Discovering Decision-Making Patterns for Security Novices and Experts

Hanan Hibshi[*] **Travis Breaux**[*] **Maria Riaz**[†]
Laurie Williams[†]

March 2015
CMU-ISR-15-101

Institute for Software Research
School of Computer Science
Carnegie Mellon University
Pittsburgh, PA 15213

[*]Institute for Software Research, School of Computer Science, Carnegie Mellon University, Pittsburgh, PA, USA
[†]Department of Computer Science, North Carolina State University, Raleigh, NC, USA

This research is supported by grants from the National Security Agency, the Software Engineering Institute, and King Abdul-Aziz University.

Keywords: Security, requirements, patterns, analysis, decision-making, situation awareness

Abstract

Security analysis requires some degree of knowledge to align threats to vulnerabilities in information technology. Despite the abundance of security requirements, the evidence suggests that security experts are not applying these checklists. Instead, they default to their background knowledge to identify security vulnerabilities. To better understand the different effects of security checklists, analysis and expertise, we conducted a series of interviews to capture and encode the decision-making process of security experts and novices during three security requirements analysis exercises. Participants were asked to analyze three kinds of artifacts: source code, data flow diagrams, and network diagrams, for vulnerabilities, and then to apply a requirements checklist to demonstrate their ability to mitigate vulnerabilities. We framed our study using Situation Awareness theory to elicit responses that were analyzed using coding theory and grounded analysis. Our results include decision-making patterns that characterize how analysts perceive, comprehend and project future threats, and how these patterns relate to selecting security mitigations. Based on this analysis, we discovered new theory to measure how security experts and novices apply attack models and how structured and unstructured analysis enables increasing security requirements coverage. We discuss suggestions of how our method could be adapted and applied to improve training and education instruments of security analysts.

1 Introduction

Each year, attackers exploit well-known vulnerabilities that have obvious, well-documented solutions. Hewlett-Packard's top cyber security risks report in 2011 presents many popular attacks against web applications, such as SQL injection attacks [14]. In addition, the OWASP Top 10 web application security vulnerabilities [19] and the SANS Top 20 Critical Security Controls [23] aim to reduce the most common vulnerabilities. Finally, high profile standards bodies publish security control catalogues, including the ISO/IEC 27000 Series standards and the U.S. National Institute of Standards and Technology (NIST) Special Publication 800 Series that contain best practice security requirements. Despite these broadly disseminated, diverse and in-depth sources of security knowledge, information systems continue to be susceptible to known vulnerabilities. Many systems continue to operate under poor security practices, such as unencrypted wireless networks, the same administrative password across multiple systems, and unexpired, outdated passwords [3].

The lack of information system security is unlikely due to an absence of security requirements or analysis methods, which are abundant: research in requirements engineering has sought to address security, including abuse and misuse cases [17, 25], anti-goals [26], and trust assumptions that are used to construct assurance arguments [12, 13]. Combined with the wealth of available security knowledge, we hypothesize insecure information systems persist because security analysts experience two challenges: a) they experience difficulty in perceiving relevant risks in the context of their information system designs; and b) they experience difficulty in deciding which requirements are appropriate to minimize risk. We propose that security requirements analysis methods evaluation should address these two difficulties, directly.

The contributions of this paper are:

- A novel coding method to apply Situation Awareness (SA) to interview data, to understand how security experts think about problems;
- New theory based on SA decision-making patterns to measure how attack models enhance security analysis and how novices and experts differ in the application of these models; and
- New evidence based on SA decision-making patterns that explains the trade-offs in structured versus unstructured analysis and the impact on increasing coverage in security analysis.

The remainder of this paper is organized as follows: we present background on situation awareness Section 2; we present our research method in Section 3; we present results of evaluating our approach in Section 4; we present the decision-making patterns in Section 5; we discuss role of expertise and the attacker model in 6;we discuss our observations across the three artifacts in Section 7; we present threats to validity in Section 7, followed by our discussion in Section 9. Finally, we conclude in Section 10.

2 Situation Awareness and Security Risk

Situation Awareness (SA) is a framework introduced by Mica R. Endsley in 1988 [8] that distinguished between a user's "perception of the elements in the environment within a volume of time and space, the comprehension of their meaning, and the projection of their status in the near future" during their engagement with a system. Perception, comprehension and projection are called the levels of SA, and a person ascends through these levels in order to reach a decision. To illustrate, consider SQL injection, in which an attacker inserts an SQL statement fragment into an input variable (often via a web form) to gain unauthorized database access. When an expert conducts a source code vulnerability assessment, they look for cues in the code to place input sanitization, which is a mitigating security requirement. Upon finding such cues (perception), analysts proceed to reason about whether the requirement has or has not been implemented (comprehension). Once understood, they can informally predict the likelihood of an SQL injection attack and the consequences on the system (projection) based on their experience and understanding of the threat and attack vector.

We believe SA can be used to explain how analysts perform risk assessments. The NIST Special Publication 800-30 [18] defines risk as the product of the likelihood that a system's vulnerability can be exploited and the impact that this exploit will have on the system, and this is the definition that we assume throughout this paper. The ability to predict likelihood and impact depend on the analyst's ability to project prospective events based on what they have perceived and comprehended about the system's specification and its state of vulnerability. If the expert succeeds in all three SA levels, then they have "good" SA and they should be able to make more accurate decisions about security risks. Failure in any level results in "poor" SA that leads to inaccurate decisions or no decisions at all. In Section 3, we describe our method to detect the SA-levels in security expert interviews.

Endsley and other researchers [10, 8, 9] go beyond the SA definition to establish a holistic framework that scientists in other fields could benefit from and apply. This framework entails details and relationships to other concepts such as: expertise effect, goals, mental models, automation, uncertainty, requirements analysis, etc. Endsley explains that expertise can facilitate the person's ability to interpret their perceptions and make necessary projections that lead to better decisions EJ03.

The SA framework is flexible and could be customized according to the needs of a system. Examples of fields in which SA has been applied include military operations [7], command and control [11], cyber security [4, 15] and many others [10, 24]. Researchers have modeled SA in intelligent and adaptive systems [7, 11, 24]. Feng et al. proposed a context-aware decision support system that models situation awareness in a command-control system [11]. Their focus was to have agents based on a rule-based inference engines that provide decision support for users. They applied Endsley's concepts and focused on shared situation awarenessÓ along with a computational model that they applied to a case study of a command and control application. Chen et al. formalized experts' experiences by extending a cyber intrusion detection system using a logical formalization of SA concepts [4]. Jakobson proposed a framework of situation aware multi-agent systems that could be cyber-attack tolerant [15]. To our knowledge, SA has not been widely adopted in security requirements engineering.

3 Research Approach

We chose an exploratory, qualitative research method that aims to understand the symbolic and cognitive processes of specific security analysts, as opposed to testing hypotheses against specific variables citeCS07. The purpose of our approach is to develop a theory of security analysis from a rich dataset that we can later test in a controlled experiment. Consequently, this theory is grounded in the domain and findings from this study are only valid for this dataset citeCS07. Our method consists of three main phases:

- The preparation phase, in which we developed the research protocol, including tailoring SA to security analysis, selecting the system artifacts to use in the analysis, and recruiting the security analysts to be interviewed;
- The interview phase, wherein we elicited responses from the selected analysts; and
- The qualitative data analysis phase, in which we coded the interview transcripts and systematically drew inferences from the data.

We employed coding theory [22] to link SA concepts to the dataset and validate whether our observations are consistent and complete with respect to that dataset. In the first cycle, we applied the hypothesis coding method to our dataset [22] using a predefined code list derived from Endsley's SA levels; this method tests the validity of the initial code list. In the second cycle, we applied theoretical coding to discover decision-making patterns from the coded dataset. We now discuss the three phases.

3.1 The Preparation Phase

The SA framework can be tailored to a field of interest by mapping SA levels to statements made by domain analysts. We tailored the framework by verbally probing the analyst during the interview process as they were asked to evaluate security risk of information system artifacts. We expected the dataset to show how analysts build SA and to help us further discover how perceptions of security risk evolve as the analyst's awareness of both potential vulnerability and available mitigations increases. The inability to perceive risk may be due to limitations in analysts' knowledge or ambiguities in the artifacts. We define the SA levels as follows:

- Level 1: Perception: the participant acknowledges perceiving security cues in the given artifact. Examples include:"there is a picture of a firewall her" or"there are SQL commands in the code snippet" Each observation excludes any deeper interpretation into the meaning of the perception.
- Level 2: Comprehension: the participant explains the meaning of cues that they perceived in Level 1. They provide synthesis of perceived cues, analysis of their interpretations, and comparisons to past experiences or situations. Examples of comprehension include: "the firewall will help control inbound and outbound traffic..." and "the SQL commands are used to access the database which might contain private information, so we need to check the input to those commands, but this is not done in the code..."

