# Binary Rewriting without Control Flow Recovery

Gregory J. Duck, Xiang Gao, and Abhik Roychoudhury. 2020. Binary Rewriting without Control Flow Recovery. In Proceedings ofthe 41st ACM SIGPLAN International Conference on Programming Language Design and Implementation (PLDI '20), June 15–20, 2020, London, UK. ACM, New York, NY, USA, 13 pages. https://doi.org/10.1145/ 3385412.3385972

# 传统重写方法 vs E9Patch

- 传统重写方法:
  - 基于控制流, 重写过程可能会移动指令, 因此需要调整跳转目标来恢复控制流.
  - 传统实现依赖于简化的试探法或假设(例如针对特定的编译器/语言/文件元信息), 因此难以扩展, 且无法处理大/复杂程序.
- E9Patch
  - 提出了指令双关的方法, 无需恢复控制流.
  - 可以扩展到大的复杂应用.

# 传统工具重写步骤

1. 反汇编器解析二进制文件的机器码指令.
2. 恢复(某种形式)控制流信息(比如跳转目标).
3. 对插入/删除/替换/重定位后的二进制代码进行转换
4. 输出修改后的二进制.

# 二进制恢复控制流遇到的困难

重写步骤中的控制流信息恢复是非常困难的. 传统有两种方法尝试解决该问题:

1. 利用文件元信息, 比如调试符号和位置. 但是不适用于剥离符号和非PIC的程序
2. 使用静态二进制分析恢复控制流, 但目前没有绝对准确的恢复算法. 因此传统工具都简化了应用情景, 比如假设间接跳转都遵循特定模式(比如像C的Switch一样). 但这样扩展性很差, 并且不够鲁棒.

# E9Patch是如何解决该问题的?

作者希望找到一种无需控制流信息也不用启发式/假设的方法, 也就找到了baseline instruction punning

- baseline instruction punning: 对于一组要重写的指令组P, 尝试用跳转指令J替换P组内的每个指令I, J会将控制流重定向到实现预期的二进制补丁/指令的蹦床, 最后将控制流归还主程序. (代码HOOK)
- 但是如果要修补的指令长度小于跳转指令J的长度(x86_64上的跳转指令长度为5字节, 1字节是操作码, 4字节是地址长度), 那么就不能直接替换.

# 指令双关

- 所以这里采用了指令双关的方法复用了指令间重叠的字节. (被LiteInst工具用于动态插装的方法)
- 因为双关跳转只是利用了原有指令中的字节, 不会对原指令I进行修改或移动, 也就保留了原来的跳转目标地址, 也就是控制流无关的.
- 但是因为依赖于指令间重叠的字节, 因此有时候跳转目标会是无效的内存位置, 因此作者想出新策略来弥补这个问题.
- 弥补的方法就是: instruction eviction, 指令逐出. 在不更改指令语义的情况下修改重叠指令的字节表示.
- 理想情况可以基于该策略将补丁覆盖率提高到100%(或接近100%)

# 蹦床空间问题

- 跳转后不一定能找到合适空间作为蹦床, 因此蹦床并非内存连续排列指令. 这样就会导致碎片过多和输出文件大小膨胀.
- 作者引入新方法来优化空间使用情况 - 物理页面分组

# 现有x86指令补丁方法: Signal Handlers

1. 用单字节 Int3 指令(0xCC)替换每个补丁位置指令.
2. Int3指令会触发中断发出SIGTRAP信号.
3. 信号处理程序接收信号后进行补丁

- 这是传统调试器实现断点的方式
- 虽然保留了跳转目标, 但是中断和信号处理需要切换内核/用户模式, 性能较差. (有时会降低几个数量级)

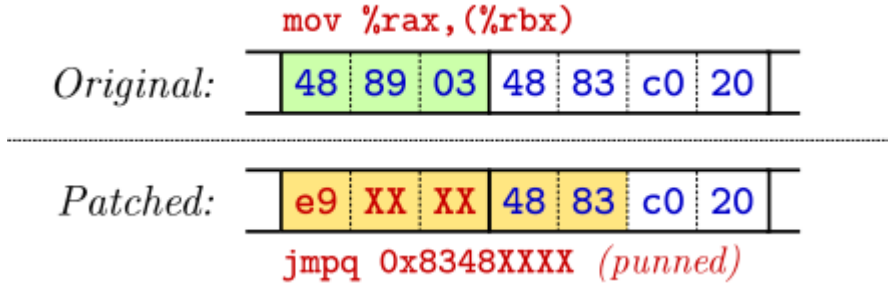
# 现有x86指令补丁方法: Jumps

1. 用跳转指令替换每个补丁位置指令.
2. 该跳转指令跳向实现补丁的蹦床
3. 蹦床修补完指令后跳回主程序.

- 比中断方式快得多, 并被大量重写工具采用.
- X86_64的跳转指令: jmpq rel32 长度为5个字节, 1个为操作码0xe9, 4个字节用于 rel32 的值, 也就是跳转目标.
- 但是长度小于5字节的指令不能用这个跳转指令直接替换而跳向蹦床.

# 现有x86指令补丁方法: Instruction Punning

- 该方法用于弥补Jumps方法不能应用于短小指令的缺陷.
- 基本思想是找到重叠指令, 复用其中相同的字节来表示偏移地址(rel32)
- 例如:
  - mov %rax,(%rbx)  add $32,%rax
  - 原字节:  48 89 03  48 83 c0 20
  - 修补字节:  e9 xx xx  48 83 c0 20
  - mov指令字节长度为3, add指令长度为4. 这里0xe9是跳转操作码.
  - 修补mov指令(3字节长度)时复用了add指令的前2字节(48 83)
  - 实现跳转 jmpq 0x8348xxxx
  - 当然0x8348xxxx并不总是有效的内存地址



# 三个策略略

- 通过上面讨论的跳转和双关的方法, 可以解决部分的指令重写问题. 但是不能做到完整的应用.
- 因此基于指令填充/双关/逐出进行组合尝试出三个新策略用于扩大指令修补的覆盖范围.
- 三个策略的介绍就以下述指令演示:

