10:58 AM EDT using RSA key ID 2D9266A6808FE067
gpg: Good signature from "James Bennett <james@b-list.org>"
Primary key fingerprint: BD47 7E2E 05F7 EF63 71B6 E8EE 2D92 66A6 808F E067
What happened here?

- Django 1.4.11 is vulnerable to 8+ RCE vulnerabilities
- But the GPG verification passed?
- Why did this happen?
The problem
Why did this happen?

- Simply put, some Git metadata is not signed
Why did this happen?

- Simply put, some Git metadata is not signed
Why did this happen?

- Simply put, some Git metadata is not signed
Why did this happen?

- Simply put, some Git metadata is not signed.
Why did this happen?

- Simply put, some Git metadata is not signed

Signed!
This is our target
Why did this happen?

- Simply put, some Git metadata is not signed
  - References, pointers to Git tags and commits, are not signed
Why did this happen?

- Simply put, some Git metadata is not signed
  - References, pointers to Git tags and commits, are not signed

- An attacker with write access to the repository can modify this information.
Why did this happen?

- Simply put, some Git metadata is not signed
  - References, pointers to Git tags and commits, are not signed

- An attacker with write access to the repository can modify this information.

- The resulting attack looks like regular git operation.
Metadata Manipulation Attack Taxonomy
Attack taxonomy

● Teleport Attacks
  ○ Branch Teleport Attack
  ○ Tag Teleport Attack

● Rollback Attacks
  ○ Branch Rollback Attack
  ○ Global Rollback Attack
  ○ Effort Duplication Attack

● Deletion Attacks
  ○ Branch Deletion Attack
  ○ Tag Deletion Attack
Attack taxonomy

- **Teleport Attacks**
  - Branch Teleport Attack
  - Tag Teleport Attack

- **Rollback Attacks**
  - Branch Rollback Attack
  - Global Rollback Attack
  - Effort Duplication Attack

- **Deletion Attacks**
  - Branch Deletion Attack
  - Tag Deletion Attack
Branch teleport attack

repository

master

do_not_merge!

Dev
Branch teleport attack

Apple’s duplicated goto

Dev

repository

master
do_not_merge!
Branch teleport attack

Repository

Master

Do not merge!

What is the latest master?

Dev
Branch teleport attack

repository

master

do_not_merge!

Uhh, just a sec

Dev
Branch teleport attack
Branch teleport attack

What!? OK, I better merge

do_not_merge!

Master
Branch teleport attack: result

repository

master

do_not_merge!
Tag teleport attack

Repository

master

v1.vuln

v1.1

give me tag v1.1!

user
Tag teleport attack

Repository

v1.vuln

v1.1

master

You got it!

user
Tag teleport attack
Tag teleport attack

repository

master

v1.1

v1.vuln

Neat!
less features!

user
Branch rollback attack
Branch rollback attack

Here's the fix!
Can you review?

Dev

Repository

Fix

Feature

Master
Branch rollback attack

Repository

master

Feature

FIX

Dev

Dev

looks good!
Ready to merge
Branch rollback attack

master

Feature

FIX

repository

Dev

Just a sec
Branch rollback attack

Repository

- master
- Feature
- FIX

Dev
Branch rollback attack
Branch rollback attack
Attack taxonomy: summary

- **Teleport Attacks**
  - Branch Teleport Attack
  - Tag Teleport Attack

- **Rollback Attacks**
  - Branch Rollback Attack
  - Global Rollback Attack
  - Effort Duplication Attack

- **Deletion Attacks**
  - Branch Deletion Attack
  - Tag Deletion Attack

- Buggy code inclusion
- Wrong version retrieved
- Critical code omission
- Critical code omission
- Coding effort increased
- Missing branch
- Missing tag
How can we fix this?
The problem with existing solutions

- We could solve fork-consistency using existing solutions
The problem with existing solutions

● We could solve fork-consistency using existing solutions

● Consistency systems, like SUNDR, could solve this issue, but they disregard Git’s distributed nature.
The problem with existing solutions

- We could solve fork-consistency using existing solutions

- Consistency systems, like SUNDR, could solve this issue, but they disregard Git’s distributed nature.

