Quantifying Upstream Integration with PaStA

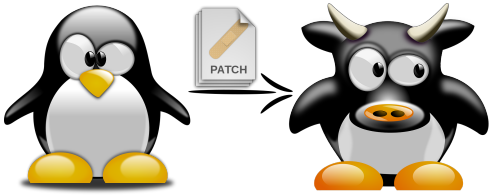
Ralf Ramsauer
Daniel Lohmann, Wolfgang Mauerer

Digitalisation Laboratory,
Technical University of Applied Sciences Regensburg (OTH)

LTSI Workshop @ ELC17

Motivation

- Body of source code provides **basic building blocks**
- Custom **modifications**
- No obligatory **mainline integration**
- Massive **out-of-tree development**
- Diverging software branches
- **Maintenance** of patch stacks requires substantial effort



Quantitative observation of patch stacks

Our Goals

- Actionable criteria for **Long-Term maintenance**
- Guideline for **successful development**
- Quantify maintenance effort/costs

Evolution of Patch Stacks

- Track patches
- Living code base
- Identify trouble spots

Quantitative observation of patch stacks

Our Goals

- Actionable criteria for **Long-Term maintenance**
- Guideline for **successful development**
- Quantify maintenance effort/costs

Evolution of Patch Stacks

- Track patches
- Living code base
- Identify trouble spots

Patch **St**ack **A**nalysis (PaStA)

- Identify similar patches on patch stacks
- Measure mainline integration
- Visualise the evolution of patch stacks
- Support development

Quantitative observation of patch stacks

Our Goals

- Actionable criteria for **Long-Term maintenance**
- Guideline for **successful development**
- Quantify maintenance effort/costs

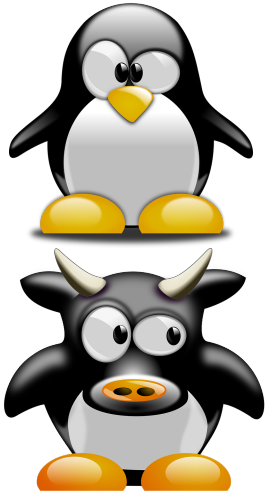
Evolution of Patch Stacks

- Track patches
- Living code base
- Identify trouble spots

Patch **St**ack **A**nalysis (PaStA)

- Identify similar patches on patch stacks
- Measure mainline integration
- Visualise the evolution of patch stacks
- Support development

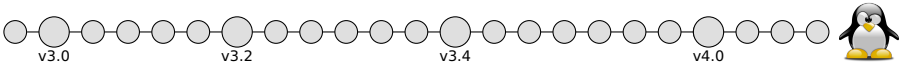
$\Rightarrow$ mine git repositories!



