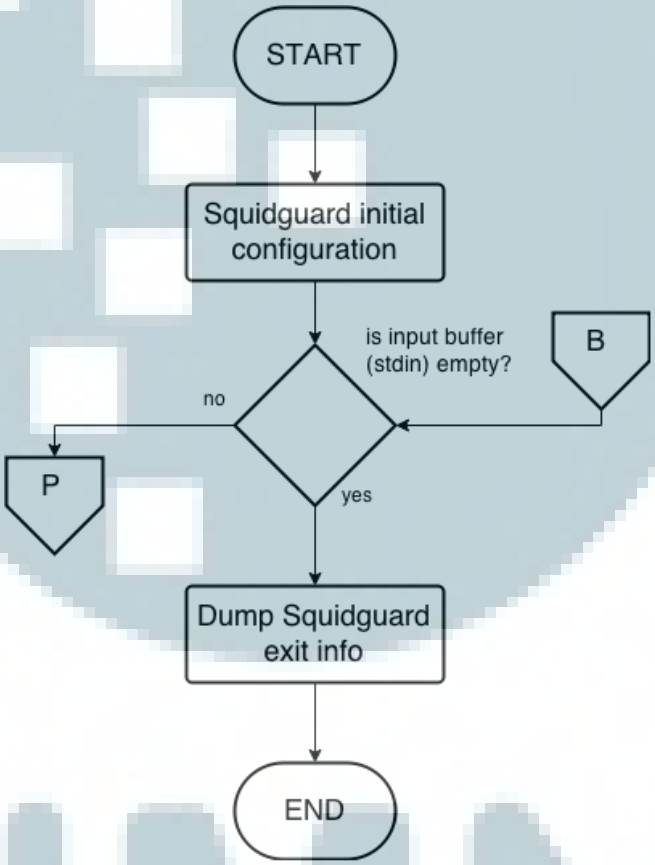
# BAB III
# METODOLOGI PENELITIAN

## 3.1 Gubahan Ulang Squidguard

### 3.1.1 Rancangan Awal Squidguard

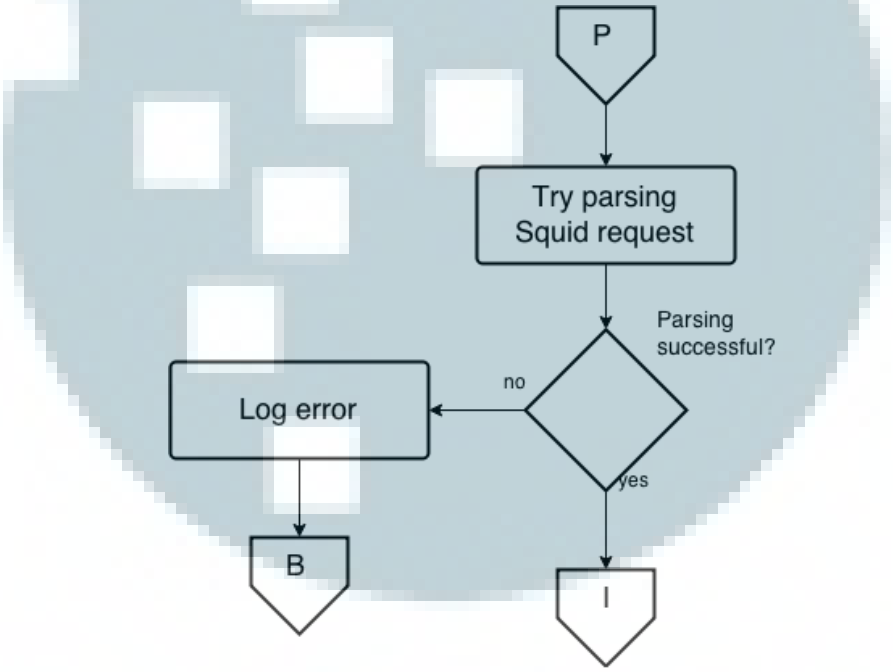
Berdasarkan pengamatan dan analisis kode sumber Squidguard, susunan bagan alir Squidguard dapat dilihat pada Gambar 8, Gambar 9, dan Gambar 10.




Gambar 8. Bagan alir Squidguard awal (bagian 1)

Pada bagan alir awal bagian pertama terlihat bahwa Squidguard memulai rutinnya dengan melakukan konfigurasi awal. Konfigurasi awal Squidguard ini termasuk mengakses log, mendeklarasikan variabel, menerjemahkan dokumen konfigurasi ke dalam memori, dan sebagainya. Konfigurasi awal dilakukan satu kali setiap Squidguard dijalankan.