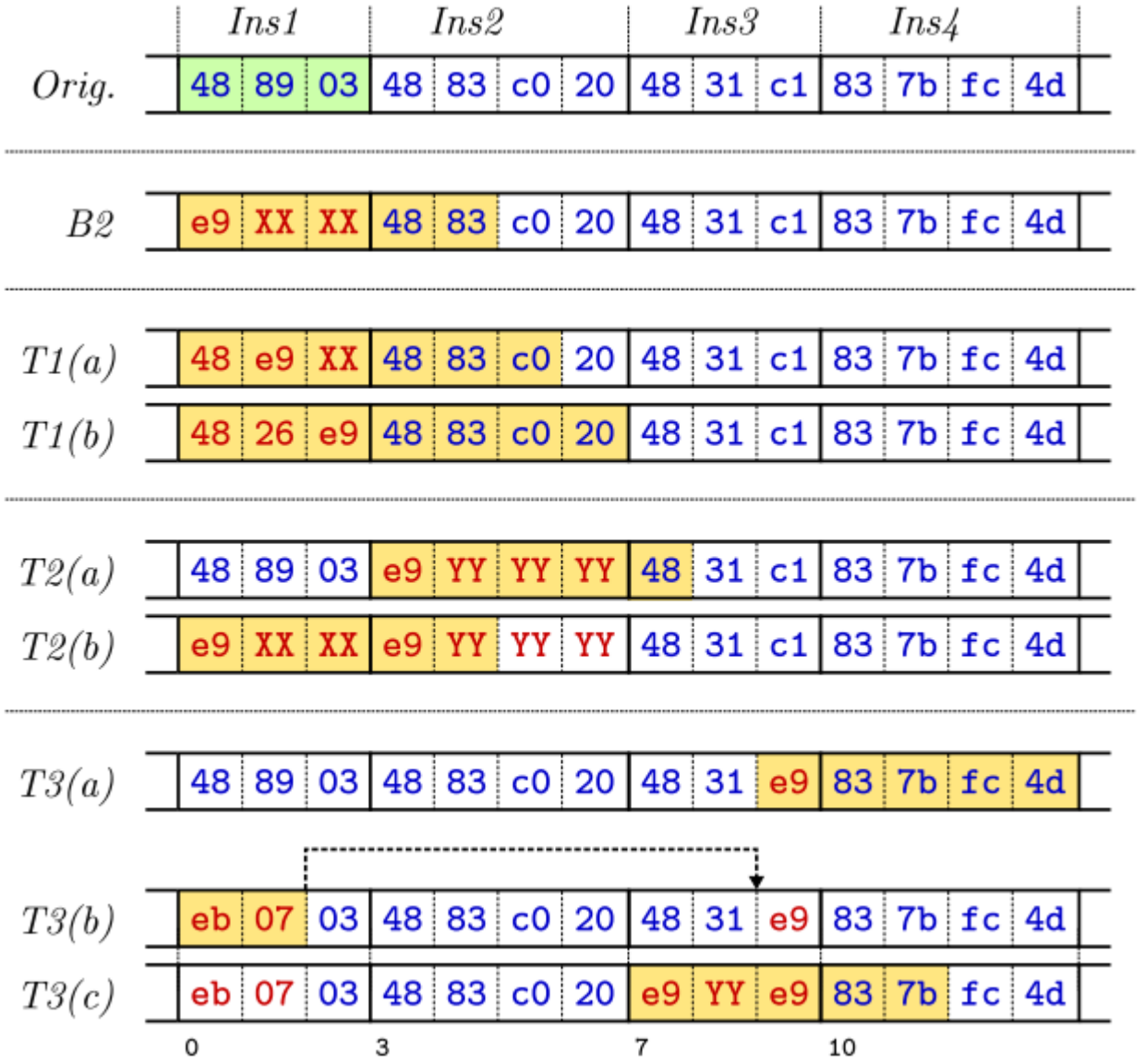
*Ins1*: `mov %rax,(%rbx)`
*Ins2*: `add $32,%rax`
*Ins3*: `xor %rax,%rcx`
*Ins4*: `cmpl $77,-4(%rbx)`

- 要修补的指令是 Ins1

# T1: Padded Jumps

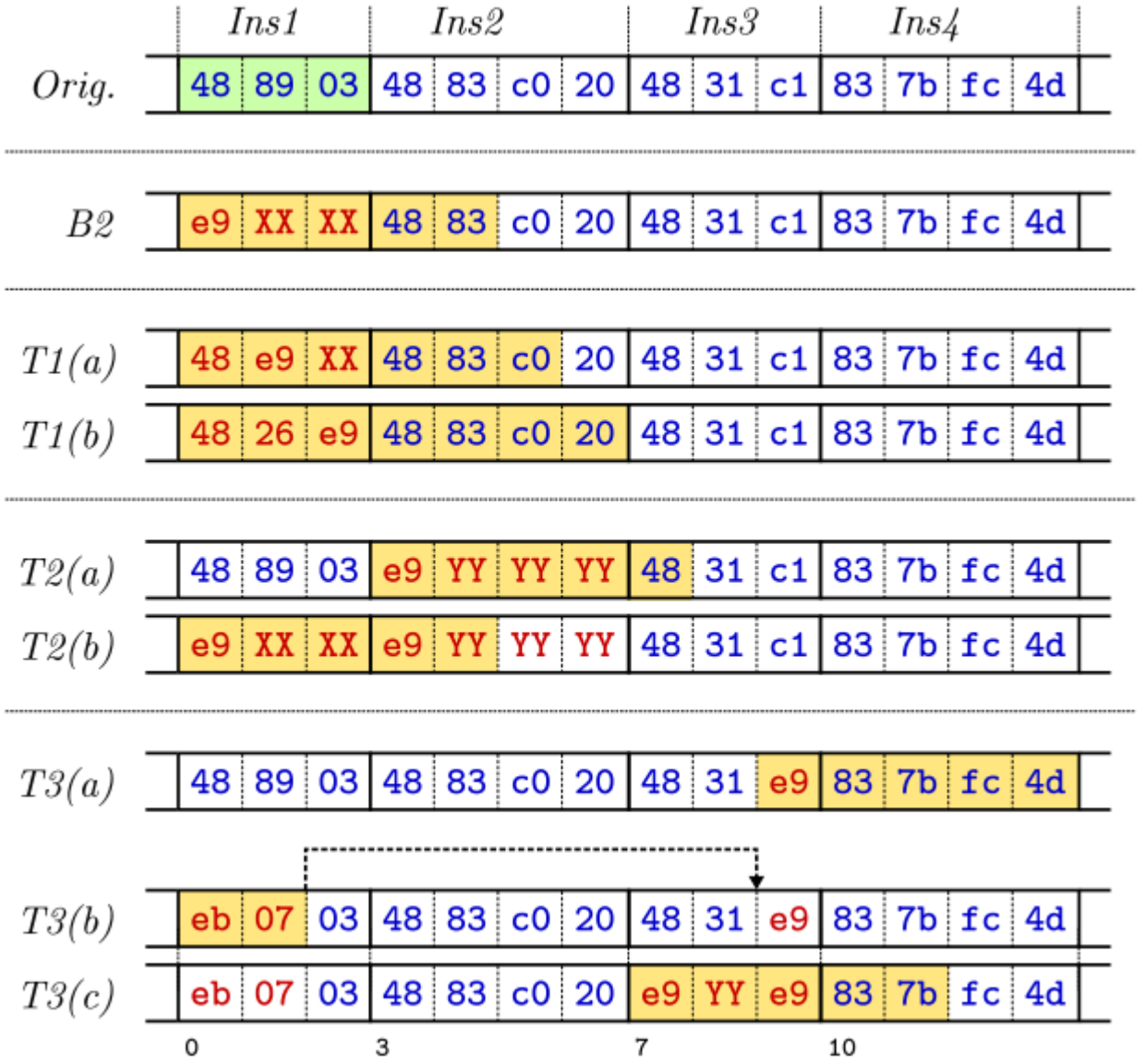
- 使用冗余的指令前缀来填充跳转指令
- 如图所示, T1(a)的冗余前缀是 48 , T2(a)的冗余前缀是 48 26 .
- 使用冗余前缀的缺点就是会限制可操控的范围, 比如B2的范围是 0x8348xxxx , 但T1(a)的范围只有 0xc08348xx , T1(b)则是一个具体的值了.

- T1的适用性取决于补丁指令的长度, 长度越大, 能右移尝试的次数也就越多.
- 同时也意味着T1不适用于 单字节指令. 同时右移会受到越多的范围约束.



