[OSSA-2025-001] (CVE-2025-44021) Can image a node with any file conductor can read

Bug #2107847 reported by Jay Faulkner on 2025-04-21

This bug affects 1 person 268

Affects	Status	Importance	Assigned to	Milestone
Ironic	Fix Released	Critical	Jay Faulkner	
OpenStack Security Advisory	Triaged	Critical	Unassigned	

Bug Description

Ironic standalone use cases permit image_source to be set to:
- A glance image UUID
- an OCI registry path
- Any valid file://, http://, or https:// URL

Currently, Ironic has no restriction on what file we'll deploy from a file:// path except that it is a file (and not, e.g. a character device).

This leads to a situation where a node owner manager could potentially get secret data (e.g. /etc/ironic/ironic.conf) onto a disk. We do not believe there is a way for a node deployed this way to reach "ACTIVE" state, however, once secret data appears on disk it is at risk of exfiltration via node management actions such as manual cleaning or
in environments which have disabled automated cleaning.

We do not believe this is a practical to exploit in most Ironic installs, including standalone installs, however all of the ingredients are present for a practical exploit to be performed when combined with insider information or downstream-modified workflows.

https://github.com/openstack/ironic/blob/master/ironic/common/image_service.py#L826-L865

A discussed fix among Julia, Jay, and Brian Rosmaita is Ironic implementing an allowlist for paths we will permit to be used inside file:// URLs. The default value will not include system paths such as /etc, /sys, /proc, or /dev, and these paths will not be allowed for images even if in the allowlist.

See original description

CVE References

2025-44021

Jay Faulkner (jason-oldos) wrote on 2025-04-21: #1

```
Since this report concerns a possible security risk, an incomplete
security advisory task has been added while the core security
reviewers for the affected project or projects confirm the bug and
discuss the scope of any vulnerability along with potential
solutions.

Changed in ossa:
    status:New → Incomplete
Changed in ironic:
    status:New → Triaged
  assignee:nobody → Jay Faulkner (jason-oldos)
importance:Undecided → Critical
```

Jay Faulkner (jason-oldos) wrote on 2025-04-21: #2

```
Per discussion with Brian and Julia, going to treat this as a Class A and
apply for a CVE number. It should be a relatively simple fix in Ironic.

summary:- Can use any file conductor can read as image
        + Can image a node with any file conductor can read
```

Jay Faulkner (jason-oldos) on 2025-04-21

```
description:updated
```

Jay Faulkner (jason-oldos) wrote on 2025-04-21: **Re: Can image a node with any file conductor can read** #3

```
Draft OSSA.
```

Jay Faulkner (jason-oldos) wrote on 2025-04-21: #4

```
CVE requested from MITRE

description:updated
```

Jay Faulkner (jason-oldos) on 2025-04-21

Andre keedy
Andrew Ruthven
Anna
Anton Arefiev
Anusha
Arpita Rathi
Aruna Kushwaha
Asghar Riahi
Ashish Kumar Singh
Austin Cormier
Barki Mustapha
Bathri Ajay Raj
Branko Vukmirovic
Bruce Martins
C Sasi Kanth
Calub Viem
Cara O'Brien
Chason Chan
Chris Glaubitz
Chris Samson
Craig Miller
Dagmawi Biru
David M. Zendzian
David Seelbach
Deepak Nair
DengBO
Dongwon Cho
Douglas Mendizábal
Dustin Lundquist
Gage Hugo
Greg Althaus
Hironori Shiina
Hosam Al Ali
Hugo Kou
Ian Y. Choi
Jared R Greene
Jay Janardhan
Jeff Ward
Jie Li
Jim Jiang
Joel wineland
John
John L. Villalovos
John Lenihan
John Stafford

Changed in ossa:
    status:Incomplete → Triaged
importance:Undecided → Critical

Jay Faulkner (jason-oldos) wrote on 2025-04-21: #5

WIP patch, still needs real testing.

Julia Kreger (juliaashleykreger) wrote on 2025-04-22: #6

I think the patch might work, that being said I guess I'm most curious about rpath == bad logic. We should discuss it once coffee has woken me up.

As for "a saner default" permitted path, I suspect that is the only one.

Thinking about it, I've personally done something like file:///httpboot/myimage.qcow2. We likely need to check actual for other project file URLs as well.

Julia Kreger (juliaashleykreger) wrote on 2025-04-22: #7

Inside of Ironic, I think we'll want to ensure we take a glance at:

doc/source/install/standalone/enrollment.rst
ironic/common/pxe_utils.py
ironic/drivers/modules/ilo/firmware_processor.py

Furthermore, I suspect this change as-is, will break bifrost's CI. Bifrost's CI jobs utilize file:///opt/cache/... to point to a cirros url. I suspect we'll just need to merge this or a similar change, break bifrost, and then fix bifrost's configuration to explicitly permit /opt/cache as a path.

