Nuvens proprietárias, "cativas" e livres: Uma breve análise da relação entre software livre e computação em nuvem

Miguel Said Vieira
UFABC
miguel.vieira@ufabc.edu.br

Abstract: Este artigo analisa a relação entre software livre e computação em nuvem, da perspectiva da segurança e da privacidade dos usuários finais. Identifica as limitações da abordagem de software livre no contexto da computação em nuvem, e descreve estratégia proposta para contorná-las — a da licença AGPL —, incluindo suas limitações.

1. Introdução: segurança e software livre[1]

Nas literatura sobre segurança de software, um tema bastante discutido é se FLOSS (*Free/Libre Open Source Software*, software livre ou de código aberto) tende a ser mais seguro que software proprietário. Embora o debate prossiga, especialistas como Bruce Schneier [1] têm defendido fortemente as vantagens de segurança do FLOSS. O argumento de Schneier é razoavelmente simples: para evitar problemas de segurança, não é suficiente avaliar sua funcionalidade (por meio de *beta-testing*, por exemplo); é necessário avaliar o seu código. E a melhor maneira de *possibilitar* que o código seja avaliado continuamente e por múltiplos experts é publicando o código e permitindo seu estudo, como ocorre com o FLOSS. Schneier qualifica essa afirmação, porém: a abertura do código facilita que ele seja mais avaliado, mas não garante isso; em teoria, ainda é possível que certos softwares proprietários sejam mais seguros (o que dependerá de seus produtores investirem ativamente na auditoria interna de seus códigos por especialistas).

Em um estudo empírico, Payne [2] confirma essa afirmação, bem como seu caráter limitado. Comparando indicadores de segurança entre três sistemas operacionais (um proprietário — Solaris — e dois FLOSS — Debian e OpenBSD), ele detecta que ambos os SOs livres são mais seguros que o proprietário, mas que há uma diferença muito significativa entre o grau de segurança de ambos: o vencedor da comparação é OpenBSD, por larga margem, ao passo que o segundo colocado, Debian, tem uma vantagem bem menor sobre o único SO proprietário: Solaris. A explicação seria que, além de ser FLOSS, OpenBSD é um projeto que valoriza particularmente a questão da segurança, e conta com uma comunidade bastante especializada no tema.

2. Privacidade

Essa comparação considera, entretanto, que produtores de software proprietário e FLOSS estarão igualmente interessados em garantir a segurança do software para usuários finais. Embora essa premissa seja razoável, hoje ela não se sustenta plenamente ao considerarmos os riscos relativos à privacidade. Essa é uma questão candente quando levamos em conta que os modelos de negócio baseados no uso de dados pessoais tem se mostrado extremamente rentáveis; Google, Apple, Facebook, Amazon e Microsoft (que Smyrnaios agrupa no acrônimo "GAFAM") [3] são exemplos de companhias extremamente bem sucedidas e cujo negócio depende cada vez mais da acumulação de dados pessoais dos usuários de seus produtos (hardwares, softwares ou serviços web). Em casos como esses, o produtor de software está ativamente interessado em fazer uso dos dados pessoais do usuário final, eventualmente violando sua privacidade.

Assim, se FLOSS oferecia uma vantagem limitada quanto a segurança, no caso específico da privacidade essa vantagem é muito mais significativa. Para usuários finais, quando um software coleta dados pessoais, ele envolve riscos de privacidade: é do interesse dos usuários, assim, eliminar ou controlar essa coleta de dados; no entanto, para outros atores — como as empresas "GAFAM" mencionadas acima — essa coleta pode ser uma funcionalidade desejada, e na prática a razão central da existência daquele software. Nesses casos, não há como se esperar que auditorias

internas mitiguem adequadamente os riscos à privacidade, e a abertura do código (que permitirá sua avaliação por atores independentes) passa a ser uma condição *sine qua non* para a ampliação da segurança de um software.

3. Computação em nuvem

A ascensão do modelo de negócio dos "GAFAM" implica logicamente o aumento da relevância dos riscos à privacidade; dessa perspectiva, o uso de FLOSS tem se tornado cada vez mais importante para a segurança. Essa tendência a favor de FLOSS guarda, no entanto, um caráter contraditório.

Para compreendê-lo, primeiramente é preciso considerar que a ascensão dos "GAFAM" está diretamente associada à ascensão da computação em nuvem. O uso de "serviços computacionais" ofertados por grandes empresas, e que dependem de processamento em servidores que não são controlados pelos usuários finais, amplia em muito a possibilidade de coleta de dados pessoais.

A contradição dessa tendência reside no fato de que os FLOSS, quando usados no contexto da computação em nuvem, têm vários dos seus possíveis benefícios (inclusive a vantagem em relação à segurança) impactados negativamente. A razão disso é que FLOSS fundamenta-se na valorização da autonomia do "usuário",[2] por meio da permissão de livre uso, estudo, modificação e compartilhamento do software; e que toda essa autonomia, no entanto, é drasticamente limitada quando o software em questão roda em um servidor que não é controlado pelo indivíduo, e sim por uma empresa provedora de nuvem.

Analisemos em termos práticos esse impacto sobre a vantagem de FLOSS no quesito segurança. Ora, ainda que determinado FLOSS tenha código amplamente auditado, no momento em que ele roda "na nuvem", é muito mais difícil que o usuário final assegure-se de que é efetivamente aquele código que está sendo executado; como garantir que a empresa provedora da nuvem não modificou o FLOSS em questão, acrescentando funcionalidades voltadas à coleta de dados pessoais?