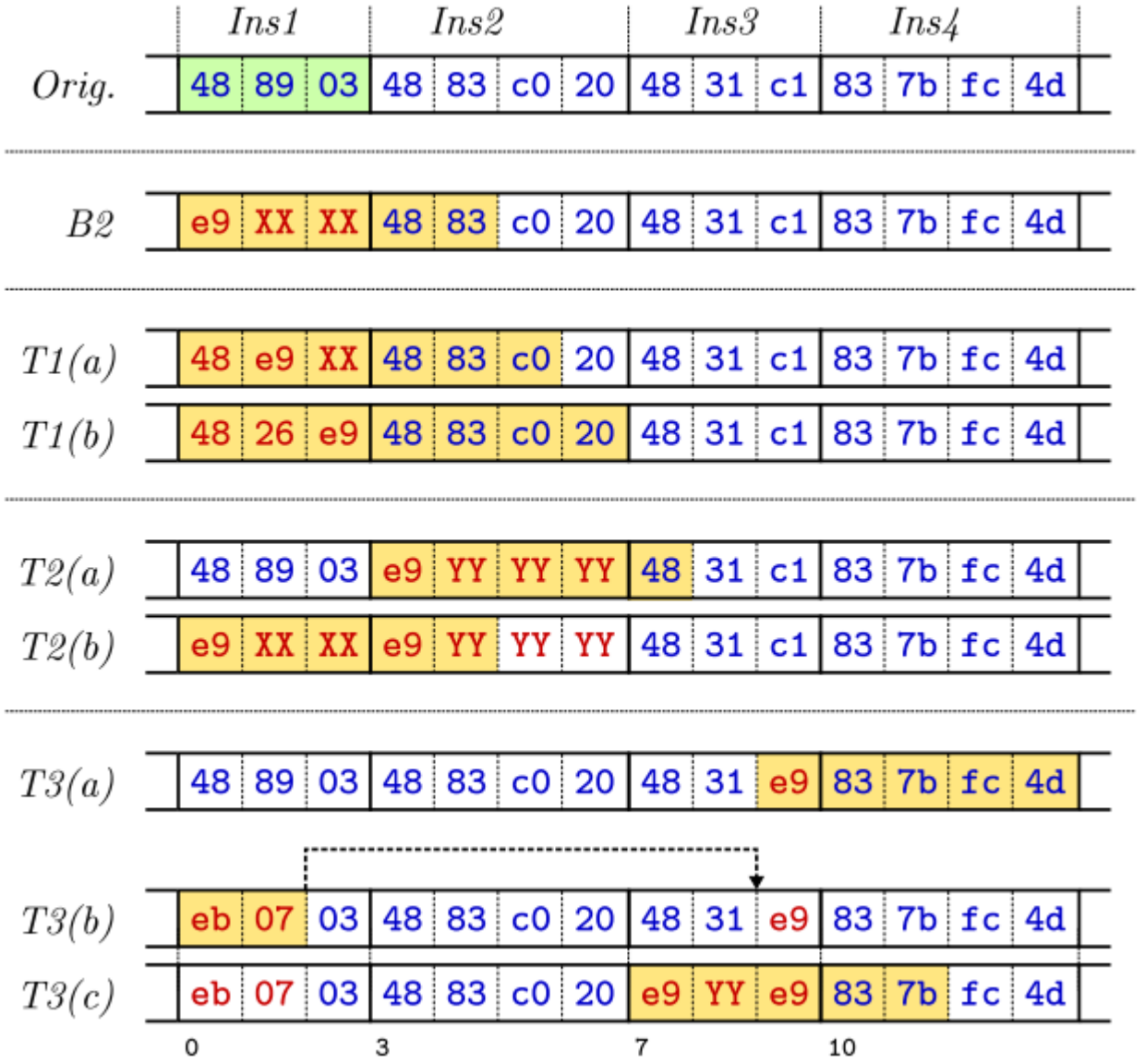
# T2: Successor Eviction

- 使用后一个指令(ins2)的pacth冗余来填充跳转指令(ins1).
- 比如利用T1策略将ins2填充为e9 YY YY YY, 那么可以再次应用T1策略让ins1利用ins2的冗余e9 YY, 那么可以控制的范围就成了0xYYe9XXXX.
- 与T1不同, T2的后继指令可以使用单字节指令.



# T3: Neighbour Eviction

- 通过短跳转(-128~127)来跳转到附近的可用指令. 操作码是0xeb, 跳转字节为0x00~0xFF.
- 到达后结合T1和T2使用得到更大的覆盖范围.
- JShort -> JPatch -> trampoline



# T3策略在CVE-2019-18408的应用

- T3短跳转到422ad3处, 该处组合利用T1和T2策略转向了两个蹦床.

# 三个策略组合带来的问题

- 许多程序需要进行许多指令的patch. 但修补策略相互干扰时就会出现复杂的问题.
- 比如应用了T1策略修补了Ins1, 那么就会锁定Ins2里的重叠字节值.
- T2和T3也有类似的锁定问题.
- 为了管理多个补丁的位置, 作者提出了逆向补丁策略S1

*Ins1*: `mov %rax,(%rbx)`  *Ins3*: `xor %rax,%rcx`

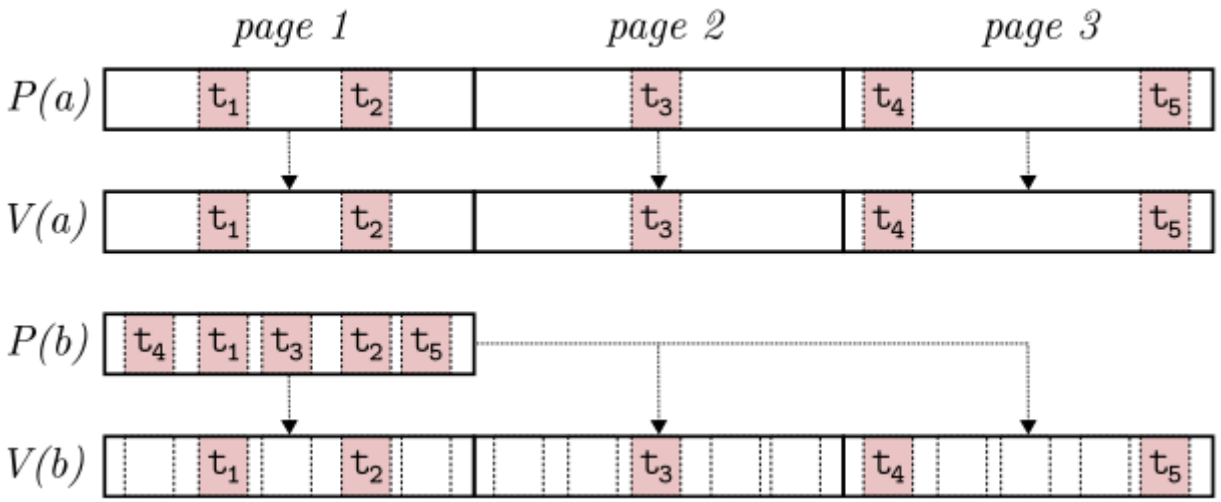
*Ins2*: `add $32,%rax`  *Ins4*: `cmpl $77,-4(%rbx)`

# S1: Reverse order patching strategy

- 基本思想: 因为指令双关后锁定后续指令, 因此修补时按”最高到最低”地址的顺序修补指令.
- 比如优先修补Ins2指令, 尽可能修改/锁定Ins3或Ins4的字节. Ins2修补完成后, 才尝试对Ins1进行修补. 这样不会Ins1的修补不会影响到Ins2
- 实现: 为修补双关指令的每个字节维护了一个bool值来记录该字节的锁定状态. 但满足以下情况时会设置为锁定状态(True).
  - Modified: 字节值被覆盖
  - Punned: 字节值未被覆盖, 但是用作了双关指令的一部分.
- 故而策略T1/2/3只能(1)修改未锁定字节, 并且(2)只能在当前补丁位置之后锁定字节. (也就是T3短跳必须为正偏移)

# 内存和文件尺寸管理

- 基本思想, 将物理页面中位置不重叠的蹦床合并来优化物理内存利用率.




- 比如P(a)中占了3个物理页, 但实际上可以将三个物理页上不冲突的蹦床合并到同一个物理页上, 这样就能达到减少物理占用的目的.
- 作者使用简单的贪心算法来合并各个蹦床分区. (性能开销也小)

# ELF文件重写

- E9Patch将新数据追加到文件末尾, 避免了移动现有数据和重新计算ELF文件偏移量.
- 程序加载期间还必须将新的物理页面映射到程序的虚拟地址空间, 为此E9Patch使用一个小型的loader替换程序入口点. 将蹦床/插装页面映射到正确的位置再返回到程序实际入口点.