Julia Kreger (juliaashleykreger) wrote on 2025-04-22: #8

Some quick testing, and everything worked as expected. That being said, I think a *good* idea is to shift the file presence check until after the path check.

Jay Faulkner (jason-oldos) wrote on 2025-04-22: #9

The patch is complete and in testing.

A question for other cores:
- Right now, we default allowed paths to /var/lib/ironic. The more I think about this, the more I'm convinced we should be setting this, by default, to an empty list -- effectively disabling the feature unless explicitly enabled. My basic logic being: folks are extremely unlikely to be using this path already for their images, meaning we're probably going to break all users anyway until they edit config... so why not default to more secure?

Julia and I pondered ^ but decided it was best to get more opinions. @Dmitry @Riccardo?

Jay Faulkner (jason-oldos) on 2025-04-22

```
summary:- Can image a node with any file conductor can read
        + [OSSA-2025-001] Can image a node with any file conductor can read
```

Jay Faulkner (jason-oldos) wrote on 2025-04-22: **Re: [OSSA-2025-001] Can image a node with any file conductor can read** #11

Full results below: tl;dr all works as expected.

One open question (in addition to the one question above in #9): should we hook this into validate? My hunch is no; we currently don't check for the existence of the file in validate, and there is a potential risk to allowing a quick feedback loop for untrusted projects to find what paths are allowed. I feel like not hooking it up to validate (existing patch) is the right move.

Testing results:

Case 1: An image in a reasonable but disallowed configuration (e.g. not in /etc but not in CONF.conductor.file_url_allowed_paths).

Trying to deploy with 'image_source=file:///opt/disallowed/not/blocked/myimage.img' gets deploy failed with a last_error of "Failed to prepare to deploy: Validation of image href file:///home/lol/blocked/myimage.img failed, reason: Security: Path /home/lol/blocked/myimage.img is not allowed for image source file urls.

Note ^^ the error does not contain which paths are valid for security reasons.

Case 2: An image in a disallowed-for-security location (e.g. in /etc)

Trying to deploy with image_source='file:///etc/lol/blocked/myimage.img' gets deploy failed with a last_error of "Failed to prepare to deploy: Validation of image href file:///etc/lol/blocked/myimage.img failed, reason: Security: The path /etc is not permitted in file urls.

Note ^^ the error INCLUDES that /etc is disallowed since that is statically coded in Ironic and is therefore public information.

Case 3: An image in an allowed location (file:///var/lib/ironic/myimage.img)

This fell through all the security checks and ended up failing to deploy because the file didn't exist. This was the expected behavior.

Case 4: No allowed locations

This was me explicitly setting [conductor]/file_url_allowed_paths to None and attempting to deploy to a default-allowed location. Trying to deploy with image_source=file:///var/lib/ironic/myimage.img now deploy failed with the following last_error: Failed to prepare to deploy: Validation of image href file:///var/lib/ironic/myimage.img failed, reason: Security: Path /var/lib/ironic/myimage.img is not allowed for image source file urls.

RaviM Singh
Riccardo Pittau
Richa
Rick Melick
Robert Budden
Robert van Leeuwen
Rochelle Grober
Ron Cannella
Ruby Loo
Rushil Chugh
Ryo Shi
SHIGEMATSU Mitsuhiro
Sandhya Balakrishnan
Satyanarayana Pat...
Sayaji Patil
Sebastian Luna-Va...
Shawn Hartsock
Shruthi Chari
Sid Sun
Songhee Kang
Soo Choi
Stephane Miller
Steve Baker
Steve Sloka
Steven Pavlon
Stuart Hart
Summer Long
Swaroop Jayanthi
Tan Lin
Tao Zhou
Taurus Cheung
Tayaa Med Amine
Tiago Everton Fer...
Uma
Vidhisha Nair
Vil Surkin
Vinu Pillai
Xiang Hui
Xin Zhong
Yongqiang Yang
Yuriy Zveryanskyy
Zahid Hasan
Ziv
aeva black
amar krishna

Jay Faulkner (jason-oldos) wrote on 2025-04-22: #12

Just changing wording around how to set the list empty (None). No code changes from previous patch posted today.

Dmitry Tantsur (divius) wrote on 2025-04-23: #13

This is going to immediately break Metal3 (which uses /shared for images, and at least the bootloader is accessed via a file:// link, also possibly IPA images) and who knows who else (probably everyone who uses containers).

I'm +2 on disabling awkward locations by default, but I'm close to -2 for disabling all locations. This is a breaking change with no early preparation possible.

As an aside, while this bug is (rightfully) embargoed, everyone who is aware of the (documented) file:// feature is aware of its implications. I'd be surprised if they already did not use SELinux/AppArmor/containers to isolate Ironic.

Dmitry Tantsur (divius) wrote on 2025-04-23: #14

> I'm close to -2 for disabling all locations

I'd be more convinced if your change applied only to instance images. Using file:// for them is presumably rather rare. Using file:// links for deploy_kernel/ramdisk/bootloader is something I'd assume a lot of people do (including Metal3 upstream, separately OpenShift too, and Bifrost).

