Penerapan Model Detection Engineering dalam Proses Threat Modeling di dalam Security Operation Center (SOC)

oleh Mangatas Tondang dan Wahyu Nuryanto

mt.infosec [at] gmail.com | wahyu [at] blueteam.id

Abstrak

Dalam dunia keamanan siber, sinergi antara berbagai proses memiliki peran yang sangat penting. Salah satu proses atau *framework* yang tengah menjadi sorotan dan menarik perhatian luas adalah *Detection Engineering*. Proses *Detection Engineering* ini bertujuan untuk meningkatkan struktur dan pengorganisasian dalam pembuatan *detection use case* atau *rules* di *Security Operation Center (SOC)*. *Detection Engineering* bisa dikatakan masih baru dalam dunia keamanan siber, sehingga terdapat banyak peluang untuk membuat keseluruhan prosesnya menjadi lebih baik. Salah satu hal yang masih terlupakan adalah integrasi antara proses *Detection Engineering* dan *Threat Modeling*. Biasanya, *Threat Modeling* lebih berfokus pada solusi pencegahan dan mitigasi resiko secara langsung dan melupakanan komponen deteksi ketika pencegahan dan mitigasi tersebut gagal dalam menjalankan fungsinya. Dalam makalah ini, kami memperkenalkan paradigma baru dengan mengintegrasikan *Detection Engineering* ke dalam proses *Threat Modeling*. Pendekatan ini menjadikan *Detection* sebagai langkah proaktif tambahan, yang dapat menjadi lapisan pertahanan ekstra ketika kontrol pencegahan dan mitigasi akhirnya gagal dalam menghadapi ancaman sesungguhnya.

Kata Kunci

detection engineering, security operation center, threat modeling, detection use case, stride, owasp, mitre att&ck, threat hunting, defensive security

1.Pendahuluan

Detection engineering dapat didefinisikan sebagai serangkaian proses yang memungkinkan ancaman potensial terdeteksi dalam suatu lingkungan. Proses-proses ini mencakup siklus dari awal hingga akhir, mulai dari pengumpulan kebutuhan deteksi, pengumpulan data telemetri sistem, penerapan dan pemeliharaan logika deteksi, hingga validasi efektivitas program [1]. Di dalam *Security Operation Center (SOC)*, proses *Detection Engineering* ini masih jauh dari kata ideal.

Dalam survey terbaru dari SANS Institute tentang *Security Operation Center* [2], kurangnya konteks atau informasi terkait *alert* yang diterima oleh *SOC Analyst* menjadi alasan utama kenapa SOC tidak dapat beroperasi secara maksimal. Hal ini tidak terlepas dari proses

Detection Engineering yang belum berjalan dengan baik atau bahkan tidak ada sama sekali. Sebagian besar SOC masih tergantung pada *detection rules* yang sudah ada pada *Security Information and Event Management (SIEM)* dimana hampir sebagian dari *rules* tersebut mungkin tidak relevan dengan organisasi mereka, dan bahkan sebagian besar di antaranya mungkin tidak memiliki sumber data yang sesuai.

Salah satu jenis *detection rules* yang paling sedikit area cakupannya adalah *detection rules* di sisi aplikasi, baik aplikasi yang dibuat sendiri ataupun aplikasi yang dibeli dari pihak ketiga. Aplikasi yang dikembangkan sendiri oleh organisasi sudah hampir pasti tidak memiliki detection rulenya jika pada tim SOC tidak terdapat proses *Detection Engineering* yang memadai. Sementara dari sisi aplikasi, fitur pencatatan *logs* masih menitikberatkan pada kebutuhan untuk identifikasi masalah dan pemantauan stabilitas aplikasi itu sendiri.

Di sisi lain, dukungan komunitas untuk kebutuhan detection rules sudah sangat baik di bandingkan beberapa tahun lalu. Beberapa proyek open source sudah bermunculan dan terbukti memberikan sumbangsih yang besar ke peningkatan keamanan siber seperti sebut saja MITRE ATT&CK, Sigma Rules, Sysmon Rules, dll. Akan tetapi, sebagian besar proyek tersebut masih berfokus pada ancaman yang mengarah ke infrastruktur dibandingkan aplikasi, karena memang ancaman tersebut lebih mudah dipetakan, didokumentasikan, dan dibagikan kepada mereka yang memiliki infrastruktur yang serupa.

Saat ini, perkembangan aplikasi berjalan sangat cepat, sementara pengembangan kontrol keamanan siber yang menyertainya seringkali tertinggal. Teknologi cloud computing telah mempercepat proses pengembangan dan peluncuran aplikasi. Penting bagi tim keamanan untuk selalu berada di jalur yang sama dengan berbagai proses dan fitur yang ada dalam aplikasi.

Oleh karena itu maka perlu adanya sebuah framework atau standar yang dapat dijadikan referensi untuk pembuatan detection use cases dan penentuan detection requirements di level aplikasi. Hal ini tentu tidak mudah, tapi dengan penelitian yang mendalam dan dukungan dari komunitas keamanan siber yang besar, hal ini tidaklah tidak mungkin untuk dicapai.

2. Tinjauan Pustaka

Pada bagian ini kami akan membahas tentang dua framework utama yang menjadi landasan pembuatan framework Detection-oriented Modeling. Kedua framework tersebut adalah OWASP Threat Modeling dan Detection Engineering.

2.1 Threat Modeling Framework

Salah satu threat modeling framework yang terkenal dan banyak dipakai di berbagai organisasi adalah OWASP Threat Modeling [3]. OWASP Threat Modeling ini dibagi ke dalam beberapa proses, antara lain

- Decompose and Model the System

  Langkah pertama dalam proses threat modeling adalah dengan memahami bagaimana aplikasi tersebut bekerja dan berbagai fitur yang dimilikinya, termasuk juga bagaimana aplikasi tersebut berinteraksi dengan sistem atau aplikasi lain.

  Langkah berikutnya setelah memahami bagaimana aplikasi bekerja adalah melakukan identifikasi berbagai cara masuk untuk mengetahui potensi dimana ancaman dapat terjadi atau diarahkan

  Kemudian melakukan identifikasi terhadap berbagai aset, informasi atau area yang mungkin akan menjadi sasaran dari ancaman tersebut

  Langkah terakhir adalah dengan menentukan batas tingkat kepercayaan antara aplikasi dan pihak luar. Hal ini akan memungkinkan untuk memahami bagaimana aplikasi memperlakukan berbagai permintaan dari pihak luar baik yang resmi atau tidak.

  Hasil dari semua langkah diatas kemudian didokumentasikan kedalam sebuah diagram yang bernama Data Flow Diagram (DFD). DFD ini kemudian akan menjadi acuan utama untuk proses threat modeling berikutnya.

- Determine and Rank Threats

  Langkah berikutnya adalah menentukan berbagai ancaman yang mungkin terjadi atau diarahkan ke aplikasi. Proses ini sangat penting untuk memetakan dan memprioritaskan berbagai ancaman terhadap aplikasi.

  Beberapa framework lain dapat diikuti dalam menjalankan langkah ini seperti STRIDE (Spoofing, Tampering, Repudiation, Information Disclosure, Denial of Service, Elevation of Privilege) dari Microsoft [4], DREAD (Damage, Reproducibility, Exploitability, Affected Users, Discoverability) [5], dan ASF (Application Security Framework)[6].

  Setelah berbagai ancaman tersebut dipetakan, langkah selanjutnya adalah melakukan penilaian dan proses penentuan prioritas dari keseluruhan threats tersebut. Beberapa faktor dapat dilibatkan seperti potensi dampak yang mungkin ditimbulkan, kontrol keamanan saat ini, seberapa mudah ancaman tersebut dilakukan, dan lain-lain.

- Determine countermeasures and mitigation

  Setelah semua ancaman dipetakan dan dibuat daftar prioritasnya berdasarkan resiko tertinggi sampai yang terendah, maka langkah selanjutnya adalah dengan memetakan berabagai langkah mitigasi untuk menghadapi ancaman tersebut. Tentu saja dimulai dari ancaman dengan tingkat resiko paling tinggi berdasarkan penilaian pada proses sebelumnya.

Pada praktiknya, tidak semua ancaman dapat dimitagasi atau akan dimitigasi, terdapat begitu banyak pertimbangan sebelum memutuskan apakah sebuah resiko akan dimitigasi atau tidak. Salah satu faktor yang paling sering muncul dan menjadi faktor penentu adalah dampak ancaman tersebut terhadap bisnis secara langsung.