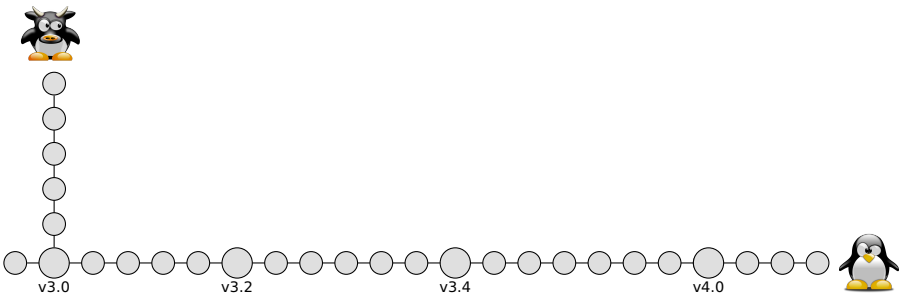
Vanilla Mainline Linux

Preempt_RT patched Linux

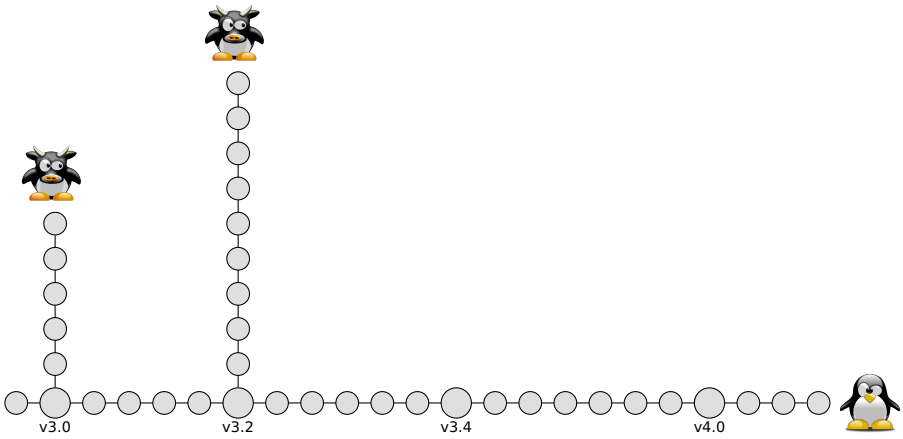
Take mainline project as foundation



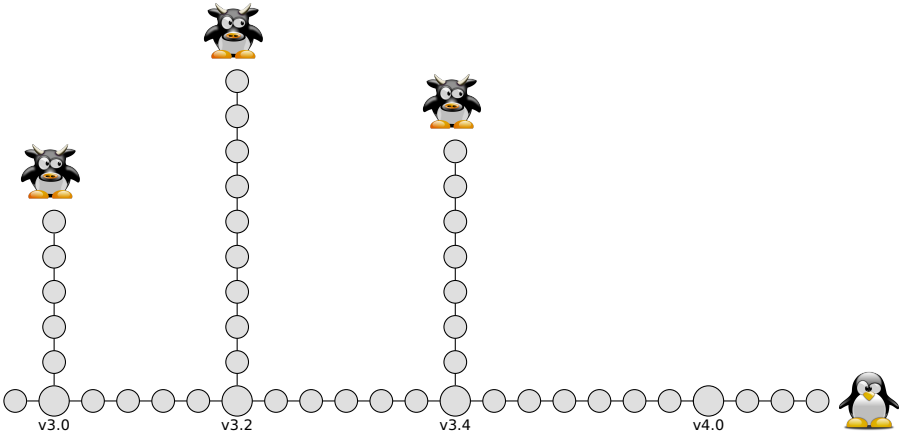
Apply releases of the patch stacks on separate branches



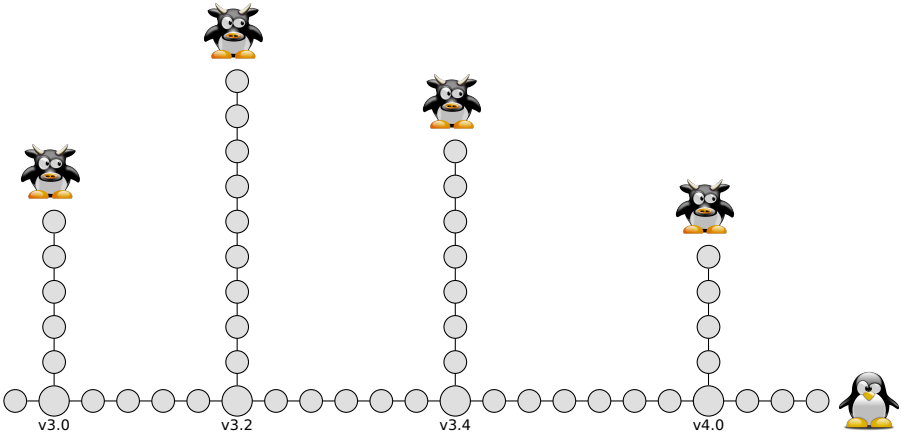
Apply releases of the patch stacks on separate branches



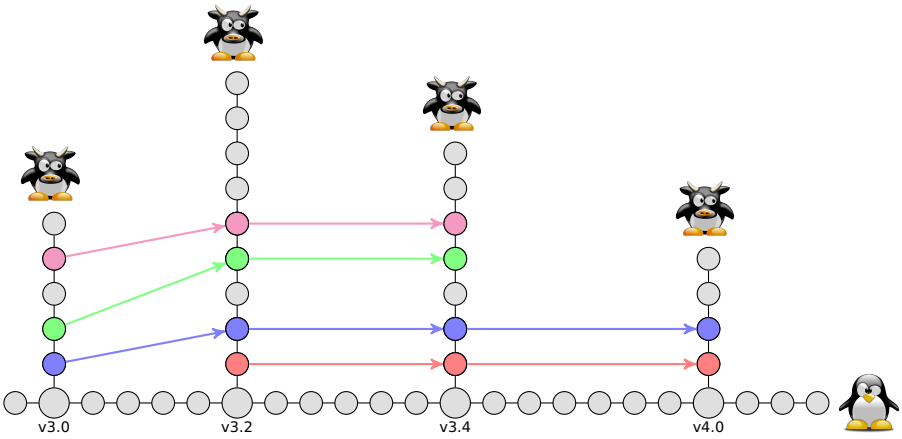
Apply releases of the patch stacks on separate branches



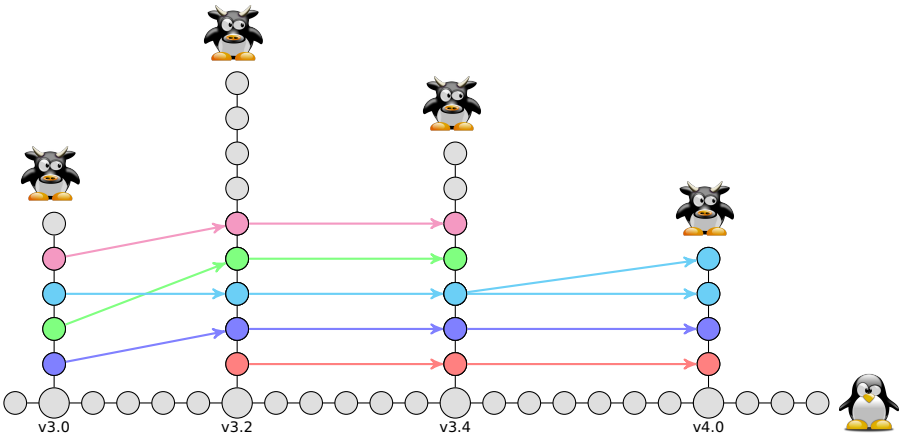
Apply releases of the patch stacks on separate branches