- Level 3: Projection: the participant has comprehended sufficient information in Level 2, so they can project future events or consequences. In security, projections include potential, foreseeable attacks or failures that result from poor security. Examples include:"this port allows all public traffic, which makes the network prone to attacks...", or "unchecked input opens the door to SQL injectionÉ"

At Level 3, we expect participants can make security related-decisions. Decisions include steps to modify the system to mitigate, reduce or remove vulnerabilities. Continuing with the firewall and SQL injection examples, one decision could be:"this port should be closed" or "a function should be added here that checks the input before passing it to the SQL statement." Closing the port in a firewall prevents an attacker from exploiting the open port in an attack, whereas checking the input can remove malicious SQL in an SQL-injection attack.

3.1.1 Security Artifacts

We presented each participant with three categories of security-related artifacts: source code, data flow diagrams, and network diagrams. We chose these artifacts to cover from low-level source codes to high-level architecture, noting that security requirements should be mapped to each artifact in different ways and analysts require different skills to do this mapping. Based on our own experience and knowledge of security expertise, we considered the effect of specialization in areas such as secure programming, network security, etc. in selecting these artifacts. Hence, the selection aims to satisfy two goals: 1) to account for diverse background and experience; and 2) to assess whether different artifacts show differences among SA levels. We discuss our findings to address these two goals in Section 6. We now describe the artifacts used in this study.

SC: **Source Code.** We present participants with JavaScript code snippets, corresponding SQL statements, and a user interface related to the snippet. The code contains two vulnerabilities, an SQL injection attack and unencrypted username and password. JavaScript is a subset of a general purpose programming language, i.e., no templates, pointers, or memory management issues. Thus, we expect analysts with general programming language proficiency and knowledge of SQL injection to be able to spot these vulnerabilities. We also list a high-level security goal to prompt participants and we ask participants if the goal has been satisfied.

DFD: **Data Flow Diagram.** We present participants with a data flow diagram for installing an application on a mobile platform. As shown in Figure 1, the diagram contains high-level information about the data flow between the user, app developer and the market. The participants are asked about possible security requirements to ensure a secure information flow, and whether they can evaluate those requirements based on this diagram.

ND: **Network Diagrams.** We present participants two network diagrams: diagram ND1 shows an insecure network, and diagram ND2 shows a network with security measures that address weaknesses in ND1. After participants are provided time to study ND1, we present ND2 and ask participants to evaluate whether ND2 is an improvement over ND1. After collecting data on participants'

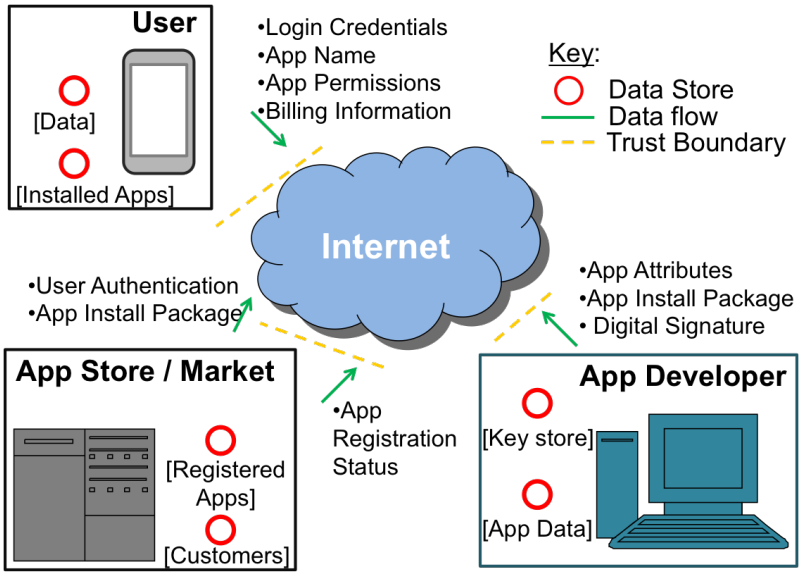

Figure 1: The data flow diagram (DFD) artifact

evaluation of ND2, we present 15 security requirements to participants, which we explain are part of a security improvement process, and we ask participants to assess whether the network in ND2 satisfies the 15 requirements (see Appendix A). All of the selected artifacts are typical examples comparable to what is taught in college-level security courses.

3.1.2 Choosing Experts for the Study

In this study, we aim to observe how security expertise affects requirements analysis. However, security analysts are not all equal in expertise: some analysts have more expertise than others in particular areas, and training in academia is different than hands-on practice. To cover a broad range of expertise, we invited industrial practitioners and Ph.D. students at different stages of matriculation, all working in security. We present demographic data to characterize their experience levels in Section 6.1

3.2 The Interview Phase

We designed the interviews to study how analysts reach a security-related decision, and not to study the correctness of the decision or degree of security improvement. We chose this design to reduce participants' anxiety about being personally evaluated. During our interviews, we only ask the following kinds of questions:

- What cues did the participant look at? (Perception)
- How were the cues interpreted? (Comprehension)
- Why did they interpret a cue that way? (Comprehension)

- What are the future consequences of each interpretation? (Projection)
- Based on those projected consequences, what is the best practice? (Decision)

Our approach differs from how SA is traditionally studied in human operator environments (e.g., airplane cockpits and nuclear power plants) that use the Situational Awareness Global Assessment Technique (SAGAT), in that our participants are not immersed in a simulation per se. Rather, we present artifacts (source code, data flow diagram, network diagrams) to participants with prompts to evaluate artifacts for vulnerabilities asking them to act as the security analyst in this setting. We observe their ability to conduct requirements analysis, their proposed modifications or decisions, and their evaluation of security requirements satisfaction.

In addition, we ask participants to share information about their decision-making, such as unstated assumptions and what artifact cues led participants to reach a decision. We were careful not to guide participants in a particular direction by keeping our questions general. In addition, we avoided questions such as: what do you perceive, comprehend, or project? For example, if a participant identified an attack scenario, we would follow with Òwhy would you think such an attack could occurÓ, or Òcould you describe how it could happen?Ó Based on our approach to limit our influence on their responses, we found participants returning to the artifact to identify cues and to explain their interpretation.

Given our interest in distinguishing novice from expert analysts, we asked participants to provide a brief description of their relevant background. Questions to elicit background information were asked twice: first, at the interview start, we ask participants about their security background, their education, industry experience, and security topics of interest. Lastly, at the end, we ask the participant about their analysis process they used during the interview and how it relates to their background. Finally, we recorded the interviews for transcription and analysis.

3.3 The Qualitative Data Analysis Phase

Grounded analysis is used to discover new theory and to apply existing theory in a new context citeCS07. We apply grounded analysis in three steps: (1) we transcribe the interviews; (2) beginning with our initial coding frame (see Table 1), we code the transcripts by identifying phrases that match our codes, while discovering new codes to further explain phrases that do not match our preconceived view of the data; and (3), we review previously coded datasets to ensure the newly discovered codes were consistently applied across all transcripts. After the pilot, we observed uncertainty among participants so we added codes to capture the uncertainty. Table 1 shows the complete coding frame: the first eight codes (P, C, J, D, including the variants that account for uncertainty U*) constitute the initial coding frame and were inspired by Endsley's terminology for the Situation Awareness [10]; the remaining four codes were discovered during our analysis to account for the interview mechanics. We employed two coders (the first and third authors) who first met to discuss the coding process and coding frame, before separately coding the transcripts, and finally meeting to resolve disagreements. The process to resolve disagreements led to improvements in the form of heuristics that explain when to choose one code over the other in otherwise ambiguous situations. To efficiently identify disagreements, we used a fuzzy string-matching algorithm [2] to align the separately coded transcripts. Finally, each coder recorded their start and stop times.

Table 1: Situation awareness annotation codes

Code Name and Acronym	**Definition and Coding Criteria Used to Determine Applicability of the Code**
Perception (P)	Participant is acknowledging that they can see certain cue(s)
Comprehension (C)	Participant are explaining the meaning of cue(s) and conducting some analysis on the data perceived
Decision (D)	Participant is stating their decision
Uncertain Perception (UP)	Uncertainty at perception level: participant is missing certain data that would help they need to analyze the artifact
Uncertain Comprehension (UC)	Uncertainty at comprehension level: participant is not missing data but they can't interpret their meaning confidently
Uncertain Projection (UJ)	Uncertainty at projection level: participant cannot predict possible future consequences confidently
Uncertain Decision (UD)	Uncertainty in decision: participant is not confident about the decision that should be made
Assumption (A)	Participant is stating assumption(s)
Ask Question (Q)	Participant is asking the interviewer questions
Probe (Pro)	Interviewer is triggering the participant's thinking with questions or guidance information
Background (BG)	Participant is providing information regarding their personal background
Null code (NA)	Statement is not applicable to code criteria above

To ensure all statements are coded, we applied the null code NA to any statements that did not satisfy the coding criteria, such as when participants request a scrap of paper to draw a figure, or when they ask how much time is remaining for the interview, and so on. We code statements, such as: "I took a course in security..." or "I saw on the news a security breach related to this artifact" as background `{B}`, which includes their personal experience and knowledge. If the participant compares and contrasts comprehended information from the artifact to their experience or knowledge, then that information is coded as comprehension `{C}`. To improve construct validity, the two raters resolved borderline cases by discussing and refining the code definitions and heuristics. The following heuristics were used to classify statements and draw clearer boundaries between coded data:

Perception: The participant verbally identifies a cue in the data (e.g. line number in code, an entity on the network diagram, a specific requirement in the text). Participants are only reporting what they see, and are not commenting or analyzing the cue.