Jika resiko dan dampaknya masih bisa diterima maka ancaman tersebut kemungkinan besar tidak akan dimitigasi.

Jika resiko dan dampaknya cukup besar, maka langkah-langkah mitigasi perlu dilakukan. Langkah mitigasi ini biasanya hanya mencakup sebagian perlindungan saja karena resiko yang dianggap tidak terlalu besar sehingga langkah mitigasi akan berhenti sampai resiko berada pada level yang dapat diterima.

Yang terakhir adalah jika resiko dan dampak yang dapat ditimbulkan akan menjadi sangat besar, maka ancaman harus dihilangkan sama sekali. Hal ini biasanya terkait vulnerability dengan tingkat keparahan kritikal yang harus segera di perbarui untuk menghilangkan vulnerability tersebut.

Tiga langkah tersebut diatas merupakan langkah atau proses threat modeling secara garis besar yang terjadi di setiap organisasi atau perusahaan. Pada praktiknya, setiap perusahaan atau organisasi memiliki kebebasan dalam menentukan langkah-langkah threat modeling yang ingin mereka ikuti dan terapkan.

Dan di setiap proses atau framework pasti ada kelebihan dan kekurangannya masing-masing. Adapun beberapa kelebihan dari proses OWASP Threat Modeling framework ini antara lain seperti

- Identifikasi ancaman yang mungkin terjadi di level aplikasi lebih dini
- Memprioritaskan ancaman berdasarkan dampak dan probabilitasnya
- Identifikasi kontrol keamanan yang diperlukan untuk mengurangi resiko ancaman
- Dokumentasi ancaman dan kontrol keamanan, sehingga dapat digunakan sebagai referensi di masa depan

Dan beberapa kekurangan yang penulis temui antara lain seperti

- Kurangnya penjelasan dan kontrol di sisi deteksi, dimana hampir semua kontrol berfokus pada pencegahan dan mitigasi ancaman. Padahal jika di kemudian hari kontrol pencegahan tersebut gagal, maka kontrol deteksi setidaknya masih akan dapat mengidentifikasi ancaman tersebut.
- Kurang adanya penekanan terhadap standar logging yang baik terutama untuk kebutuhan deteksi ancaman. Ketika ancaman sudah berhasil dimitigasi maka logging tidak lagi menjadi relevan, padahal untuk melakukan deteksi, dibutuhkan banyak sekali informasi dari fitur logging tersebut.

Hal ini lah yang kemudian mendasari perumusan framework ini untuk dikembangkan lebih lanjut. Akan tetapi sebelum membahas framework Detection-oriented Modeling alangkah lebih baiknya kita pelajari terlebih dahulu juga proses yang ada pada Detection Engineering saat ini.

2.2 Detection Engineering Framework

Seperti telah dijelaskan diawal bahwa Detection Engineering merupakan sebuah proses end to end dalam menentukan berbagai potensi ancaman yang mungkin dapat di deteksi di dalam lingkungan kita.

Secara garis besar, proses Detectiong Engineering ini dibagi ke dalam tiga fase yakni perencanaan, pengembangan, dan pemantauan dan peningkatan berkelanjutan. Untuk mempermudah dan memberikan gambaran yang lebih baik, setiap fase tersebut akan dipecah ke dalam beberapa proses lagi.

Dan berikut adalah contoh proses yang ada pada Detection Engineering yang diambil dari buku Practical Detection Engineering [1]

- Requirement Discovery

  Langkah pertama dari proses detection engineering adalah menentukan prasyarat deteksi. Hasil dari langkah pertama ini dapat berupa Input dan Output. Dimana Inputnya merupakan berbagai temuan dari sumber prasyarat deteksi. Sementara untuk outputnya berupa berbagai informasi lain yang diolah dari Input untuk kemudian diteruskan ke proses selanjutnya

  Beberapa contoh informasi yang bisa dimasukkan sebagai Output antara lain seperti

  - Asal deteksi, bisa berupa individu, tim atau dari proses yang terjadi di tempat lain
  - Deskripsi, apa yang perlu di deteksi, bisa berupa deskripsi yang cukup teknis atau masih lebih umum termasuk juga informasi pendukung dari sumber deteksi, misal seperti investigasi yang sudah dilakukan
  - Alasan kenapa perlu dibuatkan detection rules. Informasi ini nantinya akan membantu dalam proses triage ketika alarm terpicu untuk menentukan seberapa penting alarm tersebut
  - Pengecualian. Dalam kondisi seperti apa alarm tersebut dapat diabaikan
  - Ruang lingkup, menentukan dimana deteksi tersebut dapat diaplikasikan
  - Bukti. Bisa terdiri dari file malware, logs, PCAP, atau informasi lain yang bisa digunakan kemudian untuk melakukan testing terhadap deteksi

  Sementara untuk Input, dapat menggunakan beberapa sumber berikut ini seperti

  - Threat Intelligence (Internal/External TI, OSINT, ISACs, dll)
  - Business security requirements
  - Security operations center requests

- Red/Purple/Threat Hunting Team exercises
- Continuous improvement activities

- Triage

  Ketika jumlah deteksi yang ada pada fase sebelumnya sudah mulai banyak dan menumpuk, maka proses pada tahap ini menjadi penting untuk memilah dan memprioritaskan deteksi mana yang akan diproses untuk menuju ke fase selanjutnya. Input dari proses ini adalah detection requirement dari fase sebelumnya. Sementara ouputnya adalah daftar detection requirement yang sudah dipilah dan diprioritaskan

  Beberapa faktor dapat dipertimbangkan dalam penentuan skala prioritas untuk setiap deteksi, antara lain seperti

  - Tingkat severity dari ancaman
  - Relevansi ancaman dengan organisasi
  - Cakupan deteksi saat ini (tidak ada sama sekali, sebagian, atau sudah terpasang)
  - Ketersediaan exploits di luar sana

- Investigate

  Fase ini bertujuan untuk mempersiapkan deteksi yang sebelumnya sudah dipersiapkan agar lebih matang sebelum masuk ke fase pengembangan. Pada fase ini juga diharapkan dapat mengidentifikasi adanya gap atau celah terkait misalnya sumber deteksi, atau masalah lain yang dapat mempengaruhi proses pengembangan selanjutnya. Input dari fase ini adalah daftar deteksi dari proses sebelumnya. Sementara outputnya adalah berupa detections requirement yang lebih details dan lebih teknis dibanding pada fase sebelumnya yang masih lebih umum.

  Terdapat empat fase utama pada proses Investigate ini, antara lain

  - Identify the data source

    Pada fase ini, tim detection engineer perlu memahami detection requirement secara mendalam sehingga dapat memetakan kebutuhan data sources apa saja yang nantinya akan dibutuhkan. Kerjasama dengan tim lain juga diperlukan untuk memahami data source yang mungkin jarang diketahui atau digunakan oleh tim.

    Identifikasi terhadap gap informasi juga perlu dilakukan untuk meminimalisasi masalah yang akan datang, seperti contoh seperti kurangnya informasi yang terkandung di dalam logs, padahal informasi tersebut dibutuhkan untuk

pengembangan deteksi.

- Determine detection indicator types

  Pada tahap ini dilakukan identifikasi terhadap jenis indikator yang akan dipakai untuk pembuatan deteksi. Beberapa jenis indikator yang sering digunakan antara lain seperti Static, Behavioral, Statistical, dan Hybrid. Jenis indikator ini akan mempengaruhi tingkat pembuatan deteksi dan tingkat akurasi deteksi.

- Research

  Tahap ini dilakukan untuk memperkaya informasi terkait deteksi, beberapa hal lagi dilakukan pada tahap ini seperti identifikasi contoh ancaman yang pernah terjadi di masa lampau terkait deteksi ini, identifikasi deteksi yang mungkin serupa dan sudah ada pada sistem, atau deteksi yang sebenarnya hanya perlu diperbarui saja tanpa perlu membuat deteksi baru, identifikasi variasi ancaman yang mungkin dapat terjadi dan memetakan teknik mana yang mungkin dapat dilakukan dan mana yang tidak

- Establish validation criteria

  Pada fase ini dilakukan validasi dan pemetaan tentang bagaimana deteksi dapat diuji sebelum deteksi tersebut dikembangkan. Terkadang jika teknik ancaman sudah diketahui, maka akan lebih mudah untuk melakukan pengujian terhadap deteksinya. Akan tetapi jika ancamannya masih sebatas teori maka dibutuhkan identifikasi dan proses lebih lanjut untuk menentukan bagaimana deteksi tersebut dapat diuji dan divalidasi

- Develop

  Fase berikutnya adalah fase pengembangan dimana pada fase ini, semua detail teknis pada fase sebelumnya akan digunakan sebagai input untuk mulai melakukan proses pengembangan. Output dari fase adalah dapat berupa detection rule atau detection code, tergantung dari bagaimana deteksi tersebut akan diimplementasikan nantinya.

