TEST DE SECURITE EXHAUSTIF DES SYSTEMES CONNECTES

Introduction à la solution beSTORM

Beyond Security, Inc. | 19925 Stevens Creek Blvd. | US: 1-800-801-2821 Cupertino, CA | Intl: +1-408-329-6041 | USA 95014
email: info@beyondsecurity.com | www.@beyondsecurity.com

Table des matières

1. BEYOND SECURITY

Les solutions de test de Beyond Security évaluent et gèrent de manière précise les failles de sécurité sur les réseaux, dans les applications, les équipements et systèmes industriels. Nous secondons les entreprises et les administrations dans la simplification de la gestion de la sécurité de leur infrastructure, réduisant ainsi leurs vulnérabilités aux attaques et à la perte de données.

Beyond Security adresse le marché de la vulnérabilité en différenciant les vulnérabilités connues et les vulnérabilités non connues.

Vulnérabilités connues : AVDS est une solution de scan de vulnérabilité automatisée couvrant vos besoins de pilotage de la sécurité tout au long de la vie et de la production de vos infrastructures. AVDS est une solution Cloud ou bien on-premise protégeant vos infrastructures de la couche 2 du modèle OSI jusqu'aux applications web dans un même outil, vous permettant notamment des rapports de conformité type PCI, ISO, SOX, Bâle III, Sarbanes-Oxley, Isaca, etc.

Vulnérabilités inconnues : beSTORM est une solution de test -automatisée et répétable- dite en « boite noire » pour rechercher et documenter les vulnérabilités zéro day sur vos équipements et infrastructures issus de l'informatique bureautique comme des systèmes industriels ou embarqués. beSTORM permet également de certifier des équipements EDSA 2.0 pour la norme internationale CEI 62443-3.

Beyond Security édite le portail Web SecuriTeam.com, lequel est aujourd'hui considéré comme l'un des sites Internet de premier plan en matière de sécurité sur Internet. SecuriTeam.com recense plus d'un million de pages vues par mois et envoie des mises à jour aux milliers d'utilisateurs abonnés à sa liste de diffusion d'alertes de sécurité. Le portail Web de SecuriTeam.com forme une composante fondamentale des activités de l'entreprise et un avantage concurrentiel essentiel.

A l'échelle mondiale, Beyond Security s'est forgé une réputation de chef de file dans le domaine des tests de vulnérabilités. Les solutions de Beyond Security sont des composantes fondamentales dans le programme de gestion des risques de bon nombre d'entreprises.

Le siège social de Beyond Security se situe à Cupertino en Californie. Les distributeurs et revendeurs de Beyond Security sont présents en Amérique du Nord et du Sud, en Europe, en Asie, en Afrique, au Moyen-Orient et en Australie.

Nous comptons parmi nos clients des établissements financiers, industriels, des commerces de détail, des administrations, des fournisseurs d'accès Internet, des sociétés de haute technologie, œuvrant dans la sécurité des informations, les télécommunications fixes, cellulaires, des entreprises du commerce électronique …

2. beSTORM

beSTORM procède à des tests de sécurité dynamiques des produits en phase de développement ou bien finis et peut être utilisé pour certifier la sécurité des solutions, équipements et applications connectées au réseau avant leur déploiement ou commercialisation.

Les services chargés de l'assurance qualité des logiciels, firmware ou hardware qui peuvent avoir recours à une douzaine d'outils différents pour éprouver la sécurité d'une application, ont désormais la possibilité d'accomplir l'ensemble des tests de sécurité dynamiques avec une solution unique. Les experts sécurité qui doivent certifier les applications avant leur déploiement, peuvent désormais utiliser un seul outil pour tester l'ensemble des applications en réseau, même celles assorties d'un protocole propriétaire.

Auparavant mis uniquement à la disposition des très grandes administrations et entreprises, beSTORM bénéficie de développements de longue date, solidement étayés en matière de localisation des problèmes dans les logiciels et les équipements de réseau. Simple d'utilisation, mais suffisamment puissant pour avoir été déployé par différentes organisation étatiques de défense au travers du globe, beSTORM peut se substituer à des douzaines d'outils d'emploi difficile, mal documentés, médiocrement suivis en terme d'évolution, et offre un processus de test de sécurité standardisé, fiable et reproductible pouvant être intégré par les entreprises de n'importe quelle taille dans leur processus d'assurance qualité et de sécurisation de leurs logiciels, équipements, solutions.

- beSTORM® effectue une analyse complète de tous les logiciel au cours de son développement ou de la validation assurance qualité, sans nécessiter de code source.

- Découvrez des vulnérabilités dans un logiciel, un système d'exploitation, un équipement ou une application. La technologie smart-fuzzing en boite noire de beSTORM testera chaque combinaison d'attaque possible.

- beSTORM détecte les anomalies des applications et vous avertit des combinaisons d'attaque réussie. Les problèmes de sécurité sont découverts et documentées automatiquement.

- L'Institut de conformité de sécurité ISA a certifié beSTORM pour leurs tests de robustesse et de sécurité des systèmes embarqués dans le cadre du programme de Certification Assurance ISASecure (EDSA).

D'un point de vue technique, beSTORM est un outil industriel de test intelligent en boîte noire, générant des données aléatoires. Il est utilisé dans un environnement de laboratoire pour éprouver la sécurité des applications pendant leur phase de développement ou pour certifier les logiciels ou les équipements en réseau avant leur commercialisation. Il est assorti d'une assistance et support technique et de solides développements futurs.

Principales fonctionnalités

1 **Innovant**
BeSTORM effectue une analyse exhaustive, découvre et documente les failles de sécurité nouvelles et inconnues dans n'importe quel logiciel. Il ne dépend pas des « signatures d'attaques » et trouvera les vulnérabilités inconnues ou bien précédemment découvertes.

2 **Couverture fonctionnelle**
Tous les protocoles peuvent être testés y compris les protocoles propriétaires complexes grâce à ses fonctions d'auto-apprentissage. beSTORM permet de tester tous les produits dans votre laboratoire en standardisant le processus et simplifiant les compétences requises par les équipes.

3 **Priorisation des attaques**
Les algorithmes d'attaque de beSTORM permettent de hiérarchiser et prioriser les tests pour commencer avec les attaques les plus Susceptibles de réussir. Cela permet d'économiser beaucoup de temps au cours du processus de vérification et souligne tout d'abord les problèmes les plus importants.

4 **Extensibilité**
beSTORM teste la réponse aux saisies dynamiques plutôt que les fonctionnalités du produit et par conséquent peut être utilisé pour tester des produits extrêmement compliqués avec une grande base de code.

5 **Évolutivité**
beSTORM fonctionne également avec plusieurs processeurs ou machines en parallèle afin de réduire considérablement la durée des tests. Le licensing de beSTORM lui permet d'être utilisé par plusieurs équipes de test.

6 **Complète**
L'analyse fine de beSTORM enrichie le processus d'audit en lui apportant la moindre vulnérabilité : vous pouvez trouver des attaques aussi subtiles que les attaques « off-by-one », mais aussi des attaques par buffer overflow qui ne font pas tomber pour autant vos applications.

