Comment peut-on tolérer les Intrusions sur Internet ?

Yves Deswarte

LAAS/CNRS
7 Avenue du Colonel Roche
31077 Toulouse Cedex 4, France
Yves.Deswarte@laas.fr

Résumé Alors que le réseau mondial est devenu indispensable à bon nombre d'entreprises comme de particuliers, les systèmes informatiques qui sont connectés à l'Internet sont encore trop vulnérables et les attaques sont de plus en plus nombreuses. Face à cette évolution, les techniques classiques de sécurité sont insuffisantes, et il est de plus en plus intéressant d'appliquer des techniques de tolérance aux fautes. Cependant, les intrusions sont des fautes très particulières, et il faut en tenir compte dans le choix des techniques de tolérance. Dans cet article, nous montrons comment ces techniques peuvent améliorer la sécurité sur Internet.

1 Internet est vulnérable

ARPANET, le réseau expérimental qui devait donner naissance à l'Internet, a été créé par un petit nombre de chercheurs et d'informaticiens qui souhaitaient mieux communiquer entre eux et partager des ressources fragiles, rares et chères, comme les lignes d'interconnexion entre ordinateurs et les capacités de traitement et de stockage de l'information. Les protocoles et services qu'ils ont développés pour cela visaient donc en tout premier lieu à fournir la meilleure disponibilité possible, malgré des composants relativement peu fiables (lignes, routeurs, ordinateurs). Considérer la possibilité de malveillance n'était pas à l'ordre du jour, puisque seuls avaient accès au réseau un petit groupe de pionniers qui formaient une communauté soudée dans un même objectif : faire fonctionner le réseau. Aucun d'entre eux n'aurait eu l'idée saugrenue d'expérimenter des attaques, qui auraient mis en danger cet objectif commun, conduisant très probablement à son exclusion du groupe. Il est donc naturel que les protocoles et services ainsi développés ne prennent en compte que les fautes accidentelles de transmission, de matériel et de logiciel, sans s'occuper des malveillances. Par exemple, dans ces protocoles, rien ne garantit l'authenticité des adresses, ce qui facilite des attaques comme la falsification d'adresses (*spoofing* en anglais) ou le détournement de session (*hijacking*). De même, ces protocoles intègrent des fonctionnalités de maintenance du réseau, comme le routage à la source, qui peuvent être facilement exploitées pour contourner des mécanismes de protection comme les pare-feux.

Aujourd'hui, ces mêmes protocoles gouvernent l'Internet, qui est devenu un système totalement ouvert, c'est-à-dire auquel n'importe qui peut avoir accès.

Bien loin de la petite communauté d'origine, de très nombreuses catégories d'utilisateurs interagissent aujourd'hui par l'Internet : entre entreprises (*business-to-business*, ou B2B), entre marchand et consommateur (*business-to-consumer*, ou B2C), entre les entreprises et les administrations (B2A), entre les citoyens et les administations (C2A) ou le gouvernement (*e-Gouvernement*), ou simplement entre individus, qui se créent leurs propres communautés virtuelles. Les utilisations sont très diverses : commerciales, administratives, démocratiques, sociales, culturelles, ou simplement ludiques. Et toutes ces utilisations sont a priori légitimes : sous le prétexte qu'une catégorie d'utilisateurs pourrait nuire à ses objectifs, aucun groupe n'est en droit de l'exclure, ni n'y a intérêt car c'est cette variété d'utilisation qui permet à l'Internet d'exister et de se développer à un coût raisonnable. Puisque les objectifs sont différents, il est normal que les besoins de sécurité soient différents, et il ne faut pas s'attendre à ce qu'un lycéen gère son ordinateur personnel avec le même soin que ceux qui administrent les serveurs de grandes banques, par exemple. Les lycéens étant plus nombreux que les banques, il n'est pas étonnant que de nombreux systèmes connectés à l'Internet soient mal administrés, et soient donc vulnérables aux attaques de personnes mal intentionnées, qui peuvent dès lors en prendre le contrôle pour propager d'autres attaques. Il est ainsi possible à ces attaquants de multiplier leur puissance pour perpétrer des attaques ciblées contre des sites bien protégés, mais aussi d'effacer leurs traces et de rendre plus difficile la collecte d'indices permettant de les identifier. Les attaques sont nombreuses, et il ne faut pas s'en étonner : les utilisateurs d'Internet sont plus ou moins représentatifs de la population des pays développés, et parmi les centaines de millions d'internautes, il doit exister une proportion non négligeable d'attaquants potentiels. Les types d'attaques les plus fréquents, et aussi les plus faciles, visent à mettre en défaut la disponibilité, par "déni de service" : l'attaquant cherche simplement à bloquer le fonctionnement du système qu'il a choisi comme cible. D'autres visent la confidentialité : l'attaquant cherche à obtenir des informations plus ou moins sensibles comme des secrets commerciaux, industriels, politiques, voire militaires, mais aussi des informations personnelles, mettant ainsi en danger la vie privée de particuliers. Enfin, d'autres types d'attaques visent l'intégrité des informations : destruction ou modification de données sensibles, diffusion de fausses informations, altération d'informations publiées, etc. Ainsi, l'un des sports à la mode parmi les pirates consiste à "défigurer" les serveurs Web, c'est-à-dire à remplacer (ou à détourner l'accès à) des pages légitimes par des parodies humoristiques, polémiques, ou pornographiques.