- Test

  Fase ini dilakukan sebagai tahap pengujian terhadap detection rule sebelum detection rule tersebut dipindahkan ke tahap produksi. Fase ini juga menjadi tahap yang penting untuk mengidentifikasi adanya potensi false-positive dan bagaimana false-positive tersebut dapat dikurangi seminimal mungkin.

Idealnya, proses pengujian ini harus dilakukan secara terus menerus dan tidak hanya sekali setelah proses pengembangan saja karena hasil dari pengujian ini dapat menjadi masukan untuk peningkatan berkelanjutan terhadap kualitas deteksi itu sendiri.

- Deploy

  Tujuan dari fase ini adalah mengambil deteksi dari hasil pengujian dan memindahkannya ke fase produksi. Salah satu hal yang dapat dilakukan pada fase ini adalah dengan mengimplementasikan fitur tagging untuk mengidentifikasi tingkat kematangan dari suatu deteksi. Contoh tags yang dapat dipakai seperti experimental, test, dan stable.

Sebuah detection rule idealnya akan melalui proses yang panjang sebelum akhirnya dapat diimplementasikan pada lingkungan produksi, dan framework detection engineering ini sudah berhasil mencakup hampir keseluruhan proses yang dibutuhkan untuk pengembangan deteksi dari nol sampai akhir.

Akan tetapi framework ini bukanlah tanpa celah, pasti ada kekurangan dan kelebihan dari framework ini sehingga kami memiliki pemikiran untuk kemudian meningkatkan beberapa proses dari framework ini. Berikut adalah beberapa kelebihan dan kekurangan dari Detection Engineering framework.

Kelebihan Detection Engineering Framework

- Bisa dikatakan bahwa detection engineering sedang mengalami masa bulan madunya di komunitas cyber security dimana detection engineering memiliki banyak sekali pengikut dan dibicarakan dimana-mana. Hal ini membuat pengembangan framework ini menjadi begitu cepat.
- Karena kebaruan framework ini, maka bisa dikatakan bahwa tidak ada benar dan salah dalam penerapan framework ini, hal ini yang kemudian membuat framework ini begitu fleksibel dalam penerapannya di berbagai organisasi.

Kekurangan Detection Engineering Framework

- Framework yang masih baru selain dapat menjadi keuntungan juga sebenarnya dapat menjadi kekurangan karena belum adanya standardisasi yang jelas sehingga menyusahkan bagi para engineer baru yang baru mulai mengadopsinya.
- Tidak dapat dipungkiri juga dalam penerapannya framework ini masih terlalu terfokus dengan berbagai ancaman yang mengarah ke infrastruktur. Referensi-referensi yang digunakan juga sangat mengarah kesana seperti contohnya berbagai TTPs pada MITRE ATT&CK Framework.

- Hal tersebut lah yang kemudian membuat framework ini kurang begitu berkembang jika dihadapkan pada penerapan yang lebih ke arah ancaman di sisi aplikasi
- Framework ini juga terlalu sering mengambil referensi terhadap berbagai TTPs (Tactics, Techniques, dan Procedures) yang sudah ada seperti yang terkandung pada MITRE ATT&CK Framework. Hal ini dapat menjadi kekurangan karena detection engineer hanya berfokus pada TTPs yang sudah teridentifikasi saja, padahal berbagai TTPs lain mungkin saja terjadi dan menjadi ancaman nyata dan TTPs tersebut belum dipetakan sepenuhnya pada MITRE ATT&CK framework.

3. Detection-oriented Modeling Framework

Berawal dari adanya gap antara kedua framework tersebut dan kami melihat bahwa terdapat kesempatan untuk dapat mensinergikan keduanya maka akhirnya kami mencoba membuat framework turunan dari kedua framework tersebut dengan harapan bahwa akan ada lebih banyak ancaman yang berhasil di deteksi di level aplikasi.

Selain itu kami juga memiliki beberapa tujuan lain dalam pengembangan framework ini.

3.1 Tujuan Pembuatan Detection-oriented Modeling Framework

1. Peningkatan Visibilitas

   Visibilitas ini sangatlah penting dalam proses pemantauan yang terjadi pada Security Operation Center. Pada umumnya, tim SOC lebih berfokus pada ancaman yang terjadi di level infrastruktur dan cukup jarang melakukan pemantauan yang komprehensif terhadap aplikasi. Dengan adanya framework ini diharapkan lebih banyak deteksi akan tercipta untuk aplikasi-aplikasi yang digunakan oleh organisasi baik yang dibuat sendiri ataupun dari pihak ketiga. Dengan *visibility* yang baik, pemilik aplikasi dapat mengamati interaksi pengguna dengan detail, memungkinkan untuk mendeteksi aktivitas mencurigakan atau bahkan serangan yang belum pernah terjadi sebelumnya (*0-day*). Ini berarti aplikasi tidak hanya melihat apa yang biasanya diharapkan, tetapi juga dapat mengidentifikasi perilaku yang tidak normal dan memiliki potensi berbahaya.

2. Integrasi dengan proses CI/CD

   Integrasi dan sinergi menjadi kunci dalam pembuatan deteksi di level aplikasi. Proses detection engineering tidak dapat berdiri sendiri dan mengabaikan proses CI/CD yang umum dipakai dalam pengembangan aplikasi. Pada era modern seperti sekarang ini, banyak sekali organisasi telah mengimplementasi proses CI/CD yang menyeluruh untuk pengembangan aplikasi mereka. Hal ini membuat lebih mudah untuk mengimplementasikan kontrol keamanan (pencegahan dan deteksi) dan memastikan

bahwa setiap perubahan kode yang dilakukan oleh tim pengembang sudah diperiksa secara otomatis, sehingga mengurangi risiko kerentanan yang tidak terdeteksi.

3. Kewenangan tambahan untuk tim

   Dengan melibatkan tim SOC atau tim detection engineer ke dalam proses pengembangan aplikasi maka tim SOC akan memiliki kewenangan lebih besar untuk mempengaruhi proses pengembangan itu sehingga lebih menguntungkan di sisi kontrol keamanannya dimana selama ini tim SOC menjadi tim paling terakhir yang mengetahui adanya perubahan di sisi aplikasi. Dengan framework ini diharapkan tim SOC/Detection Engineer dapat mendefinisikan dan memberikan arahan dalam penentuan informasi apa saja yang perlu terkandung di dalam logs untuk kebutuhan deteksi.

4. Peningkatan kualitas deteksi dan alarm pada SOC

   Dengan melalui seluruh proses yang ada framework ini diharapkan kualitas deteksi akan menjadi lebih baik sehingga dapat mengurangi false-positive pada proses pemantauan di dalam Security Operation Center. Selain itu hal ini akan meningkatkan kualitas pemantauan secara keseluruhan karena hanya alarm yang memang layak untuk diinvestigasi lebih lanjut saja yang akan terpicu oleh sistem.

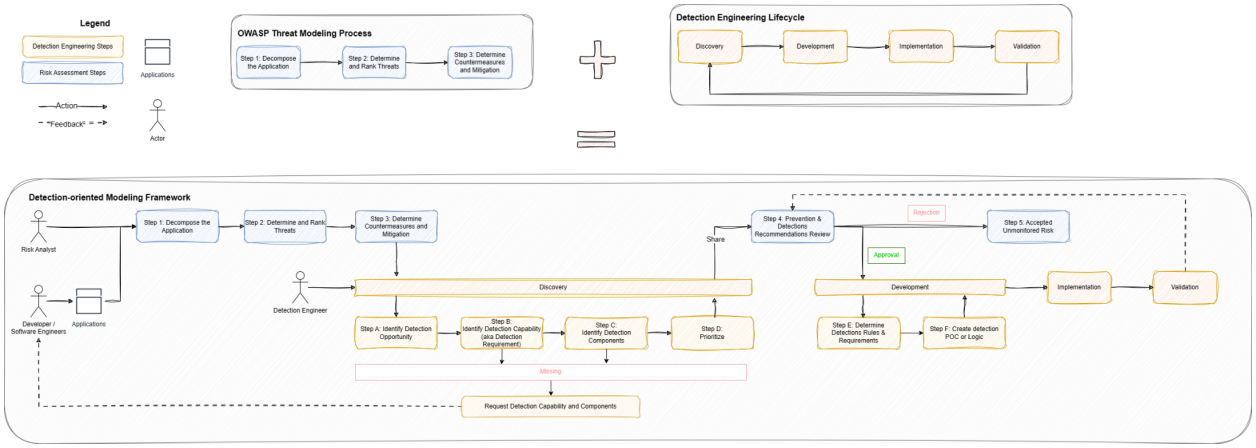
3.2 Detection-oriented Modeling Framework (DOMF)

Berikut adalah usulan kami terkait peningkatan dan sinergi dari dua framework berbeda yakni OWASP Threat Modeling dan Detection Engineering yang kemudian kami beri nama Detection-oriented Modeling Framework atau disingkat DOMF.