7 **Précision des tests**
beSTORM vérifie l'application de l'extérieur en déclenchant les attaques. Par conséquent, une vulnérabilité est signalée uniquement si une tentative a été couronnée de succès, pour exemple un buffer overflow qui a été déclenché. Autrement dit, beSTORM émule un attaquant et documente l'entrée exacte qui provoque un échec, en éliminant les faux positifs.

8 **Conformité protocolaire**
BeSTORM peut convertir le texte d'un protocole standard (la description de la BNF) en un ensemble automatique de tests. Cela garantit que l'ensemble des fonctionnalités sont vérifiés et que les bugs qui pourraient se déclarer par la suite, plusieurs mois ou années après que le produit soit commercialisé, soient trouvés et documentées rapidement.

9 **Code agnostique**
beSTORM teste l'application binaire et est donc complètement indifférent au langage de programmation ou des bibliothèques système utilisés. beSTORM signale l'interaction exacte qui déclenche la vulnérabilité et les programmeurs peuvent alors déboguer l'application avec toutes les précisions sur l'environnement du développement nécessaires pour corriger la source de la vulnérabilité.

3. LE BESOIN

Les progrès technologiques, la digitalisation, la recherche de la productivité, le développement de nouvelles offres accroissent la pénétration du monde IP et ses interconnexions dans tous les domaines de la vie civile, industrielle, militaire, administrative, privée, citoyenne…Cette évolution multiplie à une vitesse qui nous surprend nous même le nombre de systèmes connectés, qu'ils soient critiques, confidentiels ou non.

Dans le même temps, les techniques et compétences des acteurs malveillant s'accroissent et sont devenus de véritables menaces pour les Etats, les organisations, les entreprises.

Nous assistons à la fois à l'accroissement constant et exponentiel des Cyber risques ainsi qu'à la multiplication systèmes connectés critiques mis en réseaux.

Citons par exemple les domaines et métiers suivants :

- Sécurité industrielle (usines, automates, centrales nucléaires, robotique…)
- Sécurité de solutions d'électronique embarquée (aéronautique, automobile, ferroviaire…).
- Sécurité des environnements IoT (objects connectés, réseaux bas débit, domotique…).
- Sécurité des infrastructures IT critiques, notamment de type OIV (énergie, transport, communication, santé, administration, organisation militaire…).
- Validation / sécurisation d'une application, d'une nouvelle plateforme technique, d'une chaine compète de production.
- Validation d'équipements sous un processus normalisé reconnu au plan international, à la demande du fournisseur ou bien du client.

L'apport des solutions de fuzzing est considérable pour ces environnements en évolution constante, voir en complète révolution (exemple : industrie automobile avec la voiture connectée, robotisation des usines…).

4. LE TEST EN BOITE NOIRE

Le fuzzing - ou test à données aléatoires - est une technique pour tester des logiciels, firmware, hardward. L'idée est d'injecter des données aléatoires dans les entrées d'un programme. Si le programme échoue (par exemple en « plantant » ou en générant une erreur), alors il y a des défauts et vulnérabilités à corriger.

Le grand avantage du fuzzing est que l'écriture de tests est extrêmement simple, ne demande aucune connaissance du fonctionnement du système et permet de trouver des vulnérabilités rapidement. Le fuzzing est également utilisé pour traquer des failles de sécurité ou dans la rétro-ingénierie.

Le fuzzing est une méthode de test dite en « boite noire », utilisée de façon additionnel à une analyse par « boite blanche » qui se pratique lorsque l'on connaît exactement le fonctionnement du produit à valider.

Le marché du Fuzzing est encore « jeune » en termes d'offres et se structure depuis simplement quelques années.

Jusqu'à présent, les tests de validation de plateformes ou applications sont principalement réalisés par des équipes de spécialistes, experts de la cyber sécurité, de consultants (internes ou externes) utilisant à la fois leurs compétences et de multiples outils (libres, développements spécifiques sur des environnements particuliers).

Pour adresser ce besoin, Beyond Security a développé beSTORM, une solution unique afin d'apporter les bénéfices suivants :

- Validation « complète » du produit avec couverture globale des tests à effectuer lors de la recherche de vulnérabilités « zéro day ».
- Réduction considérable du temps de test pour chacune des campagnes à mener.
- Automatisation, méthodologie, processus, productivité.
- Respects des procédures des tests dans le cadre de normes spécifiques (Ex : EDSA 2.0).
- Validation de solutions avec des produits certifiés, garantissant la qualité des tests et des outils.
- Evolution des solutions en fonction des nouvelles techniques de tests et de menaces.
- Réduction de la complexité et des compétences à maintenir au sein des équipes.

Si nous mettons de côté les projets de fuzzer « libres » qui ne présentent pas harmonieusement les qualités explicité ci-dessus, beSTORM est la seule solution qui permet de tester/fuzzer des produits sur des protocoles standardisés ET propriétaires en mode industriel et réglementaire.

BeSTORM est utilisé par de nombreux grands groupes internationaux pour la validation des leurs solutions.

Quelques références : AT&T / Booz Allen Hamilton / China Mobile Labs / Defense Information Systems Agency / SNCF / Ericsson /Federal Reserve Bank / Huawei / Intel / Juniper Networks / Kaspersky Lab / Kylin Development Team / Lexmark / Lockheed Martin Space Systems / MasterCard / Singapore Ministry of Defense / Korean Ministry of National Defense / National Geospatial Intelligence Agency / National Security Bureau, Taiwan / National Security Research Institute, Korea / Parsons Corporation / US Army Cryptographic Modernization / US Army Medical Information Technology Center…

5. beSTORM – Témoignages

- beSTORM a découvert une faille fatale chez un client dans son logiciel de tracking du stock exchange en 45 minutes de temps de travail et seulement 29 secondes de temps de traitement par beSTORM :

1. beSTORM a été installé sur une machine Windows du client qui s'est loggé sur son serveur ; beSTORM est démarré en mode « Man in the middle » pour capturer le trafic du serveur,

2. Le trafic réseau est chiffré, mais possède ce qui ressemble à un 'header'. En se basant sur ces données dans le 'header' un simple module de fuzzing beSTORM est créé. Le test a pris au total 45' depuis le début de l'installation jusqu'à la découverte de la vulnérabilité, la construction du module beSTORM a été réalisée en moins de 15 minutes,

3. Ensuite beSTORM est lancé pour démarrer le fuzzing du serveur (live) comme demandé et exigé par le client (A ne pas reproduire par les enfants à la maison :°) avec un monitoring en mode « TCP echo »,

4. 29 secondes plus tard, le vecteur de test n° 206 fait tomber le serveur qui devient totalement inaccessible depuis n'importe laquelle de nos machines …

5. Le client n'y croyant pas, il relance le serveur et demande de rejouer le même test qui reproduit la faille au bout de 30 secondes !

> Cet outil a été vraiment pratique à utiliser. J'ai expérimenté des fuzzers similaires avec beaucoup de travail avant de seulement pouvoir lancer des tests, par exemple, la solution open source Sulley juste pour la mettre en route ... beSTORM est très basique et très simple d'usage. Pas d'autres questions à répondre que celles que pose le « Wizard » ... La simplicité de cet outil permet de rentrer directement dans le test de sécurité (fuzzing) du système. Bien sûr, connaître les protocoles dans le détail de leur implémentation vous sera utile mais indépendamment, beSTORM est un outil génial. Merci.CC.