Les motivations des attaquants sont variées. Ils peuvent agir par jeu, par curiosité (pour faire des expériences), par vanité (pour montrer leur compétence), par vandalisme (pour le plaisir de détruire ou de provoquer des dégâts), par esprit de vengeance (pour nuire à ceux qui ne leur plaisent pas ou pour punir ceux qui ne les considèrent pas à leur "juste" valeur), par appât du gain (chantage, extorsion de fonds), voire pour des motifs politiques, stratégiques ou terroristes. Les attaquants sont donc très divers en ténacité, en compétence et en moyens qu'ils peuvent mettre en oeuvre, selon qu'il s'agit d'adolescents

perturbés, de groupes de pirates plus ou moins structurés, de simples escrocs, d'organisations criminelles, de groupes terroristes, ou de services spécialisés dans la guerre électronique.

Les moyens utilisés pour les attaques sont, eux aussi, nombreux. Ils peuvent exploiter les vulnérabilités des réseaux et de leurs protocoles : écoute passive (*sniffing* en anglais), interception (destruction, insertion, modification ou rejeu de messages), falsification d'adresse, injection de faux messages de contrôle du réseau (de routage, par exemple), déni de service ("éblouissement" de réseau, par exemple). Il est à noter que les routeurs sont de plus en plus souvent la cible d'attaques. Les attaques peuvent aussi exploiter les failles des systèmes d'exploitation et des logiciels d'application, comme les débordements de tampons ou de pile. Il faut aussi noter que l'Internet est aussi un moyen de propagation de l'information sur les failles de sécurité, aussi bien pour les pirates que pour les administrateurs et développeurs de systèmes. Les premiers utilisent cette information pour développer et publieur des "exploits" (c'est-à-dire des méthodes pour réussir des attaques), jusqu'à des scripts automatisant les attaques, les mettant ainsi à la portée des débutants. Les autres utilisent cette même information pour développer et publier des parades. Comme la langue d'Ésope, la circulation de telles informations sur l'Internet peut donc être la meilleure et la pire des choses.

2 Les techniques classiques de sécurité sont insuffisantes

La sécurité informatique repose principalement sur l'authentification des utilisateurs et sur l'autorisation, c'est-à-dire le contrôle des droits d'accès. L'authentification est indispensable pour identifier (avec un degré de confiance suffisant) chaque utilisateur, de façon à lui attribuer les droits qui lui conviennent et aussi de façon à le rendre responsable de ses actions. L'autorisation consiste à ne permettre à l'utilisateur que de réaliser des actions légitimes. L'autorisation devrait, dans la mesure du possible, obéir au principe du moindre privilège : à un moment donné, un utilisateur ne doit pouvoir exécuter que les seules opérations nécessaires à l'exécution de sa tâche légitime. L'autorisation est mise en oeuvre par des mécanismes de protection qui permettent de détecter et de bloquer une tentative, volontaire ou involontaire, par un utilisateur d'outrepasser ses privilèges. Cette tentative peut être alors analysée par les responsables de la sécurité pour permettre éventuellement de punir le coupable, ce qui contribuera à la dissuasion d'autres tentatives. Authentification, autorisation, détection, répression et dissuasion constituent donc l'arsenal des défenseurs de la sécurité.

Malheureusement, cet arsenal est peu efficace dans la plupart des systèmes connectés à l'Internet. En effet, tout internaute, même anonyme, a des droits sur toute machine connectée : par exemple, lire les pages des serveurs Web publics, mais aussi connaître l'existence de cette machine et l'identifier par son nom ou son adresse. L'authentification des utilisateurs, lorsqu'elle existe, repose généralement sur des mécanismes faibles, comme un mot de passe qu'il est facile pour un utilisateur de transmettre à un autre, lui permettant a insi de se faire

passer pour l'utilisateur légitime ; d'ailleurs, même sans la coopération de l'utilisateur, il est souvent facile de deviner un mot de passe. Un autre problème lié à l'authentification sur l'Internet tient à sa distribution géographique qui fait que les administrateurs de systèmes n'ont généralement pas la possibilité de rencontrer physiquement leurs utilisateurs. Les utilisateurs ne sont donc identifiés que de façon indirecte, à travers une connexion virtuelle.

