**EMCS2430: Human Factors: People and Software**

Assignment: Create a STRIDE Model

Brian Russel Davis, [brian\_davis@brown.edu](mailto:brian_davis@brown.edu)

#### S.T.R.I.D.E. Model in Context

S.T.R.I.D.E. is a mnemonic for security threats that “separates them into six categories. The threats are: **S**poofing of user identity, **T**ampering, **R**epudiation, **I**nformation disclosure (privacy breach or data leak), **D**enial of service (D.o.S), **E**levation of privilege.”

Wikipedia suggests that we equate each threat with a desired security property according to the following table.

|  |  |
| --- | --- |
| **Threat** | **Desired Property** |
| Spoofing | Authenticity |
| Tampering | Integrity |
| Repudiation | Non-repudiability |
| Information Disclosure | Confidentiality |
| Denial of Service | Availability |
| Elevation of Privilege | Authorization |

PAX is launching an application that collects cannabis strain data from growers and delivers this strain information to our mobile app users. PAX is known for its commitment to safety and compliance to COPPA, so the integrity of the data and the system are paramount.

Strain Data originates with the Brand Partner ( the maker of the cannabis oil ), is submitted via the Connect Portal Web Interface, the Connect Portal save the information in the PAX Database, the information is reviewed by PAX Compliance and once it is approved the status of the content will be changed from DRAFT to PUBLISHED making it accessible it the public in the PAX mobile app ( iOS and Android ). Data that originates with the Brand Partner is outside the trust boundary which is why sanitization of this data begins during collection, long before it is saved in the PAX Backend.

There is not really a significant movement of data outside the Trust Boundary so instead of the of a table showing the flow of data, I will list the threat phenotype alongside a security phenotype.

|  |  |
| --- | --- |
| **Potential Threat** | **Ideal Security Response** |
| **Spoofing**  *Threat #1*: Attacker impersonating an authenticated user  *Threat #2*: Unauthorized insider using credentials w/o approval | **Authentication Methods**  *Counter Measure #1* : Using in-person contact to create trusted initial admin user profiles.  *Counter Measure #1* : IP logging and implementation of MFA would curtail unauthorized usage |
| **Tampering**  *Threat #1*: Attacker attempts to tamper with data already submitted by the user  *Threat #2*: Insider attempts to change the status of strain data that has not passed compliance | **Integrity Methods**  *Counter Measure #1* : While proper authentication should prevent this, the system should also log a record of every change and who made it, any anomalies should be reviewed by a security engineer  *Counter Measure #2* : Insiders without the proper role should not be able to alter records. Instituting DBAC would mitigate against this potential threat, |
| **Repudiation**  *Threat #1* : Attacker tries to bypass the compliance check and change the status of the strain information from submitted to approved  *Threat #2* : Attacker tries to Spoof a message from compliance to justify overriding a compliance decision | **Non-repudiability Methods**  *Counter Measure #1* : The system should also log a record of every change and who made it, any anomalies should be reviewed by a security engineer  *Counter Measure #2* : Proper education should be deployed to help compliance and BP’s understand that they should only trust messages on the platform |
| **Information Disclosure**  *Threat #1* : BP tries to see the competitors strain information  *Threat #2* : BP tries to see hidden compliance conversations or notes | **Confidentiality Methods**  *Counter Measure #1* : While proper authentication should prevent this, the system should also log a record of every time the service is accessed and where they accessed it from, any anomalies should be reviewed by a security engineer  *Counter Measure #2* : |
| **Denial of Service**  Threat #1 : Attacker tried to flood the authentication API with bogus requests as a way of disabling it  Threat #2 : Attacker tried to flood the strain submission API with bogus requests as a way of disabling it | **Availability Methods**  *Counter Measure #1* : Institute a max number of attempts for bad logins and a check against unknown locations  *Counter Measure #2* : Institute a WAF to prevent the endpoint from being used by anyone or anything outside of the application. |
| **Elevation of privilege** Threat #1 : BP tries to find and edit their user role and change it to admin. Threat #2 : Insider who is not an admin or super user tries to add themselves to a BP account | **Authorization Methods**  *Counter Measure #1* : Make sure that details of where the permission assignments are keep are obfuscated from the user.  *Counter Measure #2* : Disable role elevation by instituting DBAC and a requirement of live permission granting. |

S.T.R.I.D.E. opens up an opportunity to have an ongoing detailed conversation with software and product teams about security. The author of book John Wiley did a talk and stressed the importance of NOT trying to create one document with all of the S.T.R.I.D.E. analysis before deploying an application. Instead he strongly recommended the use of Agile to iterate of security needs. The STRIDE effort does not end with one document or one effort to do a top to bottom analysis. Instead STRIDE and security as a practice is integrated into every feature, every new data point, every new user role and every engineering sprint.

#### Conclusion

It’s unclear if the S.T.R.I.D.E. model will continue to be relevant at the shape of technology continues to evolve. How will S.T.R.I.D.E. deal with rouge intents in the world of Artificial Intelligence? There’s a group concerned that S.T.R.I.D.E. doesn’t adequately cover Lateral Movement and create S.T.R.I.D.E.-LM. All of these are interesting concerns and the community of security professionals that embrace S.T.R.I.D.E. should also embrace it’s evolution. In the parable at the beginning of this paper the student who was successful force to the Master to play on his terms and experience an environment that he had never seen. If organizations adopted the idea of watching the activity of users and programmatically challenging users that exhibit strange activity, it would go a long ways toward

#### 

#### A Short Parable

There once lived a Grand Master, well known for his agility, strength, speed and cunning. Every year the Grand Master would take on students and train them in his way. At the end of their time with the Master he would test the top four students, hoping one day to anoint a new Grand Master.

On the day of the test the Master instructed the top four students saying, “Here is a flag, the one who can use these stones to protect the flag from me will earn my post as Grand Master.”

The first student used the stones to build a high wall around the flag, but found quick defeat when the Grand Master used his agility to scale the wall and capture the flag.

The second student used the stones to enclosed the flag in a thick cocoon, but also found defeat when the Grand Master used his strength to break the stones and capture the flag.

The third student seeing the failure of the first two decided to challenge the Master’s intellect and built a complex labyrinth, hiding the flag deep inside a sprawling maze. However, the Master used his speed and cunning to solve the maze and capture the flag.

The fourth student did not build walls, barriers or mazes with his stones, but instead laid each stone carefully in a shallow part of a river bed with fast moving current, stood on one of the stones and blind folded himself with a flag as he beckoned the Master to come and take the flag. Unsure of his footing at first, the Master stepped out onto the stones being careful to not get caught in the rushing water as he fought with the student while balancing on the slippery stones. However, it didn’t take long for the Master to learn the art of moving from stone to stone while fighting and eventually he ripped the flag from the student’s face.

“Very creative,” said the Grand Master, “but, you underestimated my ability to learn your system and I am afraid you still lose.”

The student smiled, “No my dear Master, I think you have lost.”

“I knew that you were stronger, faster and more agile than me. But, everyday you trained me, I practiced the same techniques in this river bed, on these stones, blindfolded and now I know them like my own skin.”

“While you were focused on learning your footing, I slipped the real flag into your armour while your back was turned, knowing that your confidence and vanity are your greatest weakness.”

The student walked behind the Master and retrieved a carefully folded flag from the back of his armour and was anointed the new Grand Master of the province.

**Threat modeling is not about making an impernatable system. Threat modeling is the art of understanding your system and your opponent better than your opponent does and using this knowledge to exploit their weakness.**