- beSTORM teste avec succès la solution « eSafe 4 Mail » qui passe le feu de 55 attaques de sécurité en buffer overflow. La solution d'email sécurisée passe avec succès la suite de tests de beSTORM.

  CHICAGO - Aladdin Knowledge Systems Ltd. (NASDAQ: ALDN) a annoncé avoir passé avec succès les audits de sécurité de beSTORM pour sa suite « eSafe 4 SMTP », confirmant ainsi que son produit est exempt de vulnérabilités ou de failles de sécurité.

  Tous les tests ont été menés avec beSTORM sur le serveur SMTP de relai des mails. La suite eSafe a été monitorée à chaque instant pour vérifier qu'aucune anomalie (crash ou ralentissement) n'était observée pendant les tests de sécurité.

6. beSTORM – Cas concrets

Pendant son développement l'équipe Beyond Security qui développe beSTORM vérifie que le produit remplit toutes les exigences client. beSTORM est alors utilisé pour tester des produits disponibles sur le marché. Vous trouverez ci-dessous l'exemple de 13 produits sur lesquels au moins une vulnérabilité a été trouvée et ont incité les éditeurs à mettre au point un patch de leur produit pour éviter les risques auxquels leurs clients étaient exposés.

- TFTPD32 Buffer Overflow Vulnerability (Long filename) - http://www.securiteam.com/windowsntfocus/6C00C2061A.html
- Vulnérabilité dans WinSyslog (DoS) - http://www.securiteam.com/windowsntfocus/6L00F158KE.html
- sipD vulnérabilité sur format de chaîne - http://www.securiteam.com/unixfocus/6R00G1595S.html
- sipD gethostbyname_r DoS - http://www.securiteam.com/unixfocus/6B00F0A95O.html
- Xlight FTP Server PASS Buffer Overflow - http://www.securiteam.com/windowsntfocus/6X00R0K95E.html
- ArGoSoft FTP Server Multiple Vulnerabilities (SITE ZIP, UNZIP, COPY, PASS) - http://www.securiteam.com/windowsntfocus/5RP010KCAO.html
- WFTPD GUI DoS - http://www.securiteam.com/windowsntfocus/5JP0B20CAY.html
- GlobalSCAPE Secure FTP Server Buffer Overflow (Parameter Handling) - http://www.securiteam.com/windowsntfocus/5KP0C20CAC.html
- KPhone STUN DoS (Malformed STUN Packets) - http://www.securiteam.com/unixfocus/5PP0B1FCLY.html
- Serv-U LIST -l Parameter Buffer Overflow - http://www.securiteam.com/windowsntfocus/5ZP0G2KCKA.html
- Titan FTP Server Aborted LIST DoS - http://www.securiteam.com/windowsntfocus/5RP0215CUU.html
- Firebird Database Remote Database Name Overflow - http://www.securiteam.com/unixfocus/5AP0P0UCUO.html
- Mollensoft Lightweight FTP Server CWD Buffer Overflow - http://www.securiteam.com/windowsntfocus/5RP0L15CUM.html

7. Fonctionnement

1. Introduction

beSTORM est un framework pour le fuzzing de systèmes qui peut être utilisé pour tester les systèmes que vous fabriquez ou les solutions fournies par des tierces parties.

beSTORM peut être également utilisé comme outil de qualification de vos systèmes (établissement de règles propres à votre organisation ou standard comme EDSA de ISA Secure pour la norme 62443).

beSTORM fournit un environnement de test avec une interface graphique et une interface en ligne de commande qui permet de piloter des tests les plus simples (protocoles intégrés au produit, utilisation du « Wizard ») jusqu'aux tests les plus complexes nécessitant de dérouler des scénarios impliquant de multiples systèmes.

beSTORM est un framework générique de test (fuzzing) qui ne nécessite pas de connaissance préalable en programmation.

beSTORM est particulièrement adapté et simple d'utilisation pour le test :

- De protocoles standards comme HTTP, POP3, SMTP, SIP et protocoles similaires disposant d'un RFC, des protocoles industriels sur bus CAN, OBD, … avec ou sans fil (BT, BT/LE, Wifi, Zigbee, …), …
- De formats de fichiers BMP, TGA, Wav, MP4, AVI, …
- De librairies DLL, d'appels d'API,
- Et globalement tout type de systèmes communicants (l'architecture du produit permet une très grande flexibilité),
- Et bien sûr tout format de protocole ou système propriétaire.

beSTORM est disponible sur plateformes Windows et Linux (sans IHM sur Linux).

2. Principe général

beSTORM est en fait un générateur de vecteurs d'attaque « intelligent » livré avec :

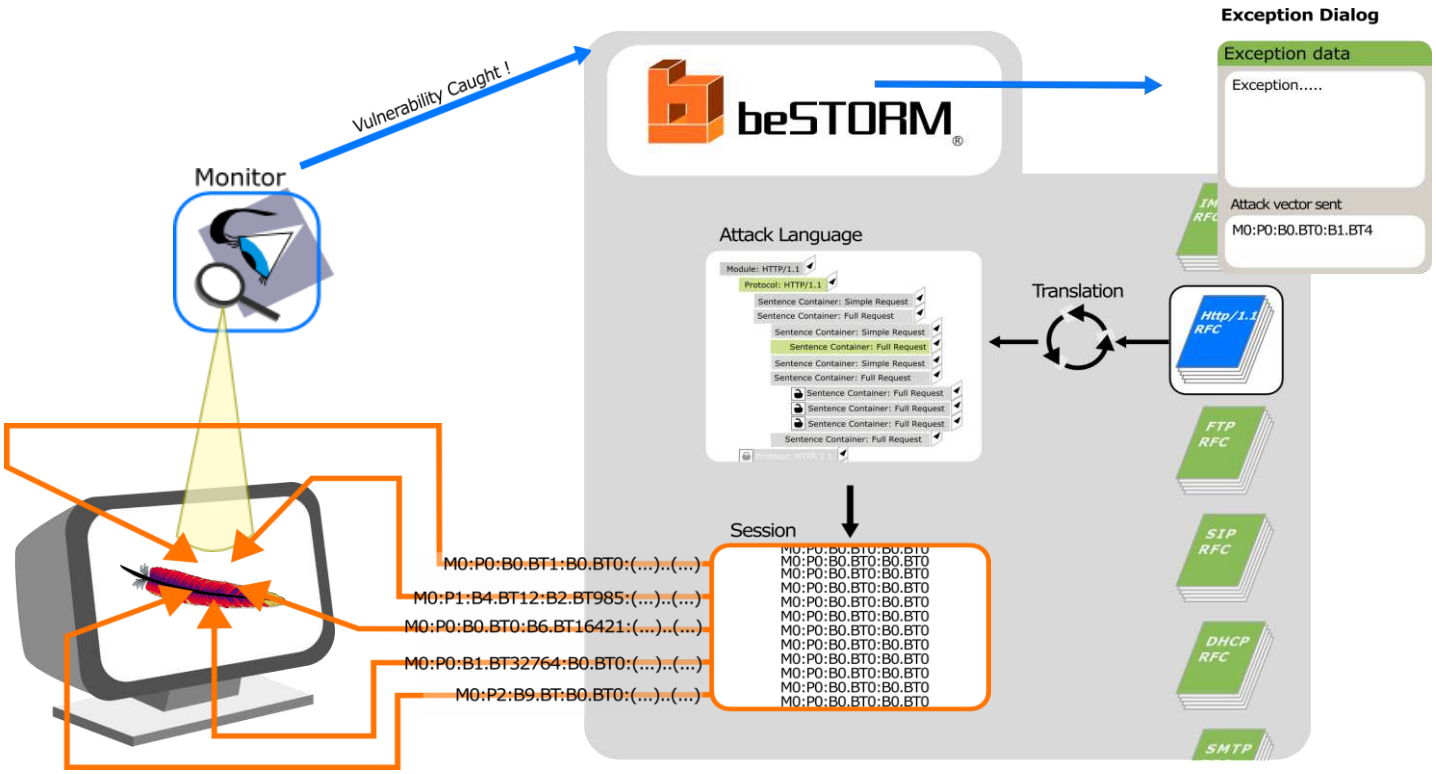
- Des descriptions des protocoles et formats de fichiers classiques (HTTP, IP, UDP, CAN, …),
- Des attaques classiques (« Repeated A », « %n », …),
- Des exemples de librairies et harnais de test.