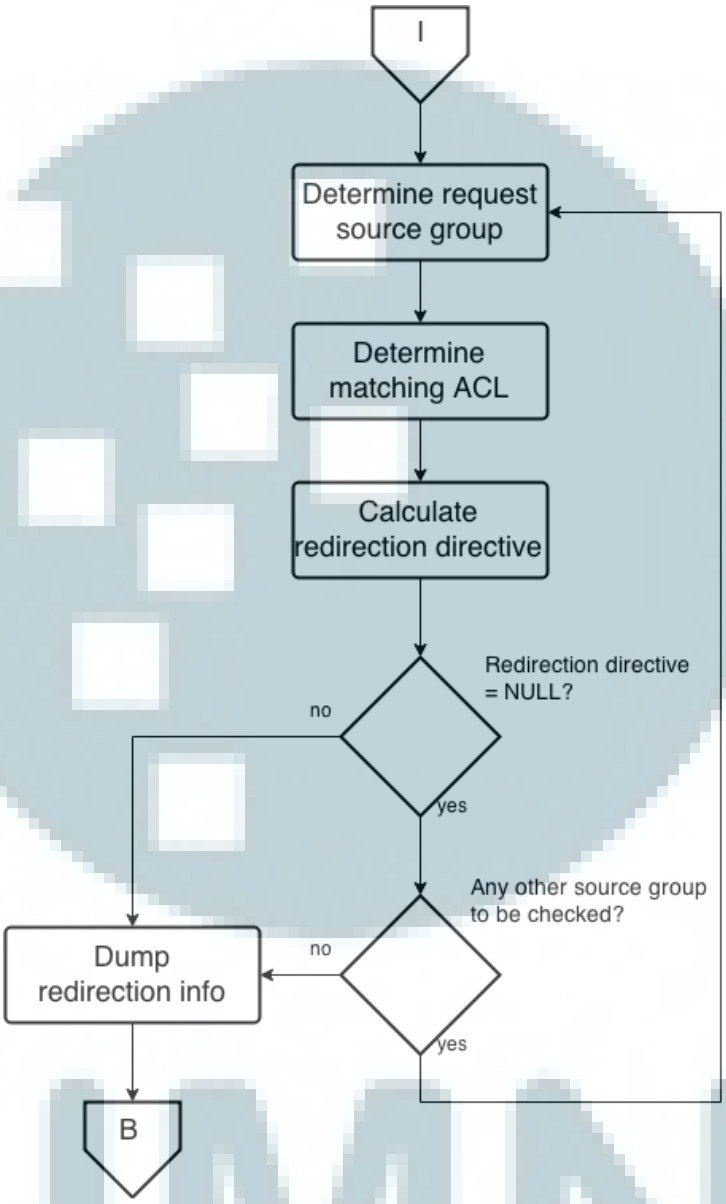
Setelah melakukan konfigurasi awal, Squidguard akan memeriksa keberadaan masukan dalam penyangga masukan, dalam hal ini `stdin`. Saat Squidguard dijalankan sebagai anak proses dari Squid, Squid akan menyangga masukan Squidguard; jalur komunikasi dari Squid bertindak sebagai `stdin` untuk Squidguard. Jika Squidguard dijalankan menggunakan terminal, akhir masukan (penanda end of line atau end of file) menjadi batas akhir masukan sehingga Squidguard dapat menghentikan dirinya. Jika ada masukan dalam penyangga, Squidguard akan mencoba melakukan penerjemahan masukan tersebut ke dalam bentuk yang dapat diproses lebih lanjut.




Gambar 9. Bagan alir Squidguard awal (bagian 2)

Penerjemahan, seperti terlihat pada bagan alir awal bagian kedua, dilakukan dengan menguji string masukan terhadap standar dan menerjemahkannya ke dalam bentuk yang dapat diproses Squidguard. Squidguard juga memiliki fitur keamanan di dalam kodenya, seperti menolak berbagai bentuk URL yang tidak sesuai dengan standar atau yang dicurigai mencoba meretas keamanan; salah satunya *HTTP double slash vulnerability* yang dijelaskan dalam Debian Bug report logs #516829.