3.2.1 Usulan Framework

3.2.1.1 Visualisasi DOMF

Berikut adalah tampilan visual secara garis besar dari keseluruhan proses yang ada pada DOMF agar lebih mudah dipahami dan dilihat keterkaitannya kembali dengan framework asalnya.



3.2.1.2 Penjelasan DOMF

Framework ini merupakan kombinasi dari OWASP threat modeling dan detection engineering, dimana keduanya merupakan framework yang paling dikenal dan diakui di bidanngya masing-masing. Pada bab ini kami akan menggunakan penomoran angka untuk proses yang terkait dengan threat modeling dan penomoran huruf untuk proses yang terkait dengan detection engineering agar memudahkan dalam membedakannya saja. Pada akhirnya keseluruhan proses dapat dilihat sebagai Detection-oriented Modeling Framework.

Step 1: Decompressing the Application

Dalam langkah ini, tujuan utamanya adalah memahami secara mendalam bagaimana aplikasi tersebut bekerja dan berinteraksi. Hal ini mencakup mengidentifikasi berbagai komponen, fungsi, dan interaksi dalam aplikasi. Dengan pemahaman aplikasi yang baik, tim pengembangan deteksi dapat mengidentifikasi potensi kerentanan dan risiko yang terkait dengan berbagai komponen dan proses dalam aplikasi tersebut.

Step 2: Determining and Ranking Threats

Setelah memahami aplikasi tersebut, tim pemodelan ancaman akan mengidentifikasi berbagai ancaman yang mungkin timbul. Ancaman-ancaman ini dapat mencakup serangan peretasan, pelanggaran data, gangguan layanan, dan lain sebagainya. Ancaman-ancaman ini kemudian akan diberi peringkat berdasarkan potensi dampak dan kemungkinannya. Hal ini membantu tim untuk lebih fokus pada ancaman dengan potensi kerentanan yang lebih tinggi.

Step 3: Determining Prevention and Mitigation Actions

Langkah ketiga melibatkan identifikasi tindakan pencegahan dan mitigasi yang diperlukan untuk mengatasi ancaman yang telah diidentifikasi. Ini termasuk pengembangan strategi dan metode untuk mengurangi risiko dari ancaman tersebut. Tim akan merancang sistem keamanan dan kontrol, termasuk implementasi mekanisme

keamanan seperti enkripsi, otentikasi, dan pemantauan, untuk mengurangi potensi kerentanan.

Step A: Identifying Detection Opportunities

Data yang dikumpulkan dalam langkah 1 dan 2 digunakan untuk mengidentifikasi peluang deteksi, juga dikenal sebagai hipotesis. Berdasarkan skenario serangan yang terkumpul, detection engineer dapat mengidentifikasi potensi deteksi menggunakan data dan log tertentu secara umum. Peluang-peluang ini ditempatkan secara strategis sesuai dengan ancaman dan kerentanan yang akan diprioritaskan, memudahkan implementasi sistem pemantauan dan peringatan yang ditargetkan untuk memperkuat pertahanan aplikasi terhadap berbagai potensi ancaman siber.

Step B: Identifying Detection Capability (Requirements)

Memastikan kemampuan deteksi yang kuat adalah hal yang sangat penting untuk langkah-langkah keamanan yang efektif. Berikut adalah beberapa aspek kunci yang perlu dipertimbangkan saat mempersiapkan dan mengidentifikasi kemampuan yang diperlukan dalam proses pembuatan deteksi:

1. **Data Sources**: Mulailah dengan mengidentifikasi sumber-sumber data dan informasi yang akan menjadi kunci dalam pembuatan deteksi, yang mungkin mencakup Log Otentikasi, Log Admin, dan log lain yang terkait dengan aplikasi web. Memahami dari mana data dan informasi yang relevan berasal adalah hal dasar dalam pengembangan deteksi yang baik.

2. **Technology Support**: Evaluasi apakah para detection engineer memiliki akses ke teknologi yang diperlukan, seperti sistem Security Information and Event Management (SIEM). SIEM dapat mengumpulkan dan menganalisis log dari berbagai sumber, dan membantu dalam deteksi insiden. Memiliki infrastruktur teknologi yang tepat sangat penting untuk pemantauan dan peringatan dini yang efektif.

3. **Data Quantity and Quality**: Evaluasi kembali apakah informasi terkait data sources yang dikumpulkan sudah cukup, baik dalam hal kuantitas maupun kualitas. Pertimbangkan faktor-faktor seperti, waktu retensi data, dan cakupan (memantau semua pengguna atau berfokus pada pengguna tertentu, misalnya). Menyeimbangkan antara kualitas dan kuantitas data source sangat penting untuk menghindari deteksi akan membebani sistem SIEM di masa depan.

4. **Data Preparation and Engineering**: Pastikan bahwa data yang dikumpulkan telah dipersiapkan dan diolah dengan baik untuk analisis lebih lanjut. Proses ini melibatkan implementasi skema secara umum yang kemudian dipakai untuk mendefinisikan atribut-atribut data. Sebagai contoh, menggabungkan data dari atribut berbeda seperti IPAddress, IPv4, dan IP ke dalam satu atribut yang

disebut UserIP dapat meningkatkan konsistensi dan kemudahan analisis. Menggunakan format log yang terstandarisasi seperti JSON atau XML juga dapat menyederhanakan pemrosesan dan analisis data di masa yang akan datang.

5. **Human Resources Requirement**: Make sure that your detection and response team has the necessary skills and sufficient personnel to manage complex monitoring and analysis tasks. Effective security monitoring requires personnel trained to identify threats and respond quickly. Also, ensure regular training to keep your team's skills up to date in dealing with rapidly evolving threats. This includes providing specialized training on the latest detections.

   Pastikan bahwa tim deteksi dan respons memiliki keterampilan yang diperlukan dan jumlah personel yang cukup untuk mengelola tugas-tugas pemantauan dan analisis yang kompleks. Proses pemantauan yang efektif memerlukan personel yang terlatih untuk mengidentifikasi ancaman dan merespons dengan cepat dan tepat. Selain itu, lakukan pelatihan secara terus menerus untuk mengasah kemampuan tim dan personel, termasuk juga pelatihan yang mencakup informasi atau kemampuan dalam mendeteksi ancaman-ancaman baru yang sudah diidentifikasi.

Dengan mempertimbangkan berbagai komponen diatas yang mencakup kemampuan deteksi, organisasi kemudian dapat membangun dasar yang kuta untuk proses pemantauan dan deteksi ancaman siber yang lebih efektif dan efisien.

Step C: Identifying Detection Components (Indicators)

Untuk mendeteksi berbagai Taktik, Teknik, dan Prosedur (TTP) yang digunakan oleh penyerang, tim deteksi akan memerlukan data-data khusus, yang sering disebut sebagai indikator. Data-data ini kemudian dapat dirujuk pada Pyramid of Pain yang dicetus oleh David Bianco [8], terutama pada empat tingkat terbawah dan seterusnya. Oleh karena itu, data-data seperti Alamat IP, Nama Pengguna, dan elemen-elemen penting lainnya dalam aktivitas aplikasi web sangatlah penting. Setiap aktivitas yang dipelajari dari langkah pertama akan memperkenalkan berbagai jenis indikator.