Il s'appuie sur un langage de description des tests à réaliser et sur un harnais de test de vos systèmes.

Le harnais de test va permettre :

- D'une part de communiquer les vecteurs d'attaque vers la cible sous test,
- D'autre part d'observer des anomalies de fonctionnement et les reporter à beSTORM.

Une fois le test décrit et le harnais en place, beSTORM va parcourir intelligemment (de façon à opérer dans des temps raisonnables) l'arbre de tous les vecteurs d'attaque possibles et enregistrer toutes les exceptions générées par les vecteurs transmis à la cible sous test.

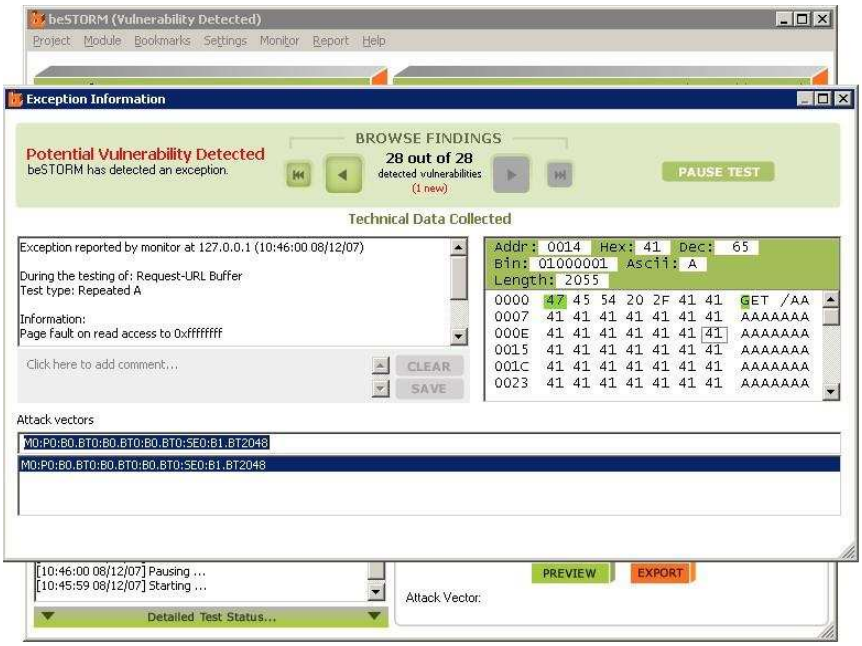


3. Interface graphique

L'interface graphique permet de créer de nouveaux projets très simplement avec un « wizard », de gérer finement les caractéristiques d'un projet, de lancer des tests et en enfin de parcourir et générer des rapports des tests effectués.

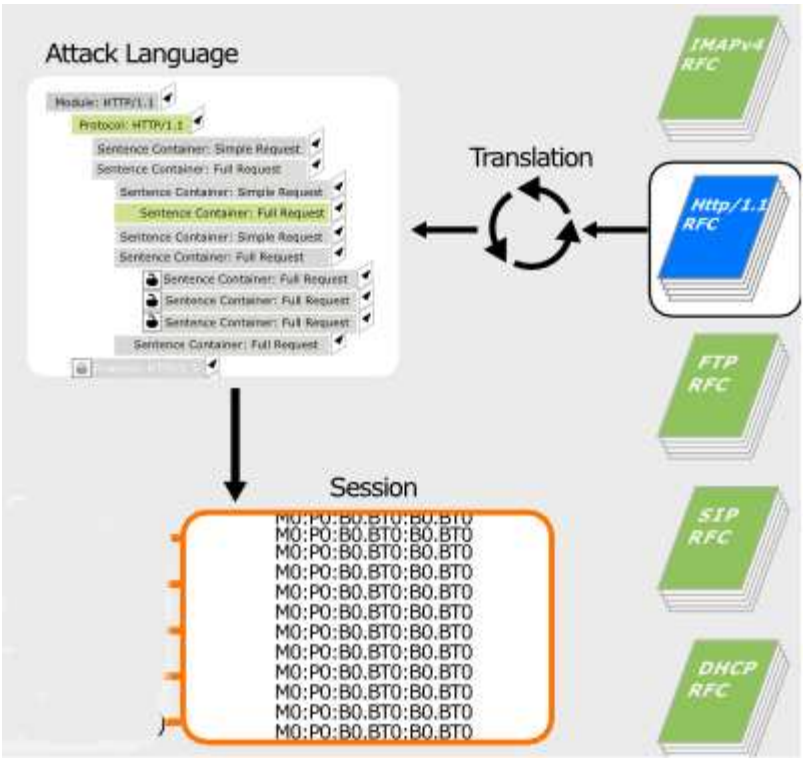
A son lancement beSTORM propose un « wizard » qui vous accompagne dans la création de votre test et permet de très vite disposer d'un environnement opérationnel.

Lorsque vous lancez vos tests et découvrez des vulnérabilités, vous pouvez observer le vecteur de test et les résultats de l'attaque en détail dans l'IHM :



4. Vecteurs d'attaque

BeSTORM permet de fuzzer des fichiers, des DLL, des protocoles réseau et à peu près tout type de systèmes ou de systèmes de systèmes grâce à la notion d'arbre couvrant de manière intelligente l'ensemble des solutions possibles pour interagir avec le système sous test.



Cette interaction est rendue possible par une grammaire exprimée en XML qui décrit finement Des protocoles, Des formats de fichier, Des paramètres d'API, …

A partir de cette description, beSTORM va automatiquement créer un arbre des donnant les différentes attaques possibles sur le système, chaque feuille de cet arbre ayant une référence unique i.e. le vecteur d'attaque.

beSTORM va lancer tous les vecteurs d'attaque sur la cible sous test, en notant à chaque moment si un vecteur d'attaque a été « efficace » ou non. BeSTORM peut être informé de l'efficacité d'un vecteur d'attaque soit par un « Moniteur » soit par timetout.