You'll also need to overcome the CI since the empty list proposal won't pass the Bifrost job, while the /var/lib/ironic idea will likely fail the Metal3 one (if it's running with virtual media).

Dmitry Tantsur (divius) wrote on 2025-04-23: #15

As a final note,

> environments which have disabled automated cleaning

are not environments that care about untrusted tenants. That's the drum you should be beating in, Jay :)

I've toyed in my head with possibilities to bypass this, e.g. by using rescue on an instance that is ACTIVE but failed to boot. But I think you're right saying that such a deployment attempt will not get the attacker to ACTIVE. It is not going to get past the _validate_partitioning call [1].

The attack has a lot of pre-requisites (wrong ACL on the Ironic service, untrusted standalone users, somehow getting the information back), I don't believe fixing it justifies breaking an unclear number of consumers.

[1] https://opendev.org/openstack/ironic-python-agent/src/commit/a15680f51eadb7e69b761bd33471ba82674559b6/ironic_python_agent/extensions/standby.py#L833-L854

avinashsau
brightson
bugtracker@devshe...
chaiwat wannaposop
chenglch
chitu
congge
fei Yang
galeido
gsccc
iopenstack
jeff wang
kalim khuang
kgrvamsi
lanpi
laoyi
lewis Liu
liaonanhai
lololmarwa255
lpmqtt
macJack
maestropandy
manish
matianliang
mershard frierson
milan k
mingi hong
miralaunchpad
nawawit kes
pawnesh kumar
raja
rasty94
satyanarayana pat...
satyanarayana pat...
shuangyang.qian
simon stephan
sivagnanam C
tangfeixiong
tomasz
vivek.ys
xiaoningli
xreuze
yangbo
yangzhenyu
yzcao

Jay Faulkner (jason-oldos) wrote on 2025-04-23: #16

> I'd be surprised if they already did not use SELinux/AppArmor/containers to isolate Ironic.

This sounds absolutely bananas to me! Maybe this is true at a service provider level, but it's not true for smaller installations -- installations that might not even be aware this file:// support exists as a ticking time bomb. This is more my concern: small standalone Ironic installations.

[re: breaking metal3+bifrost]

I'd suggest a multi-staged approach:
1) The first patch to fix the CVE can be updated to default to permitting any paths currently in use by metal3, bifrost, ironic-standalone-operator, or any other install tools that might utilize file://. This should be able to be backported unconditionally.
2) Update bifrost and metal3 (and other places as needed) to set CONF.conductor.file_url_allowed_path to the explicit path(s) they care about.
3) Update the default value **only in master** for Ironic to remove metal3/bifrost specific references if we deem it necessary (I don't know what paths this will be and if they make general sense for Ironic). My strong preference would be to ship with allowed_paths of empty, disabling this feature unless opted into (and metal3/bifrost opting into them as they configure it).

[re: exploitability]

So first of all, IMO it's a CVE-quality bug to put secrets from the conductor on a node disk, regardless of if we provide the ability to extract it. I would not want to have to explain to my downstream security team why it's OK that sensitive information was written to the disk. I do think this could represent a link in an exploit chain. A concrete example: we cannot assume the physical access restrictions are modeled how we expect; what if a central Ironic conductor is deploying racks a customer has physical access to? Now they can get secrets off the disk out of band.

In summary, I think these are the concerns and how I think we can address them:
- Operators who use file:// URLs have secured their filesystem properly (e.g. using selinux)

This doesn't apply because you get file:// support by default in standalone; so operators are vulnerable without having taken any positive action to enable it. Additionally, files we know the conductor must have access to, such as the ironic config, will contain secrets such as the mysql password. (I'm legitimately interested in seeing a config that would allow conductor to read that on startup but not use it later, that'd be super cool).

- Don't break bifrost/metal3 out of the box

I'll do the research and ensure the default config for the security patch will not be breaky by default. We can then discuss (in the light of gerrit) a reasonable moving-forward default.

- Expoitability

zhengyue

Patches

0001-OSSA-2025-001-Disallow-unsafe-image-file-paths.patch

2025-04-25-0001-OSSA-2025-001-Disallow-unsafe-image-file-paths.patch

2025-04-29-0001-OSSA-2025-001-Disallow-unsafe-image-file-paths.patch

2025-04-30-0001-OSSA-2025-001-Disallow-unsafe-image-file-paths.patch

ossa-2025-001-patches.tar.gz

2025-05-02-OSSA-2025-001.patch

BOBCAT-EOL-COURTESY-PATCH-0001-OSSA-2025-001-Disallow-unsafe-image-file-paths.patch

Add patch

Remote bug watches

Bug watches keep track of this bug in other bug trackers.

It's a major security issue to put conductor secrets on a physical device not authorized to have them, regardless of if Ironic is the vector to get that information out. A physical attack against an Ironic node's physical hardware should never lead to potential exposure of conductor secrets.