- We require a solution that understands which files are meant to be synchronized
Defense assumptions

● Developers communicate through other means
  ○ A complete fork attack will be noticed and discussed by side-channels

● A repository can be initialized with a root of trust
Our Solution
Defense goals: usability

- Preserve current Git workflows
- Ensure backwards compatibility with older Git versions
- Provide increased security in partial adoption scenarios
Defense goals: security

- Prevent modification of committed data
- Ensure consistent repository state
- Ensure repository state freshness
Defense: Overview

- Prevent modification of committed data
  ➔ Provided by Git

- Ensure consistent repository state
  ➔ Reference State Log

- Ensure repository state freshness
  ➔ Nonce Bag
Defense: Overview

- Prevent modification of committed data ➔ Provided by Git
- Ensure consistent repository state ➔ Reference State Log
- Ensure repository state freshness ➔ Nonce Bag
The Reference State Log
The Reference State Log
The Reference State Log

regular push

Repo

Push!

Dev

Dev
The Reference State Log

regular push
signed statement

repo
regular push
signed statement

Repo
Push!

Dev

Dev
The Reference State Log
The Reference State Log

regular fetch

Repo

Fetch!

Dev

Dev

RSL
The Reference State Log

- Regular fetch
- Reference consistency

Repo

Dev

Dev

Pull!
The RSL push entry

Entry
Entry
...
Entry

<table>
<thead>
<tr>
<th>Branch: master</th>
</tr>
</thead>
<tbody>
<tr>
<td>HEAD: 0xfe....ab</td>
</tr>
<tr>
<td>PREV_HASH: 0xac...89</td>
</tr>
<tr>
<td>Signature: Dev’s signature</td>
</tr>
<tr>
<td>...</td>
</tr>
<tr>
<td>...</td>
</tr>
<tr>
<td>...</td>
</tr>
<tr>
<td>...</td>
</tr>
</tbody>
</table>
The RSL push entry

- Branch: master
- HEAD: 0xfe….ab
- PREV_HASH: 0xac...89
- Signature: Dev’s signature

- references changed
- their updated locations
- hash of previous RSL entry
- authenticates whoever added this entry
Implementation: prototype

- Two extensions to git
  - `git securepush` ➢ Add an RSL entry and push
  - `git securefetch` ➢ fetch, retrieve RSL, and verify repository state

- RSL lives in repo
  - as a special branch
  - sent in-band
Synchronization

Repo

master

Dev

Dev
Synchronization

repo

master

work!
Synchronization

Repo

secure push!

Dev

Dev

Feature

master
Synchronization
Synchronization

Repo

master

merge!

Feature

Dev

Dev
Synchronization

Repo

secure push!

master

Feature

Dev

Dev
Synchronization

repo

Dev

secure pull!

master

Feature
Verification

1. Is the entry signed by a trusted party?

2. Are all the entries in the RSL correctly linked together?

3. Are all the references pointing to the right place?
Evaluation
How are attacks prevented

- **Teleport Attacks**
  - Branch Teleport Attack
  - Tag Teleport Attack
- **Rollback Attacks**
  - Branch Rollback Attack
  - Global Rollback Attack
  - Effort Duplication Attack
- **Deletion Attacks**
  - Branch Deletion Attack
  - Tag Deletion Attack
How are attacks prevented

● Teleport Attacks
  ○ Branch Teleport Attack
  ○ Tag Teleport Attack

● Rollback Attacks
  ○ Branch Rollback Attack
  ○ Global Rollback Attack
  ○ Effort Duplication Attack

● Deletion Attacks
  ○ Branch Deletion Attack
  ○ Tag Deletion Attack

⇒ Requires RSL entry with target:
  - commit
  - tag

⇒ Requires replaying RSL entry
  - Target commit must have been pushed
  - (prevented with Nonce Bag)
  - (Prevented with Nonce Bag)

⇒ Requires valid RSL entry
## RSL + Nonce Bag VS other mechanisms

<table>
<thead>
<tr>
<th>Feature</th>
<th>Commit signing</th>
<th>Push Certificate</th>
<th>RSL</th>
</tr>
</thead>
<tbody>
<tr>
<td>Commit Tampering</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Branch Teleport</td>
<td>X</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Branch Rollback</td>
<td>X</td>
<td>X</td>
<td>✓</td>
</tr>
<tr>
<td>Global Rollback</td>
<td>X</td>
<td>X</td>
<td>✓</td>
</tr>
<tr>
<td>Effort Duplication</td>
<td>X</td>
<td>X</td>
<td>✓</td>
</tr>
<tr>
<td>Tag Rollback</td>
<td>X</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Minimum Git Version</td>
<td>1.7.9</td>
<td>2.2.0</td>
<td>1.7.9</td>
</tr>
<tr>
<td>Distribution Mechanism</td>
<td>in-band</td>
<td>(no default)</td>
<td>in-band</td>
</tr>
</tbody>
</table>
Partial adoption of our defense