Jika penerjemahan berjalan dengan lancar, Squidguard akan menggunakan informasi yang telah didapat dalam penerjemahan untuk menentukan pengalihan akses laman pada iterasi selanjutnya.




Gambar 10. Bagan alir Squidguard awal (bagian 3)

Pada bagan alir awal bagian ketiga terlihat bagaimana Squidguard menentukan pengalihan URL masukan. Setelah menentukan kelompok asal permintaan, Squidguard akan mencoba mencari ACL yang bersesuaian dengan kelompok asal permintaan. Kedua hasil pencarian ini sangat tergantung pada konfigurasi. Pada Subbab 2.1.2 juga terlihat, jika Squidguard tidak menemukan kelompok asal permintaan, Squidguard secara otomatis memberi kelompok `none`

kepada masukan. Juga terlihat pada Subbab 2.1.2, Squidguard akan menggunakan ACL `default` jika tidak ada ACL yang bersesuaian dengan kelompok asal permintaan.

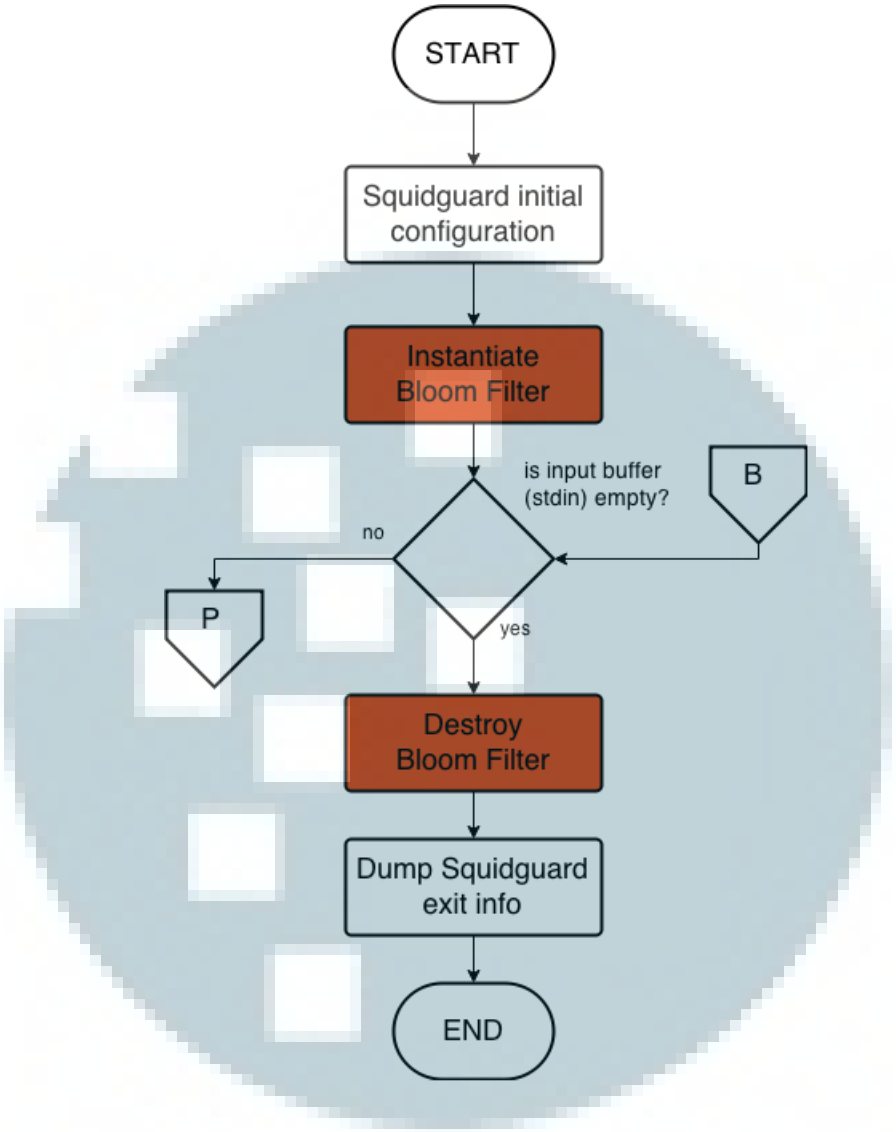
Setelah kelompok asal permintaan dan ACL ditentukan, Squidguard akan menghitung pengalihan akses laman. Jika pengalihan perlu dilakukan, Squidguard akan menuliskan URL alihan, berupa untai karakter (*string of characters*), ke dalam petunjuk pengalihan (*redirection directive*). Petunjuk pengalihan akan bernilai NULL selama penghitungan berjalan atau jika Squidguard menyimpulkan bahwa pengalihan tidak perlu dilakukan.