Comprehension: The participant analyzes, makes inferences, or makes comparisons about what they see. This may include the name of the cue (e.g. firewall), but the statement at least includes an interpretation in addition to reporting the perception of the cue.

Projection: The participant forecasts future attacks, possible threats or any events that could occur based on the context found in the artifact.

Decision: The participant makes a decision with regards to the context. This includes deciding whether the system is secure or not secure, or if a certain requirement is satisfiable. Introducing new mitigations of security threats are also considered decisions.

Uncertianty (at any SA level): To determine if the participant is uncertain, first examine the verbal cues that indicate uncertainty, including, but not limited to: ÒI guessÓ, ÒI am not sureÓ, and Òthis is not clear to meÓ. For example, the participant may indicate that they do not know what an icon represents. Alternatively, if the participant acknowledges that they see a cue, but that they cannot understand its role in the artifact, then this is an uncertain comprehension.

Assumption: The participant here needs to explicitly express that they are making an assumption. Examples of such statements include: "I am going to guess that this means", "I assume", "Based on my experience this means, but it's not necessarily what the artifact tells me" and so on. To clarify how to distinguish assumptions from comprehensions, a comprehension is when the participant is explaining a certain cue's meaning based on the information given in the artifact. Assumptions, however, provide further explanation based on the participant's experience with similar systems to compensate for missing cues or missing information in the artifact.

After the first cycle coding, we conducted a second cycle or axial coding [22] to identify decision-making patterns. We defined cut-offs between coded sequences by sequentially numbering each statement and then assigning group numbers to statements that address the same expanding idea. The groups serve to delineate transitions between units of analysis. We programmatically extracted SA-level sequences (e.g., P-C for perception followed by comprehension) that we later associated with separate, named patterns, and we searched the dataset without the cut offs to assess pattern validity (i.e., detect false-positives: does the SA-level sequence always correspond to an actual coherent pattern that we assigned). We recorded false positives in which the sequence appeared in the data, but did not conform to the pattern. We used the false positives identification to compute pattern accuracy or ratio of true positives over the sum of true and false positives. The next step in our grounded analysis includes labeling interviewee statements with entity identifiers from the specifications, such as variables and functions in the source code or servers and firewalls in the network diagram. Once labeled, we were able to sort our analysis results by entity to see how different participants react to and analyze the same entity and to link the decision patterns to corresponding entities. We report the results of the entity analysis under Section 6 with respect to the role of attack models. We also report the results in Section 7 to reflect on participants' performance among the different artifact types.

3.4 Pilot Study

We piloted the study on two experts: participant P1 is an expert with extensive hands-on and academic expertise in networks and systems security; and participant P2 is a novice who has only academic security experience. The purpose of the pilot study is to test our interview protocol and apply any needed modifications to the questions or protocol before conducting additional interviews. Both participants P1 and P2 analyzed the network diagram artifact, but P2 was unable to think deeper about certain details and reported a higher number of uncertainties. One insight that we observed in the pilot study was the ability of the more experienced participant P1 to make assumptions when faced with uncertainty. When the novice participant was faced with uncertainty, their solution was to ask the interviewer clarification questions. The following excerpt below is an example of an assumption that participant P1 made when they analyzed the requirement R9 that states implementing time synchronization for logging and auditing capabilities. Note that each statement will have an opening and closing code tags (see Table 1 for codes):

```
{UP} I don't see an NTP server on this network{/UP}{C}but I know that
Windows Domain Controller can act as NTP{/C},{A}so I am going to
assume that when they install it they'll probably leave that box
checked because it's a default option{/A}.{D}I think that is probably
happening here{/D}
```

When P2 was faced with uncertainty, however, they turned to the interviewer and asked:

```
{Q} What kind of software does this thing has?{/Q}
```

An observation during our pilot study is that, although we asked participants to verify security requirements to check consistency between the requirements and the network diagram present in the artifact, they actually performed requirements validation, where they assess if the requirements actually meets the stakeholders' system security goals). An explanation may be that security experts rely on background knowledge and apply known security requirements. In addition, we found experts often add missing requirements, explain how to apply a requirement, evaluate whether a requirement was feasible, list some needed specifications, and prioritize requirements. For example, consider the following excerpt as participant P1 is evaluating R2 in the context of diagram ND2 and pointing out that this requirement is less critical than requirement R1 that they had evaluated earlier:

```
{C}but I don't think it's as critical as say the DMZ one, but I think
its sort of whatever is the next tier of criticality{/C}.
```

Based on our pilot study experience and the participant feedback, we revised our study protocol. A major change was the order of the presentation of network diagrams ND1 before ND2, and asking participants to draw on ND1 to improve this diagram. After this modified step, we show participants the secure diagram ND2 and ask them to compare this diagram to their own solution to ND1. Finally, we ask participants to review the requirements list, and to answer the following questions for each requirement:

- Is the requirement satisfied or not satisfied based on the information given in the diagram?

- How would the participant evaluate the security requirement: is it good, bad, unnecessary, immeasurable, unrealistic, etc.

The questions above are asked in a conversational style with an open-ended fashion where participants are free to comment, explain and elaborate in their answers.

4 Evaluation of Approach

In this section, we report the results from our empirical evaluation: the artifact assignment and inter-rater reliability.

4.1 Artifact Assignment

Due to self-perceived inexperience by participants and time limitations, not every participant analyzed all artifacts in the three categories we described in Section 3. The average total interview time per participant to complete each interview was 29 minutes. Table 2 presents the participant assignment to conditions: the shaded cells show the category of artifacts that participants attempted; cells labeled with "X" indicate that the participant spent at least 15 minutes analyzing the artifact. Because participants have varying skills and expertise, some participants invested more time than others analyzing certain artifacts. The order in which the artifacts were presented to different participants was randomized and the time allowed to complete the interview was limited to 60 minutes. Thus, not all participants reviewed all artifacts. The Sum column in Table 2 presents the total number of participants who reviewed each artifact.

Table 2: Final dataset frequencies by code

Artifact	**Participant**											**Sum**
	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	
1) Source Code			X	X		X		X		X	X	
2) Data Flow		X	X			X	X	X				
3) Network	X		X	X					X		X	

4.2 Agreement and Inter-rater Reliability

Two raters (the first and third authors) applied the coding frame from Table 1 to the transcripts of participant audio recordings. We measured inter-rater reliability using Cohen's Kappa, a statistic for measuring the proportion of agreement between two raters above what might be expected by chance alone [6]. We calculated Kappa for each participant, which ranges between 0.51-0.77 with a median of 0.62. These values are considered moderate to substantial agreement [16]. The coding times were 19 and 8 hours for raters 1 and 2, respectively. Rater 1 spent more time documenting heuristics and developing the method. In addition to the above time, 6 hours were used for the

resolution of disagreements between the two coders. Table 3 shows the breakdown of the total 2,595 coded statements in our final dataset by code (including the pilot participants P1 and P2).

Table 3: Participants' assignment by artifact

Code	Total Codes	Code	Total Codes
Perception	250	Uncertain Percept.	82
Comprehension	498	Uncertain Comp.	180
Projection	215	Uncertain Proj.	13
Decision	367	Uncertain Dec.	25
Question	95	Probe	535
Background	47	Assumption	45
N/A	243		

5 Decision Making Patterns

We now present the decision-making patterns that ground the SA framework in the data. We use the acronyms introduced in Table 1 to express the patterns as a sequence of coded observations across the interview transcripts. Findings from this section are going to motivate the discussion, analysis, and impact on security analysis that is present in the remainder of this paper.

5.1 The Classic SA Patterns

Endsley suggests that experts who assess risky situations engage in a process of perceiving information, comprehending the meaning of that information, and then projecting what might occur in the future. We call this pattern the Classic SA pattern, which proceeds from P→C→J→D where the "→" means the coded statement on the left-hand side appeared adjacent and before the coded statement on the right-hand side in the transcript. In addition to the Classic SA pattern, we searched for contiguous fragments of the Classic SA pattern while the order is maintained, such as P→C→J, and C→J that indicate when a participant is move to higher levels of SA.

Table 4 presents the pattern name, number of occurrences (Freq.) and the accuracy (Accu.), which is the ratio of actual, confirmed pattern instances among the total number of observations of the sequence, and, finally, the list of participants who exhibited these patterns. We believe the pattern J→D is interesting because in combination with other patterns, we see variation fragmetns of the order appear. The results indicate that the J→D pattern only appears 31 times with 10% false positives. This observation suggests that projections and decisions, as well as other SA levels, can occur out of sequence, which motivated our search for the other pattern fragments shown in Table 4; all of these fragments are variations of the full Classic SA pattern (P→C→J→D). We observed that participants demonstrated the J→D pattern without the P→C pattern component, but this does not mean that participants did not perceive cues or comprehended those cues. Instead, participants may not be verbally reporting their perceptions and comprehension, or they may have automatized these stages of SA as part of their prior experience.