# E9Patch应用对象

- PIE文件:
  - E9Patch可以应用于PIE和非PIE文件, 但PIE文件更易于修补, 因为PIE代码段会被加载到高内存地址, 相对更安全地远离无效的负地址范围.
  - 非PIE文件通常加载到低固定地址, 例如0x400000, 这意味着大多数负偏移量都是无效的.
- 共享链接对象/库
- 混合修补/未修补的代码: E9Patch不会移动指令, 因此可以安全地混合已修补和未修补的二进制代码.

# E9Patch的局限性

- 对于以下情况无法适用:
  L1: 虚拟地址空间不足
  L2: 单字节指令
  L3: 尝试修补许多指令
- (L1) 具有很大代码/数据段的程序可能会限制蹦床的虚拟地址空间.
- (L2) 单字节指令不能用T1修补, T3仅可以针对单个短跳转地址, 从而限制了适用性. 对于x86_64主要影响ret/push/pop, 其他大多数常见指令都是2字节以上.
- (L3) 修补策略之间可能相互依赖, 因此尝试修补(或几乎)所有的指令会造成干扰并限制适用性.
- 所幸, L1不适用于大多数程序, L2和L3也与许多应用无关

# E9Patch的实验评估

- Here, we instrument the (.text) section for each application. Column (Size) is the binary size in MB, (#Loc) is the number of patch locations, (Base%) is the percentage of successful patchings using baseline methods (B1+B2), (T1/T2/T3%) is the percent- age ofsuccessful patchings for each tactic, (Succ%) is the over- all percentage of successful patchings (B1+B2+T1+T2+T3), (Time%) is the overall runtime performance overhead, and (Size%) is the overall output binary size over the original. For the latter, the physical page grouping optimization (Section 4) has been applied with a granularity ofM=1 (i.e., maximum aggression). All experiments are run on a Xeon Silver 4114 Processor (2.20GHz with 32GB of RAM).

**Table 1.** Patching Statistics. Binaries marked by (†) are *position independent executables* (PIE).

| Binary | Size | Jmp/Jcc instructions (*A1*) | | | | | | | | Heap write instructions (*A2*) | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | (MB) | #Loc | Base% | *T1%* | *T2%* | *T3%* | Succ% | Time% | Size% | #Loc | Base% | *T1%* | *T2%* | *T3%* | Succ% | Time% | Size% |
| *perlbench* | 1.25 | 36821 | 86.88 | 7.40 | 1.45 | 4.27 | 100.00 | 459.59 | 174.28 | 7522 | 71.16 | 24.42 | 1.18 | 3.23 | 100.00 | 244.90 | 116.66 |
| *bzip2* | 0.07 | 1484 | 79.85 | 13.61 | 2.22 | 4.31 | 100.00 | 280.85 | 199.45 | 1044 | 68.39 | 26.05 | 2.49 | 3.07 | 100.00 | 279.67 | 170.95 |
| *gcc* | 3.77 | 97901 | 85.66 | 8.29 | 1.62 | 4.43 | 100.00 | 364.41 | 164.50 | 14328 | 70.60 | 24.95 | 0.68 | 3.78 | 100.00 | 148.73 | 109.90 |
| *bwaves* | 0.08 | 314 | 71.34 | 2.87 | 0.32 | 25.48 | 100.00 | 107.08 | 137.01 | 1168 | 92.55 | 7.36 | 0.00 | 0.09 | 100.00 | 139.02 | 142.43 |
| *gamess* | 12.22 | 125620 | 59.91 | 15.01 | 5.05 | 19.76 | 99.73 | 226.16 | 131.14 | 279592 | 87.58 | 9.65 | 0.50 | 2.20 | 99.94 | 321.89 | 136.93 |
| *mcf* | 0.02 | 295 | 68.47 | 20.00 | 4.41 | 7.12 | 100.00 | 194.92 | 203.75 | 220 | 75.91 | 20.00 | 1.36 | 2.73 | 100.00 | 141.02 | 221.51 |
| *milc* | 0.14 | 1940 | 80.62 | 13.40 | 1.29 | 4.69 | 100.00 | 115.03 | 157.13 | 699 | 84.84 | 13.16 | 0.29 | 1.72 | 100.00 | 117.54 | 119.14 |
| *zeusmp* | 0.52 | 3191 | 53.74 | 11.66 | 2.98 | 30.30 | 98.68 | 145.34 | 125.28 | 6106 | 82.61 | 12.15 | 0.39 | 4.67 | 99.82 | 131.50 | 128.74 |
| *gromacs* | 1.20 | 12058 | 80.19 | 11.49 | 1.38 | 6.94 | 100.00 | 116.16 | 133.01 | 16940 | 93.87 | 5.50 | 0.11 | 0.53 | 100.00 | 148.07 | 123.71 |
| *cactusADM* | 0.91 | 12847 | 78.94 | 13.32 | 2.30 | 5.44 | 100.00 | 101.43 | 140.70 | 5420 | 86.85 | 11.62 | 0.41 | 1.13 | 100.00 | 119.48 | 113.45 |
| *leslie3d* | 0.18 | 2584 | 44.43 | 27.67 | 12.46 | 15.44 | 100.00 | 151.89 | 174.56 | 2761 | 91.34 | 8.22 | 0.04 | 0.40 | 100.00 | 172.08 | 138.47 |
| *namd* | 0.33 | 4879 | 73.42 | 13.88 | 2.75 | 9.96 | 100.00 | 146.78 | 154.81 | 2498 | 71.46 | 28.14 | 0.20 | 0.20 | 100.00 | 138.01 | 120.42 |
| *gobmk* | 4.03 | 17912 | 75.88 | 14.72 | 2.57 | 6.83 | 100.00 | 368.97 | 113.80 | 2777 | 79.33 | 15.56 | 0.94 | 4.18 | 100.00 | 179.24 | 102.30 |
| *dealII* | 4.20 | 61317 | 71.31 | 14.99 | 4.50 | 9.19 | 100.00 | 386.08 | 144.34 | 25590 | 80.47 | 17.83 | 0.17 | 1.52 | 99.99 | 168.86 | 112.27 |
| *soplex* | 0.49 | 10125 | 79.72 | 11.57 | 2.58 | 6.13 | 100.00 | 244.23 | 162.93 | 4188 | 83.05 | 15.28 | 0.53 | 1.15 | 100.00 | 162.98 | 121.64 |