Dmitry Tantsur (divius) wrote on 2025-04-23: #17

> This sounds absolutely bananas to me!

What, that people should not run Ironic as root? :) Even without SELinux, using a separate user gets you a very good protection, except for, yes, you're right, /etc/ironic and such.

> This is more my concern: small standalone Ironic installations.

I don't think there is a reliably correlation between the size of installations and how they read docs. In my experience (obviously biased towards standalone), the smaller scope, the more people read the docs.

> I'd suggest a multi-staged approach

I like the path of at least not immediately breaking our own deliverables. Somebody else may get upset at us... well. Hopefully, file:// is rarely used indeed.

> IMO it's a CVE-quality bug to put secrets from the conductor on a node disk

No real argument here. What I'm pointing out is that, unlike the qemu-img situation, which came as shock to many, this issue can be trivially deduced from our documentation, while is also much more difficult to exploit (I forgot how it's called in CVE-terms).

> I'll do the research and ensure the default config for the security patch will not be breaky by default.

I suspect you'll need /var/lib/ironic (bifrost), /shared/html and /templates (metal3, openshift), but please do double-check me.

Dmitry Tantsur (divius) wrote on 2025-04-23: #18

Let me try to put it a bit differently structured. We have three proposals at hand:

(I) Stop allowing file:/// URLs to unusual paths (/etc /sys and so on).

I don't think there is any disagreement to (unconditionally) do it.

(II) Add a way for an operator to tighten this list even further.

Again, a good idea. We can also, referring to your earlier comment, add an option for which schemes are allowed in case someone wants to disable file:// entirely (or, say, only allow glance).

(III) Restrict file:// by default, including on stable branches.

It is this proposal that I find questionable. Once /etc and other sensitive locations are out of the picture, what's the attack surface? Assuming people are not running Ironic as root, at the very least?

Julia Kreger (juliaashleykreger) wrote on 2025-04-23: #19

Another aspect regarding exploiting - In cases where a file might be copied (or linked) to the conductor attached http server and a url is then recorded on the API surface (such that an authenticated user with sufficient access) could see the field and grab the url. If they are directly able to talk to that endpoint, then they don't actually need to access the node which was deployed at all, it sits in "deploy fail" state or even gets moved back to available. That actually makes exploiting this even easier and should bias the Ironic project response to being even more aggressive in ensuring this issue is addressed through a defense in depth approach *and* possibly outright ability to disable support for file:// paths.

Granted, that is *entirely* dependent on the exact behavior of the deploy_interface in use and the exact construction of settings. That also means consumers and providers of opinionated tooling are more impacted, but it is improper of us to entirely insulate them of this issue. Some action should always be expected as a result of this soft of security issue.

I believe the reasonable path is to ship with reasonable defaults, but also expect we may break things and respond accordingly. Breaking things is okay, as long as there is a path to fix them.

Yes, that means possibly more painful, but at the end of the day, we can only make reasonable assumptions and attempt to do the right thing in the end.

Unfortunately there is also the right thing for the project as it exists, and then the right thing for projects and consumers which then wrap Ironic with specific opinionated settings or usage patterns. We should try to reconcile that, but understand sometimes more work will be needed and that is okay.

I know JayF wanted to move quickly to get a patch out the door, perhaps we need to slow down a tiny bit, make sure the same page is reached, and if there are impacts which hit other projects with apply specific opinions, that they are in the loop to know "hey, there is an issue here".

It seems like the last two comments reflect that process now, so really if there is any takeaway from this comment, is that I figured out a way this could be worse *and* that I'm advocating that it is okay to break things if we must.

Dmitry Tantsur (divius) wrote on 2025-04-23: #20

> I'm advocating that it is okay to break things if we must.

I'm not sold on the "must" though. We're mostly speculating how something can be in theory exploited if the stars align. We have to weigh this against some number of operators (and downstream projects like Metal3) arbitrary broken on upgrade. This reduces trust in upgrading Ironic under the usual motto of "in OpenStack, something always breaks whenever you upgrade".

Jay Faulkner (jason-oldos) wrote on 2025-04-23: #21

I don't think we have to make a choice here, we can get what we need:

We can:
1) Be extremely careful with the first patch and in stable branches to ensure we, by default, will work with existing metal3/bifrost/* setups (I'm also thinking about kayobe) and backport this.
2) Work with those projects (in the open, after #1 lands) to explicitly configure their file:// URL usage in future versions to prevent breakage.
3) Implement a more secure default *after* we've already completed step #2.

Regarding trust, I think -- at least for the users I represent -- they care more about us ensuring Ironic is secure, and being able to trust our secure-by-default stance than they do about having to set a configuration flag on upgrade.

If we can agree on the first step (#1), then lets punt on #2+#3 until after this gets in the open? The only part that needs to be done quietly is the initial fix and I think (maybe I'm wrong?) we have some consensus on a forward path there.