Contohnya, ketika membahas otentikasi, terutama login, elemen-elemen seperti alamat IP, nama pengguna, hasil aktivitas (berhasil atau gagal), dan status MFA merupakan data indikator yang baik. Di sisi lain, ketika berbicara tentang proses transaksi, mungkin perlu untuk mencari informasi lebih lanjut seperti mata uang yang digunakan, jumlah transaksi, atau bahkan kode diskon yang digunakan. Selain itu, indikator-indikator kontekstual seperti perilaku yang tidak biasa atau aktivitas mencurigakan, dan percobaan akses yang tidak sah juga perlu dipertimbangkan untuk dimasukkan sebagai indikator tambahan untuk keperluan deteksi

Dengan pengembangan dan pelatihan secara terus menerus terkait berbagai TTPs baru yang mungkin digunakan oleh penyerang akan sangat membantu dalam proses pengembangan deteksi dan respons terhadap berbagai ancaman siber.

Step D: Detection Prioritization

Memprioritaskan deteksi adalah langkah yang penting dalam proses identifikasi deteksi mana yang harus menjadi fokus utama dalam proses pengembangannya nanti. Beberapa faktor yang dapat dipertimbangkan dalam fase ini antara lain seperti:

- **Threat Severity**: Tingkat ancaman seharusnya menjadi pertimbangan utama. Ancaman dengan resiko dan dampak yang lebih tinggi pada organisasi akan mendapatkan prioritas utama dalam pengembangannya.
- **Org/Business Alignment**: Penting untuk mempertimbangkan sejauh mana deteksi tersebut sesuai dengan tujuan organisasi dan bisnis, termasuk perlindungan aset penting (crown jewels), toleransi risiko, dan kebijakan keamanan yang relevan.
- **Detection Availability**: Ketersediaan deteksi saat ini, baik melalui vendor SIEM, solusi open source seperti SIGMA, atau sumber deteksi lainnya, juga harus dipertimbangkan.
- **Active Exploit**: Ancaman yang saat ini aktif dieksploitasi di luar sana atau memiliki urgensi tinggi bisa mendapat prioritas yang lebih tinggi, karena potensi ancamannya lebih besar.
- **Detection Capability**: Evaluasi kemampuan deteksi, termasuk sejauh mana deteksi tersebut dapat mengenali taktik, teknik, dan prosedur (TTP) yang digunakan oleh penyerang.
- **Cost of Investment**: Pertimbangkan biaya investasi, termasuk biaya waktu implementasi, biaya waktu investigasi, dan biaya lain yang terkait dengan implementasi dan pemeliharaan deteksi.
- **False Positive Rate**: Tingkat false-positive juga sebaiknya dipertimbangkan, karena deteksi yang menghasilkan terlalu banyak alarm palsu dapat mengganggu operasional proses pemantauan dan mengurangi efektivitas.
- **Historical Data**: Lakukan pengkajian ulang terhadap data historis untuk mengidentifikasi tren dan pola serangan di masa lampau yang mungkin dapat menjadi masukan untuk proses penentuan prioritas

Dengan pertimbangan yang cermat, organisasi dapat mengembangkan deteksi yang sudah diprioritaskan dengan lebih efektif untuk mengatasi ancaman keamanan yang akan datang di masa depan

Step 4: Prevention & Detection Recommendations

Langkah berikutnya dari OWASP threat modeling adalah menyusun rekomendasi untuk tindakan pencegahan, penanggulangan, dan juga deteksi. Rekomendasi deteksi sangat jarang ditemui atau bahkan tidak ada pada proses threat modeling. Rekomendasi ini

berfungsi sebagai panduan bagi tim pengembang atau developer untuk meningkatkan keamanan aplikasi secara keseluruhan.

- Jika rekomendasi diterima dan diimplementasikan oleh pengembang, maka detection engineer akan melanjutkan prosesnya ke langkah E dibawah.
- Jika rekomendasi ditolak seluruhnya atau sebagian, maka perlu dibuatkan dokumentasi terpisah untuk menyatakan bahwa organisasi telah bersedia untuk menerima segala resiko di kemudian hari dari penerimaan resiko tersebut. Proses selanjutnya kemudian dapat dilihat pada langkah nomor 5 dibawah.

Step E: Determining Detection Rules & Requirements

Langkah ini sudah dijelaskan dengan cukup baik pada subbab sebelumnya di bagian Investigate

Step F: Creating Proof of Concept (POC) of Detection Logic

Langkah ini sudah dijelaskan dengan cukup baik pada subbab sebelumnya di bagian Develop dan Test

Step 5: Accepting Unmonitored Risk

Proses penerimaan resiko adalah ketika terdapat resiko yang telah diidentifikasi akan tetapi telah diterima oleh organisasi untuk tidak dilakukan tindakan mitigasi atau pencegahan terhadap ancaman yang mungkin terjadi. Resiko-resiko dapat muncul karena biaya atau kompleksitas dalam mitigasi resiko tersebut terlalu tinggi atau bisa juga karena resiko dianggap terlalu rendah sehingga masih dapat diterima dan proses pemantauan secara aktif tidak diperlukan. Meskipun resiko-resiko tersebut telah diterima, sangat penting untuk tetap memiliki dokumetasi yang menyeluruh dan lengkap sehingga dapat dijadikan acuan di masa depan jika terjadi hal-hal yang tidak diinginkan.

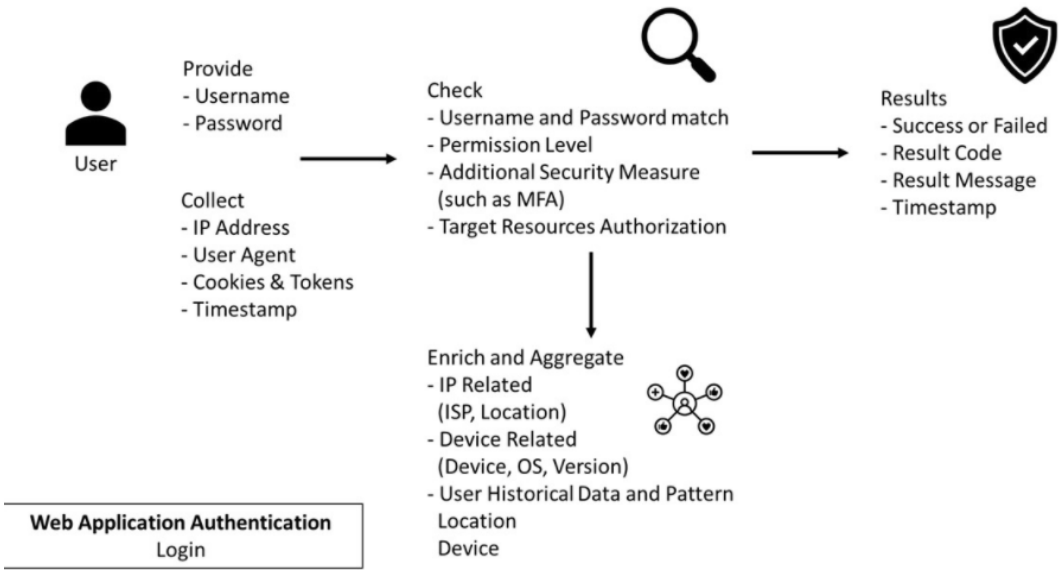
Dengan mengikuti berbagai langkah diatas melalui detection-oriented modeling framework, diharapakan organisasi dapat lebih proaktif dalam mengidentifikasi berbagai resiko di aplikasinya dan dapat menerapkan berbagai strategi pencegahan, mitigasi, dan juga deteksi sehingga aplikasi tersebut dapat lebih kuat dan dalam bertahan dari berbagai ancaman di masa depan sehingga tetap dapat melindungi berbagai informasi penting yang terkandung di dalamnya.

3.2.2 Contoh Penerapan DOMF

3.2.2.1 Fitur Login pada Aplikasi Web

Step 1: Decompressing the App

Berikut adalah contoh hasil dekompresi fitur login pada aplikasi web



Step 2: Determining and Ranking Threats

Langkah berikutnya adalah menentukan berbagai ancaman yang mungkin dapat terjadi pada fitur login di aplikasi web

Threat: Compromised Credentials

Pengguna aplikasi dapat menjadi korban dari ancaman ini dan mereka dapat menjadi korban dari berbagai macam teknik seperti phishing, penggunaan password yang sama pada berbagai platform, kebocoran informasi, dan lain-lain.