Table 4: Variations of classic SA pattern

Name	Pattern	Freq.	Accu.*	Participants
Classic w/o Decision	P→C→J	4	100%	P1, P3, P6
Projection-Decision	J→D	31	90%	All except P1
Classic Skip Projection	P→C→D	10	100%	P1, P3, P4, P6, P11
Classic Skip Perception	C→J	55	81%	All
Classic Skip Perception and Projection	C→D	56	83%	All except P2 & P5
Classic Perception Comprehension	P→C	61	81%	All except P10

*Excluding false positives

Except for the first two patterns, a common feature among the patterns in Table 4 is the skip factor. Participants could skip a level of SA before reaching the next expected SA level. Because we coded participants' verbal responses, and participants may not have verbalized each level of cognition, our dataset may be missing the expressions of some levels. Another explanation for skipping levels is the level of expertise and exposure to the problem. If the participant has seen several examples of a certain problem, they may jump to their decisions immediately without providing explicit verbal analysis of the perceived cues, meanings and possible consequences. The following is an example from P3's response to the source code artifact where they immediately projected an SQL attack without perceiving or comprehending a certain cue (we use brackets [] to explain the item of the artifact that the participant is speaking about):

```
{J} this [speaking about the line of code that shows the unsanitized
 input] is just pure SQL injection here {/J}
```

By comparison, P11 articulated moving from perception to projection while describing the same attack scenario:

```
{P}And thus, [speaking about the line of code that shows the
 unsanitized input], you use SQL query that explicitly say its
 inserting into the customer value {/P}{J}it may suffer from the SQL
 injection attack. {J}
```

In contrast, the pattern (P→C→D) from Table 4 describes how a participant moves from perception to comprehension but jumps to the decision phase without describing the projection.

The patterns (C→J) and (C→D) bypass the perception level, where participants move from comprehension to either a projection or a decision phase. Based on our analysis, it is not unusual for participants to begin verbalizing at the comprehension level. In this case, participants begin by describing the meaning of a cue without explicitly identifying the cue. Consider the following excerpt from the coded response of P9 when they were analyzing the Demilitarized Zone in the network artifact:

```
{C}É people can access this part [speaking about the DMZ subnet in the
 network diagram] but it means de-militarized zone.{/C} {J}If these
 machines are hacked, they can't affect other inner parts{/J}
```

The last pattern in Table 4 reflects that participants move from the perception to the comprehension level, but without going immediately into projection or decision levels. We find this pattern interesting because it shows that someone could move back and forth between perception and comprehension without moving higher to projection or decision. This movement could indicate that a participant found themselves "stuck" at comprehension where they could not proceed further, because they lacked the needed cues, understanding to envision what comes next or how to mitigate a threat.

5.2 The Reverse SA Patterns

In our dataset, we observed that SA patterns might occur in reverse order. This difference may be due to the participant using an inductive vs. deductive reasoning style. Up until now, we assumed that participants used a deductive reasoning style: they first report perceiving a cue, comprehending the meaning, and from this information, they deduce and report what may occur in the future (projection). In an inductive reasoning style, the participant verbalizes the possible consequences and from this information, they work backward by inducing the cues that led them to this conclusion. To accommodate the inductive reasoning style, we checked the dataset for patterns in the reverse direction of the classic SA pattern. Table 5 presents the reverse SA pattern names, their frequencies, accuracy and participants who exhibited these patterns.

Table 5: Reverse SA patterns

Name	Pattern	Freq.	Accu.*	Participants
Reverse SA w/ Decision	D→J→C→P	None	None	None
Reverse SA w/o Decision	J→C→P	1	100%	P6
Reverse SA skip projection	D→C→P	3	67%	P6, P9
Reverse SA no perception	J→C	35	67%	All
Reverse SA no perception no projection	D→C	46	75%	All

*Excluding false positives

The following excerpt illustrates the reverse pattern exhibited by participant P6 who is analyzing the source code; the participant first reports their decision to prioritize a particular part of the diagram, followed by their understanding of this part and their perception of the part's character that led to the prioritization decision:

```
{D}It's very important [speaking about using encryption for
 communication over the Internet] {/D}{C} you're sending the SSN over
 the Internet{/C} {P}The SSN is in plaintext. {P}
```

5.3 Patterns of Uncertainty and Assumptions

Uncertainty plays an important role in security, as many security risks are probabilistic and participants must estimate the likelihood of particular events when forming projections. Moreover,

analyst experience is likely to play a role in interpreting ambiguity in a specification and then deciding whether that ambiguity includes an interpretation that may lead to a security exploit. Table 6 presents the uncertainty patterns that we identified in the data. These patterns consist of statements coded with uncertainty (UP, UC, UJ, and UD) and assumptions{A}, questions {Q}, and decisions {D}. The total coded subset relevant to this discovery is comprised of 440 statements across all participants. We categorized uncertainty into three categories:

- *Propagated Uncertainty* occurs in the first three patterns, wherein the uncertainty in perception or comprehension is propagated to a subsequent comprehension, projection or decision
- *Hedged Uncertainty* occurs in all patterns where uncertainty leads to assumptions (e.g.,U$\star$ $\rightarrow$A), in which case the analyst bounds the uncertainty by interpreting an ambiguity and concluding this interpretation in the form of an assumption; and
- *Uncertainty Transfer*, in which the analyst asks a question (e.g., U$\star\rightarrow$Q), to resolve uncertainty by seeking outside assistance.

With hedged uncertainty, 5 out of the 8 participants who made assumptions after their uncertain comprehension were able to make decisions. We found 9 instances of hedged uncertainty leading to decisions, which may involve unstated assumptions. Finally, we observed that participants could move from a certain state to an uncertain one. In our dataset we found participants transitioning to uncertain comprehension from perception (P$\rightarrow$UC, 22 occurrences, 86% accuracy) or from comprehension (C$\rightarrow$UC, 25 occurrences, 68% accuracy).

Table 6: Uncertianty patterns

Pattern	Freq.	Accu.*	Participants
UP$\rightarrow$UC	8	100%	P1, P3, P5, P6, P9
UC$\rightarrow$UJ	2	100%	P2, P5
UC$\rightarrow$UD	2	100%	P1, P4
UC$\rightarrow$A	8	75%	P1, P2, P3, P9, P11
UC$\rightarrow$A$\rightarrow$D	5	100%	P1, P3, P9, P11
UC$\rightarrow$Q	7	100%	P2, P3, P4, P5, P7, P9
UP$\rightarrow$A	5	60%	P1, P3
UP$\rightarrow$Q	3	67%	P1, P3, P5
UC$\rightarrow$D	9	67%	P1, P2, P5, P6, P8, P9, P11

*Excluding false positives

5.4 Patterns Showing Redundant States

In addition to the patterns we discussed so far, we identified several patterns that appear to show the analyst is working harder to reach a decision. This includes patterns with accuracy rates above 60%: (C$\rightarrow$C$\rightarrow$C$\rightarrow$C), (C$\rightarrow$C$\rightarrow$D), (P$\rightarrow$C$\rightarrow$C$\rightarrow$J), (P$\rightarrow$C$\rightarrow$C$\rightarrow$D), and (P$\rightarrow$C$\rightarrow$P$\rightarrow$C). These patterns

appeared 21, 26, 3, 5, and 12 times, respectively. The patterns show that participants are working harder to comprehend and interpret meanings to make more informed decisions. The patterns and corresponding text indicate that, the more detailed and thorough participants' comprehensions were, the better and clearer their future projections or decisions. This may explain why a participant needs more than one comprehension to reach the projection or decision levels. Moreover, there could be situations where complex security projections rely on multiple cues and multiple comprehensions. Moreover, the comprehension level is where the analysis and interpretation begins, and projecting or forming a decision relies heavily on how well the analyst understands the vulnerability. For example, when an analyst comprehends the meaning of a firewall on the network, they consider different factors, which could lead them to verbalize more than one comprehension. Consider the following example as P3 was trying to analyze the network diagram ND2 against the first security requirement from the requirements list provided:

```
{P}your firewall{/P} {C}which is your first point of entry to both DMZ
 traffic and intranet site traffic and also to your users{/C} {P}has
 all of these on separate subnets{/P}{D}the first rule here about stuff
  being unavailable [speaking about the requirement R1]comes down to
 whether this firewall is properly configured.{\D}
```

Participant P3 in the example above cannot reach a decision without comprehending two cues: 1) the firewall is the first point of entry to multiple network segments, and 2) the firewall places the segments on different subnets. Therefore, this decision is dependent on a composition of multiple comprehensions, which explains the redundancy in the above pattern.

5.5 The SA Path to Security Analysis

From our analysis results, we extended Endsley's SA model to account for uncertainty, the role of assumptions and participant inquiry that results from uncertainty. Endlsey defines the stages of SA as they occur in the human mind, but since we are annotating participant articulations of those stages based on their verbal statements, there will be no guarantee that we will observe patterns in the data that will exactly reflect the classic or reverse SA workflow (P→C→J→D).