Selanjutnya, jika petunjuk pengalihan masih bernilai NULL dan Squidguard menganggap pembandingan kelompok asal permintaan dan ACL masih perlu dilakukan, Squidguard akan mengulang kedua proses tersebut sampai perhitungan petunjuk pengalihan tidak menghasilkan NULL atau sampai tidak ada lagi kelompok asal permintaan dan ACL untuk dibandingkan.

Kemungkinan lainnya, informasi pengalihan akan langsung dikeluarkan jika petunjuk pengalihan sudah tidak menunjukkan nilai NULL. Dalam kasus ini, informasi pengalihan akan berisi petunjuk pengalihan itu sendiri. Jika sampai akhir iterasi, petunjuk pengalihan masih menunjukkan nilai NULL, informasi pengalihan akan tetap dikeluarkan, tetapi dalam bentuk baris kosong (*empty line*).

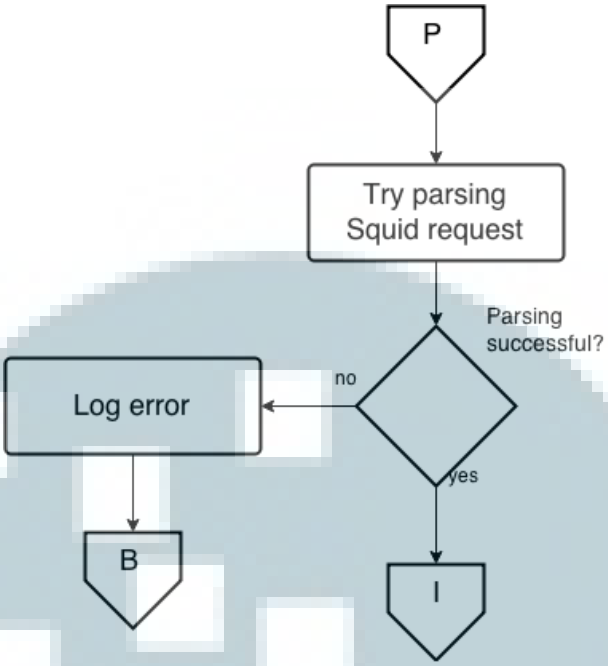
### 3.1.2 Rancangan Modifikasi Squidguard

Untuk melakukan pengalihan yang lebih selektif berdasarkan alamat IP yang telah melakukan akses laman, Bloom filter perlu disisipkan ke dalam rutin Squidguard. Rancangan penyisipan ini terlihat pada Gambar 11, Gambar 12, dan Gambar 13.




Gambar 11. Bagan alir Squidguard termodifikasi (bagian 1)

Pada bagan alir termodifikasi bagian pertama, terlihat bahwa ke dalam rutin Squidguard ditambahkan modul konfigurasi awal Bloom filter dan modul untuk menghapus Bloom filter dari memori. Bloom filter akan disusun setelah Squidguard selesai melakukan konfigurasi awal dan penghapusan Bloom filter dari memori dilakukan sebelum Squidguard menyelesaikan rutinnya. Pengujian keberadaan masukan dalam penyangga tetap bekerja seperti sebelumnya karena terletak di antara dan independen terhadap kedua modul yang ditambahkan.