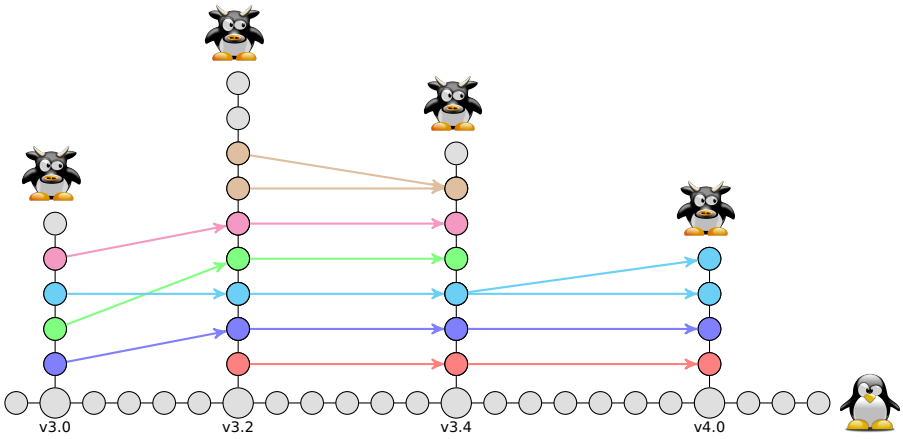
Determine similar patches on patch stacks



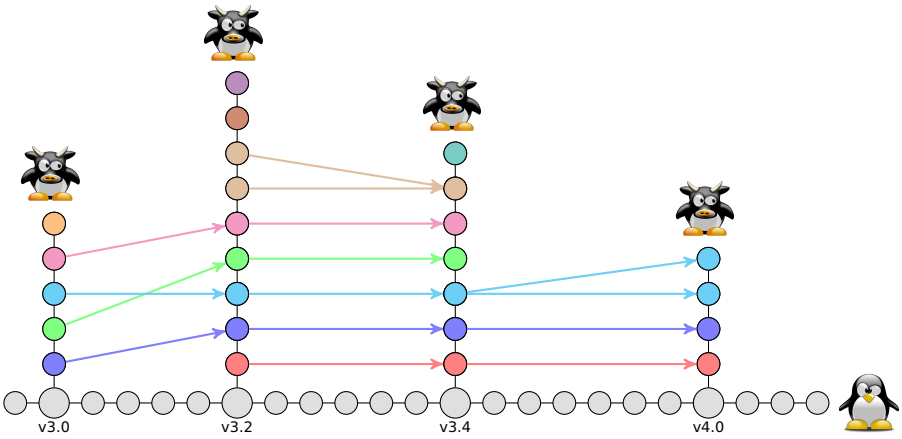
Determine similar patches on patch stacks



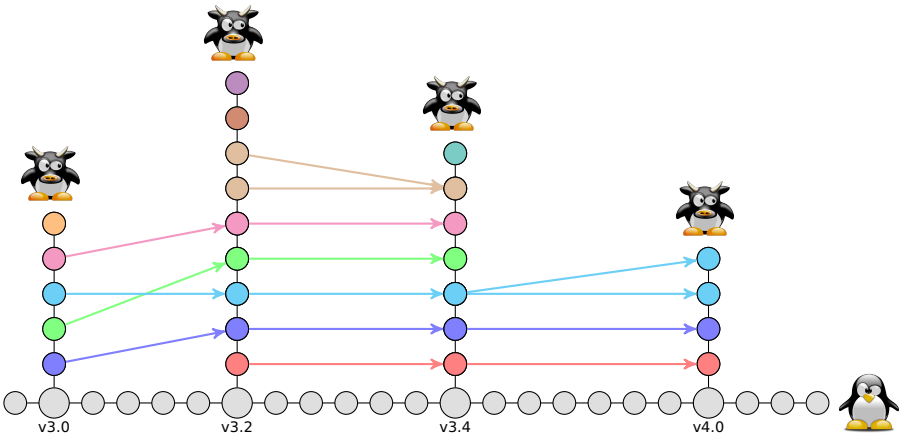
Determine similar patches on patch stacks