Hence, we decide to view SA levels as states where a security analyst could take different paths transitioning between the states. We categorize the four main stages of our extended model of SA into: *inspection*, wherein an expert is perceiving and interpreting the meaning of cues (codes P and C in our data); *evaluation*, in which an expert projects future consequences and concludes with decisions influenced by those projections (codes J and D); *ambiguity*, wherein an expert faces uncertainty during inspection and evaluation stages (codes U*); and *resolution*, wherein an expert is resolving uncertainty by making assumptions or asking questions (codes A and Q).

By our extended definition of SA, we open our analysis into other possibilities and combinations that would help understand security expert's decision-making process, and distinguish between experts and novices. We will elaborate more on this in the following sections.

6 Participant's Expertise and the Attacker Model

We investigated whether more experienced participants would exhibit better SA and, thus, be able to form more confident decisions. Herein, we report our findings drawn from demographic data including participants' background and experience, and their experiences reported as remarks during their interview that we coded as {BG}. Next, we examine the role of expertise in forming more confident decisions. Finally, we link an expert's situation awareness with the attacker model by assessing how experts are achieving security decision based on impersonating an attacker.

6.1 Participants' Background and Expertise

Table 7 summarizes participant backgrounds (including pilot participants P1 and P2): the P# is the participant number, which is used consistently throughout this paper; Years is the number of years of industry experience, including internships; Security Areas are the general topics that best describe their industry experience; Research Focus are the topics that best describe their research experience; and Degree is their highest degree earned, or in progress. Among the total eleven, four participants (P1, P3, P4, P5) have extensive industry experience in security (4-15 years) with diverse concentrations.

Table 7: Summary of participants' background

P#	Industry		Research	Degree
	Years	Security Areas		
P1	5+	Network, systems, forensics and more.	Mobile computing, forensics, systems security	Ph.D.
P2	< 1	Security protocols, social networks.	Global cyber threat	Ph.D.*(5th yr)
P3	15+	Systems, Networks, programming, and more.	NA	B.S.
P4	5+	Systems, Networks, architecture, and more	Security for real-time critical systems & architecture	Ph.D.
P5	10+	Software Architecture, Secure Programming	Software Architecture	M.S.
P6	0	NA	Cyber & system security	Ph.D.*(4th yr)
P7	0	NA	Android security, malware, static analysis.	Ph.D.*(4th yr)
P8	1	Infrastructure security, log visualization	Security and Privacy	Ph.D.*(5th yr)
P9	0	NA	Security analysis, network traffic	Ph.D.*(2nd yr)
P10	0	NA	Anomaly Detection	Ph.D.*(1st yr)
P11	0	NA	Network traffic	Ph.D.*(4th yr)

*PhD student, followed by year of matriculation in parentheses

P1, and P4 hold a Ph.D. in security and specialize in systems and infrastructure. These two Ph.Ds and P5 have teaching experience in which they taught advanced security courses. The remaining seven participants were all Ph.D. students with research specialties in security. The Ph.D. students had varying levels of experience, from a student who completed security courses, but who did not apply these lessons in practice beyond class projects, to students who had completed internships with a reputable company working on infrastructure security and log visualizations.

According to Endsley & Jones [10], an increase in experience may affect participants' ability to project future consequences and, hence, may lead to more confident decisions. In our study, we observe that more experienced participants were able to make more assumptions compared to those with less experience. For example, participants with more than 5+ years of industry experience made an average of 7 assumptions, while participants with less than 5 years of experience made an average of 1 assumption. We coded statements with assumptions when the participant explicitly mentions that they are missing relevant details and that they have to assume or guess to complete their understanding.

Difference in artifacts presentation and notation could possibly affect situation awareness. Certain portions of an artifact were likely more unclear than others, so we may only expect to see assumptions when participants encountered less clear portions of the artifact. The pattern (UC→A →D) in Table 6 was observed for experts P1, P3, and P9, when they analyzed the network artifact, and was observed for P11 when they analyzed the source code artifact. Participant P11 demonstrates advanced understanding when analyzing the source code artifact by reaching 24 decisions and this participant was the only participant to make 2 assumptions in that artifact.

6.2 The Attacker Threat Model

Experts Security analysis entails projecting future attack scenarios, and then deciding on how to mitigate them. This aligns very well in SA as we are already coding projections and decisions. In security analysis, projection and decision are closely related, because security analysts may be trained to think like an attacker and have an attack model in mind [26, 21]. With an attack model in mind, the analyst decomposes a future attack scenario into multiple steps that exploit vulnerabilities. Under SA, we expect this decomposition to first appear as perceptions and comprehensions of the vulnerabilities, which then lead to the conclusion or projected exploitation, and finally a commensurate decision to mitigate the vulnerabilities. For example, Participant P3, notes: "what could I do since I am looking at this code to do bad stuff", which is their reflection on trying to walk through threat models that could be relevant to the code segment under review.

We analyzed our dataset to measure how often security analysts employed the attacker perspective. In our study, five participants (P1, P2, P6, P8, P10) demonstrated the need to think like an attacker as demonstrated by the word "attack" in their statements while referring to how an intruder would act.

Our results show 45 instances of attack words used where participants demonstrate knowledge of an attack; out of which only 29 instances describe an application of the attacker model where participants describe how the attack is taking place. The total 45 statements include instances where participants are explaining attacks that they knew about from their background, but without relating that knowledge to the artifact being analyzed. For example, the word attack could show

up in a {BG} statement without a relevant SA pattern. For our analysis, we are interested in the 29 instances where participants are actually thinking like an attacker by demonstrating an attack scenario. Table 8 shows our results from this analysis: the participant number (P#) who described the attack scenario; the frequency (Freq.) that the term attack appears, the security artifact (Art.); and the relevant in-context patterns associated with the word - the SA code of the statement containing the attack word is highlighted in bolded text to show the position within the pattern. Each participant can exhibit multiple, separate instances of thinking like an attacker, which we separated by artifact and in-context pattern.

Table 8: Participants use of the term attack

P#	Freq.	Art.	In-Context Pattern
P1	5	ND1	P→C→C→Pro→**J**
		ND2	P→C→**J**→C
		ND2	D→D→Pro→C→C→**J**→C→C
		ND2	U→J→Pro→**UJ**→Pro→J
		ND2	Pro→UJ→Pro→**J**
P3	3	ND1	P→C→D→Pro→C→D→C→D→J→D→D→Pro→**J**
		ND2	D→J→Pro→J→Pro→**J**→Pro→**J**→C
P4	2	ND2	D→C→C→**J**
		SC	D→C→Pro→**J**→Pro→C→C→P→C
P6	4	SC	**J**→D→**J**→J→C→C→J→Pro→C→C→Pro→P→J
		SC	C→C→**J**
		SC	D→**J**→D→D→J→Pro→C→P→J
P8	3	SC	C→Pro→**J**→Pro→J→D
		DFD	C→C→D→**J**→Pro
		DFD	C→J→**J**→C→C
P9	1	SC	Pro→**J**→J→D→UP→D
P10	7	SC	D→Pro→**J**→J→**J**→D→C
		SC	**J**→Pro→Pro→J→**J**→D
		SC	**J**→J→Pro→**J**→J→**J**
P11	4	SC	P→**J**→J→**D**
		SC	C→C→**D**
		ND2	D→**C**→C→UC

Among the 29 instances of the word *attack*, we observe that most instances (25/29) occurred in the projection stage of SA. In less than half of the instances (12/29), the projection was observed after the interviewer probed the participant to explain why they were perceiving, comprehending or projecting prior to describing the attack scenario (coded as Pro→**J**). Participants P2, P5, P7 are absent from Table 8, as they failed to demonstrate the attacker model.

Attack scenarios can be simple, meaning a single vulnerability is exploited to achieve an attacker's goal, or complex, meaning that multiple exploits are needed. In our results, we may observe and measure the complexity of attack scenarios as a series of different SA stages need to

demonstrate how an attack occurs within an artifact. In Table 8, participant P1 presents the pattern (P→C→J→C) in ND2 by first perceiving the server names, such as Alpha, Lima, Bravo, etc. and then by comprehending the server naming scheme and subsequently projecting that an attacker discovering these names alone cannot tell the role or function of the servers. Based on our entity analysis (see Section 3.3) that links SA codes to these servers across participants, we found that participant P11 perceived the same naming scheme in their analysis (Q→P→C→UC→C), but they were unable to project based on the meaning of the scheme and thus were unable to see the attack scenario. Instead, P11 asks questions, experiences uncertain comprehension due to the meaning of the naming scheme and whether the scheme has any relevance to network security. Unlike P1, participant P11 stops at comprehension and does not proceed to projection or decisions.

