CS 2001: Research Topics

Writing up and Presenting Your Research

Prof. Adriana Kovashka
University of Pittsburgh
September 9, 2025

Plan for this lecture

- Writing
  - Writing timeline
  - Writing style: Consistency, aesthetics, emphasis, inclusivity
  - Submission logistics (paper limit, metadata)
  - Writing rebuttals
- Presenting
  - Planning how you present depending on audience
  - Oral delivery tips

Writing

Reminder: Structure of a Paper

- Introduction
  - Motivation
  - Gaps in science, deficiencies in prior work
  - Brief overview of key ideas
- Related work
  - Group by topic, state how proposed work different
- Approach
  - Explain clearly each step
  - Distinguish standard aspects from your contribution
- Experiments
  - Describe setup, data, metrics, etc.
  - Compare to papers from the recent literature
  - Verify contributions of each method component through ablations

Getting started

- Download conference/journal template
- Set up Overleaf.com project directory
- Write extended outline
  - What will each paragraph say? – summarize with one bullet
  - (Show to advisor, iterate based on feedback)
- Then fill in details
  - Show to advisor, iterate based on feedback

Timeline

- You don’t have to wait for all results to be finished before you start writing
  - Write parts of intro, including motivation and claims, as early as you have a clear idea—helps you identify the work you still need to prove claims
  - Write related work—you might find more relevant baselines, methods to compare to
  - Write parts of approach—good way to make sure approach is sound, and your advisor is on the same page
  - Write plan for experiments–to help identify what experiments still need to be conducted
- Iterating on writing with advisor can take 2-4 weeks for a first-time submission, so leave plenty of time!

Writing Style

- Your advisor may be used to writing papers in a particular way
- A certain style may be common in your research community (and not others)
- Talk to you advisor about their writing style
- Talk to them about their pet peeves
  - Pet peeve: “minor annoyance that an individual finds particularly irritating” (Wikipedia)
  - These might be about consistency, aesthetics, etc.

Consistency

- Strive to describe and run things in a way that is consistent across methods etc.
  - Don’t run one method with one set of settings and another with other settings, unless you have a reason
- Strive to describe and format text consistently
  - Don’t use Capitalization in one section Title but not Another
  - Don’t use bold for emphasis in **one section** and underline *for another*
  - Use the same string for the same conference, not “CVPR” and “32nd CVPR” and “cvpr”

Aesthetics

- Make sure tables and figures are legible
  - Font large enough
  - Colors in lines in figures different enough
- Make sure it's easy to navigate through paper
  - Provide section headers, paragraph headers, where appropriate

Inclusivity

- Use neutral pronouns – “they” rather than “he”/”she”
- Avoid potentially hurtful terms, e.g. “master and slave”
- Consider including examples that a broader audience will find intuitive: do not assume understanding of any particular culture
- Use colors everyone will read easily; avoid red/green and blue/purple

Emphasis

- Writing is a craft—think of yourself as a woodcarver—every detail matters
- Avoid sentences (in intro) that are not crucial for the point you're trying to make—you're just losing the reader's attention and an opportunity to emphasize your contribution
- Be careful about *how* you say what you—what is the accent?

Making Paper Fit the Page Limit

- Your draft is 10 pages but limit is 8 pages—no problem!
- Remove papers from related work that aren't all that related, or trim your descriptions of them
- Read through whole paper, and rephrase sentences as needed to make your narrative more concise, *punchier*
- Revise paragraphs ending in a few words (trailing lines)
- Remove unnecessary words and characters
  - Examples: "first X and second X" -> "first and second X", "that"
  - Change tense to lose characters, replace subsection headers with paragraph headers, etc. — it's easy, but it takes practice!
- Last resort: use negative vspace, \vspace{-2mm}
  - May be disallowed by conference
- You will take many passes through your paper, and each time, you'll find ways you can say it more concisely

Note on Equations

- Explain all variables
- Explain purpose, idea in English
- Refer to equations by name rather than just ID later in the text

https://www.microsoft.com/en-us/research/wp-content/uploads/2016/07/How-to-write-a-great-research-paper.pdf

More from readings: Jones

- Don’t wait: write
  - Idea -> write -> research
- Identify your (one, sharp) key idea
  - “infect mind of reader with your idea, like a virus”
  - “can you hear the ping?”
- Tell a story
- Nail your contributions to the mast
  - Contributions should be listed, and should be refutable
- Related work later
- Put readers first
  - Present the intuition, examples, then details
- Listen to your readers
- Use simple, direct language

Common Mistakes in Writing

- Starting introduction with platitudes (“deep learning has greatly improved the state of the art…”)
- Too long related work, plus not saying what’s different about proposed work
- Method mixes standard techniques with novel contribution
- Not clear what motivation, key contribution are
- Results—unclear where gains in performance come from
- Results—focusing just on the negative outcomes—results won’t be perfect, but the question is whether there is evidence that your proposed idea has benefits

Should you get writing help from AI?

- Check conference rules
- Even if allowed, I strongly recommend doing the writing yourself
- Gives you a chance to read and reflect on what you are stating, may offer an opportunity to reframe the work better

Other Paper Submission Logistics

- Paper ID (may need to register paper before submission deadline)
- Title
- Author list and order
- Subject areas

Writing Rebuttals: Audience

"We write rebuttals for two different audiences

1. The reviewers, who have read your paper (to varying degrees), but may have forgotten some of the details or didn't understand them in the first place.
2. The AC, who is likely even less familiar with your work, and a good guiding principle is to assume that all they will read is the set of reviews and the rebuttal."