Step 3: Determining Prevention and Mitigation

Prevention and Mitigation: Compromised credentials cukup susah untuk dicegah karena terdapat terlalu banyak faktor yang dapat menjadi penyebabnya. Salah satu langkah strategis yang dapat diimplementasikan adalah dengan memaksa pengguna untuk menggunakan Multi Factor Authentication (MFA) sebagai langkah verifikasi tambahan saat mereka login ke aplikasi.

Step A: Identifying Detection Opportunities

Opportunity for Detection: Saat melakukan aktivitas login, pengguna akan menghasilkan berbagai macam informasi dan informasi tersebut akan tercatat pada logs aplikasi. Informasi tersebut kemudian dapat dimanfaatkan untuk kebutuhan deteksi ketika ada indikasi aktivitas mencurigakan atau tidak normal berdasarkan data historis pengguna tersebut.

Contoh deteksi lain yang dapat diimplementasikan meliputi

Impossible Travel: mendeteksi pengguna ketika melakukan aktivitas login dari dua lokasi yang berbeda dalam waktu yang hampir berdekatan. Berdasarkan kalkulasi jarak antara kedua lokasi dan perbedaan waktu yang terjadi dari aktivitas pertama dan kedua, hal ini bisa dikatakan tidak mungkin, sehingga perlu di deteksi dan dapat menjadi indikasi telah terjadinya compromised credentials di sisi pengguna.

Step B: Identifying Detection Capability (Requirements)

Detection Capability Requirements: Untuk kebutuhan deteksi yang efektif, informasi yang lengkap dari aktivitas login akan dibutuhkan. Untuk mengimplementasikan ini setidaknya dibutuhkan perangkat yang dapat mendukung seperti Security Information and Event Management (SIEM), User and Entity Behavior Analytics (UEBA), atau sebuah Big Data platform yang mampu menyimpan dan mengolah logs setidaknya dalam kurun waktu 7 hari kebelakang atau lebih.

Deteksi jenis ancaman yang cukup umum biasanya sudah ada pada SIEM seperti ancaman bruteforce dan serangan SQL Injection. Meskipun begitu tetap perlu dilakukan penilaian dan identifikasi kembali apakah deteksi bawaan pada SIEM telah mencakup aplikasi baru yang akan dikembangkan ini, atau apakah sistem deteksinya memiliki cakupan untuk mengurai informasi penting dari logs aplikasi.

Step C: Identifying Detection Components (Indicators)

Detection Components (Indicators): Beberapa komponen penting yang dapat dipertimbangkan untuk kebutuhan deteksi meliputi:

- **Username**: dibutuhkan untuk mengidentifikasi pengguna yang terdampak atau menjadi target dari ancaman
- **Timestamp**: dibutuhkan untuk memberikan gambaran waktu kejadian menggunakan waktu universal atau waktu lokal.
- **IP Address and User Agent**: digunakan untuk mendeteksi adanya anomali dengan diperkaya berbagai informasi tambahan seperti nama ISP, lokasi alamat IP, jenis sistem operasi dan versinya, dan lain-lain.
- **Result Code and Message**: dibutuhkan untuk mengetahui apakah aktivitas login yang dilakukan berhasil atau gagal.
- **Session information**: dibutuhkan untuk mengidentifikasi sesi aktif yang dimiliki oleh pengguna, ada berapa jumlah sesi aktif, untuk kemudian dikombinasikan dengan informasi pendukung lain.

Step D: Detection Prioritization

Ketika fitur Multi Factor Authentication sudah berhasil diimplementasikan dan diadopsi sepenuhnya oleh pengguna maka akan terdapat beberapa perubahan dalam sisi deteksi yang dapat diimplementasikan.

Ancaman bruteforce yang menargetkan username dan password pengguna dapat dikesampingkan untuk sementara waktu atau diberikan nilai resiko lebih kecil karena setiap pengguna telah menerapkan MFA. Meskipun username dan password telah berhasil ditebak oleh penyerang, penyerang masih perlu untuk mendapatkan kode MFA untuk dapat login ke dalam aplikasi

Deteksi terhadap Impossible Travel masih dapat dilakukan akan tetapi dapat lebih berfokus pada aktifitas login yang berhasil dan session yang telah terbentuk. Hal ini untuk menghindari adanya kemungkinan banyaknya alarm false-positive yang mungkin akan terpicu karena deteksi hanya melihat alamat IP dari pengguna. Penggunaan MFA setelah login yang berhasil juga perlu dilihat untuk memastikan bahwa pengguna telah memasukkan kode MFA yang sesuai.

Step E: Determining Detection Rules and Requirements

Detection Rules and Requirements:

- Jika seorang pengguna berhasil login dari perangkat atau sistem operasi baru, maka sebuah alarm baru dapat dipicu untuk aktivitas login yang tidak biasa dan langkah investigasi lebih lanjut dapat segera dimulai terhadap pengguna tersebut
- Jika seorang pengguna terdeteksi melakukan aktivitas login yang berhasil dari dua tempat berbeda atau memiliki dua session yang aktif secara bersamaan dari dua lokasi berbeda maka deteksi Impossible Travel dapat dipicu yang mengindikasikan adanya kegiatan login yang tidak normal.
- Penggunaan sistem machine learning dan pemrosesan statistik tingkat tinggi juga dapat dipertimbangkan untuk melakukan baselining terhadap aktivitas login pengguna untuk kemudian memicu alarm ketika terdapat aktivitas yang diluar standar/baselinenya.

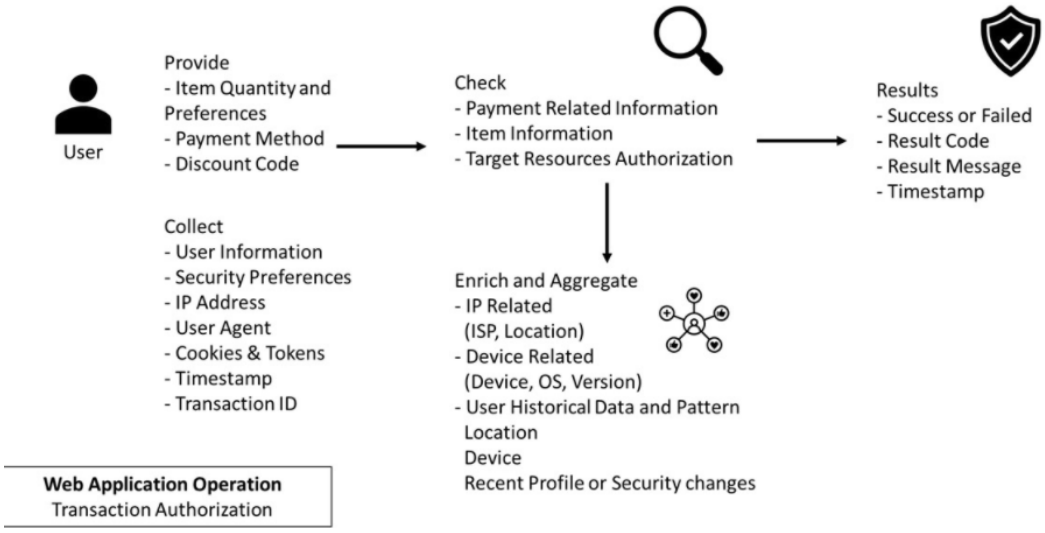
Step F: Creating POC of Detection Logic

Description: Proses berikutnya adalah pembuatan Proof of Concept (PoC) untuk logika deteksi yang akan dikembangkan. Implementasinya kemudian dapat berbeda-beda tergantung dari sistem dimana deteksi tersebut akan diimplementasikan, seberapa kompleks logika deteksinya, dan lingkungan atau aset pendukung apa saja yang dibutuhkan.

3.2.2.2 Fitur Transaksi pada Aplikasi Web

Step 1: Decompressing the App

Diagram berikut menjelaskan bagaimana cara kerja aplikasi yang akan menjalani proses detection-oriented modeling



Step 2: Determining and Ranking Threats

Threat: Unauthorized Modification of Transaction Data (Changing Amount Paid by Modifying Price Value)

Description: Dalam alur transaksi di dalam sebuah aplikasi web, penyerang biasanya akan mencoba untuk melakukan modifikasi terhadap data transaksi untuk keuntungan pribadi. Data yang dikirimkan ke server akan dibuat berbeda dengan data yang pertama kali dimasukkan oleh user.