| *povray* | 1.19 | 20520 | 86.92 | 7.39 | 1.49 | 4.20 | 100.00 | 408.33 | 146.34 | 9377 | 84.50 | 13.46 | 0.37 | 1.66 | 100.00 | 186.36 | 116.37 |
| *calculix* | 2.17 | 30343 | 70.48 | 17.75 | 2.89 | 8.88 | 100.00 | 132.78 | 141.24 | 32197 | 85.62 | 13.02 | 0.38 | 0.98 | 100.00 | 126.13 | 128.26 |
| *hmmer* | 0.33 | 6748 | 77.71 | 13.96 | 1.99 | 6.34 | 100.00 | 182.94 | 174.52 | 3061 | 75.11 | 22.64 | 0.65 | 1.60 | 100.00 | 468.53 | 129.85 |
| *sjeng* | 0.16 | 3473 | 83.01 | 10.14 | 1.79 | 5.07 | 100.00 | 444.13 | 177.02 | 683 | 84.77 | 12.74 | 0.15 | 2.34 | 100.00 | 134.78 | 123.32 |
| *GemsFDTD* | 0.58 | 9120 | 41.62 | 17.28 | 21.44 | 19.66 | 100.00 | 104.78 | 166.74 | 10345 | 93.23 | 6.54 | 0.04 | 0.18 | 100.00 | 111.64 | 132.30 |
| *libquantum* | 0.05 | 732 | 75.55 | 15.85 | 3.42 | 5.19 | 100.00 | 325.81 | 190.57 | 186 | 76.34 | 17.74 | 0.00 | 5.91 | 100.00 | 269.68 | 139.82 |
| *h264ref* | 0.58 | 9920 | 80.30 | 13.58 | 1.22 | 4.90 | 100.00 | 206.61 | 151.60 | 4981 | 81.87 | 15.42 | 0.80 | 1.91 | 100.00 | 178.89 | 122.04 |
| *tonto* | 6.21 | 48247 | 52.65 | 22.84 | 8.63 | 15.88 | 100.00 | 196.21 | 125.54 | 164788 | 90.05 | 9.09 | 0.15 | 0.71 | 100.00 | 192.72 | 141.53 |
| *lbm* | 0.02 | 106 | 67.92 | 17.92 | 3.77 | 10.38 | 100.00 | 103.80 | 193.33 | 111 | 93.69 | 6.31 | 0.00 | 0.00 | 100.00 | 110.13 | 148.74 |
| *omnetpp* | 0.79 | 9568 | 78.08 | 13.96 | 2.16 | 5.79 | 100.00 | 203.90 | 135.45 | 5020 | 74.12 | 18.57 | 3.01 | 4.30 | 100.00 | 144.81 | 117.53 |
| *astar* | 0.05 | 769 | 78.54 | 13.78 | 2.21 | 5.46 | 100.00 | 287.64 | 180.98 | 491 | 72.91 | 23.01 | 0.61 | 3.46 | 100.00 | 137.64 | 152.03 |
| *sphinx3* | 0.21 | 3500 | 79.20 | 12.17 | 2.03 | 6.60 | 100.00 | 196.27 | 170.99 | 1159 | 73.94 | 22.95 | 0.78 | 2.33 | 100.00 | 129.17 | 123.55 |
| *xalancbmk* | 5.99 | 81285 | 75.66 | 14.10 | 3.50 | 6.74 | 100.00 | 474.07 | 137.04 | 32761 | 79.51 | 17.61 | 0.43 | 2.45 | 100.00 | 130.16 | 111.38 |
| #Total/Avg% | 47.74 | 613619 | 72.79 | 13.95 | 3.73 | 9.48 | 99.94 | 210.81 | 157.43 | 636013 | 81.63 | 15.68 | 0.60 | 2.09 | 99.99 | 164.71 | 130.90 |
| *inkscape*† *0.91* | 15.44 | 195731 | 97.83 | 1.31 | 0.86 | 0.00 | 100.00 | – | 130.40 | 105431 | 99.96 | 0.03 | 0.01 | 0.00 | 100.00 | – | 109.58 |
| *gimp 2.8.16* | 5.75 | 71321 | 71.75 | 18.69 | 2.49 | 7.08 | 100.00 | – | 135.74 | 15730 | 84.83 | 12.59 | 0.64 | 1.95 | 100.00 | – | 106.00 |
| *vim*† *7.4* | 2.44 | 72221 | 99.18 | 0.23 | 0.60 | 0.00 | 100.00 | – | 173.31 | 13279 | 99.92 | 0.02 | 0.06 | 0.00 | 100.00 | – | 110.77 |
| *git 2.7.4* | 1.87 | 44441 | 80.06 | 11.91 | 2.14 | 5.88 | 100.00 | – | 169.16 | 9072 | 68.06 | 27.62 | 1.16 | 3.16 | 100.00 | – | 113.60 |
| *pdflatex 2.6* | 0.91 | 22105 | 82.05 | 10.46 | 2.06 | 5.42 | 100.00 | – | 168.72 | 6060 | 70.61 | 24.97 | 1.25 | 3.17 | 100.00 | – | 118.70 |
| *xterm 322* | 0.54 | 11593 | 79.12 | 12.45 | 3.04 | 5.39 | 100.00 | – | 166.23 | 2681 | 89.11 | 9.40 | 0.41 | 1.08 | 100.00 | – | 113.16 |
| *evince*† *3.18.2* | 0.42 | 3636 | 99.59 | 0.30 | 0.11 | 0.00 | 100.00 | – | 131.63 | 716 | 99.86 | 0.00 | 0.14 | 0.00 | 100.00 | – | 107.86 |
| *make 4.1* | 0.21 | 4807 | 79.34 | 12.96 | 1.71 | 5.99 | 100.00 | – | 182.78 | 1383 | 74.98 | 20.46 | 0.94 | 3.62 | 100.00 | – | 125.48 |
| *libc.so 2.23* | 1.87 | 52393 | 81.19 | 11.55 | 2.23 | 5.03 | 100.00 | – | 247.67 | 24686 | 74.32 | 21.98 | 1.05 | 2.64 | 100.00 | – | 203.87 |
| *libc++.so 6.0.21* | 1.57 | 20593 | 75.14 | 13.02 | 4.60 | 7.24 | 100.00 | – | 184.99 | 15442 | 67.56 | 27.76 | 0.99 | 3.68 | 100.00 | – | 168.80 |
| *Chrome*† *78.0* | 152.51 | 3800565 | 93.20 | 4.68 | 1.87 | 0.25 | 100.00 | – | 226.31 | 2624800 | 99.38 | 0.49 | 0.11 | 0.01 | 100.00 | – | 197.68 |
| *FireFox*† *70.0* | 0.52 | 13971 | 98.02 | 0.54 | 1.44 | 0.00 | 100.00 | – | 269.22 | 7355 | 99.90 | 0.10 | 0.00 | 0.00 | 100.00 | – | 208.06 |
| *libxul.so 70.0* | 115.03 | 1463369 | 68.55 | 15.08 | 5.26 | 11.10 | 99.99 | – | 194.55 | 666109 | 75.72 | 20.61 | 0.62 | 3.06 | 100.00 | – | 174.22 |