Dmitry Tantsur (divius) wrote on 2025-04-24: #22

I agree with the entire plan. If we seriously consider doing #3, let's add a release note to #1 saying that we're planning on it.

Jay Faulkner (jason-oldos) wrote on 2025-04-24: #23

A note, I checked

>doc/source/install/standalone/enrollment.rst

This has been updated.

>ironic/common/pxe_utils.py

I was unable to find anything that downloaded a file here.

>ironic/drivers/modules/ilo/firmware_processor.py

This downloads files using the image service, and so should be covered by the patch as written.

I'm finishing up the revised patch now, it should be posted in the next two hours. Please prioritize reviewing of this, as I'd like to get this patch out Monday (we will not announce on a Friday by policy).

Jay Faulkner (jason-oldos) wrote on 2025-04-24: #24

0001-OSSA-2025-001-Disallow-unsafe-image-file-paths.patch (13.1 KiB, text/plain)

This is the patch, updated for:
- defaults. We have a *lot* of file:// paths in bifrost docs and repo, I've added them all.
- having >1 value in the conf exposed a bug; that's been fixed up

- reno updated to indicate additional defaults and to encourage operators to explicitly set this config value

Julia Kreger (juliaashleykreger) wrote on 2025-04-25: #25

Overall, the patch looks good to me. The only aspect I see weird is the release note, where it says to set the value to "None". I think the intent is for the value to be none, which the only way to reach an override of a defaulted value to None is to have it set like so:

```
[section]
option =
```

I don't know a good way to disambiguate that, just feel it is necessary to highlight.

Julia Kreger (juliaashleykreger) wrote on 2025-04-25: #26

Discussed with Jay, he is going to revise again since his perception on one of the methods was different from what it actually does when handed "None".

Jay Faulkner (jason-oldos) wrote on 2025-04-25: #27

Yeah, to summarize:

we're calling os.path.realpath() which will make a relative path outta *anything*.

The only reason "= None" appears to have worked is that `pwd` + /None doesn't exist. Setting it to empty *did* permit images for the reason that it effectively set the value to `pwd`.

I need to ensure these config values are fully qualified paths before checking against them.

Jay Faulkner (jason-oldos) wrote on 2025-04-25: #28

2025-04-25-0001-OSSA-2025-001-Disallow-unsafe-image-file-paths.patch (17.0 KiB, text/plain)

I have reworked the patch a bit, adding:
- a regexp that is applied to config items in file_url_allowed_paths to ensure they are *absolute* paths to avoid the realpath() problem
- tests to validate ^^ that all works
- moved to using abspath() in validate_href -- this will prevent it from resolving symlinks, which is likely our desired behavior here

This appears to have fixed the weirdness around how to be an empty config:

```
stack@ubuntu:~$ cat /etc/ironic/ironic.conf |grep allow
file_url_allowed_paths = ''
stack@ubuntu:~$ openstack baremetal node show node-1 -c provision_state -c last_error
```

```
+-----------------+-------------------------------------------------------------
--------------------------------+
| Field | Value |
+-----------------+-------------------------------------------------------------
--------------------------------+
| last_error | Failed to prepare to deploy: Validation of image href file://
/var/lib/ironic/myimage.img |
| | failed, reason: Security: Path /var/lib/ironic/myimage.img is not
allowed for image |
| | source file urls. |
| provision_state | deploy failed |
+-----------------+-------------------------------------------------------------
--------------------------------+
```

I will test the patch more thoroughly over the weekend.

Dmitry Tantsur (divius) wrote on 2025-04-28: #29

> a regexp that is applied to config items in file_url_allowed_paths to ensure they are *absolute* paths to avoid the realpath() problem

Why not (path == os.path.abspath)? This way you'll also rule out using ../ and other stuff that should not be there

Dmitry Tantsur (divius) wrote on 2025-04-28: #30

Btw, do you have any opinions on banning /run/ ?

> if rpath == bad or rpath.startswith(bad + os.sep):
> if rpath == allowed or rpath.startswith(allowed + os.sep):

nit: could use https://docs.python.org/3/library/pathlib.html#pathlib.PurePath.is_relative_to

In fact, your logic may not work if "allowed" contains a trailing slash (like /etc/), while is_relative_to automatically normalizes paths.

+ good = True
+
+ if not good:

nit: you could use break combined with an else: block

+ABSPATH_REGEXP = (r'^/(?!.*(?:/\.\.|/\./|/\.$|/\.\.$))(?!/\.)(?:[^/\0]+/)*[^/\0]*$') # noqa

Noooooooo :) Seriously though, please use https://docs.python.org/3/library/pathlib.html#pathlib.PurePath.is_absolute. Or use https://docs.python.org/3/library/pathlib.html#pathlib.Path.from_uri which will get you this check for free.

Jay Faulkner (jason-oldos) wrote on 2025-04-29: #31

So the reason I used a regexp is the shape of the oslo.config ListOpt (or Opt) interface.