Il existe bien sûr une tendance à vouloir renforcer l'authentification de tous les internautes par des mécanismes forts, à la charge des fournisseurs d'accès à l'Internet. Mais ceci se ferait au détriment de la protection de la vie privée, qui fait partie des droits fondamentaux de nos démocraties. Ainsi, selon des lois françaises récentes [1], les opérateurs (en particulier les fournisseurs d'accès à l'Internet) peuvent être tenus d'enregistrer "certaines catégories de données techniques" liées à toute communication et de conserver ces informations pour une durée ne pouvant excéder un an. Même si la loi précise que ces informations ne doivent pas porter sur le contenu des communications, mais seulement sur les données techniques, et même si la définition précise de ce que sont ces données techniques ne sera établie que par un décret non encore paru, on peut deviner qu'elles devront contenir l'identité de celui qui établit la connexion, le type de connexion, les adresses des deux extrémités de la connexion, ainsi que la date, l'heure et la durée de la connexion. En fait, il est impossible de distinguer "contenu" et "données techniques". Par exemple, si Monsieur Smith se connecte régulièrement sur un serveur Web ayant l'adresse Internet 12.96.163.40, on pourra déduire que cette personne a probablement des problèmes d'alcoolisme, puisqu'il s'agit du serveur de l'association des alcooliques anonymes aux États-Unis. Cet exemple montre qu'il existe un risque que le renforcement de la sécurité sur l'Internet se fasse au détriment de notre vie privée.

L'arsenal classique de la sécurité est également peu efficace pour une autre raison : comme on l'a vu, la plupart des attaques sur l'Internet exploitent les vulnérabilités des réseaux ou des logiciels, et contournent ainsi les mécanismes d'authentification et d'autorisation. Il est donc nécessaire d'implémenter les parades spécifiques à chaque type d'attaque, développées à la suite d'attaques réussies de systèmes similaires, ou simplement à la suite de la découverte de nouvelles vulnérabilités. En particulier, les fournisseurs de logiciels sont maintenant fortement sensibilisés aux problèmes de sécurité, et aux risques qu'ils encourent pour leur image de marque. Ils essaient donc de fournir dès que possible des "rustines" (*patches* en anglais) corrigeant les vulnérabilités identifiées. Néanmoins, il n'est pas si facile d'appliquer des rustines : cela requiert du temps et de la compétence de la part des administrateurs, qui sont une ressource rare et chère. D'abord, les annonces de nouvelles vulnérabilités sont nombreuses, et il est parfois délicat d'identifier si un système donné présente ces vulnérabilités et nécessite qu'on applique les rustines. D'autre part, l'application de rustines modifie parfois les fonctionnalités des systèmes, empêchant ainsi le fonctionnement de certaines applications qui peuvent être vitales pour les entreprises. Enfin, comme cela a déjà été mentionné, de nombreux systèmes sont mal administrés, et ne sont que rarement mis à jour. Certains fournisseurs prévoient une mise

à jour automatique, grâce à une connexion Internet, mais cette automatisation elle-même n'est pas exempte de risque.

Il vaut mieux, dit-on, prévenir que guérir, et il faudrait donc éviter d'introduire des failles ou des vulnérabilités dans la conception, la réalisation ou l'exploitation des systèmes. Malheureusement, l'objectif "zéro défaut" est encore loin d'être atteint en matière d'informatique. Les systèmes actuels sont trop complexes pour être totalement maîtrisables, et leur sécurité n'est pas toujours un objectif primordial. Ainsi un constructeur pourra considérer plus rentable de mettre rapidement sur le marché un système imparfait plutôt que de lui faire subir les méthodes de développement et de validation nécessaires pour obtenir un bon niveau de sécurité, comme ceux qui sont définis dans les "critères communs" [2]. D'ailleurs, même les systèmes les plus sûrs ne résistent pas longtemps à des attaques menées par des professionnels compétents. Il faut donc explorer d'autres voies.

3 La tolérance aux intrusions

3.1 La tolérance aux fautes

La tolérance aux fautes [3] est une technique qui a fait ses preuves pour réaliser des systèmes informatiques capables de fournir un service correct, même en présence de phénomènes accidentels, comme des perturbations de l'environnement (fautes externes), des défaillances de composants matériels (fautes physiques internes), voire des fautes de conception, en particulier logicielles (bogues).

Selon la terminologie de la sûreté de fonctionnement [3], les fautes sont les causes des erreurs, les erreurs sont des parties anormales de l'état du système informatique, et lorsque les erreurs se propagent jusqu'à l'interface du système, c'est-à-dire lorsque le service fourni par le système est incorrect, il y a défaillance. Lorsque les fautes sont accidentelles, et suffisamment rares, il est possible de les tolérer. Il faut pour cela détecter les erreurs avant qu'elles ne produisent de défaillance et les corriger : c'est le traitement des erreurs. Il faut aussi effectuer un diagnostic, c'est-à-dire identifier la ou les fautes, isoler les composants fautifs, les remplacer ou les réparer, et enfin remettre le système dans sa configuration nominale : tout cela constitue le traitement des fautes.