# E9Patch的实验评估

- 覆盖率: 即成功修补指令的比率
- 单独使用B1和B2的修补覆盖率在72.79%和81.63%
- 随后的每个策略T1-T3都会提高覆盖率
- 如果没有T3, A1的总覆盖率仅为90.5%
- PIE的修补基准覆盖率在93%以上, 比非PIE更容易成功修补.

**Table 1.** Patching Statistics. Binaries marked by (†) are *position independent executables* (PIE).

| Binary | Size | Jmp/Jcc instructions (*A1*) | | | | | | | | Heap write instructions (*A2*) | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | (MB) | #Loc | Base% | *T1%* | *T2%* | *T3%* | Succ% | Time% | Size% | #Loc | Base% | *T1%* | *T2%* | *T3%* | Succ% | Time% | Size% |
| *perlbench* | 1.25 | 36821 | 86.88 | 7.40 | 1.45 | 4.27 | 100.00 | 459.59 | 174.28 | 7522 | 71.16 | 24.42 | 1.18 | 3.23 | 100.00 | 244.90 | 116.66 |
| *bzip2* | 0.07 | 1484 | 79.85 | 13.61 | 2.22 | 4.31 | 100.00 | 280.85 | 199.45 | 1044 | 68.39 | 26.05 | 2.49 | 3.07 | 100.00 | 279.67 | 170.95 |
| *gcc* | 3.77 | 97901 | 85.66 | 8.29 | 1.62 | 4.43 | 100.00 | 364.41 | 164.50 | 14328 | 70.60 | 24.95 | 0.68 | 3.78 | 100.00 | 148.73 | 109.90 |
| *bwaves* | 0.08 | 314 | 71.34 | 2.87 | 0.32 | 25.48 | 100.00 | 107.08 | 137.01 | 1168 | 92.55 | 7.36 | 0.00 | 0.09 | 100.00 | 139.02 | 142.43 |
| *gamess* | 12.22 | 125620 | 59.91 | 15.01 | 5.05 | 19.76 | 99.73 | 226.16 | 131.14 | 279592 | 87.58 | 9.65 | 0.50 | 2.20 | 99.94 | 321.89 | 136.93 |
| *mcf* | 0.02 | 295 | 68.47 | 20.00 | 4.41 | 7.12 | 100.00 | 194.92 | 203.75 | 220 | 75.91 | 20.00 | 1.36 | 2.73 | 100.00 | 141.02 | 221.51 |
| *milc* | 0.14 | 1940 | 80.62 | 13.40 | 1.29 | 4.69 | 100.00 | 115.03 | 157.13 | 699 | 84.84 | 13.16 | 0.29 | 1.72 | 100.00 | 117.54 | 119.14 |
| *zeusmp* | 0.52 | 3191 | 53.74 | 11.66 | 2.98 | 30.30 | 98.68 | 145.34 | 125.28 | 6106 | 82.61 | 12.15 | 0.39 | 4.67 | 99.82 | 131.50 | 128.74 |
| *gromacs* | 1.20 | 12058 | 80.19 | 11.49 | 1.38 | 6.94 | 100.00 | 116.16 | 133.01 | 16940 | 93.87 | 5.50 | 0.11 | 0.53 | 100.00 | 148.07 | 123.71 |
| *cactusADM* | 0.91 | 12847 | 78.94 | 13.32 | 2.30 | 5.44 | 100.00 | 101.43 | 140.70 | 5420 | 86.85 | 11.62 | 0.41 | 1.13 | 100.00 | 119.48 | 113.45 |
| *leslie3d* | 0.18 | 2584 | 44.43 | 27.67 | 12.46 | 15.44 | 100.00 | 151.89 | 174.56 | 2761 | 91.34 | 8.22 | 0.04 | 0.40 | 100.00 | 172.08 | 138.47 |
| *namd* | 0.33 | 4879 | 73.42 | 13.88 | 2.75 | 9.96 | 100.00 | 146.78 | 154.81 | 2498 | 71.46 | 28.14 | 0.20 | 0.20 | 100.00 | 138.01 | 120.42 |
| *gobmk* | 4.03 | 17912 | 75.88 | 14.72 | 2.57 | 6.83 | 100.00 | 368.97 | 113.80 | 2777 | 79.33 | 15.56 | 0.94 | 4.18 | 100.00 | 179.24 | 102.30 |
| *dealII* | 4.20 | 61317 | 71.31 | 14.99 | 4.50 | 9.19 | 100.00 | 386.08 | 144.34 | 25590 | 80.47 | 17.83 | 0.17 | 1.52 | 99.99 | 168.86 | 112.27 |
| *soplex* | 0.49 | 10125 | 79.72 | 11.57 | 2.58 | 6.13 | 100.00 | 244.23 | 162.93 | 4188 | 83.05 | 15.28 | 0.53 | 1.15 | 100.00 | 162.98 | 121.64 |
| *povray* | 1.19 | 20520 | 86.92 | 7.39 | 1.49 | 4.20 | 100.00 | 408.33 | 146.34 | 9377 | 84.50 | 13.46 | 0.37 | 1.66 | 100.00 | 186.36 | 116.37 |
| *calculix* | 2.17 | 30343 | 70.48 | 17.75 | 2.89 | 8.88 | 100.00 | 132.78 | 141.24 | 32197 | 85.62 | 13.02 | 0.38 | 0.98 | 100.00 | 126.13 | 128.26 |
| *hmmer* | 0.33 | 6748 | 77.71 | 13.96 | 1.99 | 6.34 | 100.00 | 182.94 | 174.52 | 3061 | 75.11 | 22.64 | 0.65 | 1.60 | 100.00 | 468.53 | 129.85 |
| *sjeng* | 0.16 | 3473 | 83.01 | 10.14 | 1.79 | 5.07 | 100.00 | 444.13 | 177.02 | 683 | 84.77 | 12.74 | 0.15 | 2.34 | 100.00 | 134.78 | 123.32 |
| *GemsFDTD* | 0.58 | 9120 | 41.62 | 17.28 | 21.44 | 19.66 | 100.00 | 104.78 | 166.74 | 10345 | 93.23 | 6.54 | 0.04 | 0.18 | 100.00 | 111.64 | 132.30 |
| *libquantum* | 0.05 | 732 | 75.55 | 15.85 | 3.42 | 5.19 | 100.00 | 325.81 | 190.57 | 186 | 76.34 | 17.74 | 0.00 | 5.91 | 100.00 | 269.68 | 139.82 |
| *h264ref* | 0.58 | 9920 | 80.30 | 13.58 | 1.22 | 4.90 | 100.00 | 206.61 | 151.60 | 4981 | 81.87 | 15.42 | 0.80 | 1.91 | 100.00 | 178.89 | 122.04 |
| *tonto* | 6.21 | 48247 | 52.65 | 22.84 | 8.63 | 15.88 | 100.00 | 196.21 | 125.54 | 164788 | 90.05 | 9.09 | 0.15 | 0.71 | 100.00 | 192.72 | 141.53 |
| *lbm* | 0.02 | 106 | 67.92 | 17.92 | 3.77 | 10.38 | 100.00 | 103.80 | 193.33 | 111 | 93.69 | 6.31 | 0.00 | 0.00 | 100.00 | 110.13 | 148.74 |
| *omnetpp* | 0.79 | 9568 | 78.08 | 13.96 | 2.16 | 5.79 | 100.00 | 203.90 | 135.45 | 5020 | 74.12 | 18.57 | 3.01 | 4.30 | 100.00 | 144.81 | 117.53 |
| *astar* | 0.05 | 769 | 78.54 | 13.78 | 2.21 | 5.46 | 100.00 | 287.64 | 180.98 | 491 | 72.91 | 23.01 | 0.61 | 3.46 | 100.00 | 137.64 | 152.03 |
| *sphinx3* | 0.21 | 3500 | 79.20 | 12.17 | 2.03 | 6.60 | 100.00 | 196.27 | 170.99 | 1159 | 73.94 | 22.95 | 0.78 | 2.33 | 100.00 | 129.17 | 123.55 |
| *xalancbmk* | 5.99 | 81285 | 75.66 | 14.10 | 3.50 | 6.74 | 100.00 | 474.07 | 137.04 | 32761 | 79.51 | 17.61 | 0.43 | 2.45 | 100.00 | 130.16 | 111.38 |
| #Total/Avg% | 47.74 | 613619 | 72.79 | 13.95 | 3.73 | 9.48 | 99.94 | 210.81 | 157.43 | 636013 | 81.63 | 15.68 | 0.60 | 2.09 | 99.99 | 164.71 | 130.90 |
| *inkscape*† *0.91* | 15.44 | 195731 | 97.83 | 1.31 | 0.86 | 0.00 | 100.00 | – | 130.40 | 105431 | 99.96 | 0.03 | 0.01 | 0.00 | 100.00 | – | 109.58 |
| *gimp 2.8.16* | 5.75 | 71321 | 71.75 | 18.69 | 2.49 | 7.08 | 100.00 | – | 135.74 | 15730 | 84.83 | 12.59 | 0.64 | 1.95 | 100.00 | – | 106.00 |
| *vim*† *7.4* | 2.44 | 72221 | 99.18 | 0.23 | 0.60 | 0.00 | 100.00 | – | 173.31 | 13279 | 99.92 | 0.02 | 0.06 | 0.00 | 100.00 | – | 110.77 |
| *git 2.7.4* | 1.87 | 44441 | 80.06 | 11.91 | 2.14 | 5.88 | 100.00 | – | 169.16 | 9072 | 68.06 | 27.62 | 1.16 | 3.16 | 100.00 | – | 113.60 |
| *pdflatex 2.6* | 0.91 | 22105 | 82.05 | 10.46 | 2.06 | 5.42 | 100.00 | – | 168.72 | 6060 | 70.61 | 24.97 | 1.25 | 3.17 | 100.00 | – | 118.70 |
| *xterm 322* | 0.54 | 11593 | 79.12 | 12.45 | 3.04 | 5.39 | 100.00 | – | 166.23 | 2681 | 89.11 | 9.40 | 0.41 | 1.08 | 100.00 | – | 113.16 |
| *evince*† *3.18.2* | 0.42 | 3636 | 99.59 | 0.30 | 0.11 | 0.00 | 100.00 | – | 131.63 | 716 | 99.86 | 0.00 | 0.14 | 0.00 | 100.00 | – | 107.86 |
| *make 4.1* | 0.21 | 4807 | 79.34 | 12.96 | 1.71 | 5.99 | 100.00 | – | 182.78 | 1383 | 74.98 | 20.46 | 0.94 | 3.62 | 100.00 | – | 125.48 |
| *libc.so 2.23* | 1.87 | 52393 | 81.19 | 11.55 | 2.23 | 5.03 | 100.00 | – | 247.67 | 24686 | 74.32 | 21.98 | 1.05 | 2.64 | 100.00 | – | 203.87 |
| *libc++.so 6.0.21* | 1.57 | 20593 | 75.14 | 13.02 | 4.60 | 7.24 | 100.00 | – | 184.99 | 15442 | 67.56 | 27.76 | 0.99 | 3.68 | 100.00 | – | 168.80 |
| *Chrome*† *78.0* | 152.51 | 3800565 | 93.20 | 4.68 | 1.87 | 0.25 | 100.00 | – | 226.31 | 2624800 | 99.38 | 0.49 | 0.11 | 0.01 | 100.00 | – | 197.68 |
| *FireFox*† *70.0* | 0.52 | 13971 | 98.02 | 0.54 | 1.44 | 0.00 | 100.00 | – | 269.22 | 7355 | 99.90 | 0.10 | 0.00 | 0.00 | 100.00 | – | 208.06 |
| *libxul.so 70.0* | 115.03 | 1463369 | 68.55 | 15.08 | 5.26 | 11.10 | 99.99 | – | 194.55 | 666109 | 75.72 | 20.61 | 0.62 | 3.06 | 100.00 | – | 174.22 |