Basically I can provide a regexp to have oslo.config validate these values on load. That's why I wrote it that way. Alternatives would be to fail later (not ideal), allowing relative paths (I am -2 to this unless someone has a compelling argument), or write a config validity check method that gets called somewhere during startup. IMO that's not worth getting out of one ugly regexp that is applied exactly once on system startup -- but I'm happy to have the conversation.

Dmitry Tantsur (divius) wrote on 2025-04-29: #32

> IMO that's not worth getting out of one ugly regexp that is applied exactly once on system startup -- but I'm happy to have the conversation.

The problem is unreadable error message if they do manage to provide a relative path. Also, I cannot review this, sorry. I can trust you that it's right, but if you made a subtle mistake.. too bad.

On the other hand, we're talking about something like

```
class AbsolutePath(oslo_config.types.String):

  def __call__(self, value):
      if not pathlib.Path(value).is_absolute():
          raise ValueError(f"expected an absolute path, got {value}")
      return super().__call__(value)
```

(may make sense to also check for existence btw)

Jay Faulkner (jason-oldos) wrote on 2025-04-29: #33

A custom oslo.config Opt instance is a great idea, I'll head down that path.

Jay Faulkner (jason-oldos) wrote on 2025-04-29: #34

2025-04-29-0001-OSSA-2025-001-Disallow-unsafe-image-file-paths.patch (20.0 KiB, text/plain)

This version of the patch:
- Updated to use the for: else syntax suggested by Dmitry
- Updated to use a custom oslo.config type to do validation instead of a giant terrible regexp (also suggested by Dmitry)
- Moved around some test cases (removed some redundant ones, added ones where relevant)

It's been tested working as expected in devstack.

Dmitry Tantsur (divius) wrote on 2025-04-30: #35

```
+ * File images must not be located in ``/dev``, ``/sys``, ``/proc``,
+ ``/etc``, ``/boot``, or other system paths.
```

Needs /run now

```
+ reason=_("Security: The path %s is not permitted in file "
+ "urls." % bad)
```

nit: no dot in the end of the message (same in at least one more place)

```
+ if len(value) > 1:
+ value = value.rstrip('/')
```

The "if" check is unnecessary (also it's more idiomatic to use `if value:`)

```
+ 'containing no path traversal mechanisms. Config'
+ f'item was: {value}, but resolved to {absvalue}.')
```

Missing space after "Config"

```
+ def _trypath(tpath):
+ try:
+ eap(tpath)
+ except ValueError:
+ return False
+ else:
+ return True
+
+ for path in good_paths:
+ self.assertTrue(_trypath(path),
+ msg=f"Improperly disallowed path: {path}")
```

You should be able to replace this with

```
 self.assertRaisesValueError, eap, tpath)
```

```
+
+ for path in bad_paths:
+ self.assertFalse(_trypath(path),
+ msg=f"Improperly allowed path: {path}")
```

and here just call eap(path), ValueError will cause the test to fail anyway

Mostly nits, so feel free to propose the patch without waiting for me.

Jay Faulkner (jason-oldos) wrote on 2025-04-30: #36

Responding bit by bit:

> Needs /run now

This change is already made locally, I noticed it this morning.

> nit: no dot in the end of the message (same in at least one more place)

ack

> The "if" check is unnecessary (also it's more idiomatic to use `if value:`)

You misread the code. I'm not checking for existence; I'm checking to see if it's *longer than* one character, mainly trying to avoid truncating "/" to "". I believe if I remove the if, I'm going to introduce that bug.

> You should be able to replace this with

Honestly I wrote it this way because I wanted to return a good error because it's REALLY confusing to get this error when you're mocking

os.path.abspath(). I can change it up if you feel strongly about it; but even though it looks weird it's a much more pleasant experience when it fails.

Jay Faulkner (jason-oldos) wrote on 2025-04-30: #37

(revised patch incoming)

Jay Faulkner (jason-oldos) wrote on 2025-04-30: #38

2025-04-30-0001-OSSA-2025-001-Disallow-unsafe-image-file-paths.patch (20.1 KiB, text/plain)

Julia Kreger (juliaashleykreger) wrote on 2025-05-01: #39

I took a look at the latest revision. Overall, it looks good. I like the approach. One super minor item: in test_explicit_absolute_path there is a reference to regex which I believe was a prior iteration. I didn't explicitly test this patch manually, but can on Monday.