Our SA attack model shows how we can use SA to detect a certain expertise skill: thinking like an attacker. A conclusion that is based on the background data alone that is shown in Table 7 above, might indicate that participants: P1, P3, P4, and P5 are the more experts compared to the remaining participants in the table who could be treated as novices. This classification, which we will refer to from now on as the industry classification, is based on participants' clearly combining years of practical industry experience along with academic degrees. However, this classification does not take into account the personal skills that a security analyst might acquire through their job or academic learning. Our attack threat model, on the other hand, help address this limitation by identifying the experts who demonstrate who can *think like an attacker*. Table 8 shows that in addition to P1, P3, P4, who are already identifies experts based on their industry experience, P6, P8, P9, P10 P11 can also demonstrate the skill of thinking like an attacker.

Going back to Table 8, we observe that except for P11's ND2 pattern, all participants had their "attack" keyword appearing in a projection or a decision statement, which resonates with the definition of our projection statements where a future attack is described, and our decision statements where mitigations to an attack is explained. By looking into the details of P11's pattern (D→C→C→UC), we observe how the participant is stuck at the comprehension level where they demonstrate a level of uncertainty.

7 Observations Across the Three Artifact Categories

The three categories of artifact - source code, data flow diagrams and network diagrams Ð were chosen to vary specificity in system design and operation in order to surface variations in analyst performance. We now discuss those variations based on our SA results.

7.1 The Source Code (SC)

Eight participants were presented with the source code artifact, of whom seven agreed to analyze it. Six out of the seven participants identified at least two major concerns: the risk of SQL injection attack and of unencrypted user data. The remaining one participant, who was P10 by the way, could not spot the SQL injection vulnerability although he was reminded by the interviewer more than once to look at the artifact and provide any possible security concerns they might have, or if they have further comments, etc.

The level of analysis and the proposed solutions varied in detail between the participants. While some were able to explain what languages to use and what libraries to call, some found it sufficient to explain that there are more secure measures that exist and good programmers should know about it. To investigate this more, we looked at the coded statements of participants; and compared participant P10 to others who were able to spot the vulnerabilities. For this specific source code artifact, P10 had only 4 perceptions compared to 12, 9, 13 perceptions for P6, P8, P11 respectively. However, P10 had 30 comprehension statements, which is the same as P11 who had more perceptions. When we read some of the statements, we found that P10 spent more time comprehending the 4 perceptions and deviated away from the intended attack to demonstrate other types of attacks that could occur such as phishing. Although Table 8 indicates that P10 can actually demonstrate thinking like an attacker, results from our entity analysis showed that P10 was demonstrating possible attacks other than the SQL injection attack.

7.2 The Data Flow Diagram (DFD)

We found 4/7 occurrences of the (UC→Ask) pattern in the data flow diagram (DFD), as participants report being confused about the chronological order of diagram entities. In addition, the DFD shows higher comprehension uncertainties (49 UC statements compared to 24 UC statements for source code). From the participant responses, we infer that all seven participants agree that the diagram lacks specific details needed for analysis. This result was expected when we chose the artifact: we deliberately chose the diagram showing fewer details to assess how ambiguity could affect the results. In our data, we observe two participants (P2, P5) responding differently to the ambiguity although they have perceived the same cue. Participant P2 states that they do not understand the role of the digital signature shown on the diagram (UC). In contrast, the participant P5 responds to the same entity by challenging the uncertainty with a perception and scaffolding their analysis with an assumption to reach a decision:

```
{UC}Okay. So presumably I'm not sending my digital signature in the
 clear. It's an encrypted session, right?{/UC}{P} But again that doesn'
 t really show that here{/P} {A}so if we assume that's an encrypted
 session and that I am not sharing my digital signature with somebody{/
 A} {D}then this is trusted{/D} {J}but if my machine's been compromised
  and someone has my digital signature they could potentially publish
 things as me, right?{/J}
```

7.3 The Network Diagrams (ND1 and ND2)

The network artifacts illustrate how expertise areas and job role affect decision-making. Recall from Section 6.2 how participant P1, and P11 reacted differently to the same perceived cue of the server-naming scheme. When we matched participant background information from Table 7 with their decision-making patterns, we observed that a job role, such as P1's hands-on experience in networking, might improve the participant's comprehension of cues and lead them to better decision-making.

Contrary to the SC artifact, where participants look at a code snippet showing one distinctive vulnerability: the SQL injection, network diagrams describe a composition of IT components (servers, routers, etc.) in which each component may have its own vulnerabilities. Thus, participants must view these vulnerabilities together to reach certain categories of decision. These interactions can be overwhelming for participants, if no structure is imposed on how they conduct their security analysis. We observed three modes of security analysis: unstructured, semi-structured and structured, which we now discuss.

7.3.1 The Unstructured Mode

Participants were provided the least amount of structure when they were presented with the insecure network diagram (ND1) that had minimal cues, text and legends. Every participant began their analysis with a different cue or entity, and each participant arrived at their own concerns and threat models. Table 8 shows that P1 and P3 demonstrated an attacker threat for ND1, but the entity analysis shows that the two participants were looking at different entities and demonstrating different attackers. Participant P1 began their analysis from the firewall and its possible rules for open ports and participant P3 was more focused on the insecure layout of the DNS, e-mail and web servers. Both participants reached similar mitigation techniques, such as using a DMZ, and network segmentation in order to reduce the attack surface.

7.3.2 The Semi-Structured Mode

The diagram ND2 has more legends and cues. The icons are distinguished by type of entity and the text and legends provide more detail, such as IP address, server name, OS type, etc. When participants analyzed ND2, they showed more structured analysis than they did with ND1. Contrary to ND1, all participants here, novices and experts, started at the same cue: network segmentation. They recognized the network segmentation of users, administration, management and DMZ, and explained the security advantages of such designs. The diagram in ND2 clearly shows the segmentation using legends and color-codes that the network segmentation becomes very obvious. However, some participants weren't able to explain by the diagram alone some of the network design decisions such as the reason for having two separate DNS servers one of which is present in the DMZ. We will show next how structured analysis helped address this problem.

7.3.3 The Structured Mode

After presenting the diagram ND2 to the participants, we presented the security requirements list. We observed individual differences among experts and novices when assessing a certain requirement and linking it to the diagram entities, but in general participants had more insights compared to the two modes above. However, we observed that participants P1, P3, P4, who organize their thoughts and follow a more structured approach in their analysis of the requirements list, tend to provide more insights and recognize entities that affect security analysis that they did not mention before looking at the requirements list. Using our entity analysis, we compared participants' responses across entities in diagram ND2. Our analysis results indicate that the requirements list

could help both experts and novices: the experts' attention was focused towards a specific security component and help them reach better-informed decisions, and the novices became aware of a requirement and/or its security justification. Consider requirement R12 that requires a split DNS policy: expert participants P1, P3, P4, and P9 were able to map requirement R12 to the split DNS servers shown on the diagram and to state that the network satisfies the requirement, and they were also able to explain why such requirement is important from a security standpoint. Participants P1, P3, P4, P9 demonstrated the patterns: (P→P→UP→P→UP→D),(P→Q→Pro→D→J→J→J→A→J), (Q→C→C→C→J→J),(C→P→J→D→Pro→D→UC→C→A→C→C→J→C→D) respectively. We investigated why P3 and P9 had longer patterns, and we found that they were demonstrating an attacker's attempt against the DNS server and how the split DNS increases the difficulty for attackers to break into the system. Towards the middle of participant P9's pattern, the participant exhibits uncertainty about why this requirement in needed for the system's security and thus they made an assumption in order better comprehend and project before reaching their final decision. Participant P11 was able to state that the requirement R12 is satisfied based on the diagram, but was unclear why a split DNS policy is needed. This is an example of how introducing structure to security analysis, could help analysts become aware of essential security requirements.

Participant P4 took an alternative and more highly structured approach to analysis by drawing a table on a blank piece of paper, listing the requirements numbers, and documenting how the requirement could be satisfied given the information shown on the diagram. During the interview process, P4 has shows more depth when analyzing the results and had confidence in their security analysis. We use the word depth here because P4 was able to refine requirements into specification levels and write down system specification that are essential to satisfy the requirement, and this observation did not occur with any of the other participants.

8 Threats to Validity

In this section, we address threats to construct, internal and external validity.

Construct validity is whether measures actually measure the construct of interest [29]. In our study, the construct of interest is SA, which is comprised of the four levels previously mentioned. One threat to construct validity is the definitions of the codes for each level in the coding frame are ambiguous and not mutually exclusive, such that the codes are inaccurately applied to the wrong statements (i.e., the perception code, if misapplied, may not be measuring instances of perception). To address this threat, we had two researchers (the first and third authors) meet to first discuss the coding frame before applying it to the dataset, after which we identified points of disagreement and reconciled these differences in a subsequent meeting. Recall from Section 4, we computed the inter-rater reliability statistic Cohen's Kappa that showed a moderate to high agreement. Unfortunately, we cannot know when participants are making implicit or unstated assumptions before reaching their decisions. Personality may be a co-factor that can effect whether or not participants make assumptions, since assumption making may be related to over-confidence.