La notion de vecteur d'attaque est très importante dans beSTORM. Elle permet par exemple de rejouer des séquences de test à partir d'une certaine position en ayant au préalable enregistré le vecteur de test qui vous intéresse.

5. Langage ligne de commande

En plus de l'interface graphique, beSTORM peut être utilisé via un langage de pilotage qui permet de lancer des campagnes de test automatiques, engageant des scénarios éventuellement complexes et permettant l'écriture de tests de systèmes ou de systèmes de systèmes.

Les ports 6969 et 6970 de la machine beSTORM permettent de piloter beSTORM :

- Port 6969 : utilisé pour déclarer qu'une exception/vulnérabilité/problème a été identifié sur la cible sous test ; le message envoyé sera reproduit dans le rapport beSTORM (limité à 64 k),
- Port 6970 : utilisé pour piloter beSTORM avec un langage de commande.

Le langage permettant de piloter beSTORM est le suivant :

- NOOP : watchdog, permet de prévenir beSTORM que le système sous test est toujours « up and running »,
- LOAD : charge un projet beSTORM donné en paramètre (ex : « LOADc:\project\settings.bsp »)
- SAVE : sauvegarde le projet en cours,
- EXCEPTION : permet de déclarer une exception auprès de beSTORM,
- ERROR : informations à reporter à l'utilisateur comme une erreur durant les tests,
- UP : augmente la vitesse d'attaque de beSTORM,
- DOWN : diminue la vitesse d'attaque de beSTORM,
- START : démarre les attaques beSTORM,
- STOP : met en pause les attaques beSTORM, on reste dans le projet en cours,
- RUNFOR : jouer « n » vecteurs d'attaque et stopper, à utiliser en combinaison avec VECTORS et RETURNVECTORS,
- FINISH : termine la session d'attaques beSTORM, on peut alors charger un autre projet,
- STATUS : demande le statut beSTORM (en cours de test par exemple ou en mode pause),
- CURRENTVECTOR : retourne le vecteur d'attaque actuel,
- VECTOR : demande à beSTORM de charger un vecteur d'attaque spécifique – cette fonction permet de jouer à un endroit choisi dans l'arbre des attaques beSTORM plutôt que de démarrer depuis 0,
- INCREMENTVECTOR : passe au vecteur d'attaque suivant (possibilité de faire du fast forward en augmentant de n vecteurs à la fois),
- RETURNVECTORS : enregistre dans un fichier le nombre de vecteurs d'attaque spécifiés, exemple : « RETURNVECTORSC:\TEMP\DUMP.TXT\t1000 »,
- COUNTEXCEPTION : nombre d'exceptions rencontrées depuis le début du test,
- RETURNEXCEPTION : retourne les informations provenant d'une exception en particulier,
- SETTINGSCHANGED : précise si le projet en cours a été modifié (par beSTORM ou par l'utilisateur),
- STATS : retourne le nombre total de sessions générées et le nombre d'attaques par secondes actuel,
- EXIT : termine les tests et termine l'application beSTORM.

6. Langage de description des tests

BeSTORM propose un langage de description des attaques sous format XML (on parle de modules) qui peut être généré depuis l'interface graphique ou que vous pouvez éditer directement avec un éditeur si vous devenez familier avec ce langage.

Exemple de description d'une attaque avec login/password sur un serveur supportant SSL :

```
<SC Name="URL">
 <S Name="First Part">
  <C Name="Method with URI" ASCIIValue="GET /cgi-bin/login.cgi?username=" />
  <VB Description="Username for FORM based authentication" ASCIIDefault="username" Name="Buffer0" />
  <C Name="Password section" ASCIIValue="&password=" />
 </S>
 <SP Library="OpenSSL Interface.dll" Name="HASH" Procedure="MD5Encode" >
  <S Name="Encoder S">
   <VB Description="Password for FORM based authentication" ASCIIDefault="password" Name="Buffer1" />
  </S>
 </SP>
 <S Name="Second Part">
  <C Name="Ender" ASCIIValue=" HTTP/1.1" />
  <C Name="CR" Value="0x0D,0x0A" />
  <C Name="Another line" ASCIIValue="Host: " />
  <VB Description="Host field for HTTP" ASCIIDefault="192.168.4.5" Name="Buffer2" />
  <C Name="CR" Value="0x0D,0x0A" />
  <C Name="CR" Value="0x0D,0x0A" />
 </S>
</SC>
```

Le langage permet d'utiliser des librairies internes au produit comme des librairies externes (Math, OpenSSL, …) comme d'utiliser vos propres librairies et vous aide également à construire vos propres DLL.

Parmi les balises importantes documentées dans le manuel beSTORM, la balise BT permet de créer des attaques spécifiques. Des attaques « type » sont fournies avec les modules intégrés du logiciel. Exemple : « A répété », buffer overflow,%n, …

Vous pouvez étendre ces attaques en définissant vos propres heuristiques soit de manière générale dans un bloc en début de description de votre fichier xml, soit en local sur un type de données particulière.

```
<GeneratorOptSettings>
<BT>
<T Name="Repeated A" Max="65536" ASCIIValue="A" />
<T Name="Repeated %n" Max="512" ASCIIValue="%n" />
<T Name="Repeated %25n" Max="256" ASCIIValue="%25n" />
<T Name="Repeated Base64A" Max="16384" Type="Base64" ASCIIValue="A" />
<T Name="BiggerSmaller" Max="32768" ASCIIValue="&lt;&gt;" />
<T Name="Repeated %00" Max="21846" ASCIIValue="%00" />
<T Name="Number Generating DEC" Max="4294967295" Type="DecimalPositive" />
<T Name="Negative Number Generating DEC" Max="2147483648" Type="DecimalNegative" />
<T Name="Number Generating HEX" Max="4294967295" Type="DecimalPositive" />
<T Name="Repeated Space" Max="65536" ASCIIValue=" " />
</BT>
</GeneratorOptSettings>
```

Le langage vous permet également de donner des éléments variables qui pourront être définis via l'IHM afin de « paramétriser » vos tests.

7. Exemple de rapport

Les rapports reprennent les informations globales connues par beSTORM pour la cible sous test, la liste des attaques prédéfinies pour la cible et donnent ensuite la couverture du test (quelques sont les tests réalisés entièrement, en cours, à réaliser, …) et une liste des exceptions observées par beSTORM avec le vecteur de test au moment où l'exception s'est produite.

Pour les opérationnels ou équipes de test, le rapport est un moyen d'envoyer le moyen de reproduire (en laboratoire ou en R&D) les attaques ayant mené à un comportement incohérent du système. Dans les rapports, un script perl est fourni qui permet de reproduire le vecteur d'attaque.