Pour détecter les erreurs, on peut utiliser deux classes de techniques. La première classe est constituée par des contrôles de vraisemblance : il faut observer l'état du système, en particulier certaines valeurs ou certains événements, et vérifier s'ils sont plausibles. Cela nécessite en général peu de matériel ou logiciel supplémentaire (redondance). Parmi les contrôles de vraisemblance par matériel, notons que la plupart des microprocesseurs détectent des codes d'instruction inexistants ou interdits, des commandes inexistantes ou interdites, et que des "chiens de garde" permettent de détecter des dépassements de durée de certaines exécutions. Les contrôles de vraisemblance par logiciel consistent à insérer dans des programmes des tests sur les valeurs de certaines variables ou sur les instants de certains événements ou leur séquence. On parle alors de "programmation

défensive" [4]. Un cas particulier des contrôles sur les valeurs est constitué par les codes détecteurs d'erreurs.

La seconde classe des techniques de détection d'erreur consiste à comparer plusieurs exécutions, soit successives sur le même matériel, soit sur des matériels différents. Cette deuxième classe nécessite donc plus de redondance (on parle de "redondance massive"). Mais elle suppose aussi qu'une faute unique n'ait pas les mêmes effets, c'est-à-dire ne produise pas les mêmes erreurs, sur les différentes exécutions. S'il s'agit de fautes matérielles internes, on peut exécuter les mêmes traitements sur des matériels identiques, puisqu'il est peu probable que les deux exemplaires de matériels subissent chacun une faute interne identique en même temps. En revanche, s'il s'agit de fautes de conception, les mêmes traitements sur des matériels identiques produiront les mêmes erreurs, et la comparaison ne détectera rien. Dans ce cas, il faut "diversifier" la conception des matériels (si on suppose qu'il peut y avoir des fautes de conception dans le matériel), mais surtout des logiciels, où les fautes de conception sont fréquentes [5]. Dans ce cas, les différents exemplaires doivent être conçus et développés indépendamment, de façon à réduire le risque qu'une faute identique ne provoque les mêmes erreurs dans les divers exemplaires.

Pour corriger les erreurs, on peut tenter de remettre le système dans un état qu'il a eu avant qu'il y ait des erreurs, c'est-à-dire effectuer une reprise. Pour cela, il faut avoir construit des points de reprise, c'est-à-dire des sauvegardes de l'état du système. Une autre technique, la poursuite, consiste à remplacer l'état erroné du système par un état sain artificiel, et à poursuivre le traitement. Ceci est possible, par exemple, dans certains systèmes de commande-contrôle, où l'on peut réinitialiser le système et réacquérir les paramètres de tous les capteurs avant de poursuivre le traitement. Enfin, il est possible aussi de "masquer" les erreurs, lorsqu'on a suffisamment de redondance pour reconstituer un état correct à partir de l'état erroné, par exemple par vote majoritaire sur trois exécutions (ou plus).

Dans la plupart des cas, l'efficacité des techniques de tolérance aux fautes repose sur le fait que les fautes sont des phénomènes rares et distribués aléatoirement dans le temps. Ainsi, par exemple, lorsqu'on utilise une redondance massive, on suppose qu'il est peu probable qu'un exemplaire défaille pendant qu'on en répare un autre. Cette hypothèse n'est malheureusement pas valable lorsqu'on considère les intrusions : si un attaquant réussit à s'introduire dans un système, il essaiera de poursuivre son attaque sur le même système, mais aussi sur d'autres systèmes similaires.

3.2 Vulnérabilités, attaque, intrusion

Une intrusion résulte de l'exploitation d'une vulnérabilité (faute de conception ou d'exploitation, dans la terminologie de la sûreté de fonctionnement) par une attaque (faute externe) [6]. L'intrusion peut être considérée comme une faute interne, qui peut produire des erreurs pouvant provoquer une défaillance vis-à-vis de la sécurité, c'est-à-dire une violation de la politique de sécurité du système. On a vu précédemment qu'il était illusoire d'éviter les attaques sur

l'Internet. De même il est impossible d'éliminer toutes les vulnérabilités. Par exemple, on peut considérer que de connecter un système sur l'Internet est en soit une vulnérabilité, mais à quoi servirait un serveur Web s'il n'était pas connecté sur Internet ?

Il serait donc intéressant de tolérer les intrusions, c'est-à-dire de faire en sorte que l'intrusion dans une partie du système n'ait pas de conséquence sur sa sécurité. Pour cela, on peut utiliser les techniques développées dans le cadre plus général de la tolérance aux fautes. Mais cela pose deux problèmes principaux :

- Si une attaque a réussi (au moins partiellement) sur une partie du système, il ne doit pas être trop facile de réussir la même attaque sur une autre partie. Pour cela, il faut que chaque partie soit suffisamment sécurisée et, de préférence, que les parties soient diversifiées.
- Il ne faut pas qu'une seule intrusion dans une partie du système fournisse à l'attaquant des informations sensibles (confidentialité). Ceci est d'autant plus important que la redondance, nécessaire à la tolérance aux fautes, peut fournir plus d'occasions d'attaques aux pirates éventuels.

Si on est capable de résoudre ces problèmes, on peut appliquer les techniques de la tolérance aux fautes : traitement des erreurs (détection et correction) et traitement des fautes (diagnostic, isolation, réparation, reconfiguration). Notons cependant que pour ce qui concerne la détection des erreurs dues à des intrusions, des techniques de détection spécifiques ont été développées, improprement appelées "de détection d'intrusions", alors qu'elles ne détectent pas directement les intrusions, mais seulement leurs effets, c'est-à-dire les erreurs dues à des intrusions (ou mêmes seulement à des attaques, sans qu'il n'y ait intrusion).