Source: Devi Parikh et al., https://deviparikh.medium.com/how-we-write-rebuttals-dc84742fece1

Writing Rebuttals: Goals

"1. For the reviewers: clarify doubts, answer questions, correct misunderstandings, push back on mischaracterizations, and make a good-faith effort to incorporate feedback and improve your work.

2. For the AC: convince them that you have made a good-faith effort, present a representative summary of the reviews, help them understand if the reviewer concerns were addressed, call out bad-faith reviewing, and ultimately, *help them make a decision.*

[…] Would a neutral third-party be able to tell if the reviewer concerns were addressed *purely based on your rebuttal* (without reading the paper or the reviews again)?"

Source: Devi Parikh et al., https://deviparikh.medium.com/how-we-write-rebuttals-dc84742fece1

Writing Rebuttals: Example Process

1. Itemize reviewer comments
2. Brain dump possible responses
3. Write a draft rebuttal
4. Review and revise

Source: Devi Parikh et al., https://deviparikh.medium.com/how-we-write-rebuttals-dc84742fece1

Writing Rebuttals: Example Process

Reviewer comment	Author 1	Author N
1. There existing several related papers discussion of the using human attention map in image captioning and visual question answering. For example, (1) Liu et al. Attention correctness in neural image captioning. (2) Qiao et al. Exploring human-like attention supervision in visual question answering. Please illustrate the differences with these papers.	The papers mentioned provide attention supervision over the attention layer. Our central argument for this and the next point will be that Grad-CAM is more faithful than attention. In order to show this I am planning on doing occlusion studies in the proposal space and compare that with the attention weights and the Grad-CAM proposal importance weights. Also with attention supervision only the layers before the attention layer can be updated, but with HINT all layer weights can be updated. Include lines from the paper. Att supervision doesn't work.	We should first very clearly say what you say in the first sentece of your response. And expand on that a bit if needed to make the point clearly. Does the paper differentiate our work from these work R2 cites or other such works? If so, we should clearly say in the response "As discussed in LXYZ-ABC..." You can then make the point about which layers can be updated. We can then additionally make the "central argument" point. But the direct response should be clear / not confused with the description of a new experiment and such.
2. It seems that the ground-truth attention map is used for the VQA task. For the captioning task, although no ground-truth attention map is used, the segmentation maps are used. As such compare with other methods, strong information about the image are incorporated, which should results in performance improvements.	Human attention or segmentation maps are used only during training and not during testing. While we agree that this is extra information used during training, we show why other approach fail to utilize this information to achieve improvements in performance during test time. Only a fraction of images in VQA have Human attention. Also if it is possible to such a good boost with just human attention, people would start collecting. Also HATs are important to know if models are making the right decision for the right reasons.	"we show why other approach fail to utilize this information to achieve improvements in performance during test time." You'll have to point to a specific experiment in the paper / lines in the paper / table in the paper and reproduce the curcial numbers here to support this claim. Then you can say this is only at training time, not at test time. (I think the reviewer already knows this. So starting with this response is not a strong start.)
3. For the alignment between human attention and network importance, it seems that there only exists one human attention map. However, for the network importance, specifically the VQA task, when the question is different, we should pay different attentions to different local regions. Therefore, how to make the corresponding alignments? The human attention is stable, while the network importance varies dynamically.	This is incorrect. HAT is question dependent. i.e. there exists different maps for different questions.	Yup. Say there are different human attention maps for different questions. So the human attention map is also question dependent.
The state of the art and beyond in the field is moving away from such human guided approaches. Localization is already being done in a wholly unsupervised fashion using embeddings for example. . Also the proposed approach is not scalable. Verification is done through human studies which is fine but again not scalable.	Highly opinionated with no citations. I don't agree that state of the art "and beyond" (whatever that means) is moving away from human guided approaches. Disagree that Localization is done in a wholly unsupervised way. Approaches for semi-supervised localizations exist, but they are still significantly worse than fully supervised approaches (xx% diffence in ILSVRC localization). Also disagree with the comment that verification is done only through human studies. In section 5 we quantitatively evaluate task performance , and in section 6 we quantitativly evaluate grounding, both of which show the effectiveness of HINT without requiring human workers. Our Human studies are required to show that our HINTed models are more trustworthy to humans than base models, which is needed not just for generalization but also necessary as more algorithmic decisions are made in the society	Also maybe make some point about how without human guidance sure models have good accuracies, but they can be heavily biased (and give examples from VQA maybe)?
It would be helpful to have an ablation like study in which you increase or decrease the level of HINT's and see what happens to get a deeper insight into what you are doing.	I can set this up. We can vary the amount of HATs used and examine how the performance varies	Yup

Source: Devi Parikh et al., https://deviparikh.medium.com/how-we-write-rebuttals-dc84742fece1

Writing Rebuttals: Tips

1. **Start positive.** Provide a summary of the reviews, highlighting positive things that reviewers said about your work. Rebuttals focus mostly on responding to perceived negatives, don't let RACs forget about the strengths along the way.
2. **Order matters.** Start with the biggest concerns that you have good answers for and work your way to less clear-cut responses and minor points.
3. **Let reviewers speak for themselves, then respond directly.** Quote the core of the reviewer's question or concern concisely and completely. Then before saying anything else, respond to it directly. And then give details, describe context, or explain your position.
4. **Be conversational.** Notice the conversational nature of the example responses above. It makes it easier for RACs to follow, and the responses are less likely to be perceived as being combative.