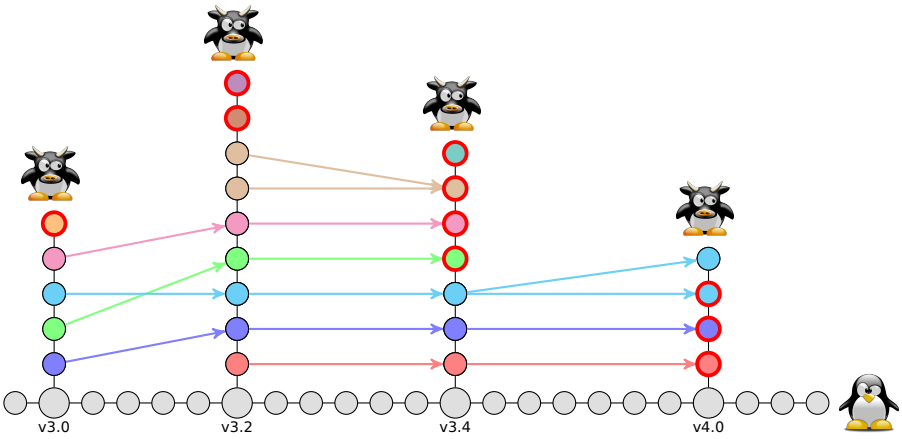
Determine similar patches on patch stacks



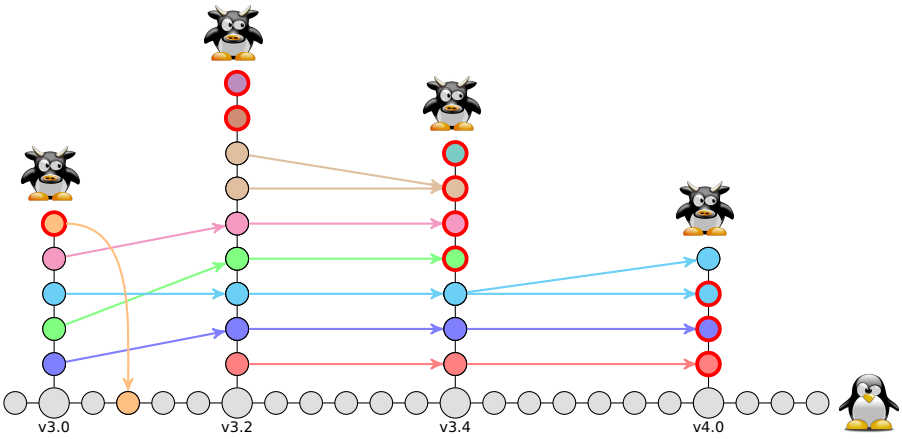
Patches with no successors



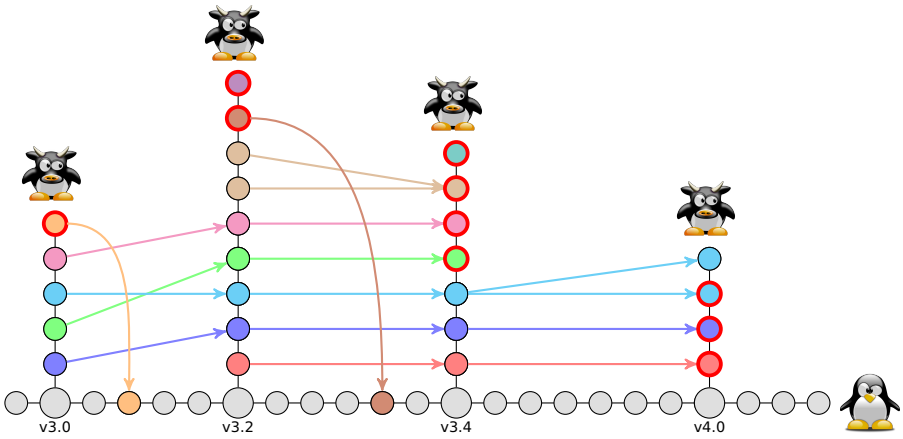
Patches with no successors



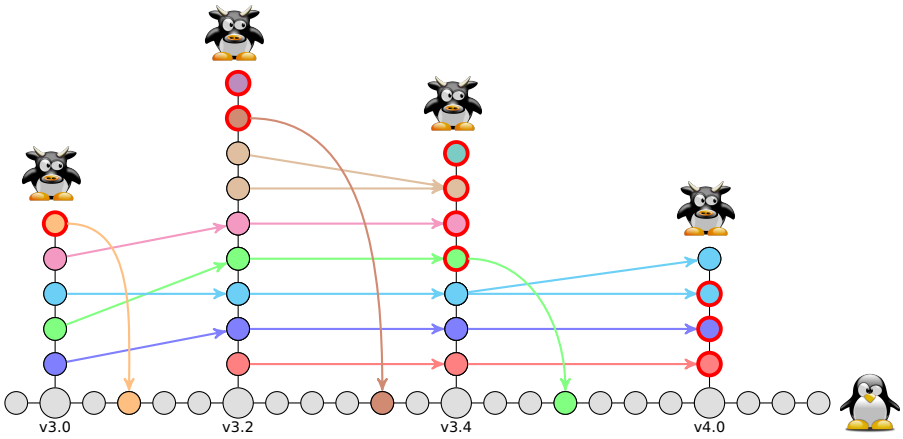
Patches with no successors: Were they integrated?