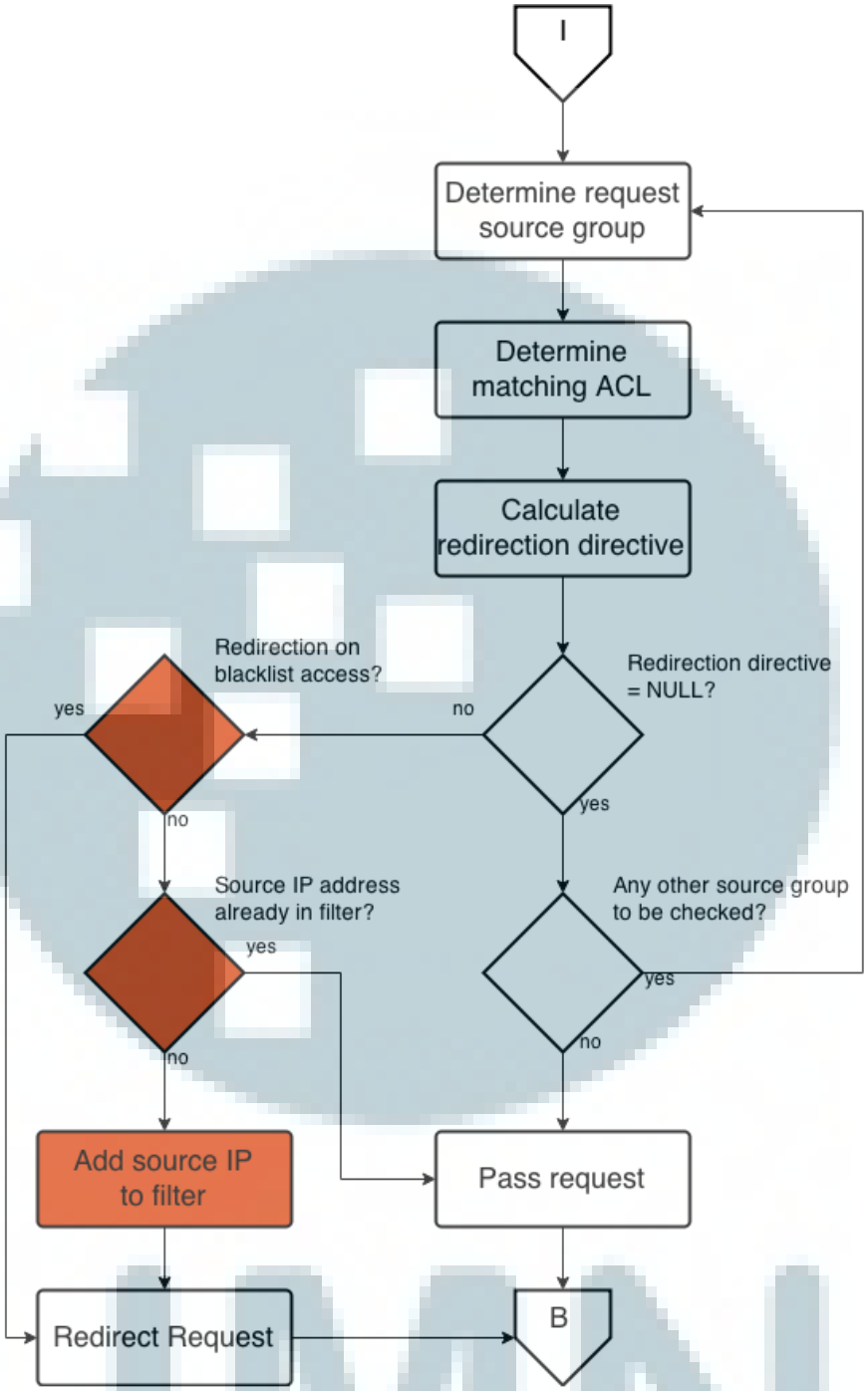



Gambar 12. Bagan alir Squidguard termodifikasi (bagian 2)

Pada bagan alir termodifikasi bagian kedua, tidak ada modul yang perlu ditambahkan. Fungsi keamanan juga tetap dijalankan seperti sebelumnya.

Hal pertama yang perlu diperhatikan pada bagan alir termodifikasi bagian ketiga adalah modul *Redirecting request* dan *Pass request*. Keduanya adalah hasil pecahan dari modul *Dump redirection info* pada bagan alir awal Squidguard bagian ketiga (Gambar 10). Pasangan modul *Redirecting request* dan *Pass request* berada pada posisi logis yang sama dengan *Dump redirection info*: sebelum program kembali menguji penyangga masukan (tautan *B* dalam bagan alir). Pemisahan ini dilakukan untuk lebih jelas membedakan kapan Squidguard meloloskan dan mengalihkan permintaan.

Secara logis, laman tujuan dibagi ke dalam dua bagian besar: laman yang tidak termasuk dalam *blacklist* dan laman yang termasuk dalam *blacklist*. Akses pertama menuju laman-laman yang tidak termasuk dalam *blacklist* dialihkan. Sedangkan laman yang termasuk dalam *blacklist* akan diblokir tanpa kecuali.