Les techniques dites de détection d'intrusion se répartissent en deux classes : détection d'anomalies et détection d'attaques (voir la figure 1). La détection d'anomalies (*anomaly detection* en anglais) consiste à comparer le comportement observé d'un utilisateur à une référence de comportement normal de cet utilisateur. Toute déviation entre les deux comportements déclenche une alerte. En revanche, la détection d'attaques (*misuse detection*) est basée sur la comparaison du comportement observé avec une référence correspondant à des scénarios d'attaques connus. Ces deux types de détection se distinguent par leurs taux de fausses alertes (appelées "faux positifs") et de non-détections (appelées "faux négatifs"). Dans le cas de la détection d'anomalies, on peut en général "régler le gain" du détecteur (par analogie avec les systèmes radar), pour choisir un point de fonctionnement correspondant à un bon compromis entre ces deux taux. Les techniques de détection d'attaques, quant à elles, ont l'avantage d'identifier quel type d'attaque est en cours (avec relativement peu de faux positifs), mais ne permettent de détecter que les symptômes d'attaques connues. Dans les deux cas, il ne s'agit que de contrôle de vraisemblance.

Pour corriger les dégâts causés par l'intrusion, on peut, comme pour la tolérance aux fautes classique, effectuer une reprise (si on dispose de sauvegardes à jour), ou une poursuite (si on peut construire un état sain artificiel), mais il est souvent plus facile et plus efficace de masquer les erreurs, en utilisant

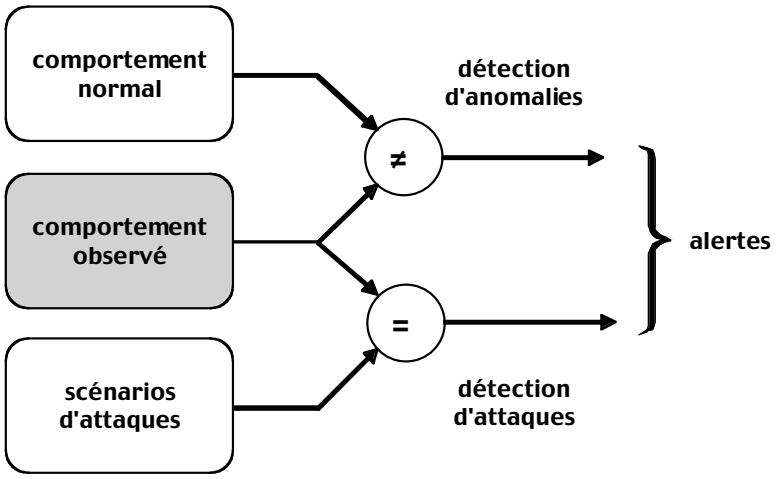

Fig. 1. Détection d'intrusions

une redondance massive. C'est ce que montrent les exemples décrits dans les trois sections suivantes.

3.3 Fragmentation, redondance, dissémination

Nous avons développé il y a quelques années une technique particulière de masquage d'erreurs qui garantit la confidentialité des informations sensibles : la fragmentation-redondance-dissémination, ou FRD [7]. Cette technique est basée sur le principe d'utiliser la répartition d'un système informatique sur un réseau local de façon à ce qu'une éventuelle intrusion ne mette pas en défaut la confidentialité, l'intégrité ou la disponibilité du système. La fragmentation consiste donc à découper les informations sensibles en fragments de telle sorte qu'un fragment isolé ne contienne pas d'information significative (confidentialité). On ajoute de la redondance à ces fragments de façon à ce que la modification ou la destruction de fragments ou de copies de fragments n'empêche pas la reconstruction de l'information (intégrité et disponibilité). Enfin, la dissémination vise à ce qu'une intrusion ne donne accès qu'à des fragments isolés. La dissémination peut être topologique, en utilisant des sites de stockage différents, ou en transmettant les fragments sur des canaux de communications indépendants. Elle peut être temporelle, en transmettant des fragments dans un ordre aléatoire et en y ajoutant éventuellement des faux fragments de bourrage. La dissémination peut aussi porter sur les privilèges, en exigeant la coopération de plusieurs personnes ayant des privilèges différents pour accomplir une opération (séparation des pouvoirs).