Internal validity refers to whether the conclusions drawn from the data are valid [29]. Based on our coding of the data, we inferred several decision-making patterns in the data set that we report in Section 5. The completeness of the data threatens internal validity, because participants have unspoken perceptions, comprehension, etc. To address this threat, we employed probing questions to prompt participants to make explicit their SA levels, and we checked our observed patterns for accuracy across the dataset, i.e., how many instances of the pattern were consistent with our definition of the pattern. This process led us to discover the reverse SA pattern reported in Section 5.2, which corresponds to differences between western deductive and eastern inductive reasoning styles previously studied in psychology [1, 5, 20].

External validity refers to the extent to which the results of this study can be generalized to other situations [29]. This study is based on grounded analysis, which limits generalizations beyond the data set. While some might argue that our findings are thus too limited, we identified several prospects for future research. This includes whether we can transfer expert assumptions to novices to facilitate transitioning novices from comprehension to decision-making, or how can we improve perception to reduce uncertainty. We plan to study these questions in generalizable, controlled experiments.

9 Discussion and Future Work

In this section, we discuss our results in the context improving the evaluation of security notation in artifacts used in security analysis, and provide suggestions moving forward explaining how hour method could be adapted to improve the design of security training.

9.1 Identifying Effective Cues

Throughout the paper, we discussed how certain analysts were able to perceive certain cues in the artifacts, comprehend them, and then, project and decide on mitigations, accordingly. However, we also showed cases where novice analysts were facing uncertainty during comprehension about a cue, e.g., trying to make sense of its meaning or its possible consequences. In Section 7.1, we showed how one analyst, P10, did not even reach perception; P10 failed to perceive the cue that leads analysts to project the SQL injection attack.

In addition to measure where analysts struggled to move past perception and comprehension, we assessed the effect of improving notations and visual cues by comparing performance between the two network artifacts, ND1 and ND2 (see Section 7), and also by comparing the analysis results of the DFD artifact. Recall from Table 8 how only one participant P8, was able to demonstrate an attack on the diagram. In Section 7.2 we showed how participants exhibited increased uncertainty analyzing the DFD artifact, which indicates how notational elements (or lack thereof) introduce ambiguity, which has a negative impact on analysis.

These observations lead to the following question: How can we avoid situations where experts fail to perceive or comprehend a cue? The SA methodology that we applied helps surface the cues that likely to need support. While experts may have little difficult reaching projection and decision,

novices may need additional information to aid them in reaching these higher levels. In addition to identifying the cues, comparing the results could help find ways to redesign the artifacts in a way that makes the cues either more explicit (improve perception) or more meaningful (improve comprehension). We even envision an adaptive security analysis system that can adapt to the training needs of a security trainee based on their perception and comprehension of cues. If a trainee fails to identify a cue, then the system could provide deeper training with further cues in order to help the trainee perceive vulnerabilities, comprehend its risk, project the impact, and decide on the proper mitigation.

In addition, deciding the appropriate cues could help inform future security experimental designs. For example, consider a study that tests how security analysts evaluate a certain system artifact for threats. In order to draw correct conclusions from the experiment, first we need to evaluate the cues used in the experiment materials (online application, paper, etc.) during a pilot study. Cues can be selected that participants perceive and understand well, and others can be improved if they are misleading or ambiguous.

9.2 Structured Analysis Trade-offs

It is arguable whether or not to provide structured approaches to security analysis. Although our findings in this work are in favor of structured analysis, we think that the decision of favoring structured vs. unstructured analysis is based on realizing the trade-offs between the two approaches, and future research examining those trade-offs is beneficial. Recall from Section 7.3 how a structured approach improved the experts' security analysis of ND2. Only after going through the requirements list, participants P1, and P2, P3, P4 noticed the split DNS design in ND2, which was an improvement over the insecure diagram shown in ND1, but they did not point it out by looking at the diagram alone.

9.3 Ambiguity and Resolution

We intentionally chose the ND1 with minimal cues and information displayed to study the role of ambiguity in decision-making. Consequently, participants interpreted a router icon differently, as a router or firewall. Figure 2 shows the different interpretations of the same entity by four participants, including their statements in order of articulation coded by the SA method. When the notation was improved in ND2, we observed a positive effect on P1 for example. After later seeing the firewall icon in diagram ND2, participant P1 returned to ND1 to correct their prior interpretation to conclude that the ND1 icon was a router.

Participants could not comprehend effectively if they did not perceive appropriate cues that lead to a comprehension, and that could explain having uncertainty patterns appear in our dataset (see Section 5.3), which leads an expert to transition to the ambiguity stage in Figure 2. When analyzing the DFD artifact, for example, one participant attempted to think of all possible interpretations given the absence of specific details from the diagram. In the excerpt below, we show how participant P3 assumed that encryption existed:

{UC}that doesn't really show that here [speaking about encryption

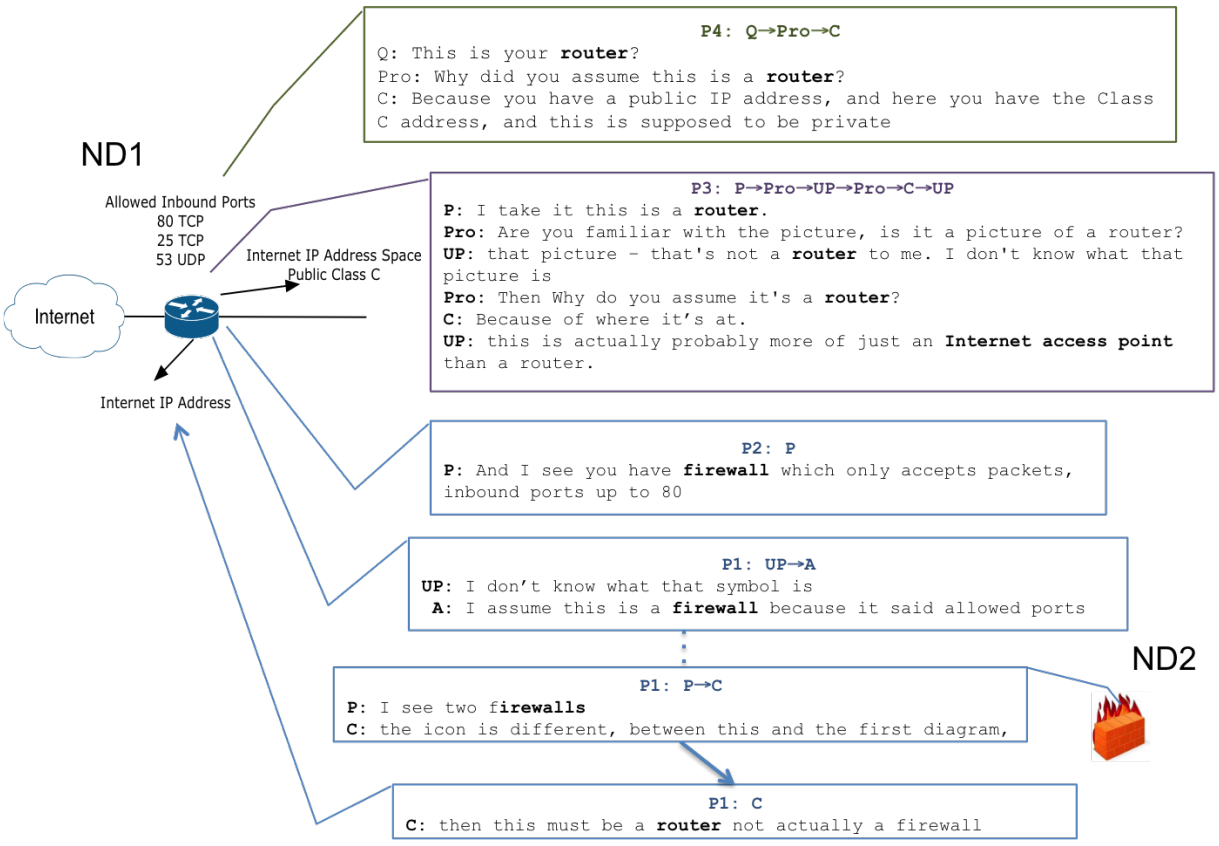

Figure 2: Participants' perceptions of the router icon in diagram ND1

session for sending the digital signature] {/UC}, {A}so if we assume that's an encrypted session and that I am not sharing my digital signature with somebody{/A} {D}then this is trusted{/D}

In a few cases of uncertainty, assumptions helped participants resolve the ambiguity and reach their decisions. Those assumptions were not arbitrary; they were based on former experience and best practices adopted for network security that experts had been exposed to.

The following coded excerpt that was taken from participant P1 and illustrates such an assumption:

{UP}I donÕt see an NTP server on this network{/UP} {C}but I know that Windows Domain Controller can act as NTP{/C}, {A}so I am going to assume that when they install it theyÕll probably leave that box checked because itÕs a default option{/A}. {D}I think that is probably happening here{/UD}

The above assumption is an example of a trust assumption first defined by [28] and then applied to security requirements by Haley et al. [13]. Trust assumptions describe desired behaviors and may be outside the control of the system designer. Based on the background-coded data BG (see Table 1 for a definition of this code), participant P1 has extensive hands-on experience in network

security, which could explain why P1 was comfortable making assumptions about the system. The example above shows an interesting pattern (UP→C→A→UD). Although we did not observe the exact same pattern with other participants, we were able to observe the latter half of the pattern: A→UD as it occurred once for P5 and P11, and twice for P3 and P9. These participants reported significant experience in network security, so one would expect them to be more confident in reaching certain decisions with respect to network artifacts. However, we must not ignore the personality effect: an expert may hesitate to make confidant decisions based on assumptions so they express a level of uncertainty with their decision to be more cautious.