Gambar 13. Bagan alir Squidguard termodifikasi (bagian 3)

Setelah petunjuk pengalihan didapat (tidak NULL), program akan menguji apakah pengalihan dilakukan karena laman tujuan terbentur blacklist. Jika ya, berarti program telah memberi pengalihan blokir kepada URL tersebut dan program langsung mengeluarkan perintah pengalihan tersebut.

Jika pengalihan tidak disebabkan oleh *blacklist*, program akan kembali mempertimbangkan untuk (tetap) mengalihkan permintaan ini, tetapi kali in dengan syarat yang berbeda. Jika alamat IP sumber permintaan belum tercatat di dalam Bloom filter, pengalihan akan dilakukan setelah pencatatan alamat IP dilakukan. Dengan demikian, saat ada permintaan datang dari alamat IP yang sama untuk kali berikutnya, program akan mendeteksi bahwa alamat IP tersebut sudah terdapat di dalam Bloom filter dan akan meloloskan permintaan tersebut tanpa pengalihan (dengan catatan permintaan ini tidak terbentur *blacklist*).

Tentunya, dengan menggunakan Bloom filter, alamat IP sumber permintaan tidak serta merta disimpan ke dalam struktur data. Hanya hash dari alamat IP bersangkutan yang disimpan ke dalam Bloom filter. Mekanisme Bloom filter dapat dilihat kembali pada Subbab 2.2.

## 3.2 Rancangan Bloom Filter

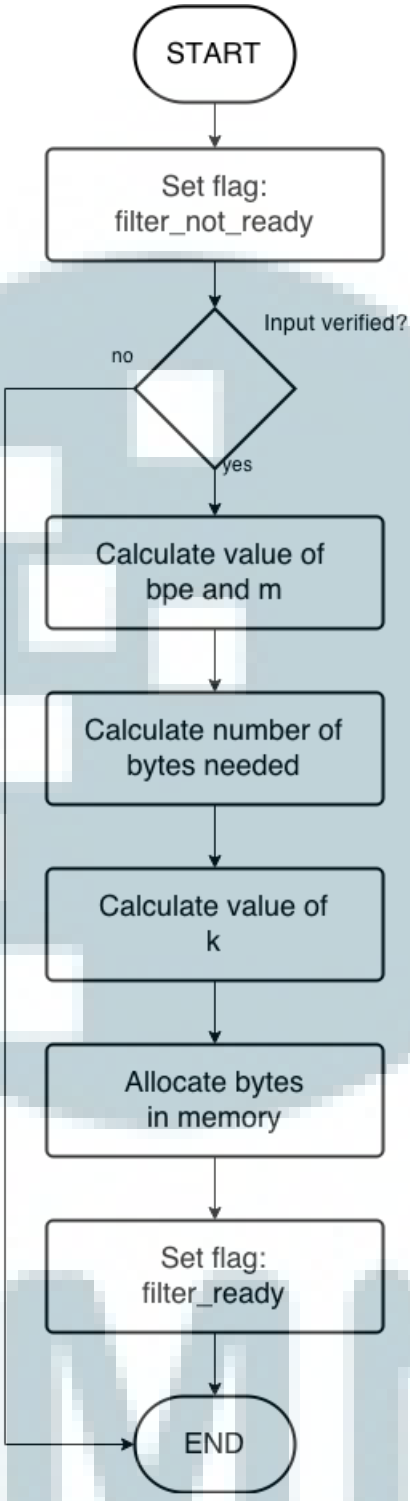
Secara tersirat, Subbab 2.2 telah menguraikan rancangan Bloom filter yang diperlukan oleh Squidguard. Bloom filter yang akan digunakan harus memunyai:

a. prosedur inisialisasi untuk menghitung jumlah bit yang diperlukan untuk Bloom filter serta jumlah fungsi hash berdasarkan banyak elemen yang diharapkan dapat ditampung Bloom filter serta batas atas galat isbat yang diinginkan,
b. prosedur untuk menguji keberadaan elemen dalam Bloom filter,
c. prosedur untuk menambahkan elemen ke dalam Bloom filter, dan
d. prosedur untuk menghapus Bloom filter dari memori.