Source: Devi Parikh et al., https://deviparikh.medium.com/how-we-write-rebuttals-dc84742fece1

Writing Rebuttals: Tips

5. **Respond to the intent of the questions.** Don't feel trapped to only discuss the quoted concern — also address the intent of the comment. For example, "Why didn't you evaluate on GLORP3?" may generally be calling your experiments into question. Answer, but then point out that you've already evaluated on X,Y, and Z which should be sufficient! Note that it is useful for other RACs to be reminded of your extensive experimental evaluation. A first glance at a reviewer comment suggesting otherwise could leave a false impression.
6. **Don't be afraid of emphasis**. "Row 2 in Table 4 shows *exactly* that." "We do NOT need a human-in-the-loop at test time." Notice that many of the responses above are not just direct, but also have emphasis (in tone if not formatting of text).
7. **Feel free to set the stage.** If it seems like all reviewers missed a central point, a concise, crisp recap of the main point could help.
8. **Keep things self-contained.** Assume RACs don't remember much about your paper and that they likely won't read it again in detail. Re-introduce any acronyms, remind them of relevant details of an experimental setup. Notice that all the responses above likely make sense to you even though you may be unfamiliar with the papers (and in some cases the names and details are made up).

Source: Devi Parikh et al., https://deviparikh.medium.com/how-we-write-rebuttals-dc84742fece1

Writing Rebuttals: Tips

9. **But get credit for details you already included.** That said, if something a reviewer asked for was already in the paper, say so. Give them line/Table/Figure numbers, and then restate it in the rebuttal. The references back to the main paper are to establish credibility with all RACs that the paper was not lacking important details. (They are not necessarily to have RACs go back and look at the paper.)
10. **Consolidate common concerns.** Save space by responding to multiple reviewers at once if they share related concerns.
11. **Color-code reviewers.** Notice above the trick to color-code reviewers. Make it as easy as possible for reviewers to spot responses that are relevant to them — even when things are merged or not in reviewer order.
12. **Stats speak louder than words.** Rather than argue with RACs, give them data/stats to back your claim up. These can be statistics/analysis of your data or results. Or the results of additional experiments you run to respond to their concern (if allowed by the venue). Every time you find yourself having a different opinion than the reviewer, ask if you can establish that with data. You can always provide the intuitive arguments after you've settled the issue with data.

Source: Devi Parikh et al., https://deviparikh.medium.com/how-we-write-rebuttals-dc84742fece1

Writing Rebuttals: Tips

13. **Don't promise, do.** Instead of saying "We will discuss Singh et al. in the paper.", provide a discussion in the rebuttal. Instead of saying "We will explain what D_{RT} stands for in the paper", explain what it stands for in the rebuttal. And then also add that you will add it to the paper. It makes it significantly easier for RACs to trust that you will make the promised changes.
14. **Be receptive and reasonable.** Most RACs will appreciate it. Plus, it is just the better thing to do — these are your colleagues! :)
15. **Be transparent.** Reviewers hinted at an additional experiment but the venue doesn't allow it? Say so. They asked about intuitions about a trend but you don't have any? Say that you've thought about it but don't have any good ideas, and will continue to investigate it. Don't have enough GPUs to run the experiment they asked for? Say so.
16. **Shine a spotlight on reviewers acting in bad-faith.** In some circumstances, a reviewer may not be adhering to reviewing best practices or may not have taken the reviewing role seriously. It can be important to make sure the other RACs realize this and appropriately discount their review. Pointing out unreasonable or unsubstantiated comments and referencing other reviewers that disagree can help. This can also include confidential comments to the AC (where applicable).

Source: Devi Parikh et al., https://deviparikh.medium.com/how-we-write-rebuttals-dc84742fece1

Writing Rebuttals: Tips

17. **Acknowledge reviewer efforts.** On the other hand, if a reviewer goes above and beyond to be constructive, thank them for it. Typo list? Thank you. Pointers to relevant work? Thank you. Detailed musings about future work? Thank you. Add at least a short blurb acknowledging these things!
18. **Don't forget the humans on the other end.** Keep in mind that this is not just a scientific but a sociopolitical interaction with other humans :) So decide whether you'd like to be argumentative and risk your reviewer taking a strong stand against the paper, or if you'd like to work towards a common ground. Finding points where you do agree with the reviewer and acknowledging them can help with the latter.

Source: Devi Parikh et al., https://deviparikh.medium.com/how-we-write-rebuttals-dc84742fece1

Presenting

Some Guidelines

1. Consider the audience & event
2. Structure based on audience and time
3. How you say it matters
4. Don’t bury the lead
5. Be concise
6. Give credit
7. Q&A: the unknown unknowns
8. Delivery tips
9. Clear slides

Slide credit: Kristen Grauman, Adam Lee

Consider Audience and Event

- Your group meeting?
- CVPR?
- Job talk to entire CS department?
- Interdisciplinary?
- K-12?
- Formal? Casual?