# E9Patch的实验评估

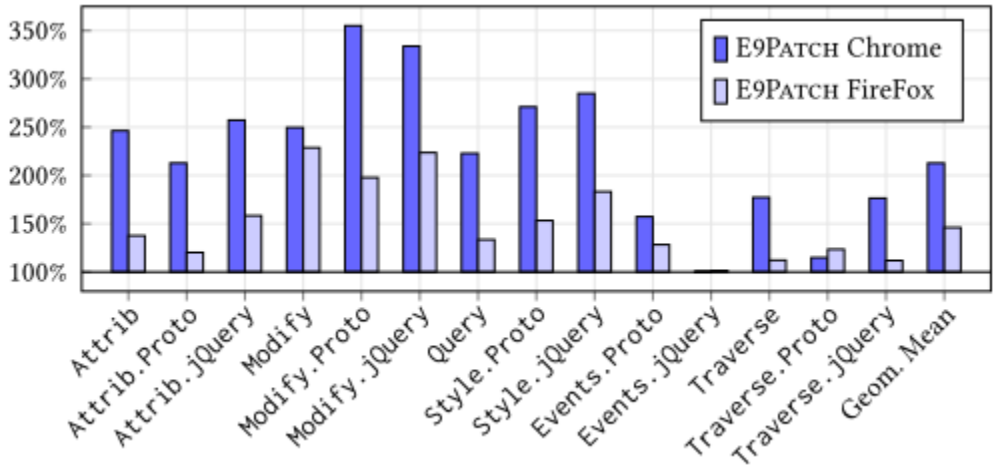
- 启用物理页面分组后, 文件大小分别增加了57.43%/30.90%.
- 而未启用分组方法. 文件大小分别增加到2239.83%/568.96%
- 突出了物理内存优化的重要性