Exception caught on 00:35:20 03/30/16

The following information was reported:

Exception reported on 127.0.0.1 by monitor located at 127.0.0.1 (00:35:20 03/30/16)

During the testing of: Accept-Language value
Test type: Repeated Base64 (A)

Information:
Page fault on read access to 0x46555136
Register dump:
CS:0023 SS:002b DS:002b ES:002b FS:0053 GS:002b
EIP:7773876c ESP:005efab8 EBP:005efc70 EFLAGS:00000206(- 00 I - -P1)
EAX:00000000 EBX:00dbce70 ECX:00000000 EDX:00000000
ESI:00dbccb0 EDI:00db36a8

Stack trace:
ERROR: SymGetSymFromAddr64, GetLastError: 487 (Address: 62CF3D3A)
ERROR: SymGetLineFromAddr64, GetLastError: 487 (Address: 62CF3D3A)

The following data was used to generate the exception: (show perl script hide perl script)

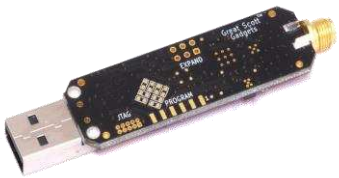
GET / HTTP/1.1
Host: 127.0.0.1
User-Agent: Mozilla/5.0 (X11; U; Linux i686; en-US; rv:1.8.1.6) Gecko/20070723 Iceweasel/2.0.0.6 (Debian-2.0.0.6-1)
Accept: text/xml,application/xml,application/xhtml+xml,text/html;q=0.9,text/plain;q=0.8,image/png,*/*;q=0.5
Referer: http://127.0.0.1/
Accept-Language:
QUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFB
Accept-Encoding: gzip,deflate

```
$host    = $opt{H} ? $opt{H} : $host;
$port    = $opt{P} ? $opt{P} : $port;
$timeout = $opt{t} ? $opt{t} : $timeout;

my @commands = (
{Command => 'Send',
 Data => "GET / HTTP/1.1\x0D\x0AHost: 127.0.0.1\x0D\x0AUser-Agent: Mozilla/5.0 (X11; U; Linux i686; en-US; rv:1.8.1.6) Gecko/20070723 Icewea

);
```

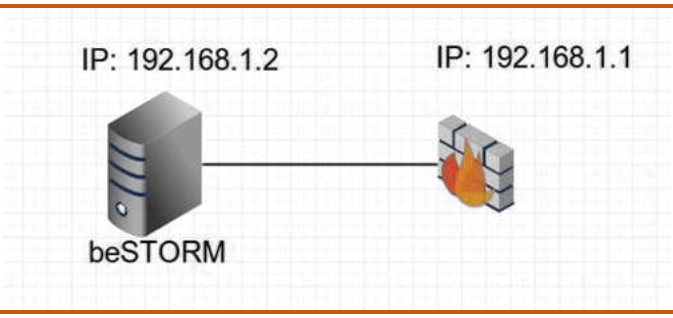
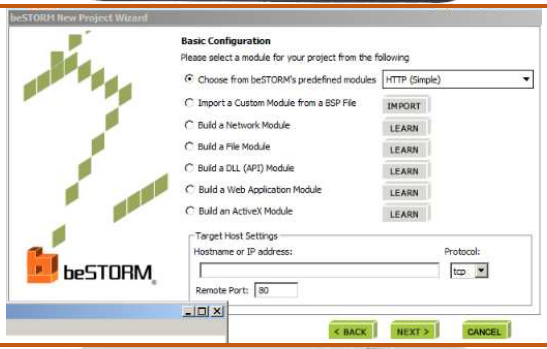
Ci-dessus un extrait de rapport beSTORM qui montre 1 le vecteur d'attaque ayant provoqué une anomalie sur le système sous test, 2 la trame qui a provoqué l'anomalie, 3 le script de test qui permet de reproduire l'anomalie (et qui peut être utilisé par des tiers ne disposant pas de la solution pour reproduire l'anomalie).

8. Exemple d'interfaces utilisées avec beSTORM

beSTORM permet de tester des applications par envoi de fichier, des DLLs, des APIS, des interfaces réseau distantes sur des protocoles classiques mais permet également de tester différentes formes d'équipements à travers des liaisons spécifiques.

WIFI/802.11	• IEEE 802.11 • 802.11: Beacon, Association, DeAssociation, … • 802.11u: Action, Advertisement, Query, …	
Bluetooth	• Bluetooth (Low Energy and "Regular") • L2CAP, BNEP, RFCOMM, • SDP (Service Discovery), • TCP (Telephone Control), OBEX, ATT, SMP	
RFID et NFC	• ISO/IEC 18092 / ECMA-340 • ISO/IEC 21481 / ECMA-352 • ISO/IEC 14443	
2G / 3G	• Cellular Testing • GTP-U • MTP3 • M3UA • SCTP • SCCP	
OSB	• Token Packets • Data Packets • Handshake Packets • Endpoints – Storage Device, HID, Printer, ...	
CAN	• OBEXII • ISO 11898-1 • ISO 11898-2 • CAN 2.0A 11bit/29bit • J1939, CANopen, CCP/XCP	
ZigBee	• IEEE 802.15.4 • ZDO (ZigBee Device Object) • KVP (Key Value Pair)	

9. Exemples de tests beSTORM

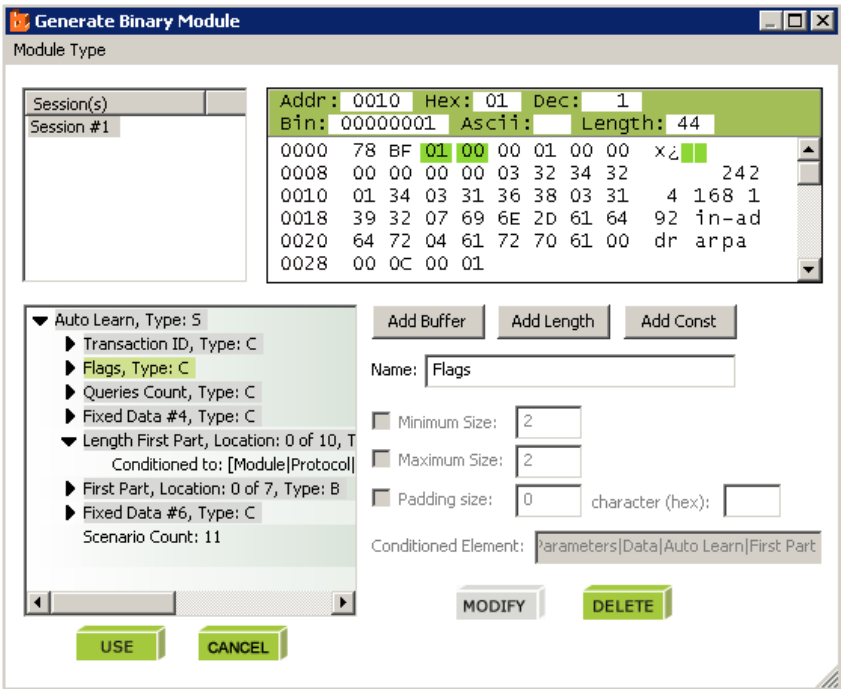
Test de Firewall HW	
Test d'injection de fichiers pdf	
Test de bus automobile CAN	
Test de serveurs web (SSL, http, HTTPS, …)	
Test de systèmes industriels suivant des processus maison ou suivant les critères EDSA	