Step 3: Determining Prevention and Mitigation

Prevention and Mitigation: Untuk mencegah jenis ancaman ini berhasil dilakukan maka perlu dilakukan pengecekan terhadap data yang diterima oleh user dengan data-data lain terkait informasi transaksi tersebut seperti harga, jumlah, kode diskon, dan lain-lain. Ketika diketahui terdapat perbedaaan, maka proses reset harus dijalankan untuk menolak transaksi yang dilakukan, dan aktivitas seperti ini harus di catat, di pantau dan di investigasi secara terus menerus.

Step A: Identifying Detection Opportunities

Normalnya data yang digunakan untuk melakukan validasi transaksi adalah data dari informasi transaksi itu sendiri, hal ini memungkinkan penyerang untuk dapat memodifikasi informasi lain seperti alamat IP dan session pengguna. Penyerang dapat mencuri session aktif dari pengguna dan mencoba melakukan transaksi atas nama pengguna untuk keuntungan si penyerang. Hal seperti ini harus dapat di identifikasi dan di deteksi sedini mungkin sehingga tidak menimbulkan kerugian di sisi penyedia layanan ataupun pengguna.

Pemantauan dan deteksi terhadap aktivitas diluar normal seperti ini biasanya tidak bisa dijadikan sebagai alasan utama untuk menggagalkan transaksi dari sisi sistem karena

rentan adanya false positive. Tindakan yang lebih tepat ketika hal seperti ini terjadi adalah dengan memberikan tanda terhadap transaksi dari sisi server untuk kemudian dapat di tinjau ulang terkai transaksi tersebut secara manual.

Step B: Identifying Detection Capability (Requirements)

Detection Capability Requirements:

- Data Sources: Kemampuan deteksi akan sangat dipengaruhi oleh informasi transaksi dan logs dari proses otorisasi baik dari sisi pengguna ataupun dari sisi server.
- Technology Support: Sistem deteksi atau pemantauan yang memadai untuk menyimpan dan memantau aktivitas transaksi secara terus menerus sangat dibutuhkan.
- Data Quantity and Quality: Data transaksi sangat diperlukan, termasuk juga data informasi lain dari pengguna seperti alamat, status keanggotaan, alamat IP, jenis perangkat yang biasa digunakan, dan lain-lain.

Step C: Identifying Detection Components (Indicators)

Detection Components (Indicators):

- Transaction Timestamp: Untuk mendeteksi waktu terjadinya transaksi dan melihat secara lebih jelas dalam rentang waktu tertentu terkait aktivitas lainnya
- Transaction Data: Harga, jumlah pembelian, jenis pembayaran, alamat pengiriman, nama penerima, jenis perangkat yang dipakai untuk transaksi, dan alamat IP.
- User Authorization Logs: untuk kebutuhan korelasi dengan data transaksi sehingga dapat dipakai untuk mengidentifikasi adanya kemungkinan aktivitas yang tidak normal atau diluar kebiasaan.
- Payment Server Logs: Untuk membandingkan data yang dikirimkan oleh pengguna dengan data yang diproses oleh server termasuk juga hasilnya sehingga integritasnya dapat dipastikan dan sudah sesuai.
- Invalid Authorization Status: Mengidenfitikasi berbagai jenis transaksi yang sudah ditolak oleh sistem untuk kemudian di analisis lebih lanjut terkait pola dan jenis transaksi yang gagal yang mungkin dapat membantu untuk identifikasi transaksi mencurigakan berikutnya.
- Logging of Data Modifications: Menyimpan informasi terkait perubahan data yang mungkin coba dilakukan oleh pengguna baik secara sengaja atau tidak sebagai bukti telah terjadi percobaan transaksi yang tidak sah.

Step D: Detection Prioritization

Description: Deteksi terhadap aktivitas transaksi yang diluar normal harus menjadi prioritas karena berdasarkan dari tingkat keparahan dari ancaman tersebut, potensi

kerugian secara finansial, dan mungkin sesuai dengan arahan dari kebutuhan bisnis di organisasi.

Step E: Determining Detection Rules and Requirements

Detection Rules and Requirements:

- Rule 1: Terdapat beberapa percobaan transaksi yang memiliki informasi berbeda, seperti penggunaan jenis perangkat yang berbeda, alamat IP yang berbeda, dan lokasi yang berbeda
- Rule 2: Transaksi yang berhasil lolos dari sistem pencegahan modifikasi data tapi terindikasi berasal dari aktivitas yang diluar normal harus bisa di deteksi lebih awal, sehingga tim pembayaran mungkin akan membutuhkan waktu lebih untuk dapat menerima atau menyelesaikan transaksi di sisi mereka.
- Rule 3: Kumpulkan semua data atau informasi terkait transaksi dalam rentang waktu tertentu untuk kebutuhan penanganan insiden atau forensik jika dibutuhkan pada waktu yang akan datang atau terdapat sanggahan dari pengguna.

Step F: Creating POC of Detection Logic

Description: Langkah berikutnya adalah mulai dengan pembuatan skenario pengujian dan pengembangan untuk deteksi. Fase pengujian ini sudah harus menghasilkan aturan deteksi yang sudah diimplementasikan pada sistem, memiliki sistem peringatan atau pemberitahuan, dan termasuk juga proses penanganan dan peninjauan lebih lanjut yang perlu untuk dilakukan selanjutnya.

Pada fase ini, deteksi belum diaktifkan untuk berjalan pada lingkungan produksi, akan tetapi masih berjalan pada lingkungan pengujian untuk mengetahui sejauh mana kemampuan deteksi dan mengidentifikasi adanya kemungkinan false positive dari deteksi yang terpicu.

3.2.3 Keunggulan Proses Detection-oriented Modeling Framework

Tidak diragukan lagi bahwa proses *Detection-oriented Modeling Framework* (DOMF) akan membawa manfaat yang signifikan dalam meningkatkan keamanan, seperti:

- Dengan menggabungkan *framework detection engineering* dengan *risk assesment*, *business context* akan lebih mudah di pahami oleh sisi *detection engineering*. *Framework* ini memperjelas fungsi dan kemampuan dari sebuah aplikasi yang akan dipergunakan untuk membangun *detection* yang diperlukan.
- Dengan informasi penting (seperti *detection capability* dan *components*) yang tersedia di awal proses *detection engineering*, *detection engineer* akan lebih mudah untuk merencanakan, membangun konsep dan mengaplikasikan sebuah *detection*. Hal ini juga akan memudahkan proses *prioritization detection* oleh tim *detection engineering*.

- Framework ini juga akan membuka media komunikasi baru antara tim *Risk Assesment, Developer* atau *Software Engineering* dan *Detection Engineering.* Dengan terbukanya jalur komunikasi ini, tim tersebut dapat bekerja sama dengan erat, tanpa pembatas dan lebih mengenal fungsi satu sama lain.

3.2.4 Kekurangan Proses Detection-oriented Modeling Framework

Proses *Detection-oriented Modeling Framework* (DOMF) membawa manfaat yang signifikan dalam meningkatkan keamanan, namun, beberapa tantangan juga harus diatasi:

- Kebutuhan akan tim yang berdedikasi penuh untuk *detection engineering* atau individu dengan peran senior yang memiliki tanggung jawab khusus dalam hal ini.
- Waktu tambahan untuk koordinasi dengan berbagai pihak, terutama *software engineer* dan *developer*, yang dapat mengakibatkan kesan bahwa proses ini dapat menjadi penghambat dalam merilis aplikasi ke produksi, terutama jika *Threat Modeling* sudah dilaksanakan sebelumnya.
- Pentingnya *Threat Modeling* yang sudah ada sebelumnya juga menjadi faktor kunci karena *Detection Modeling* memerlukan dasar tersebut.

Masalah lain yang akan dihadapi adalah *volume log* yang berlebihan atau kurangnya standardisasi log. Untuk mengatasi ini, konsep *Funnel of Fidelity oleh SpecterOps* [8] dan *Data Engineering harus* diperkenalkan untuk membantu memilah dan mengolah log yang diperlukan.

Keberhasilan penerapan proses ini sangat bergantung pada kehadiran orang yang berdedikasi untuk melakukan *detection modeling* serta ketersediaan waktu *engineer* untuk mengimplementasikan rekomendasi yang berasal dari proses *detection modeling.* Kerjasama harus dilakukan agar organisasi bisa meningkatkan keamanan menggunakan *Detection-oriented Modeling Framework.*

3.3 Masa Depan Detection-oriented Modeling Framework

Makalah ini hanya awal mula dari proyek Detection-oriented Modeling Framework. Dua contoh yang dituliskan di atas adalah *pioneer* dari *Detection-oriented Modeling Framework repository,* dimana proyek ini akan merangkum lebih banyak lagi skenario serangan dan deteksi, di berbagai area *web application.*

Beberapa area yang akan dikembangkan di masa depan adalah:

- *Login*
- *Registration*
- *Profile Management*
- *File Management*
- *Data Management*
- *Search*

- *Messaging*
- *Dashboard*
- *Notification*
- *Comments*
- *Payment/Transaction*

Penulis berharap bahwa framework ini dapat digunakan seperti MITRE ATT&CK untuk kebutuhan penilaian resiko dan deteksi **di ranah aplikasi web.