- Degree of detail
- Degree of jargon
- Depth vs. breadth
- Latest vs. arc of progress

Slide credit: Kristen Grauman

Structure based on Audience and Time

Your audience: Generally smart individuals

- Computer Scientists? Yes
- In your area? Maybe
- Knowledgeable about your problem? Probably not

Time is usually limited

- Conference talk: 20 minutes or so
- Job talk: < 1 hour

This is not a lot of time…

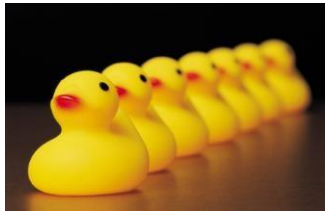
Bottom line: *Your talk should be an advertisement for your paper(s)*

Slide credit: Adam Lee

Structure based on Audience and Time

This is a hard problem…

… with interesting applications…



… that builds on prior work…

… in a verifiable way

Two sub-parts:

- You do something that has not been done
- You use neat technological advancements to do this

Hint: Try to give audience one good take-home point

Slide credit: Adam Lee

How You Say It Matters

Body language says a lot

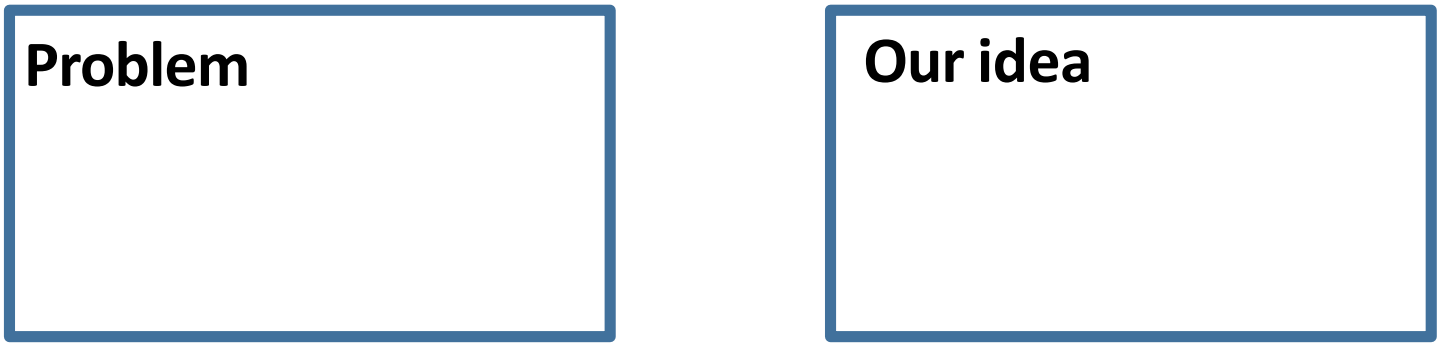
- Make eye contact with your audience
  - *Corollary*: Face your audience
- Some movement is good
- Don't speak too fast (or too slow!)

Make useful slides

- Provide a topic outline to structure your talk
- One primary idea per slide
- Use slide titles to convey take-away message
- Do not read your slides!
- A picture is worth a thousand words…

Slide credit: Adam Lee

Don’t Bury the Lead



- Don’t leave contribution implicit
- Reiterate and rephrase message throughout
- Verbally: give salient markers; “Important”…”stress that”...
- “Punchlines” for results

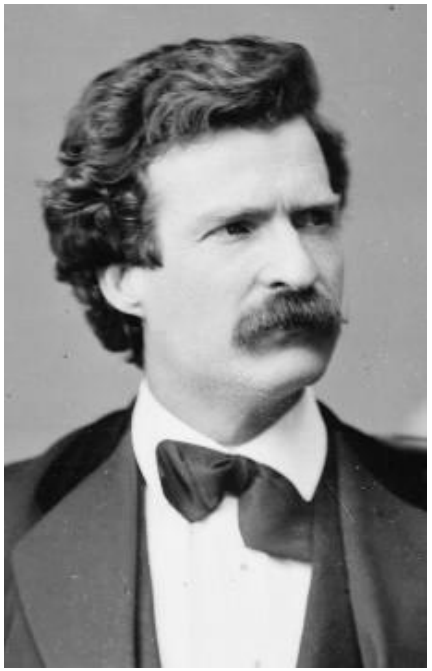
Slide credit: Kristen Grauman

Be Concise

"I didn't have time to write a short letter, so I wrote a long one instead."

— **Mark Twain**

- Prep the "concept bullets"
- Breathe, and use fewer words
- Short text phrases (not sentences)
- Not every detail needs to surface



https://en.wikipedia.org/wiki/Mark_Twain

Slide credit: Kristen Grauman

Give Credit

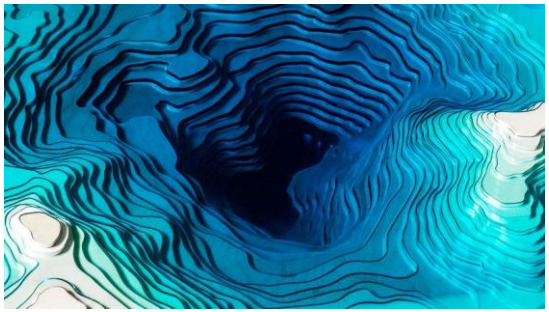
- Paint the big picture of literature for context
- (Clusters of) related work & key contrasts
- Give credit for borrowed slides, per slide

[Slide credit: Jane Smith]