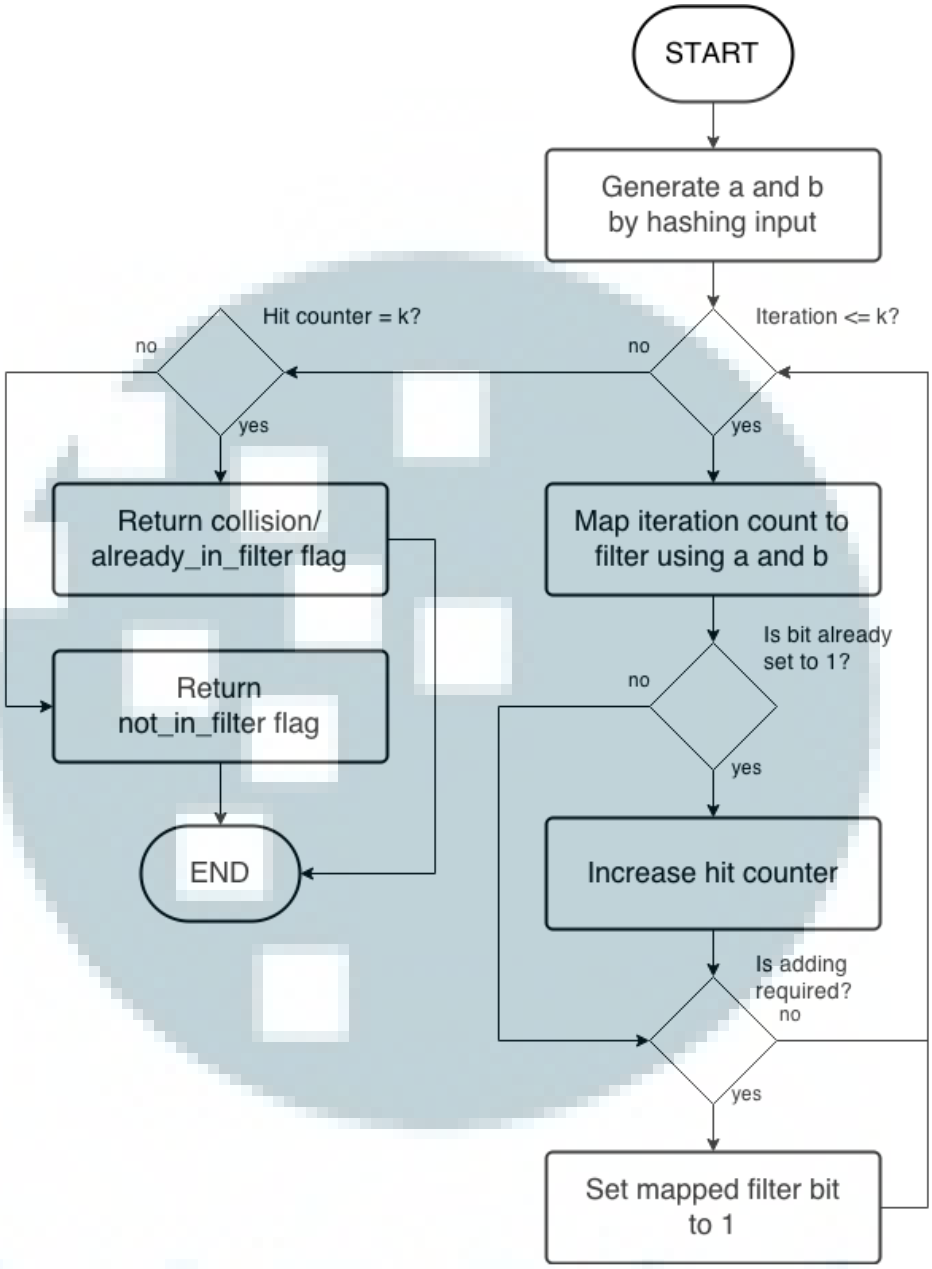
Jyri J. Virkki mengembangkan sebuah implementasi Bloom filter dalam bahasa C di bawah lisensi BSD (Virkki, 2012). Bloom filter yang dikembangkan Virkki menambahkan prosedur untuk mencetak informasi Bloom filter, seperti jumlah elemen maksimum, nilai galat isbat, jumlah bit dan bita yang digunakan oleh Bloom filter, serta jumlah fungsi hash yang diutilisasi oleh sistem. Bloom filter hasil kembangan Virkki digunakan karena kesederhanaan implementasinya, sehingga lebih sedikit elemen yang perlu dipertimbangkan dalam pengujian. Implementasi yang sederhana juga meningkatkan kecepatan proses.