Cette technique de FRD a été développée dans le cadre du projet Delta-4 [8] pour le stockage de fichiers, la gestion de la sécurité et le traitement de

données (voir figure 2). Pour le stockage de fichiers, la fragmentation est réalisée par des techniques cryptographiques simples et le nommage des fragments par une fonction à sens unique avec une clé secrète. Les fragments sont transmis sur le réseau dans un ordre aléatoire, la difficulté pour un intrus éventuel étant principalement de remettre les fragments dans l'ordre avant d'effectuer une cryptanalyse. Pour la gestion de la sécurité, le principe est de répartir les fonctions d'authentification et d'autorisation sur un ensemble de sites administrés par des personnes différentes, de façon à ce que la défaillance d'un petit nombre de ces sites ou la malveillance d'un petit nombre de ces administrateurs ne mette pas en danger les fonctions de sécurité. Sur ces sites, les informations non sensibles sont répliquées, alors que les informations confidentielles sont fragmentées par des fonctions de chiffrement à seuil. Enfin, pour le traitement de données, on considère deux types d'informations : les données numériques ou logiques, dont la sémantique est définie par le traitement, et des données contextuelles (par exemple des chaînes de caractères) qui ne font l'objet que d'opérations simples (saisie, affichage, concaténation, ...). Dans ce schéma, les données contextuelles sont chiffrées et déchiffrées uniquement sur le site de l'utilisateur lors de la saisie ou de l'affichage. En revanche, les traitements donnent lieu à une décomposition en fragments de plus en plus fins, jusqu'à ce qu'un fragment ne contienne plus d'information significative. Une méthode de décomposition orientée-objet à été définie pour cela.

3.4 Le projet MAFTIA

Les techniques développées dans le projet Delta-4 sont bien adaptées à des applications réparties dans un réseau local, plus ou moins homogène. Elles ne sont pas toutes directement transposables à des applications réparties sur l'Internet, surtout lorsque ces applications mettent en jeu des entreprises ou des organisations ne pouvant se faire mutuellement confiance. Il ne peut dès lors y avoir une gestion homogène de la sécurité.

Le projet européen MAFTIA [6] visait directement à faciliter les développements d'applications Internet tolérant les intrusions. Pour cela, des protocoles et des intergiciels (*middleware* en anglais) ont été développés pour gérer plus facilement les communications de groupe tolérant les fautes (y compris les fautes byzantines), avec ou non des contraintes de temps réel, et avec ou non des garanties de confidentialité et d'intégrité. Ces protocoles et intergiciels ont en particulier permis le développement de tierces parties de confiance (par exemple, une autorité de certification) qui tolèrent les intrusions (y compris de certains des administrateurs). Des méthodes de détection d'intrusions réparties ur Internet ont été étudiées avec une attention particulière, puisque la détection d'intrusions contribue à la tolérance aux intrusions, mais c'est aussi l'une des cibles privilégiées des attaquants. Il faut donc faire en sorte que ces mécanismes de détection tolèrent eux-mêmes les intrusions. Enfin des schémas d'autorisation ont été développés pour des applications mettant en jeu des organisations mutuellement méfiantes. Dans ce cadre, un serveur d'autorisation, tierce partie de confiance tolérant les intrusions, vérifie que chaque transaction (mettant en jeu

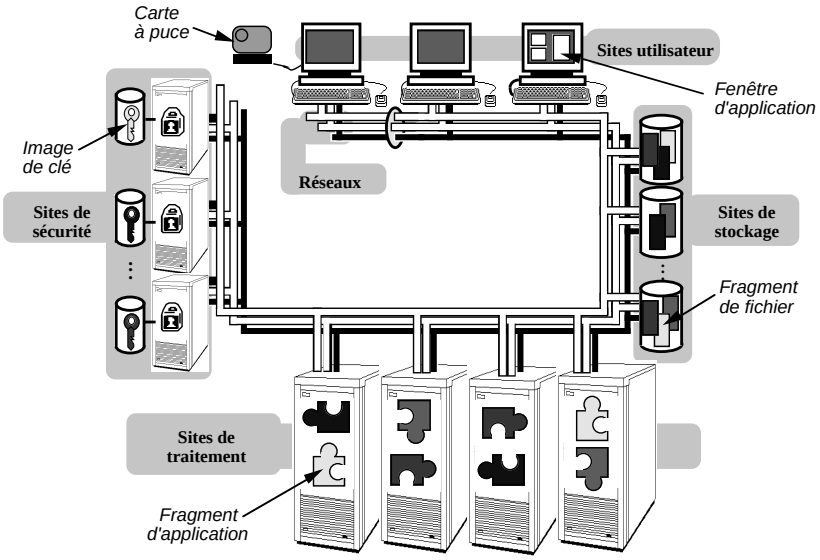

Fig. 2. Fragmentation-Redondance-Dissémination dans Delta-4

plusieurs parties) est autorisée, et dans ce cas génère les preuves d'autorisation nécessaires pour l'exécution de chaque élément de la transaction (invocation de méthodes sur des objets élémentaires). Sur chacune des machines participant à ces schémas, un moniteur de référence, implémenté par une carte à puce Java, vérifie que chaque invocation de méthode est accompagnée d'une preuve d'autorisation valide.

3.5 Le projet DIT

Dans le cadre d'une coopération avec SRI International, nous participons au développement de l'architecture DIT (Dependable Intrusion Tolerance) [9]. L'objectif est de fournir des serveurs Web capables de continuer à fournir un service correct en présence d'attaques. Pour ce type d'application, la confidentialité n'est pas essentielle, en revanche l'intégrité et la disponibilité doivent être assurées, même en cas d'attaque par des services compétents. Il est donc primordial qu'une attaque réussie sur un des éléments du système ne facilite pas

l'attaque d'autres éléments. C'est donc un souci de diversification qui a été au centre de la conception de l'architecture.