Slide credit: Kristen Grauman

Q&A: The Unknown Unknowns

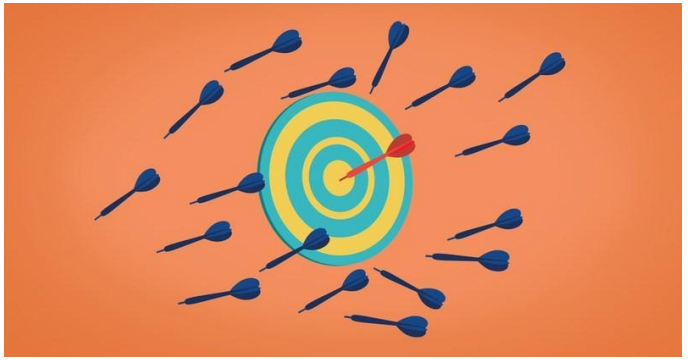
- Guess likely questions & prepare
- Don’t skip to backup slides unless necessary
- Sometimes it’s better to defer a question
- Answer, then stop
- Share feedback with co-authors afterwards



https://duffylondon.com/product/tables/abyss-horizon/

Slide credit: Kristen Grauman

Delivery Tips



- Practice and get feedback; iterate
- The first slide - what will you say?
- Be loud enough
- Use pauses
- Flow: think through transition *in* and *out* of each slide
- Qualitative examples: say something about one or two
- Manage time: stopwatch, prevent derail
- If you're skipping something, then skip it
- Think about where you want to stand / test the room
- Check the laptop, AV
- Nerves: "If you're nervous, it means you care" ~Trevor Darrell

https://www.ted.com/playlists/497/practice_makes_perfect

Slide credit: Kristen Grauman

Carefully Done Slides…

… reduce cognitive load:

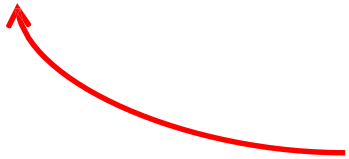
- Animation – to focus attention
- Font size – 28+ for main text
- Simplest visual possible to make the point
- Consistency: font size, capitalization, alignment…
- Avoid jitter of text placement in consecutive slides
- One liners where possible
- Use color to link pieces of equations
- Delete “Hi my name is …” from notes of first slide!
- Avoid content-free “Thank you!” slide

Slide credit: Kristen Grauman

More Tips and Tricks

Practice makes better

- *Alone*: Work on your "script," smooth out transitions
- *Research group*: Get used to other people being around
- *Broader population*: Assess comprehensibility to outsiders



e.g., other grad student friends, department seminars, etc…

Do you *really* want that laser pointer?

"Flash" is good, but too much flash is distracting

- *Good*: Animations to progressively build large diagrams or equations
- *Bad*: Animating every slide transition and every line of text…

Get out of your head and into your talk ☺

Slide credit: Adam Lee

Example: A Mediocre Talk

Towards Quantitative Analysis of Proofs of Authorization: Applications, Framework, and Techniques

- Adam J. Lee (University of Pittsburgh)

Ting Yu (North Carolina State University)

Slide credit: Adam Lee

Proofs of Authorization

- Trust management systems are used for access control in open systems
- Logical proofs are constructed at runtime to determine whether a given principal is allowed to access some specific resource
- Rather than simply interpreting a proof as a binary decision, we aim to analyze these proofs in a more quantitative manner

Slide credit: Adam Lee

Framework

Conceptually, a trust management system contains

- A set *P* of principals
- A set *S* of resources
- A set *C* of credentials that make policy statements
  - Abstraction: s <- q, signed by p
    - P says that anyone that satisfies q can access s
    - P must control s
- An inference scheme $F : P \times S \times 2^C \rightarrow \{true, false\}$

Slide credit: Adam Lee

Views

- We assume principals have some *view* of the system.

$$res(s \leftarrow q) \mapsto s \quad (1)$$

$$ac(s) \mapsto \{c \in \mathcal{C} \mid res(c) = s\} \quad (2)$$

Definition 1 (View): The *view* that some principal $p \in \mathcal{P}$ has of the protection state of a trust management system is defined as a three tuple $v_p = \langle S \subseteq \mathcal{S}, C \subseteq \mathcal{C}, \mathcal{A} \rangle$, where for each $s \in S$, $ac(s) \subseteq C$, and $\mathcal{A}$ is the abstraction of any auxiliary information that p has about the system.

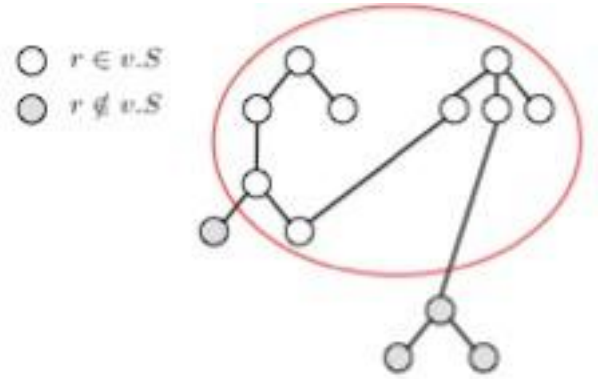


Figure 1. Graphical representation of a system view.