Pada inisialisasi Bloom filter, program akan menghitung karakteristik-karakteristik Bloom filter sesuai masukan yang diberikan: batas atas galat isbat dan jumlah maksimum elemen yang dapat ditampung. Penghitungan dilakukan sesuai dengan persamaan-persamaan yang telah dijelaskan pada Subbab 2.2. Prosedur inisialisasi juga menempatkan sebuah variabel tanda yang mengindikasikan bahwa Bloom filter siap digunakan. Bagan alir prosedur inisialisai Bloom filter dapat dilihat pada Gambar 14.

Prosedur untuk menguji dan menambahkan elemen ke dalam Bloom filter digabungkan oleh Virkki. Alir untuk menambahkan elemen ke dalam Bloom filter hanya menambahkan satu instruksi ke dalam alir uji elemen, yaitu instruksi untuk merubah nilai bit yang bersangkutan pada Bloom filter. Bagan alir prosedur uji dan tambah elemen dapat dilihat pada Gambar 15.




Gambar 14. Bagan alir prosedur inisialisasi Bloom filter




Gambar 15. Bagan alir prosedur uji elemen dan tambah elemen ke dalam Bloom filter

Prosedur uji dan tambah elemen akan menggunakan metode hash pada input untuk mendapatkan dua buah angka yang akan digunakan dalam pemetaan bit. Posisi pemetaan dihitung sebagai berikut.

$$x = (a + i \cdot b) \bmod m \qquad (7)$$

Variabel-variabel pada persamaan di atas dijelaskan sebagai berikut.

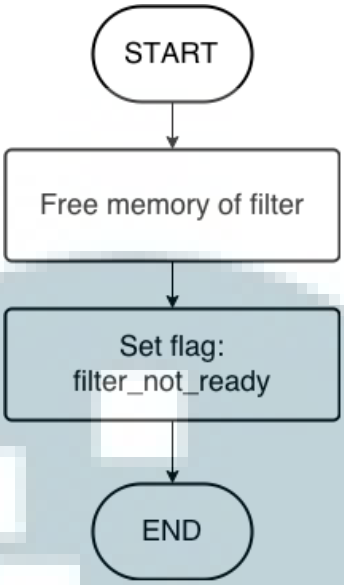
a. $i$ menunjukkan nomor putaran pengulangan (pengulangan dilakukan sebanyak jumlah pemetaan),

b. $a$ dan $b$ menunjukkan angka hasil hash,
c. $m$ menunjukkan jumlah bit memori dalam filter, dan
d. $x$ menunjukkan posisi bit dalam Bloom filter.

Selama iterasi berlangsung, program akan menghitung banyak bit, pada posisi yang ditunjukkan oleh $x$, yang telah bernilai 1. Dalam iterasi, jika prosedur digunakan untuk menambahkan elemen ke dalam Bloom filter, bit pada posisi $x$ akan ditimpa nilainya dengan 1. Jika pada akhir iterasi, jumlah bit yang telah bernilai 1 sama dengan jumlah pemetaan yang dilakukan, program menyimpulkan bahwa elemen yang sedang diuji benar ada di dalam Bloom filter atau elemen yang sedang coba ditambahkan mungkin telah ada dalam Bloom filter. Jika jumlah bit yang telah bernilai 1 kurang dari jumlah pemetaan yang dilakukan, program menyimpulkan bahwa elemen yang sedang diuji tidak terdapat dalam sistem atau elemen yang sedang coba ditambahkan belum terdapat sebelumnya dalam Bloom filter. Hal ini sesuai dengan teori Bloom filter yang dijabarkan pada Subbab 2.2.

Prosedur untuk mencetak informasi Bloom filter hanya membaca nilai-nilai variabel yang tertera dalam struktur data dan menampilkannya. Variabel-variabel tersebut adalah

a. jumlah maksimum elemen yang dapat ditampung,
b. batas atas nilai galat isbat,
c. jumlah bit yang digunakan Bloom filter secara keseluruhan,
d. jumlah bit yang digunakan Bloom filter per elemen,
e. jumlah bita yang digunakan oleh Bloom filter secara keseluruhan, dan
f. jumlah pemetaan (fungsi hash) yang digunakan Bloom filter.




Gambar 16. Bagan alir prosedur untuk menghapus filter

Prosedur untuk menghapus Bloom filter dari dalam memori membebaskan area memori yang berhubungan dengan Bloom filter, kemudian memberi tanda variabel bahwa Bloom filter tersebut menjadi belum siap dipakai. Bagan alir prosedur untuk menghapus Bloom filter dapat dilihat pada Gambar 16.