4. Kesimpulan

Aplikasi selalu menjadi target utama bagi para penyerang, namun sering kali kurang mendapatkan perhatian yang cukup dalam proses pemantauan keamanan. Oleh karena itu, perlu ada pendekatan *end-to-end* yang komprehensif untuk mendefinisikan setiap kontrol keamanan di sisi pencegahan, deteksi, dan respons yang berkaitan dengan aplikasi. Hal ini memastikan bahwa aplikasi dilindungi dengan baik dari serangan, ancaman dapat dideteksi dengan efektif, dan respons yang cepat dapat diambil jika terjadi insiden keamanan. Dengan pendekatan ini, keamanan aplikasi dapat ditingkatkan secara signifikan, menjadikannya lebih tahan terhadap serangan dan risiko yang mungkin terjadi.

5. Daftar Pustaka

[1] Roddie, Megan, et al. *Practical Threat Detection Engineering: A Hands-on Guide to Planning, Developing, and Validating Detection Capabilities*. Packt Publishing, 2023.

[2] Crowley, Christopher, et al. SANS, 2023, SANS 2023 SOC Survey.

[3]“Threat Modeling Process.” Threat Modeling Process | OWASP Foundation, owasp.org/www-community/Threat_Modeling_Process. Accessed 21 Sept. 2023.

[4] Garg, Praerit, and Loren Kohnfelder . “Threats - Microsoft Threat Modeling Tool - Azure.” Threats - Microsoft Threat Modeling Tool - Azure | Microsoft Learn, Microsoft , learn.microsoft.com/en-us/azure/security/develop/threat-modeling-tool-threats. Accessed 21 Sept. 2023.

[5] Shostack, Adam. “Experiences Threat Modeling at Microsoft - Shostack.” *Experiences Threat Modeling at Microsoft*, adam.shostack.org/modsec08/Shostack-ModSec08-Experiences-Threat-Modeling-At-Microsoft.pdf. Accessed 22 Sept. 2023.

[6] “What Are Application Security Frameworks? .” Pathlock, 18 May 2023, pathlock.com/learn/what-are-application-security-frameworks/.

[7] Bianco, David J. *The Pyramid of Pain*, detect-respond.blogspot.com/2013/03/the-pyramid-of-pain.html. Accessed 21 Sept. 2023.

[8] Atkinson, Jared. “Introducing the Funnel of Fidelity.” *Medium*, Posts By SpecterOps Team Members, 3 Dec. 2019, posts.specterops.io/introducing-the-funnel-of-fidelity-b1bb59b04036.

Lampiran

Lampiran berikut memberikan gambaran lebih luas tentang potensi detection-oriented modeling ketika diimplementasikan secara langsung secara berdampingan dengan metode threat modeling lain seperti ASF (Application Security Framework) dan STRIDE (Spoofing, Tampering, Repudiation, Information disclosure, Denial of Service, Elevation of privilege).

ASF Threat & Countermeasures Examples

Threat Type	Countermeasure / Prevention	Detection-oriented
Authentication	1. Credentials and authentication tokens are protected with encryption in storage and transit 2. Protocols are resistant to brute force, dictionary, and replay attacks 3. Strong password policies are enforced 4. Trusted server authentication is used instead of SQL authentication 5. Passwords are stored with salted hashes 6. Password resets do not reveal password hints and valid usernames 7. Account lockouts do not result in a denial of service attack	1. Encryption key/token/session stolen 2. Impossible Travel detection 3. Trust authentication abuse attempt 4. Anomalous number of password reset attempt 5. Anomalous number of account lockouts 6. Password spraying attempt detection
Authorization	1. Strong ACLs are used for enforcing authorized access to resources 2. Role-based access controls are used to restrict access to specific operations 3. The system follows the principle of least privilege for user and service accounts 4. Privilege separation is correctly configured within the presentation, business and data access layers	1. ACLs bypass attempt 2. Multiple AccessDenied Logs 3. Privilege escalation attempt

Configuration Management	1. Least privileged processes are used and service accounts with no administration capability 2. Auditing and logging of all administration activities is enabled 3. Access to configuration files and administrator interfaces is restricted to administrators	1. Unauthorized admin privilege assignment 2. Unauthorized admin activity 3. Config file exfiltration attempt (failed/success)
Data Protection in Storage and Transit	1. Standard encryption algorithms and correct key sizes are being used 2. Hashed message authentication codes (HMACs) are used to protect data integrity 3. Secrets (e.g. keys, confidential data) are cryptographically protected both in transport and in storage 4. Built-in secure storage is used for protecting keys 5. No credentials and sensitive data are sent in clear text over the wire	1. Stolen secret key usage observed 2. Secret key leaked in response or logs 3. Command injection attempts from input field 4. Exploitation for Credential Access (against certain vulnerable technology stack)
Data Validation / Parameter Validation	1. Data type, format, length, and range checks are enforced 2. All data sent from the client is validated 3. No security decision is based upon parameters (e.g. URL parameters) that can be manipulated 4. Input filtering via allow list validation is used 5. Output encoding is used	1. Data type bypass 2. Parameter bypass 3. Brute-force attempt 4. Data mismatch between user and server sides 5. Modified user response (such as using Burp or similar tool)
Error Handling and Exception Management	1. All exceptions are handled in a structured manner 2. Privileges are restored to the appropriate level in case of errors and exceptions 3. Error messages are scrubbed so that no sensitive information is revealed to the attacker	1. Error messages leaking sensitive information (such as token, password or secret key) 2. Broken error handling flow

User and Session Management	1. No sensitive information is stored in clear text in the cookie 2. The contents of the authentication cookies is encrypted 3. Cookies are configured to expire 4. Sessions are resistant to replay attacks 5. Secure communication channels are used to protect authentication cookies 6. User is forced to re-authenticate when performing critical functions 7. Sessions are expired at logout	1. Cookies stolen 2. Sessions stolen 3. Replay attacks detection 4. Expired session used in any activity
Auditing and Logging	1. Sensitive information (e.g. passwords, PII) is not logged 2. Access controls (e.g. ACLs) are enforced on log files to prevent un-authorized access 3. Integrity controls (e.g. signatures) are enforced on log files to provide non-repudiation 4. Log files provide for audit trail for sensitive operations and logging of key events 5. Auditing and logging is enabled across the tiers on multiple servers	1. Logging configuration modification detection (Log stoped, modified, restarted, tampered, etc) 2. Credential leak in Logging 3. Logging components exploitation (such as Log4j vulnerability)

STRIDE Threat & Countermeasures Techniques

Threat Type	**Mitigation Techniques / Prevention**	**Detection-oriented Countermeasure**
Spoofing Identity	1. Appropriate authentication 2. Protect secret data 3. Don’t store secrets	1. Secrets stolen/leaked 2. Sessions stolen/re-use 3. Anomalous/Unusual GetSecrets attempts 4. Failed secrets usage

Tampering with data	1. Appropriate authorization 2. Hashes 3. MACs 4. Digital signatures 5. Tamper resistant protocols	1. Data mismatch between user and server sides 2. Response modification detection
Repudiation	1. Digital signatures 2. Timestamps 3. Audit trails	1. Timestamp bombing 2. Audit trails modification 3. Fake signatures
Information Disclosure	1. Authorization 2. Privacy-enhanced protocols 3. Encryption 4. Protect secrets 5. Don't store secrets	1. Secrets stolen/leaked 2. Logging exposing sensitive information 3. Error message exposing sensitive information
Denial of Service	1. Appropriate authentication 2. Appropriate authorization 3. Filtering 4. Throttling 5. Quality of service	1. Sudden traffics spike 2. Anomalous number of traffics received 3. Malicious payload received 4. Availability disruption detected after anomalous number of traffics received
Elevation of privilege	1. Run with least privilege	1. Privilege escalation attempt (success/failed) 2. Historical regular user performing high privilege activity