Trust assumption reported by Haley et al. [12, 13] help restrict the domain by narrowing the attention span of the analyst. In SA, a narrowed focus is beneficial for projection, but it can also lock-in the analyst and prevents them from perceiving alarming cues in the environment [10]. Moreover, incorrect assumptions about a system can lead to erroneous requirements specification [27]. Our work could be extended by distinguishing which assumptions are trust assumptions to distinguish the volatility of decisions that depend on assumptions about actors that are outside the system boundary. If those trust assumptions turn out to be untrue, then the security analysis that depends upon those assumptions should be revisited for possible inconsistencies

While our dataset is small in the number of participants, we did observe that experts were more likely to use assumptions to control uncertainty and to reach a decision. In future experiments, we could test if assumptions could provide another metric to distinguish between novices and experts. Being able to distinguish users based on expertise level could have an important impact on designing intelligent and interactive tools to help novice analysts cover more security scenarios in a problem description or specification.

10 Conclusion

In this paper, we present a new approach to assess security expertise and decision-making processes. Our contribution is: 1) a systematic method to apply the Situation Awareness (SA) framework to distinguish security experts effective analysis based on their differences in recognizing attack threat models; and 2) an explanation of the trade-offs of introducing structure to the security analysis process. We summarize our results to show traces across the SA levels in the form of patterns that could be used to distinguish experts from novices, and we plan to further test our theory in future user experiments. We believe that other researchers can use this methodology to evaluate their technical solutions to security analysis by improving notation, presentation, training materials, and most importantly understanding how those solutions improve novice decision making in comparison to experts.

References

[1] J. A. Anderson. Cognitive styles and multicultural populations. *Journal of Teacher Education*, 39(1):2–9, 1988.

[2] A. Arasu, S. Chaudhuri, K. Ganjam, and R. Kaushik. Incorporating String Transformations in Record Matching. In *Proceedings of the 2008 ACM SIGMOD International Conference on Management of Data*, SIGMOD '08, pages 1231–1234, New York, NY, USA, 2008. ACM.

[3] T. D. Breaux and D. L. Baumer. Legally âĂIJreasonableâĂİ security requirements: A 10-year FTC retrospective. *computers & security*, 30(4):178–193, 2011.

[4] P.-C. Chen, P. Liu, J. Yen, and T. Mullen. Experience-based cyber situation recognition using relaxable logic patterns. In *2012 IEEE International Multi-Disciplinary Conference on Cognitive Methods in Situation Awareness and Decision Support (CogSIMA)*, pages 243–250, Mar. 2012.

[5] I. Choi, R. E. Nisbett, and A. Norenzayan. Causal attribution across cultures: Variation and universality. *Psychological bulletin*, 125(1):47, 1999.

[6] J. Cohen. Weighted kappa: Nominal scale agreement provision for scaled disagreement or partial credit. *Psychological bulletin*, 70(4):213, 1968.

[7] G. Digioia and S. Panzieri. INFUSION: A system for situation and threat assessment in current and foreseen scenarios. In *2012 IEEE International Multi-Disciplinary Conference on Cognitive Methods in Situation Awareness and Decision Support (CogSIMA)*, pages 316–323, Mar. 2012.

[8] M. R. Endsley. Design and evaluation for situation awareness enhancement. In *Proceedings of the Human Factors and Ergonomics Society Annual Meeting*, volume 32, pages 97–101. SAGE Publications, 1988.

[9] M. R. Endsley. Toward a theory of situation awareness in dynamic systems. *Human Factors: The Journal of the Human Factors and Ergonomics Society*, 37(1):32–64, 1995.

[10] M. R. Endsley and D. G. Jones. *Designing for situation awareness: An approach to user-centered design*. Taylor & Francis US, 2003.

[11] Y.-H. Feng, T.-H. Teng, and A.-H. Tan. Modelling situation awareness for Context-aware Decision Support. *Expert Systems with Applications*, 36(1):455 – 463, 2009.

[12] C. Haley, R. Laney, J. Moffett, and B. Nuseibeh. The effect of trust assumptions on the elaboration of security requirements. In *Proceedings of 12th IEEE International Requirements Engineering Conference*, pages 102–111, Sept. 2004.

[13] C. B. Haley, R. C. Laney, J. D. Moffett, and B. Nuseibeh. Using trust assumptions with security requirements. *Requirements Engineering*, 11(2):138–151, 2006.

[14] HP. HP top cyber security risks report. Technical Report, Hewlett-Packard Development Company, L.P., 2011.

[15] G. Jakobson. Using federated adaptable multi-agent systems in achieving cyber attack tolerant missions. In *2012 IEEE International Multi-Disciplinary Conference on Cognitive Methods in Situation Awareness and Decision Support (CogSIMA)*, pages 96–102, Mar. 2012.

[16] J. R. Landis and G. G. Koch. The measurement of observer agreement for categorical data. *biometrics*, pages 159–174, 1977.

[17] J. McDermott and C. Fox. Using abuse case models for security requirements analysis. In *Computer Security Applications Conference, 1999. (ACSAC '99) Proceedings. 15th Annual*, pages 55–64, 1999.

[18] NIST. NIST/ITL Special Publication (800), Jan. 2015. 00000.

[19] OWASP. OWASP Top Ten Project - OWASP, Oct. 2014.

[20] K. Peng and R. E. Nisbett. Culture, dialectics, and reasoning about contradiction. *American Psychologist*, 54(9):741, 1999.

[21] B. Potter and G. McGraw. Software security testing. *Security & Privacy, IEEE*, 2(5):81–85, 2004.

[22] J. SaldaÃśa. *The coding manual for qualitative researchers*. Number 14. Sage, 2012.

[23] SANS. SANS 20 Critical Security Controls Solutions Directory, Oct. 2014.

[24] K. Schaefer, D. Billings, and P. Hancock. Robots vs. machines: Identifying user perceptions and classifications. In *2012 IEEE International Multi-Disciplinary Conference on Cognitive Methods in Situation Awareness and Decision Support (CogSIMA)*, pages 138–141, Mar. 2012.

[25] G. Sindre and A. L. Opdahl. Capturing security requirements through misuse cases. *NIK 2001, Norsk Informatikkonferanse 2001, http://www. nik. no/2001*, 2001.

[26] A. Van Lamsweerde, S. Brohez, R. De Landtsheer, and D. Janssens. From system goals to intruder anti-goals: attack generation and resolution for security requirements engineering. *Proc. of RHAS*, 3:49–56, 2003.

[27] A. Van Lamsweerde and E. Letier. From object orientation to goal orientation: A paradigm shift for requirements engineering. In *Radical Innovations of Software and Systems Engineering in the Future*, pages 325–340. Springer, 2004.

[28] J. Viega, T. Kohno, and B. Potter. Trust (and Mistrust) in Secure Applications. *Commun. ACM*, 44(2):31–36, Feb. 2001.

[29] R. K. Yin. *Case study research: Design and methods*, volume 5. Sage, 2009.

Appendix

A. LIST OF REQUIREMENTS USED IN ARTIFACT ND2

R 1: Company X's network, with the exception of the publicly available services which will reside in a demilitarized zone (DMZ), will be unavailable for connections initiated from the Internet to Company X's network

R 2: The employees of Company X will be required to use a web proxy server for connections to the World Wide Web.

R 3: Company X will harden and secure the services and operating systems of critical systems

R 4: Company X will implement web content filtering and shall block inappropriate (pornographic) web sites

R 5: Company X will implement a Windows domain, and will manage server and user system configurations through group policy centrally on the network

R 6: Company X will implement a electronic mail relay, relaying mail from the Internet through a mail filter, which will filter spam and malware as mail enters Company X's network.

R 7: Company X will require strong passwords (8 characters with complexity) for all user accounts.

R 8: Company X will implement multiple networks (management, user, data center), and will implement strict access controls between each network.

R 9: Company X will deploy system logging capabilities at all critical systems and will gather the logs centrally for review and response

R 10: Company X will implement system time synchronization on the network for logging and auditing capabilities.

R 11: Company X will implement multiple Intrusion Detection Systems (IDS) in multiple places on the network and shall audit regularly

(a) File System Integrity IDS sensors shall be implemented

(b) Network packet pattern matching IDS sensors shall be implemented.

R 12: Company X shall implement split Domain Name System (DNS) services.

R 13: Company X will monitor network traffic with packet sniffers.

R 14: Company X will implement centralized system/service availability monitoring.

R 15: Company X will administer all systems either interactively from the console or remotely from an isolated management network.