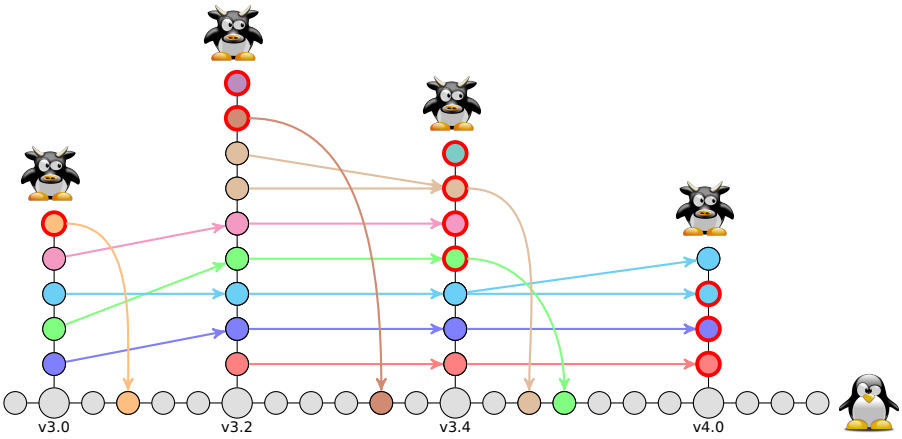
Patches with no successors: Were they integrated?



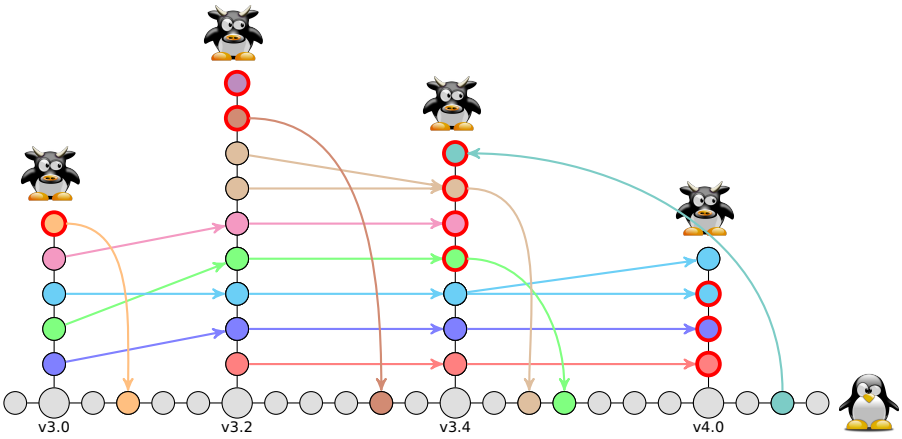
Patches with no successors: Were they integrated?



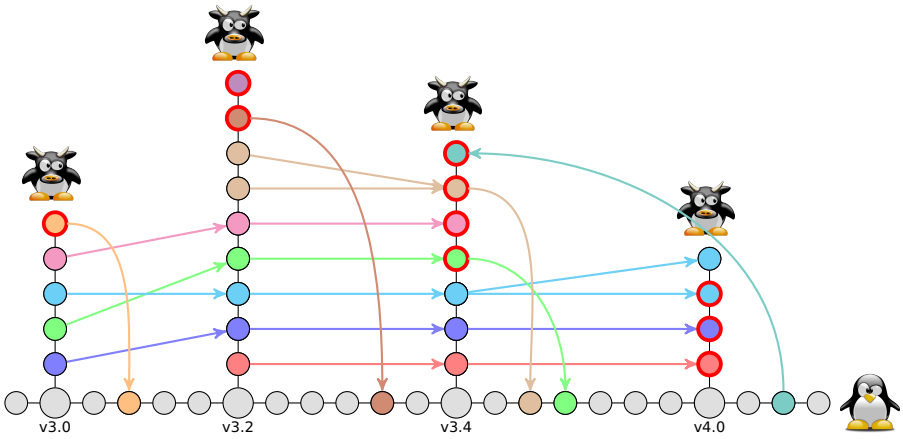
Patches with no successors: Were they integrated?



Patches with no successors: Were they backported?

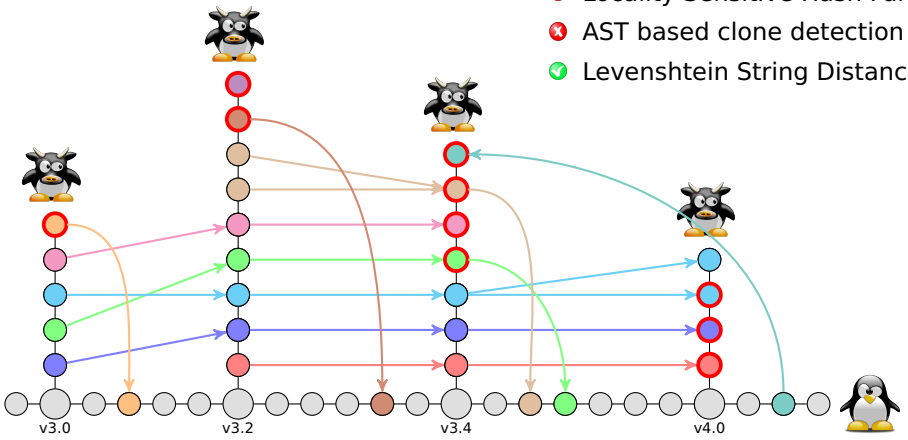


How to detect similar patches?