**Table 1.** Patching Statistics. Binaries marked by (†) are *position independent executables* (PIE).

| Binary | Size | Jmp/Jcc instructions (*A1*) | | | | | | | | Heap write instructions (*A2*) | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | (MB) | #Loc | Base% | *T1%* | *T2%* | *T3%* | Succ% | Time% | Size% | #Loc | Base% | *T1%* | *T2%* | *T3%* | Succ% | Time% | Size% |
| *perlbench* | 1.25 | 36821 | 86.88 | 7.40 | 1.45 | 4.27 | 100.00 | 459.59 | 174.28 | 7522 | 71.16 | 24.42 | 1.18 | 3.23 | 100.00 | 244.90 | 116.66 |
| *bzip2* | 0.07 | 1484 | 79.85 | 13.61 | 2.22 | 4.31 | 100.00 | 280.85 | 199.45 | 1044 | 68.39 | 26.05 | 2.49 | 3.07 | 100.00 | 279.67 | 170.95 |
| *gcc* | 3.77 | 97901 | 85.66 | 8.29 | 1.62 | 4.43 | 100.00 | 364.41 | 164.50 | 14328 | 70.60 | 24.95 | 0.68 | 3.78 | 100.00 | 148.73 | 109.90 |
| *bwaves* | 0.08 | 314 | 71.34 | 2.87 | 0.32 | 25.48 | 100.00 | 107.08 | 137.01 | 1168 | 92.55 | 7.36 | 0.00 | 0.09 | 100.00 | 139.02 | 142.43 |
| *gamess* | 12.22 | 125620 | 59.91 | 15.01 | 5.05 | 19.76 | 99.73 | 226.16 | 131.14 | 279592 | 87.58 | 9.65 | 0.50 | 2.20 | 99.94 | 321.89 | 136.93 |
| *mcf* | 0.02 | 295 | 68.47 | 20.00 | 4.41 | 7.12 | 100.00 | 194.92 | 203.75 | 220 | 75.91 | 20.00 | 1.36 | 2.73 | 100.00 | 141.02 | 221.51 |
| *milc* | 0.14 | 1940 | 80.62 | 13.40 | 1.29 | 4.69 | 100.00 | 115.03 | 157.13 | 699 | 84.84 | 13.16 | 0.29 | 1.72 | 100.00 | 117.54 | 119.14 |
| *zeusmp* | 0.52 | 3191 | 53.74 | 11.66 | 2.98 | 30.30 | 98.68 | 145.34 | 125.28 | 6106 | 82.61 | 12.15 | 0.39 | 4.67 | 99.82 | 131.50 | 128.74 |
| *gromacs* | 1.20 | 12058 | 80.19 | 11.49 | 1.38 | 6.94 | 100.00 | 116.16 | 133.01 | 16940 | 93.87 | 5.50 | 0.11 | 0.53 | 100.00 | 148.07 | 123.71 |
| *cactusADM* | 0.91 | 12847 | 78.94 | 13.32 | 2.30 | 5.44 | 100.00 | 101.43 | 140.70 | 5420 | 86.85 | 11.62 | 0.41 | 1.13 | 100.00 | 119.48 | 113.45 |
| *leslie3d* | 0.18 | 2584 | 44.43 | 27.67 | 12.46 | 15.44 | 100.00 | 151.89 | 174.56 | 2761 | 91.34 | 8.22 | 0.04 | 0.40 | 100.00 | 172.08 | 138.47 |
| *namd* | 0.33 | 4879 | 73.42 | 13.88 | 2.75 | 9.96 | 100.00 | 146.78 | 154.81 | 2498 | 71.46 | 28.14 | 0.20 | 0.20 | 100.00 | 138.01 | 120.42 |
| *gobmk* | 4.03 | 17912 | 75.88 | 14.72 | 2.57 | 6.83 | 100.00 | 368.97 | 113.80 | 2777 | 79.33 | 15.56 | 0.94 | 4.18 | 100.00 | 179.24 | 102.30 |
| *dealII* | 4.20 | 61317 | 71.31 | 14.99 | 4.50 | 9.19 | 100.00 | 386.08 | 144.34 | 25590 | 80.47 | 17.83 | 0.17 | 1.52 | 99.99 | 168.86 | 112.27 |
| *soplex* | 0.49 | 10125 | 79.72 | 11.57 | 2.58 | 6.13 | 100.00 | 244.23 | 162.93 | 4188 | 83.05 | 15.28 | 0.53 | 1.15 | 100.00 | 162.98 | 121.64 |
| *povray* | 1.19 | 20520 | 86.92 | 7.39 | 1.49 | 4.20 | 100.00 | 408.33 | 146.34 | 9377 | 84.50 | 13.46 | 0.37 | 1.66 | 100.00 | 186.36 | 116.37 |
| *calculix* | 2.17 | 30343 | 70.48 | 17.75 | 2.89 | 8.88 | 100.00 | 132.78 | 141.24 | 32197 | 85.62 | 13.02 | 0.38 | 0.98 | 100.00 | 126.13 | 128.26 |
| *hmmer* | 0.33 | 6748 | 77.71 | 13.96 | 1.99 | 6.34 | 100.00 | 182.94 | 174.52 | 3061 | 75.11 | 22.64 | 0.65 | 1.60 | 100.00 | 468.53 | 129.85 |
| *sjeng* | 0.16 | 3473 | 83.01 | 10.14 | 1.79 | 5.07 | 100.00 | 444.13 | 177.02 | 683 | 84.77 | 12.74 | 0.15 | 2.34 | 100.00 | 134.78 | 123.32 |
| *GemsFDTD* | 0.58 | 9120 | 41.62 | 17.28 | 21.44 | 19.66 | 100.00 | 104.78 | 166.74 | 10345 | 93.23 | 6.54 | 0.04 | 0.18 | 100.00 | 111.64 | 132.30 |
| *libquantum* | 0.05 | 732 | 75.55 | 15.85 | 3.42 | 5.19 | 100.00 | 325.81 | 190.57 | 186 | 76.34 | 17.74 | 0.00 | 5.91 | 100.00 | 269.68 | 139.82 |
| *h264ref* | 0.58 | 9920 | 80.30 | 13.58 | 1.22 | 4.90 | 100.00 | 206.61 | 151.60 | 4981 | 81.87 | 15.42 | 0.80 | 1.91 | 100.00 | 178.89 | 122.04 |
| *tonto* | 6.21 | 48247 | 52.65 | 22.84 | 8.63 | 15.88 | 100.00 | 196.21 | 125.54 | 164788 | 90.05 | 9.09 | 0.15 | 0.71 | 100.00 | 192.72 | 141.53 |
| *lbm* | 0.02 | 106 | 67.92 | 17.92 | 3.77 | 10.38 | 100.00 | 103.80 | 193.33 | 111 | 93.69 | 6.31 | 0.00 | 0.00 | 100.00 | 110.13 | 148.74 |
| *omnetpp* | 0.79 | 9568 | 78.08 | 13.96 | 2.16 | 5.79 | 100.00 | 203.90 | 135.45 | 5020 | 74.12 | 18.57 | 3.01 | 4.30 | 100.00 | 144.81 | 117.53 |
| *astar* | 0.05 | 769 | 78.54 | 13.78 | 2.21 | 5.46 | 100.00 | 287.64 | 180.98 | 491 | 72.91 | 23.01 | 0.61 | 3.46 | 100.00 | 137.64 | 152.03 |
| *sphinx3* | 0.21 | 3500 | 79.20 | 12.17 | 2.03 | 6.60 | 100.00 | 196.27 | 170.99 | 1159 | 73.94 | 22.95 | 0.78 | 2.33 | 100.00 | 129.17 | 123.55 |
| *xalancbmk* | 5.99 | 81285 | 75.66 | 14.10 | 3.50 | 6.74 | 100.00 | 474.07 | 137.04 | 32761 | 79.51 | 17.61 | 0.43 | 2.45 | 100.00 | 130.16 | 111.38 |
| #Total/Avg% | 47.74 | 613619 | 72.79 | 13.95 | 3.73 | 9.48 | 99.94 | 210.81 | 157.43 | 636013 | 81.63 | 15.68 | 0.60 | 2.09 | 99.99 | 164.71 | 130.90 |
| *inkscape*† *0.91* | 15.44 | 195731 | 97.83 | 1.31 | 0.86 | 0.00 | 100.00 | – | 130.40 | 105431 | 99.96 | 0.03 | 0.01 | 0.00 | 100.00 | – | 109.58 |
| *gimp 2.8.16* | 5.75 | 71321 | 71.75 | 18.69 | 2.49 | 7.08 | 100.00 | – | 135.74 | 15730 | 84.83 | 12.59 | 0.64 | 1.95 | 100.00 | – | 106.00 |
| *vim*† *7.4* | 2.44 | 72221 | 99.18 | 0.23 | 0.60 | 0.00 | 100.00 | – | 173.31 | 13279 | 99.92 | 0.02 | 0.06 | 0.00 | 100.00 | – | 110.77 |
| *git 2.7.4* | 1.87 | 44441 | 80.06 | 11.91 | 2.14 | 5.88 | 100.00 | – | 169.16 | 9072 | 68.06 | 27.62 | 1.16 | 3.16 | 100.00 | – | 113.60 |
| *pdflatex 2.6* | 0.91 | 22105 | 82.05 | 10.46 | 2.06 | 5.42 | 100.00 | – | 168.72 | 6060 | 70.61 | 24.97 | 1.25 | 3.17 | 100.00 | – | 118.70 |
| *xterm 322* | 0.54 | 11593 | 79.12 | 12.45 | 3.04 | 5.39 | 100.00 | – | 166.23 | 2681 | 89.11 | 9.40 | 0.41 | 1.08 | 100.00 | – | 113.16 |
| *evince*† *3.18.2* | 0.42 | 3636 | 99.59 | 0.30 | 0.11 | 0.00 | 100.00 | – | 131.63 | 716 | 99.86 | 0.00 | 0.14 | 0.00 | 100.00 | – | 107.86 |
| *make 4.1* | 0.21 | 4807 | 79.34 | 12.96 | 1.71 | 5.99 | 100.00 | – | 182.78 | 1383 | 74.98 | 20.46 | 0.94 | 3.62 | 100.00 | – | 125.48 |
| *libc.so 2.23* | 1.87 | 52393 | 81.19 | 11.55 | 2.23 | 5.03 | 100.00 | – | 247.67 | 24686 | 74.32 | 21.98 | 1.05 | 2.64 | 100.00 | – | 203.87 |
| *libc++.so 6.0.21* | 1.57 | 20593 | 75.14 | 13.02 | 4.60 | 7.24 | 100.00 | – | 184.99 | 15442 | 67.56 | 27.76 | 0.99 | 3.68 | 100.00 | – | 168.80 |
| *Chrome*† *78.0* | 152.51 | 3800565 | 93.20 | 4.68 | 1.87 | 0.25 | 100.00 | – | 226.31 | 2624800 | 99.38 | 0.49 | 0.11 | 0.01 | 100.00 | – | 197.68 |
| *FireFox*† *70.0* | 0.52 | 13971 | 98.02 | 0.54 | 1.44 | 0.00 | 100.00 | – | 269.22 | 7355 | 99.90 | 0.10 | 0.00 | 0.00 | 100.00 | – | 208.06 |
| *libxul.so 70.0* | 115.03 | 1463369 | 68.55 | 15.08 | 5.26 | 11.10 | 99.99 | – | 194.55 | 666109 | 75.72 | 20.61 | 0.62 | 3.06 | 100.00 | – | 174.22 |

# E9Patch的实验评估

- 总的来说, 开销分别增加了110.81% / 64.71%
- E9Patch为了最大程度提高可扩展性, 没有采用内联代码, 因此相比其他二进制重写工具, 开销更大




- 拿chrome和firefox实验.
- 总的来说, 为chrome带来了113%的开销. 为firefox带来了46%的开销.
- Firefox可能更多运行JIT代码或非插装共享对象上