- This allows us to define proof scoring functions, score: *P* x *S* x *V* -> *T*

Slide credit: Adam Lee

Properties of Scoring Functions

Required Properties

1. Deterministic
2. Simple ordering
   - $F(A,s,C)=T \wedge F(B,x,C)=F \rightarrow score(A,s,v) > score(B,s,v)$
3. Authorization relevant

Optional Properties

4. Interpretable
5. Bounded
6. Monotonic

Slide credit: Adam Lee

Overview of RT_0

Basics

- Public keys identify users
- Roles group users

Four types of rules

- Simple member: A.R <- B
- Simple containment: A.R <- B.R'
- Linking containment: A.R <- A.R1.R2
- Intersection containment: A.R <- $B_1.R_1 \cap \ldots \cap B_n.R_n$

Policies built up using combinations of these rules

Slide credit: Adam Lee

Scoring Functions: Take 1

- Assumptions
  - Simplified model
  - User designing function only knows about A.R
    - Knows all rules defining A.R
    - Understands semantics of every role “used” in these rules
  - Each credential associated with a vector w_i
    - All entries > 0
    - $||w_i||_1 = 1$

Slide credit: Adam Lee

Scoring Functions: Take 1

Algorithm 1 A simple recursive scoring scheme.

Function score($p \in \mathcal{P}, A.R \in \mathcal{R}, v \subseteq \mathcal{V}$) : $\mathbb{R}$
// Filter credentials and initialize storage vector
$C = \{c_i \mid c_i \in v.C \wedge \mathsf{head}(c) = A.R\}$
Discard all $c_i \in C$ of the form $A.R \leftarrow P', P' \neq P$
$\overline{s} = [1, 0, \ldots, 0]$ // vector in $\mathbb{R}^{|C|+1}$

for all $c_i \in C$ **do**
    $\overline{w_i} = v.\mathcal{A}.\mathsf{weight}(c_i)$ // weight vector for c_i
    if $c_i = A.R \leftarrow P$ **then**
        $\overline{t} = [1, 1]$
    else if $\mathsf{body}(c_i) = B_1.R_1 \cap \cdots \cap B_k.R_k$ **then**
        $\overline{t} = [1, B_1.\mathsf{score}(p, B_1.R_1), \ldots, B_k.\mathsf{score}(p, B_k.R_k)]$
    else if $\mathsf{body}(c_i) = A.R_1.R_2$ **then**
        Find $B \subseteq A.R_1$ such that $\forall B_j \in B : P \in B_j.R_2$
        $\overline{t} = [1, max_{B_j \in B}(B_j.\mathsf{score}(p, B.R_2))]$
    if $\overline{t}$ contains any 0 entries **then**
        $\overline{s}[i] = 0$
    else
        $\overline{s}[i] = \overline{t} \cdot \overline{w_i}$

// Get master weight vector and combine all weights
$\overline{w} = v.\mathcal{A}.\mathsf{weight}(A.R)$
return $\overline{s} \cdot \overline{w}$

Slide credit: Adam Lee

Scoring Functions: Take 1

Theorem 1: The function score : $\mathcal{P} \times \mathcal{R} \times \mathcal{V} \rightarrow \mathbb{R}$ defined in Algorithm 1 satisfies the *deterministic*, *simple ordering*, *authorization relevant*, *bounded*, and *monotonic* properties.

Proof sketch:

- Deterministic: Obvious
- Simple ordering: Members scored with a positive value, non-members not scored (Line 16)
- Authorization relevant: Only credentials defining A.R used when computing a score (Line 3)
- Bounded: $||w_i||_1 = 1$ for all credentials c_i, so bounded above by 1. All entries in each $w_i > 0$, so bounded below by 0.
- Monotonic: No negative entries in any w_i, so score can never decrease by getting more information

Slide credit: Adam Lee

Scoring Functions: Take 2

- Assumptions
  - More general system model
  - User knows nothing about policies
    - Structural information is discovered at runtime
    - Like RT, SecPAL, Gray, etc.
- Basic idea: Compute score based on number of ways that a policy can be satisfied

Slide credit: Adam Lee

Scoring Functions: Take 2

$$\text{score}(p, A.R, v) = \sum_{(C_i, w_i) \in \text{osets}_\omega(v.C, A.R)} w_i \cdot \frac{1}{2}^i$$

Weighting functions ω : 2^C x 2^{2C} -> [0,1] weight the contribution of each proof

$$\omega_{len}(C_s, \_) = \gamma^{\max_{p \in \text{paths}(C_s)}(\text{length}(p))}$$

$$\omega_{ind}(C_s, C) = 1 - \frac{\max_{C_i \in C \setminus \{C_s\}}(|C_s \cap C_i|)}{|C_s|}$$

$$\omega_{li}(C_s, C) = \alpha \cdot \omega_{len}(C_s, \_) + \beta \cdot \omega_{ind}(C_s, C)$$

Slide credit: Adam Lee

Scoring Functions: Take 2

Theorem 2: The class of scoring functions score : $\mathcal{P} \times \mathcal{R} \times \mathcal{V} \rightarrow \mathbb{R}$ represented by Equation 6 satisfies the *deterministic*, *simple ordering*, *authorization relevant*, *bounded*, and *monotonic* properties, provided that the scaling function $\omega : 2^{\mathcal{C}} \times 2^{2^{\mathcal{C}}} \rightarrow [0, 1]$ used to parameterize the osets function is deterministic.

- Proof sketch
  - Deterministic: ω is deterministic, so score is too
  - Simple ordering: Same as function #1
  - Authorization relevant: trivial by def'n of proofs of authorization
  - Bounded: Based on geometric series in score converging to 1 when summed infinitely

Slide credit: Adam Lee

Scoring Functions: Take 2

- Proof Sketch (cont)

Monotonic. To prove the monotonicity of Equation 6, we proceed by induction. We first assume that principal p has previously discovered the (ordered) collection of proofs and weights $(C_1, w_1), \ldots, (C_n, w_n)$ for the role $A.R$. The base case that we must consider is that a new pair (C_s, w_s) is discovered such that no weight w_i is less than w_s. In this case, this new pair will introduce a new term to the end of the summation calculated by Equation 6, thereby increasing principal p's score for the role $A.R$.