How to detect similar patches?

- Locality Sensitive Hash Functions [**?**]
- AST based clone detection [**?**][**?**]
- Levenshtein String Distances



Example of similar patches

```
commit 91824d74d6d85f58c63a66b8f2c7993ae246181b
Author: Thomas Gleixner <tglx@linutronix.de>
Date:   Mon Sep 12 21:45:49 2011 +0200

sched-cure-utter-idle-accounting-madness.patrch

Signed-off-by: Thomas Gleixner <tglx@linutronix.de>

diff --git a/kernel/sched.c b/kernel/sched.c
index 205499a..1121a97 100644
--- a/kernel/sched.c
+++ b/kernel/sched.c
@@ -5037,7 +5037,13 @@ EXPORT_SYMBOL(task_nice);
  */
 int idle_cpu(int cpu)
 {
-       return cpu_curr(cpu) == cpu_rq(cpu)->idle;
+       struct rq *rq = cpu_rq(cpu);
+
+#ifdef CONFIG_SMP
+       return rq->curr == rq->idle && !rq->nr_running && !rq->
+              wake_list;
+#else
+       return rq->curr == rq->idle && !rq->nr_running;
+#endif
 }

 /**
```

```
commit 908a3283728d92df36e0c7cd63304fd35e93a8a9
Author: Thomas Gleixner <tglx@linutronix.de>
Date:   Thu Sep 15 15:32:06 2011 +0200

sched: Fix idle_cpu()

On -rt we observed hackbench waking all 400 tasks to a single
cpu. This is because of select_idle_sibling()'s interaction
with the new ipi based wakeup scheme.
[..snip..]
Signed-off-by: Thomas Gleixner <tglx@linutronix.de>
Signed-off-by: Peter Zijlstra <a.p.zijlstra@chello.nl>
Link: http://lkml.kernel.org/n/tip-3b30p18b2[...]
Signed-off-by: Ingo Molnar <mingo@elte.hu>

diff --git a/kernel/sched.c b/kernel/sched.c
index 1874c74..4cdc91c 100644
--- a/kernel/sched.c
+++ b/kernel/sched.c
@@ -5138,7 +5138,20 @@ EXPORT_SYMBOL(task_nice);
  */
 int idle_cpu(int cpu)
 {
-       return cpu_curr(cpu) == cpu_rq(cpu)->idle;
+       struct rq *rq = cpu_rq(cpu);
+
+       if (rq->curr != rq->idle)
+               return 0;
+
+       if (rq->nr_running)
+               return 0;
+
+#ifdef CONFIG_SMP
+       if (!llist_empty(&rq->wake_list))
+               return 0;
+#endif
+
+       return 1;
 }

 /**
```

Example of similar patches

commit 91824d74d6d85f58c63a66b8f2c7993ae246181b
Author: Thomas Gleixner <tglx@linutronix.de>
Date: Mon Sep 12 21:45:49 2011 +0200

sched-cure-utter-idle-accounting-madness.patrch

Signed-off-by: Thomas Gleixner <tglx@linutronix.de>

```
+       struct rq *rq = cpu_rq(cpu);
+
+#ifdef CONFIG_SMP
+       return rq->curr == rq->idle && !rq->
    nr_running && !rq->wake_list;
+#else
+       return rq->curr == rq->idle && !rq->
    nr_running;
+#endif
```

```
+       return rq->curr == rq->idle && !rq->nr_running;
+#endif
 }

 /**
```

commit 908a3283728d92df36e0c7cd63304fd35e93a8a9
Author: Thomas Gleixner <tglx@linutronix.de>
Date: Thu Sep 15 15:32:06 2011 +0200

sched: Fix idle_cpu()

On -rt we observed hackbench waking all 400 tasks to a single cpu. This is because of select_idle_sibling()'s interaction with the new ipi based wakeup scheme.
[..snip..]
Signed-off-by: Thomas Gleixner <tglx@linutronix.de>

```
+       struct rq *rq = cpu_rq(cpu);
+
+       if (rq->curr != rq->idle)
+               return 0;
+
+       if (rq->nr_running)
+               return 0;
+
+#ifdef CONFIG_SMP
+       if (!llist_empty(&rq->wake_list))
+               return 0;
+#endif
+
+       return 1;
```

```
+               return 0;
+#endif
+
+       return 1;
 }

 /**
```