Outro grande benefício potencial de FLOSS é a aceleração da melhoria do software, graças à colaboração em rede possibilitada pelo caráter "viral" das licenças *copyleft* (o caso do avanço vertiginoso do núcleo Linux, licenciado sob GPL, é o exemplo paradigmático); e esse benefício também sofre impacto negativo com a transição para a computação em nuvem. O motivo é que, nas licenças mais usadas para FLOSS, a obrigatoriedade de se compartilhar o código de um FLOSS só se manifesta quando esse software é *distribuído*. Se executo um FLOSS (ou uma versão melhorada que produzi dele) em meu computador pessoal... ou em um *servidor*, não ocorre distribuição. Com isso, todo provedor de nuvem pode fazer uso do *pool* comum de FLOSS, sem ser obrigado a contribuir de volta para ele, e nem mesmo a garantir para o usuário final as liberdades oferecidas pela licença daquele software — ao contrário do que ocorria em paradigmas anteriores, com o uso do software em dispositivos locais.

4. AGPL: solução parcial

Uma das tentativas para enfrentar esses impactos negativos da computação em nuvem sobre FLOSS foi a construção de uma licença nova (a Affero GPL, ou AGPL), que não limitasse as exigências de reciprocidade à ocorrência da distribuição. A AGPL exige que seja oferecido o código do software não só quando ele é distribuído, mas também quando ele é executado remotamente, em um servidor.

De certa forma, isso implica a permanência dos benefícios de FLOSS discutidos na seção anterior, mesmo quando o software é usado na computação em nuvem. Como é oferecido o acesso ao código, é possível auditá-lo, e dessa maneira volta a manifestar-se a vantagem quanto à segurança; e a possibilidade da colaboração em rede retorna pela exigência de que toda melhoria seja compartilhada (mesmo sem ter havido "distribuição").

Embora a proposta seja engenhosa e bem intencionada, a solução que ela oferece ainda é parcial — pois uma vez que o software roda numa máquina que não é controlada pelo usuário final, ele continua sem ter como garantir de forma absoluta que a licença está sendo respeitada. Ao rodar um software sob AGPL em um provedor de nuvem, como o usuário pode confirmar que o código disponibilizado é de fato o que está sendo executado? No limite, não há nada que garanta que não foram acrescentadas funcionalidades de coleta de dados pessoais ao código que efetivamente é executado no servidor (ainda que elas não apareçam no código que é *disponibilizado* ali).

5. Considerações finais

Se a abordagem de software livre tem vantagens limitadas no que diz respeito à segurança, no caso específico da privacidade essas vantagens são mais amplas — e as violações de privacidade são um risco de segurança cada vez

[2] A razão das aspas é que, no contexto de FLOSS, a fronteira entre "usuário" e "produtor" é borrada: dadas as liberdades garantidas pelas licenças livres, todo usuário de um FLOSS a princípio tem a capacidade de tornar-se seu co-produtor.

mais significativo no contexto atual. Paradoxalmente, o cenário atual também envolve um uso cada vez mais intenso da computação em nuvem, que limita os potenciais benefícios do software livre (uma vez que, como não ocorre "distribuição" do software, as liberdades garantidas pelas licenças mais comuns de FLOSS não são obrigatórias).

A licença AGPL propõe uma evolução do conceito de FLOSS que tenta resolver esse paradoxo; mas embora ela sirva como um sinalizador de intenções (indicando que a comunidade em torno daquele software deseja que suas liberdades sejam garantidas mesmo quando usado na computação em nuvem), ela é limitada em relação a mecanismos de observância. Isto é: ainda que, em teoria, a AGPL possa dissuadir violadores pelo risco de serem processados (com base nos termos da licença), o fato é que é muito difícil provar que houve violação quando não se controla o servidor em que um software AGPL é executado, e portanto garantir que a licença seja respeitada.

Em função disso, embora a AGPL seja um avanço positivo, os FLOSS só manifestam plenamente seus benefícios quando os usuários têm controle sobre os dispositivos em que eles são executados — só aí poderíamos falar de uma nuvem "livre". Uma nuvem que roda software livre mas não é controlada por seus usuários é uma espécie de nuvem "cativa": não há como se assegurar de forma absoluta que o software ali é de fato livre. Mas mesmo nesse caso, porém, a situação sempre será igual ou melhor que aquela vivida pelos usuários de nuvens "proprietárias", em que o usuário fica plenamente à mercê de seu provedor.

6. Referências

[1] SCHNEIER, B. *Crypto-Gram: September 15, 1999*. Disponível em: <https://www.schneier.com/crypto-gram/archives/1999/0915.htmlOpenSourceandSecurity>.

[2] PAYNE, C. On the security of open source software. v. 12, n. 1, p. 61–78. ISSN 1365-2575. Disponível em: <http://onlinelibrary.wiley.com/doi/10.1046/j.1365-2575.2002.00118.x/full>.

[3] SMYRNAIOS, N. L'effet GAFAM : stratégies et logiques de l'oligopole de l'internet. v. 2016, n. 188, p. 61–83. ISSN 0336-1500. Disponível em: <http://www.necplus.eu/abstract$_S$0336150016012047>.