Jay Faulkner (jason-oldos) wrote on 2025-05-02: #40

ossa-2025-001-patches.tar.gz (9.7 KiB, application/x-tar)

This is a tarball of patches for all branches, ready for the embargo email:

master-0001-OSSA-2025-001-Disallow-unsafe-image-file-paths.patch
2025.1-0001-OSSA-2025-001-Disallow-unsafe-image-file-paths.patch
2024.2-0001-OSSA-2025-001-Disallow-unsafe-image-file-paths.patch
2024.1-0001-OSSA-2025-001-Disallow-unsafe-image-file-paths.patch
2023.2-0001-OSSA-2025-001-Disallow-unsafe-image-file-paths.patch
2023.1-0001-OSSA-2025-001-Disallow-unsafe-image-file-paths.patch
zed-0001-OSSA-2025-001-Disallow-unsafe-image-file-paths.patch
yoga-0001-OSSA-2025-001-Disallow-unsafe-image-file-paths.patch
xena-0001-OSSA-2025-001-Disallow-unsafe-image-file-paths.patch

The stable patches are different, as discussed, in that they omit the static blocklist and only rely on the configuration method to block unpleasant URLs. I also updated the release notes and documentation to reflect that when upgrading to 2025.2 those static blocks will apply.

I'm going to edit the OSSA notice itself one more time, but right now, unless I hear otherwise, I am planning for us to announce to embargo list either Monday or Tuesday (once Julia gives her signoff after local testing).

Jay Faulkner (jason-oldos) wrote on 2025-05-02: #41

CVE-2025-44021 has been assigned, but it hasn't synced into Launchpad yet

Jay Faulkner (jason-oldos) wrote on 2025-05-02: #42

2025-05-02-OSSA-2025-001.patch (2.4 KiB, text/plain)

Here's the updated OSSA. I have a question though: if we wanted to provide a patch for bobcat, which only recently EOL'd, we obviously can't do it via a gerrit review or commit to a repo.

What should we do? I suggest we publish the patch somewhere, but I don't know where "somewhere" is in this case.

```
summary:- [OSSA-2025-001] Can image a node with any file conductor can read
        + [OSSA-2025-001] (CVE-2025-44021) Can image a node with any file
        + conductor can read
```

Jeremy Stanley (fungi) wrote on 2025-05-03: #43

The patch is technically "published" the moment we switch this bug report to public. You could simply note in the OSSA that a patch for the EOL 2023.2/Bobcat version is attached to the original bug report (and maybe mention the filename for clarity).

Jay Faulkner (jason-oldos) wrote on 2025-05-05: #44

Notification sent out to embargo-notice and linux-distros. This will be made public on Thursday, 2025-05-08 at 1700UTC.

Jay Faulkner (jason-oldos) wrote on 2025-05-06: #45

Adding:
fricker (yesterday)
damien-rannou
arnaud-morin
jlejeune

per request from embargo list

Jay Faulkner (jason-oldos) wrote on 2025-05-08 (last edit on 2025-05-08): #46

removed

Jay Faulkner (jason-oldos) wrote on 2025-05-08: #47

BOBCAT-EOL-COURTESY-PATCH-0001-OSSA-2025-001-Disallow-unsafe-image-file-paths.patch (18.7 KiB, text/plain)

Bobcat / 2023.2 courtesy patch. It is not supported but you are welcome to use it as a base if you're still running EOL releases.

Jay Faulkner (jason-oldos) on 2025-05-08

description:updated
information type:Private Security → Public Security

OpenStack Infra (hudson-openstack) wrote on 2025-05-08: **Fix merged to ironic (master)** #48

Reviewed: https://review.opendev.org/c/openstack/ironic/+/949172
Committed: https://opendev.org/openstack/ironic/commit/5fddef982c718631563bf4ec788619bdc378df24
Submitter: "Zuul (22348)"
Branch: master

commit 5fddef982c718631563bf4ec788619bdc378df24
Author: Jay Faulkner <email address hidden>
Date: Mon Apr 21 15:57:37 2025 -0700

    OSSA-2025-001: Disallow unsafe image file:// paths

    Before this change, Ironic did not filter file:// paths when used as an
    image source except to ensure they were a file (and not, e.g. a
    character device). This is problematic from a security perspective
    because you could end up with config files from well-known paths being
    written to disk on a node.

    Now, we forbid any path that provides access to system configuration,
    including /dev, /sys, /proc, /boot, /run, and /etc. Additionally, we've
    added an allowlist configuration item which limits the acceptable paths
    under which images will be pulled to a list provided by the operator.

    The allowlist default list is huge, but it includes all known usages of
    file:// URLs across Bifrost, Ironic, Metal3, and OpenShift in both CI
    and default configuration.

    Generated-by: Jetbrains Junie
    Closes-bug: 2107847
    Change-Id: I2fa995439ee500f9dd82ec8ccfa1a25ee8e1179c

Changed in ironic:
status:Triaged → Fix Released

OpenStack Infra (hudson-openstack) wrote on 2025-05-08: **Fix merged to ironic (stable/2025.1)** #49

Reviewed: https://review.opendev.org/c/openstack/ironic/+/949173
Committed: https://opendev.org/openstack/ironic/commit/0506aae0c8033c68a8786753b4bbfa92b6d5a896
Submitter: "Zuul (22348)"
Branch: stable/2025.1

commit 0506aae0c8033c68a8786753b4bbfa92b6d5a896
Author: Jay Faulkner <email address hidden>
Date: Mon Apr 21 15:57:37 2025 -0700

OSSA-2025-001: Disallow unsafe image file:// paths

Before this change, Ironic did not filter file:// paths when used as an image source except to ensure they were a file (and not, e.g. a character device). This is problematic from a security perspective because you could end up with config files from well-known paths being written to disk on a node.