Example of similar patches

```
commit 91824d74d6d85f58c63a66b8f2c7993ae246181b
Author: Th                                    >
Date:   Mo

sched-cure                                    atrch

Signed-off                                    ronix.de>

 diff --git                                   c
 index 2054
--- a/kern
+++ b/kern
@@ -5037,7                                   nice);
    */
   int idle_
   {
-          re                                 >idle;
+          st
+
+#ifdef CO
+          re                                 >nr_running && !rq->
           wa
+#else
+          return rq->curr == rq->idle && !rq->nr_running;
+#endif
   }

   /**
```

```
[...]
#else
#endif
#ifdef
CONFIG_SMP
cpu_rq(cpu)
return
rq
rq->curr
rq->idle
rq->nr_running
rq->wake_list
struct
```

```
commit 908a3283728d92df36e0c7cd63304fd35e93a8a9
Author: Thomas Gleixner <tglx@linutronix.de>
Date:   Thu Sep 15 15:32:06 2011 +0200

sched: Fix idle_cpu()

On -rt we observed hackbench waking all 400 tasks to a single
cpu. Th                                       ing()'s interaction
with the
[..snip
Signed-                                       utronix.de>
Signed-                                       ra@chello.nl>
Link: h                                       18b2[...]
Signed-                                       u>

 diff --                                      ed.c
 index 18
--- a/k
+++ b/k
@@ -513                                       k_nice);
    */
   int idl
   {
-                                             pu)->idle;
+
+
+
+
+
+
+                 return 0;
+
+#ifdef CONFIG_SMP
+         if (!llist_empty(&rq->wake_list))
+                 return 0;
+#endif
+
+         return 1;
   }

   /**
```

```
[...]
#endif
#ifdef
if
CONFIG_SMP
cpu_rq(cpu)
llist_empty
return
rq
rq->curr
rq->idle
rq->nr_running
rq->wake_list
struct
```

Example of similar patches

```
commit 91824d74d6d85f58c63a66b8f2c7993ae246181b
Author: Th                                    >
Date:   Mo

sched-cure                                    atrch

Signed-off                                    ronix.de>

 diff --git                                   c
 index 2054
--- a/kern
+++ b/kern
@@ -5037,7                                   nice);
    */
  int idle_
  {
-         re                                  >idle;
+         st
+
+#ifdef CO
+         re                                  >nr_running && !rq->
          wa
+#else
+         return rq->curr == rq->idle && !rq->nr_running;
+#endif
  }

  /**
```

```
[...]
#else
#endif
#ifdef
CONFIG_SMP
cpu_rq(cpu)
return
rq
rq->curr
rq->idle
rq->nr_running
rq->wake_list
struct
```

```
commit 908a3283728d92df36e0c7cd63304fd35e93a8a9
Author: Thomas Gleixner <tglx@linutronix.de>
Date:   Thu Sep 15 15:32:06 2011 +0200

sched: Fix idle_cpu()

On -rt we observed hackbench waking all 400 tasks to a single
cpu. Th                                       ing()'s interaction
with the
[..snip
Signed-                                       utronix.de>
Signed-                                       ra@chello.nl>
Link: h                                       18b2[...]
Signed-                                       u>

 diff --                                      ed.c
 index 18
--- a/k
+++ b/k
@@ -513                                       k_nice);
    */
  int idl
  {
-                                             pu)->idle;
+
+
+
+
+
+
+                 return 0;
+
+#ifdef CONFIG_SMP
+         if (!llist_empty(&rq->wake_list))
+                 return 0;
+#endif
+
+         return 1;
  }

  /**
```

```
[...]
#endif
#ifdef
if
CONFIG_SMP
cpu_rq(cpu)
llist_empty
return
rq
rq->curr
rq->idle
rq->nr_running
rq->wake_list
struct
```