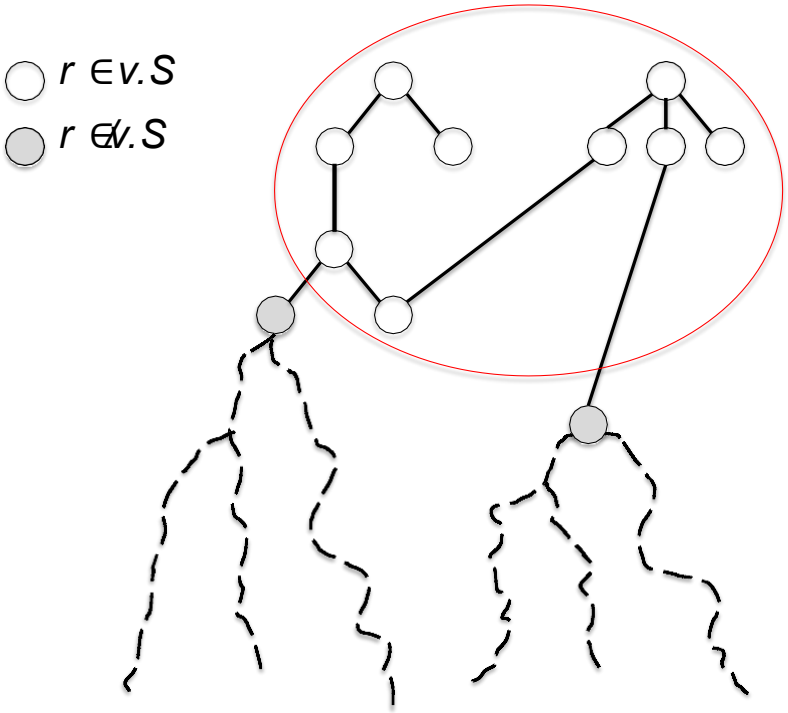
Assume that (C_s, w_s) can be inserted before up to n terms in the sequence of (c_i, w_i) pairs while still preserving the monotonicity requirement. Now, assume that p has previously found proofs of authorization with the sequence of weights $S = (C_1, w_1), \ldots, (C_i, w_i), \ldots, (C_{i+n}, w_{i+n})$ and has now discovered a (C_s, w_s) pair such that $w_s > w_i$, thereby needing to be inserted before $n + 1$ terms in the sequence S. We first note that replacing (C_i, w_i) with (C_s, w) will generate a sequence S' that—when used in conjunction with Equation 6—will produce a score greater than that produced using S, since $w_s > w_i$ and all other terms are the same. By the inductive hypothesis, (C_i, w_i) can then be re-inserted before the n final terms of S' while still preserving monotonicity.

Slide credit: Adam Lee

Composing Scoring Functions

Motivation

- Perfect information known within a security domain
- Less information known outside of security domain



Slide credit: Adam Lee

Definitions

Definition 7 (Horizon): The *horizon* of a view v is defined as the set of resources $\mathsf{horizon}(v) = \{r \mid \exists c \in ac(v.S) : r \in \mathsf{body}(c) \wedge r \notin v.S\}$. That is, $\mathsf{horizon}(v)$ contains all resources mentioned in the body of policies protecting resources in $v.S$ that are not themselves in $v.S$.

Definition 8 (Sequential Composition): Assume that we have a view v, a principal p, a resource r, and two authorization scoring functions score_1 and score_2. We say that score_1 is *sequentially composed* with score_2 if there exists a resource $r' \in \mathsf{horizon}(v)$, a principal p', and a view v' such that $\mathsf{score}_2(p', r', v')$ is calculated when calculating $\mathsf{score}_1(p, r, v)$.

Definition 9 (Order-Preserving Homomorphism): Let $\mathsf{score}_1 : \mathcal{P} \times \mathcal{S} \times \mathcal{V} \rightarrow \mathcal{T}_1$ and $\mathsf{score}_2 : \mathcal{P} \times \mathcal{S} \times \mathcal{V} \rightarrow \mathcal{T}_2$ be two authorization scoring functions. Let $t_1 \in \mathcal{T}_1$ (resp. $t_2 \in \mathcal{T}_2$) be a threshold such that if $\mathsf{score}_1(p, s, v) \leq t_1$ (resp. $\mathsf{score}_2(p, s, v) \leq t_2$) then p cannot access resource s. Similarly, if $\mathsf{score}_1(p, s, v) > t_1$ (resp. $\mathsf{score}_2(p, s, v) > t_2$) then p can access resource s. A function $f : \mathcal{T}_2 \rightarrow \mathcal{T}_1$ is an *order-preserving homomorphism* from $\mathcal{T}_2$ to $\mathcal{T}_1$ if and only if (i) $t \leq t_2 \rightarrow f(t) \leq t_1$, (ii) $f(t_2) = t_1$, and (iii) $t > t_2 \rightarrow f(t) > t_1$.