The allowlist default list is huge, but it includes all known usages of file:// URLs across Bifrost, Ironic, Metal3, and OpenShift in both CI and default configuration.

For the backportable version of this patch for stable branches, we have omitted the unconditional block of system paths in order to permit operators using those branches to fully disable the new security functionality.

Generated-by: Jetbrains Junie
Closes-bug: 2107847
Change-Id: I2fa995439ee500f9dd82ec8ccfa1a25ee8e1179c

OpenStack Infra (hudson-openstack) wrote on 2025-05-08: **Fix merged to ironic (bugfix/28.0)** #50

Reviewed: https://review.opendev.org/c/openstack/ironic/+/949184
Committed: https://opendev.org/openstack/ironic/commit/52800c8ad3d57c16a98ec95823452e2ee355231f
Submitter: "Zuul (22348)"
Branch: bugfix/28.0

commit 52800c8ad3d57c16a98ec95823452e2ee355231f
Author: Jay Faulkner <email address hidden>
Date: Mon Apr 21 15:57:37 2025 -0700

OSSA-2025-001: Disallow unsafe image file:// paths

Before this change, Ironic did not filter file:// paths when used as an image source except to ensure they were a file (and not, e.g. a character device). This is problematic from a security perspective because you could end up with config files from well-known paths being written to disk on a node.

The allowlist default list is huge, but it includes all known usages of file:// URLs across Bifrost, Ironic, Metal3, and OpenShift in both CI and default configuration.

For the backportable version of this patch for stable branches, we have omitted the unconditional block of system paths in order to permit operators using those branches to fully disable the new security functionality.

Generated-by: Jetbrains Junie
Closes-bug: 2107847
Change-Id: I2fa995439ee500f9dd82ec8ccfa1a25ee8e1179c

OpenStack Infra (hudson-openstack) wrote on 2025-05-08: **Fix merged to ironic (bugfix/27.0)** #51

Reviewed: https://review.opendev.org/c/openstack/ironic/+/949185
Committed: https://opendev.org/openstack/ironic/commit/458457a338e4fb87c15c1f6dad057148a5ca561a
Submitter: "Zuul (22348)"
Branch: bugfix/27.0

commit 458457a338e4fb87c15c1f6dad057148a5ca561a
Author: Jay Faulkner <email address hidden>
Date: Mon Apr 21 15:57:37 2025 -0700

    OSSA-2025-001: Disallow unsafe image file:// paths

    Before this change, Ironic did not filter file:// paths when used as an
    image source except to ensure they were a file (and not, e.g. a
    character device). This is problematic from a security perspective
    because you could end up with config files from well-known paths being
    written to disk on a node.

    The allowlist default list is huge, but it includes all known usages of
    file:// URLs across Bifrost, Ironic, Metal3, and OpenShift in both CI
    and default configuration.

    For the backportable version of this patch for stable branches, we have
    omitted the unconditional block of system paths in order to permit
    operators using those branches to fully disable the new security
    functionality.

    Generated-by: Jetbrains Junie
    Closes-bug: 2107847
    Change-Id: I2fa995439ee500f9dd82ec8ccfa1a25ee8e1179c

OpenStack Infra (hudson-openstack) wrote on 2025-05-09: **Fix included in openstack/ironic 29.0.2** #52

This issue was fixed in the openstack/ironic 29.0.2 Epoxy release.

OpenStack Infra (hudson-openstack) wrote 1 hour ago: **Fix merged to ironic (stable/2024.2)** #53

Reviewed: https://review.opendev.org/c/openstack/ironic/+/949174
Committed: https://opendev.org/openstack/ironic/commit/e18bbe3af00bce90c0a32a6d6ce3112e01ad9ed4
Submitter: "Zuul (22348)"
Branch: stable/2024.2

commit e18bbe3af00bce90c0a32a6d6ce3112e01ad9ed4
Author: Jay Faulkner <email address hidden>
Date: Mon Apr 21 15:57:37 2025 -0700

    OSSA-2025-001: Disallow unsafe image file:// paths

    Before this change, Ironic did not filter file:// paths when used as an
    image source except to ensure they were a file (and not, e.g. a
    character device). This is problematic from a security perspective
    because you could end up with config files from well-known paths being
    written to disk on a node.

The allowlist default list is huge, but it includes all known usages of file:// URLs across Bifrost, Ironic, Metal3, and OpenShift in both CI and default configuration.

For the backportable version of this patch for stable branches, we have omitted the unconditional block of system paths in order to permit operators using those branches to fully disable the new security functionality.

Generated-by: Jetbrains Junie
Closes-bug: 2107847
Change-Id: I2fa995439ee500f9dd82ec8ccfa1a25ee8e1179c

See full activity log