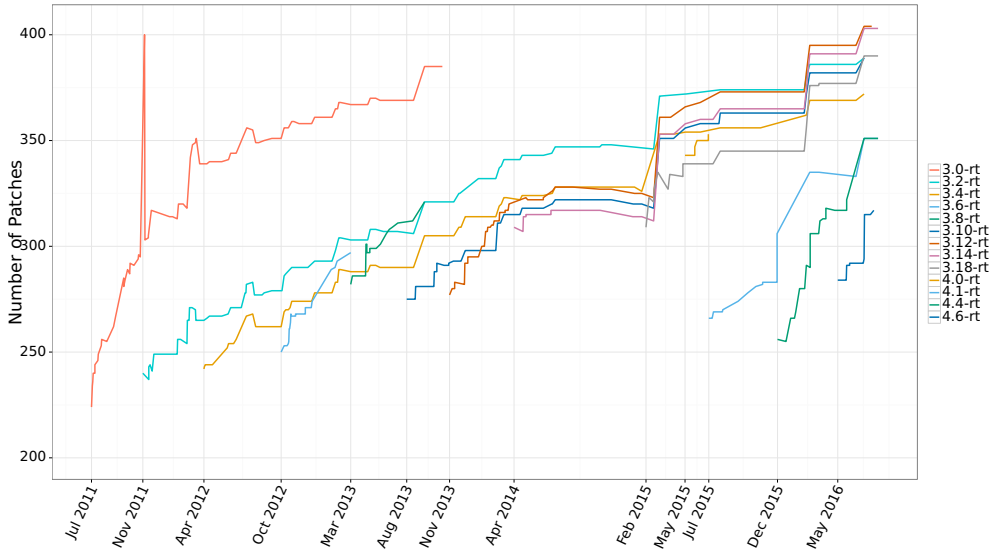
Diff similarity: 0.875

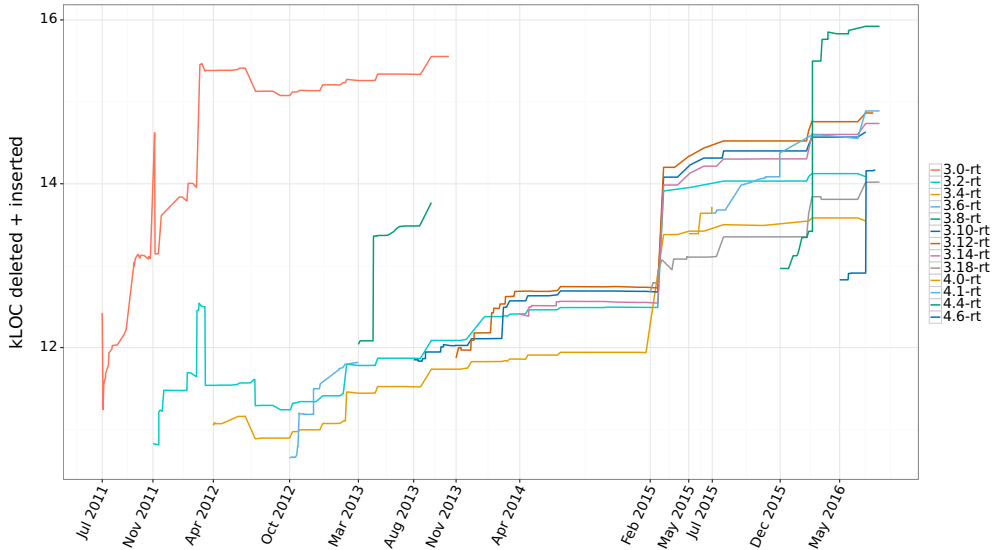
PaStA by the Numbers

Preempt_RT (Jul ’11 - Aug ’16):

582	patch stack releases
183,000	patches
1,098	patch groups
156	forward ports
219	backports

Preempt_RT: Patch Stack Size




kLOC deleted + inserted
16
14
12
Jul 2011
Nov 2011
Apr 2012
Oct 2012
Mar 2013
Aug 2013
Nov 2013
Apr 2014
Feb 2015
May 2015
Jul 2015
Dec 2015
May 2016
3.0-rt
3.2-rt
3.4-rt
3.6-rt
3.8-rt
3.10-rt
3.12-rt
3.14-rt
3.18-rt
4.0-rt
4.1-rt
4.4-rt
4.6-rt

Case Study Preempt_RT: Evolution of Patch Stacks

Flow of Patches

inflow

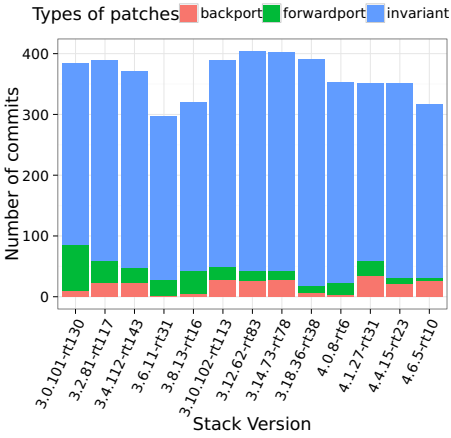
backports and **new patches**

outflow

dropped and **upstream patches**

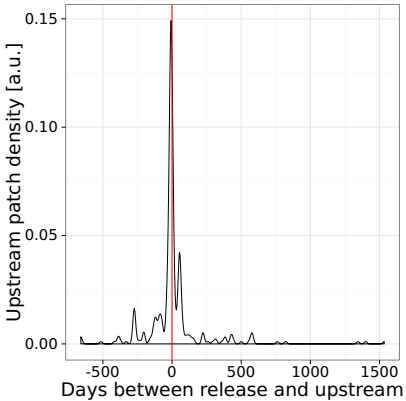
invariant

patches that remain on the stack



Preempt_RT: Composition of different versions

Case Study Preempt_RT: Mainline Integration



Preempt_RT: Duration of mainline integration

Distribution of integration times (in days)

Positive: forward ports
Negative: backports

Summary

- **Detect similar patches** with semi-automatic algorithm
- Determine **patch flow** between different releases of the patch stack
- Quantify **mainline integration** of a given patch stack
- **Support development** of patch stacks

Future Work

- Measure **invasiveness** of a patch stack
- Use results to **identify patches with high maintenance effort**
- Derive **successful development and maintenance strategies**

Thank you!

ralf.ramsauer@othr.de[1]

Get a fresh clone at github.com/lfd

[1] PGP: 8F10049B