L'architecture se compose d'un banc de serveurs Web ordinaires, mais aussi diversifiés que possible en ce qui concerne la plateforme matérielle (processeurs SPARC, PENTIUM, PowerPC, etc.), les logiciels d'exploitation (Solaris, Microsoft Windows, Linux, MacOS, etc.) et les logiciels d'application Web (Apache, IIS, Enterprise Server, Openview Server, etc.) (voir la figure 3). Seul le contenu des pages Web est identique sur chaque serveur. Les serveurs d'application sont en nombre suffisant pour assurer un temps de réponse satisfaisant sur un taux de requêtes nominal dans un régime de redondance donné (voir plus bas). Les serveurs sont isolés de l'Internet par des "mandataires", eux-mêmes composés d'ordinateurs diversifiés pour ce qui concerne le matériel, mais animés par un logiciel développé spécifiquement. Les requêtes venant de l'Internet, filtrées par un pare-feu, sont prises en compte par l'un des mandataires qui joue rôle de leader. Le leader répartit les requêtes venant d'Internet vers les différents serveurs et vérifie leurs réponses avant de les envoyer à l'émetteur de la requête. Les mandataires de secours surveillent le fonctionnement du leader en observant les réseaux pare-feu/mandataires et mandataires/serveurs, et en cas de défaillance du leader, élisent entre eux un nouveau leader. Les mandataires traitent également les alarmes issues par les capteurs de détection d'intrusions installés sur les serveurs Web et sur les deux réseaux.

Selon le niveau d'alerte, le leader envoie chaque requête à un serveur (régime simplex), deux (régime duplex), trois (régime triplex) ou la totalité des serveurs. Chaque serveur élabore alors sa réponse, puis calcule une somme de contrôle cryptographique (MD5) sur cette réponse et l'envoie au leader. En régime simplex, le serveur envoie également sa réponse au leader, qui recalcule la somme de contrôle et la compare à celle transmise précédemment. En régime duplex, le leader compare les deux sommes et si elles sont égales demande l'une des réponses, qu'il vérifie en recalculant la somme de contrôle. En régime triplex (et au-delà), un vote majoritaire est effectué sur les sommes de contrôle renvoyées par les serveurs, et la réponse est demandée à l'un des serveurs majoritaires.

Le niveau d'alerte est défini en fonction soit des récentes alarmes déclenchées par les mécanismes de détection d'intrusions ou les autres mécanismes de détection d'erreurs (comparaison entre les résultats, test d'intégrité, etc.), soit d'informations transmises par des sources externes (CERTs, autres centres de confiance, etc.). Le régime de redondance évolue vers un régime plus sévère (redondance plus élevée) dès que des alarmes sont reçues, mais revient à un régime moins sévère (redondance moindre) lorsque les éléments défaillants ont été diagnostiqués et réparés, et lorsque la fréquence des alarmes diminue. Cette adaptation de régime est donc très liée aux mécanismes de détection, de diagnostic, de reconfiguration et de réparation.Dans le cas de serveurs d'informations, tels que des serveurs Web, la réparation peut consister simplement à réinitialiser le serveur à partir d'une version sûre (une copie validée sur disque non-réinscriptible).

En raison de la diversification, le travail de l'attaquant est rendu aussi difficile que possible : lorsqu'il envoie une requête pour une page Web (seul moyen

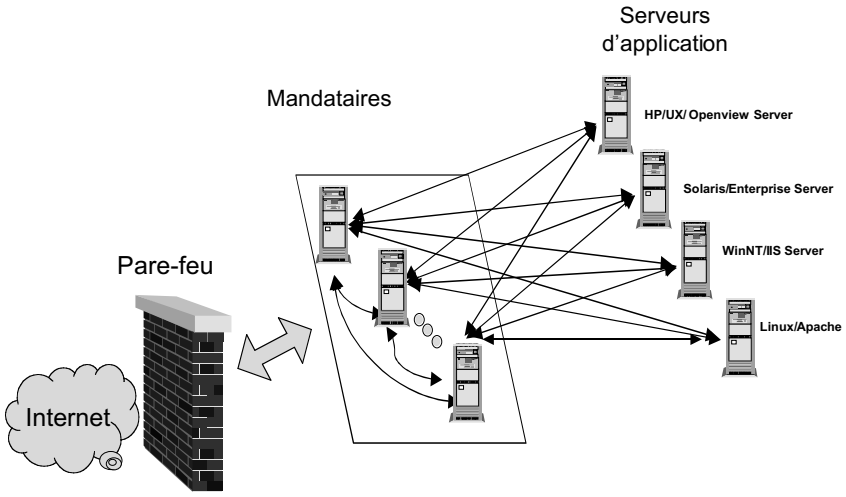

Fig. 3. Architecture DIT

pour lui de se connecter aux serveurs d'application), il ne sait pas quel type de matériel ni de logiciel va prendre en compte sa requête. Même s'il était capable de concevoir une attaque ayant le même effet sur tous les types de serveurs (ce qui difficile aujourd'hui, sauf peut-être pour des dénis de service, faciles à détecter), il est peu probable que les serveurs fournissent les mêmes résultats en régime duplex et au-delà.