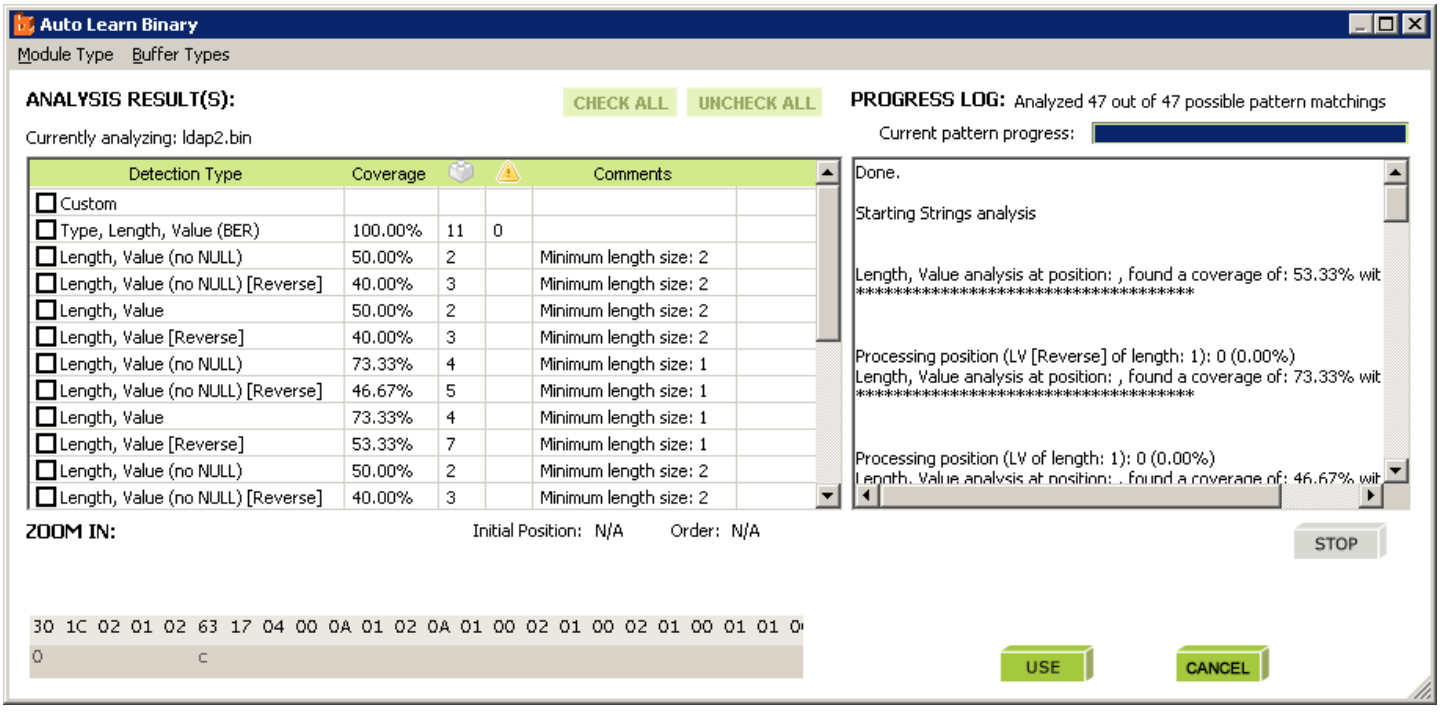
10. Couverture protocolaire

Bundles	Description
Basic IPv4	IP, TCP, UDP, ICMP, ARP
Basic IPv6	IP, TCP, UDP, ICMP, ARP
VOIP	SIP UAS, ISUP (SIP-I, SIP-T), SIP, UAC, RTP, STUN, MGCP (Megaco, H.248)
Routing	IGMP (v0, v1, v2 and v3), BGP4 Server, BGP4 Client, DVMRP,GRE, IS_IS, MPLS/LDP, NHRP, OpenFlow, OSPF v2, OSPF v3, PIM-SM, RIP, RIPng, RSVP, VRRP, IGMP
TLS	TLS/SSL Client, TLS/SSL Server, TLS 1.1 Client, TLS 1.1, Server, SSL v2 Client, SSL v2 Server, SSL v3 Client, SSL v3 server
EDSA	401 Ethernet, 402 ARP, 403 IPv4, 404 ICMPv4, 405 UDPv4, 405 UDPv6, 406 TCPv4
VPN	IPSec (AH, ESP), ISAKMP/IKEv1, Server/Client Ikev2, L2TPv2/3, OCSPclient / server,X.509 Certificates
Basic Network Clients	SSH, FTP, HTTP, DNS, NTP, DHCP
Basic Network Servers	SSH, FTP, HTTP, HTTPS, DNS, NTP, DHCP/bootp, SMTP
NetBIOS Server	NetBIOS Server
Mobile	PMIPv6, 3GPP, GTPv1 (GTP-U), M3UA (MTP3)
Tunneling	PPPoE, RGMP, SCTP, LLC, Teredo, TPKT (RFC 1006)
Full Network Clients	BVLC (BACnet transport), DHCP, Diameter, DNS, FTP,HTTP/1.0, HTTP/1.1, HTTPS, IMAP, LDAP, NNTP, NTP, POP3, Radius, RSH, RTSP, SDP, SNMP (v1, v2 and v3 with/without MD5, SHA and DES), SMTP, SYSLOG, SSH, TFTP, Telnet
SCADA	DNP3, Modbus
Files	ANI, BMP, DOC (MS Word), GIF, ICO JASC PAL, PAL,PCM, TGA, UPX
XML	XML/SOAP server and client, XML Parser
WIFI	802.11, WLAN-AP, WLAN-AP client, WPA-AP, WPA-AP CLIENT
Metro Ethernet	BFD, CFFM, E-LMI, Ethernet, GARP, LLDP, OAM, PBT IPBB-TE, Sync Ethernet
Bluetooth	L2CAP, SDP, RFCCMM, OBEX, OPP, FTP, IrMC Sync, BIP, BPP, BNEP, HFP, HSP, DUN, PBAB, FAX, AVRCP, A2DP, HCRP, HID, SAP, HFP Client, HSP Client, MDP/HDP, 2.1 Compliant
Email	POP3 Client, POP3 Server, IMAP4 Client, IMAP4 Server,SMTP Client, SMTP Server, MIME Server
IPTV	MPEG4, MPEG2, IPSEC, TLS/SSL, PRT/RTCP/RTSP/RTSP, HTTP, FTP, TFTP, IPV4, PIM SM /DM, RSVP, IGMP, CWMP (TR-69), SIP-UAC/UAS
CAN Bus	CANbuster interface device, CANbus, OBDII, Developer's package
Developer's package	Custom models development

11. Protocoles propriétaires

BeSTORM propose un module d'apprentissage automatique de protocoles propriétaires/inconnus.
Un wizard est intégré à l'outil pour des formats binaires, comme pour les formats texte pour définir les protocoles, puis générer le langage d'attaque de beSTORM.