<table>
<thead>
<tr>
<th></th>
<th>Possible Attacks</th>
<th>Time window of attack</th>
<th>Vulnerable commit objects</th>
</tr>
</thead>
<tbody>
<tr>
<td>Commit signing</td>
<td>All attacks</td>
<td>Any time</td>
<td>Any object</td>
</tr>
<tr>
<td>RSL (full adoption)</td>
<td>No attacks</td>
<td>None</td>
<td>No object</td>
</tr>
<tr>
<td>RSL (partial adoption)</td>
<td>All attacks</td>
<td>After latest RSL and before the next RSL entry</td>
<td>Objects added after the latest RSL entry</td>
</tr>
</tbody>
</table>
## Storage overhead

<table>
<thead>
<tr>
<th>Repository</th>
<th>No. of commits</th>
<th>Number of pushes</th>
<th>Repository size (MB)</th>
<th>Storage Overhead</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bootstrap</td>
<td>11,666</td>
<td>1,345</td>
<td>78.85</td>
<td>.4%</td>
</tr>
<tr>
<td>Angular.js</td>
<td>7,521</td>
<td>26</td>
<td>66.96</td>
<td>.009%</td>
</tr>
<tr>
<td>D3</td>
<td>3,510</td>
<td>255</td>
<td>32.91</td>
<td>.17%</td>
</tr>
<tr>
<td>jQuery</td>
<td>6,031</td>
<td>194</td>
<td>15.79</td>
<td>.22%</td>
</tr>
<tr>
<td>oh-my-zsh</td>
<td>3,841</td>
<td>1,170</td>
<td>3.52</td>
<td>6.5%</td>
</tr>
</tbody>
</table>
Network overhead

1. Additional ~25KB per push/fetch (less than 1% in some cases)
Network overhead

1. Additional ~25KB per push/fetch (less than 1% in some cases)

2. Double round trip time
Network overhead

1. Additional ~25KB per push/fetch (less than 1% in some cases)

2. Double round trip time

3. These issues go away when RSL becomes part Git’s pack protocol
Turning Theory Into Practice
Interaction with the Git community

1. Refactored Git tag PGP verification code
Interaction with the Git community

1. Refactored Git tag PGP verification code
   - Yes, you are running our code starting on 2.9.0
   - 6 patches, over 8 iterations
Interaction with the Git community

1. Refactored Git tag PGP verification code

2. Discussed a plan for the git-tag issue
Interaction with the Git community

**git** git change-tracking tool

2016-08-01 - 2016-09-01 (483 messages)

1. 2016-06-09 Re: [RFC/PATCH] verify-tag: add --check-name flag
git Michael J Gruber
2. 2016-06-08 Re: [RFC/PATCH] verify-tag: add --check-name flag
git Junio C Hamano
3. 2016-06-08 Re: [RFC/PATCH] verify-tag: add --check-name flag
git Santiago Torres
git Junio C Hamano
5. 2016-06-07 Re: [RFC/PATCH] verify-tag: add --check-name flag
git Jeff King
git Junio C Hamano
git Santiago Torres
git Jeff King
git Junio C Hamano
git Jeff King
git Junio C Hamano
git Jeff King
git Junio C Hamano
git Santiago Torres
15. 2016-06-07 Re: [RFC/PATCH] verify-tag: add --check-name flag
git Santiago Torres
git Jeff King
17. 2016-06-07 Re: [RFC/PATCH] verify-tag: add --check-name flag
git Jeff King
git Santiago Torres
git Jeff King
git Junio C Hamano
git santiago
Interaction with the Git community

1. Refactored Git tag PGP verification code

2. Discussed a plan for the git-tag issue

3. Discussed the plan to address the rest
## Other version control systems

<table>
<thead>
<tr>
<th>System</th>
<th>Signed revisions (commits)</th>
<th>prevents MM attacks</th>
</tr>
</thead>
<tbody>
<tr>
<td>Git</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>Bitkeeper</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>Mercurial</td>
<td>Yes (via plugin)</td>
<td>Yes</td>
</tr>
<tr>
<td>Monotone</td>
<td>Yes (mandatory)</td>
<td>Yes</td>
</tr>
</tbody>
</table>
Conclusions
To wrap up

1. Do not trust the infrastructure
To wrap up

1. Do not trust the infrastructure

2. GPG signatures on git objects is currently not enough...
   ○ ...but do it anyway!
   ○ Do not use references, but the object’s SHA1 when possible
To wrap up

1. Do not trust the infrastructure

2. GPG signatures on git objects is currently not enough...
   ○ ...but do it anyway!
   ○ Do not use references, but the object’s SHA1 when possible

3. Update Git!
Thanks

Questions?
Thanks

Questions?