4 Conclusion

Compte tenu de la fréquence des attaques sur l'Internet, et du grand nombre de vulnérabilités des systèmes informatiques actuels, la tolérance aux intrusions apparaît comme une technique prometteuse pour réaliser des applications plus sures, en particulier en diversifiant les plateformes matérielles et logicielles. Cela a bien sûr un coût, puisqu'il est plus onéreux d'exploiter des systèmes informatiques hétérogènes. Mais c'est sans doute le prix à payer pour plus de sécurité dans un monde ouvert, donc incertain.

Remerciements

Le projet Delta-4 a été mené dans le cadre du programme européen ESPRIT, de 1986 à 1992. Le projet regroupait, outre le LAAS-CNRS, Ferranti Computer Systems Ltd (GB) (principal contractant), Bull (F), le Crédit Agricole (F), IEI-CNR (I), IITB-Fraunhofer Institüt FhG (RFA), INESC (P), LGI-IMAG (F), MARI Advanced Electronics Ltd (GB), NCSR-AEA Technology (GB), Renault (F), Sema-Group (F), et l'Université de Newcastle (GB). Le projet MAFTIA (Malicious-and Accidental-Fault Tolerance for Internet Applications) était un projet du programme européen IST, mené de janvier 2000 à février 2003. Nos partenaires étaient les universités de Newcastle (coordinateur du projet), de Lisbonne et de Saarbrück, Quinetiq (GB) et le laboratoire de recherche d'IBM à Zurich. Le projet DIT est l'objet d'une coopération entre SRI International, principal contractant du projet, et le LAAS-CNRS. Il est partiellement financé par le programme OASIS (Organically Assured and Survivable Information Systems) de la DARPA (Defense Advanced Research Projects Agency) des États-Unis. Parmi les nombreux chercheurs qui ont contribué à ces travaux, il convient de remercier tout particulièrement Jean-Claude Laprie, Jean-Charles Fabre, Vincent Nicomette et surtout David Powell, pour leurs importantes contributions à la tolérance aux intusions.

Références

1. *Loi relative à la sécurité quotidienne* 2001-1062 du 15/11/2001, révisée par la Loi pour la sécurité intérieure 2003-239 du 18 mars 2003. Ces lois ont introduit l'article L-32-3-1 dans le Code des Postes et Télécommunications.
2. *Common Criteria for Information Technology Security Evaluation*, norme ISO/CEI 15408, version 2.1, août 1999.
3. J.-C. Laprie et al., *Guide de la sûreté de fonctionnement*, Cépaduès Éditions, 1995-1996, 368 p.
4. C. Rabéjac, *Auto-surveillance logicielle pour applications critiques : méthode et mécanismes*, Thèse de Doctorat de l'Institut National Polytechnique de Toulouse, N^o1095, novembre 1995, Rapport LAAS n 95449.
5. Y. Deswarte, K. Kanoun, J.-C. Laprie, "Diversity against accidental and deliberate faults", in *Computer Security, Dependability and Assurance : From needs to Solutions*, P. Amman, B.H. Barnes, S. Jajodia, E.H. Sibley Eds., IEEE Computer Society Press, 1999, pp.171-181.
6. D. Powell & R. Stroud (Eds.), *Malicious-and Accidental-Fault Tolerance for Internet Applications : Conceptual Model and Architrecture*, Final Version, Rapport LAAS n^o03011, Projet IST-1999-11583 MAFTIA, Deliverable D21, janvier 2003, 123 pp., disponible à <http ://www.research.ec.org/maftia/deliverables/>.
7. J.-C. Fabre, Y. Deswarte, L. Blain, "Tolérance aux fautes et sécurité par fragmentation-redondance-dissémination", *Technique et Science Informatiques (TSI)*, vol. 15, n^o4, pp.405-427, 1996.
8. D. Powell (Ed.), *Delta-4 : A Generic Architecture for Dependable Distributed Computing*, Springer-Verlag, ISBN 3-540-54985-4, 484 pp., 1991.

9. A. Valdes, M. Almgren, S. Cheung, Y. Deswarte, B. Dutertre, J. Levy, H. Saïdi, V. Stavridou, T. Uribe, "An Adaptative Intrusion-Tolerant Server Architecture", *10th International Workshop on Security Protocols*, Cambridge (UK), avril 2002, LNCS N^o 2845, Springer h, 2003, pp. 158-178.

Une version préliminaire de cet article a paru dans le numéro de septembre 2003 de la REE (Revue de l'Électricité et de l'Électronique), la revue de la Société de l'Électricité, de l'Électronique, et des Technologies de l'Information et de la Communication (SEE), 17 rue Hamelin, 75783 Paris cedex 16, `http://www.see.asso.fr/`