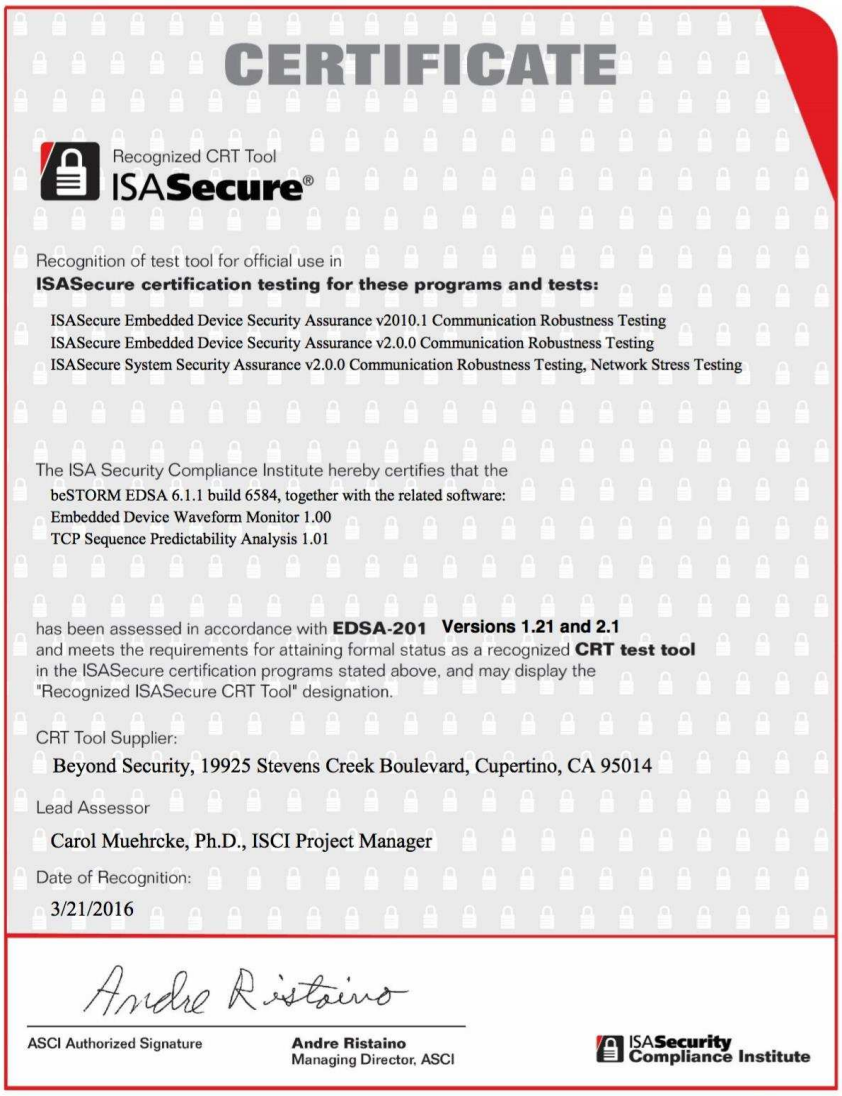


12. Certificat

Pour vos certifications EDSA – IEC 62443-3, il est nécessaire de disposer d'un outil ayant passé les tests de certification du programme ISA Secure « CRT Test Tool ».

beSTORM vous permet de garantir que vos systèmes sont conformes aux tests CRT de sécurité du programme ISA Secure et ainsi garantir à vos parties prenantes (actionnaires, direction, règlementaire, clients, …) que vos produits répondent à un standard de sécurité avec un niveau d'exigence industriel.

beSTORM est le premier et seul outil certifié EDSA 2.0.



CERTIFICATE

Recognized CRT Tool
ISASecure®

Recognition of test tool for official use in
ISASecure certification testing for these programs and tests:

ISASecure Embedded Device Security Assurance v2010.1 Communication Robustness Testing
ISASecure Embedded Device Security Assurance v2.0.0 Communication Robustness Testing
ISASecure System Security Assurance v2.0.0 Communication Robustness Testing, Network Stress Testing

The ISA Security Compliance Institute hereby certifies that the
beSTORM EDSA 6.1.1 build 6584, together with the related software:
Embedded Device Waveform Monitor 1.00
TCP Sequence Predictability Analysis 1.01

has been assessed in accordance with **EDSA-201 Versions 1.21 and 2.1**
and meets the requirements for attaining formal status as a recognized **CRT test tool**
in the ISASecure certification programs stated above, and may display the
"Recognized ISASecure CRT Tool" designation.

CRT Tool Supplier:
Beyond Security, 19925 Stevens Creek Boulevard, Cupertino, CA 95014

Lead Assessor
Carol Muehrcke, Ph.D., ISCI Project Manager

Date of Recognition:
3/21/2016

Andre Ristaino

ASCI Authorized Signature

Andre Ristaino
Managing Director, ASCI

ISASecurity Compliance Institute

Exemple de rapport de confirmité EDSA 2.0 produit par beSTORM.

Project Settings

Name	Value
beSTORM Version:	5.2.1 (6408)
Project Name:	EDSA
Thread Count:	1
Remote Monitor IP Address / Hostname:	192.168.1.4

Module Settings

Module name:	EDSA-405 UDPv4 (1.31)
Fuzzing conditioned elements:	Yes
Generator type:	Binary
Increment order:	Normal
Overflow elements once:	No
Scale type:	Base2+/-1
AVPS (Threshold):	999999
Kbps (Threshold):	124999875.000
Batch Mode:	yes
Report Connectivity Issues as Exceptions:	no
Minion Enabled:	no
Minion Host and Port:	unset (6980)
Module Type:	Raw Network
Total Bytes Sent:	0

* AVPS - currently applicable attack-vector per second rate
* Kbps - Kilobits per second. note: actual capping takes effect at the lower resulting se

Running time

beSTORM has been running for (total): **0 months, 1 days 3 hours 46 minutes 43 seconds**

Started	Ended	Duration
12:31:52 08/31/15	16:18:35 09/01/15	0 months, 1 days 3 hours 46 minutes 43 seconds

Exception Report

beSTORM has found a total of **0** exception(s):

No exceptions found.

Test list

Test Name	Attack Types	Status
Robustness Test	Default	Done
UDPv4.T00 Comment: (Baseline Operation)	Default	Done
PCAP Send	Default	Done
Virtual UDP	Default	Done
UDP Headers and Data T00	Default	Done
UDP Body T00	Default	Done
UDP Body value	Default	Done
UDPv4.T01 Comment: (Truncated TPDU Header with "non-negative" Length Field)	Default	Done
PCAP Send	Default	Done
Virtual UDP	Default	Done
UDP Headers and Data T01	Default	Done
UDP Headers T01	Default	Done
Length of Data T01	Default	Done
UDP Body T01	Default	Done
UDP Body value	Default	Done
UDPv4.T02 Comment: (Truncated TPDU Header with "negative" Length Field)	Default	Done
PCAP Send	Default	Done
Virtual UDP	Default	Done
UDP Headers and Data T02	Default	Done
UDP Body T02	Default	Done
UDP Body value	Default	Done
UDPv4.T03 Comment: (Valid TPDU Shorter than IP NPDU Payload)	Default	Done
PCAP Send	Default	Done
Virtual UDP	Default	Done
UDP Headers and Data T03	Default	Done

Présentation G-echo – powered

http://www.g-echo.fr

contact@g-echo.fr