Slide credit: Adam Lee

Composition Theorem

Theorem 3: Let $\mathsf{score}_1 : \mathcal{P} \times \mathcal{S} \times \mathcal{V} \rightarrow \mathcal{T}_1$ and $\mathsf{score}_2 : \mathcal{P} \times \mathcal{S} \times \mathcal{V} \rightarrow \mathcal{T}_2$ be two authorization scoring functions that satisfy the *deterministic*, *simple ordering*, *authorization relevant*, *bounded*, and *monotonic* properties. If there exists an order-preserving homomorphism f between $\mathcal{T}_2$ and $\mathcal{T}_1$, then the sequential composition of score_1 with score_2 is also *deterministic*, *simple ordering*, *authorization relevant*, *bounded*, and *monotonic*.

Slide credit: Adam Lee

Neat Corollaries

Corollary 1: The result of sequentially composing the authorization scoring functions defined by Algorithm 1 and Equations 6–9 using the order-preserving homomorphism $f(x) \mapsto x$ is an authorization scoring function that is also *deterministic*, *simple ordering*, *authorization relevant*, *bounded*, and *monotonic*.

Corollary 2: Let score : $\mathcal{P} \times \mathcal{S} \times \mathcal{V} \rightarrow \mathcal{T}_1$ be a authorization scoring function that satisfies the *deterministic*, *simple ordering*, *authorization relevant*, *bounded*, and *monotonic* properties and let v be a view. The result of sequentially composing score with an arbitrary any number of other *deterministic*, *simple ordering*, *authorization relevant*, *bounded*, and *monotonic* authorization scoring functions along horizon(v) is an authorization scoring function that is also *deterministic*, *simple ordering*, *authorization relevant*, *bounded*, and *monotonic*.

Arbitrary composition along horizon

Corollary 3: Let $\text{score}_1 : \mathcal{P} \times \mathcal{S} \times \mathcal{V} \rightarrow \mathcal{T}_1, \ldots, \text{score}_n : \mathcal{P} \times \mathcal{S} \times \mathcal{V} \rightarrow \mathcal{T}_n$ be proof scoring functions that satisfy the *deterministic*, *simple ordering*, *authorization relevant*, *bounded*, and *monotonic* properties, and let $f_{n-1} : \mathcal{T}_n \rightarrow \mathcal{T}_{n-1}, \ldots, f_1 : \mathcal{T}_2 \rightarrow \mathcal{T}_1$ be order-preserving homomorphisms mapping between the ranges of these functions. The result of sequentially composing $\text{score}_1, \ldots, \text{score}_n$ using $f_1, \ldots, f_{n-1}$ is also authorization scoring function that is also *deterministic*, *simple ordering*, *authorization relevant*, *bounded*, and *monotonic*.

Arbitrary depth of composition

Slide credit: Adam Lee

Scoring Functions: Take 3

- Preliminaries

Definition 10 (Canonical Proof of Authorization): A *canonical proof of authorization* for a principal p and a role $A.R$ is a minimal set of credentials C from the universe of all possible credentials such that $F(p, A.R, C) = TRUE$. We denote by $\mathsf{csets}(p, A.R)$ the set of all canonical proofs for the principal p and the role $A.R$.

$$\mathsf{psets}(p, A.R, v) = \{(C_p, C_c) \mid C_c \in \mathsf{csets}(p, A.R) \wedge C_p = v.C \cap C_c \wedge C_p \neq C_c\}$$

$$\mathsf{leaves}(C) = \{c \in C \mid c \text{ of the form } A.R \leftarrow p\}$$

$$\psi(C_p, C_c) = \frac{|\mathsf{leaves}(C_p \cap C_c)|}{|\mathsf{leaves}(C_c)|}$$

Slide credit: Adam Lee

Scoring Functions: Take 3

Goal: Score role membership, as well as non-membership

- Membership: Obvious reasons
- Non-membership: Approximate pricing

$$\phi(x) = \begin{cases} 1 & \text{if } x \geq 1 \\ 0 & \text{otherwise} \end{cases} \tag{20}$$

$$\text{score}(p, A.R, v) = \phi(|\text{sets}(v.C)|) \tag{21}$$

$$+\alpha \sum_{(w_i, C_i) \in \text{osets}_\omega(v.C, A.R)} w_i \cdot \frac{1}{2}^i$$

$$+\beta \sum_{(w, C_p, C_c)_i \in \text{opsets}(p, A.R, v)} w \cdot \frac{1}{2}^i$$

Slide credit: Adam Lee

Scoring Functions: Take 3

Theorem 4: The class of non-member scoring functions score : $\mathcal{P} \times \mathcal{R} \times \mathcal{V} \rightarrow \mathbb{R}$ represented by Equations 17–21 satisfies the *deterministic*, *simple ordering*, *authorization relevant*, *bounded*, and *monotonic* properties, provided that the scaling function $\omega : 2^{\mathcal{C}} \times 2^{2^{\mathcal{C}}} \rightarrow [0, 1]$ used to parameterize the osets function is deterministic.

Proof is similar to previous case

Interesting observation: Meets properties needed by composition theorem

Slide credit: Adam Lee

Conclusions

- Proofs have a lot more information than the binary yes/no decision that we use them for
- We developed a formal framework for scoring these proofs of authorization
- Cases explored
  - Perfect information a priori
  - No information a priori
  - Arbitrary combinations
  - Incomplete proofs

Slide credit: Adam Lee

What was Wrong with this Talk?

Issues with content:

- Why should we care about the problem?
- How will the results be useful in practice?
- Had no idea where talk was going!
- Missing context to understand problem setup

Issues with delivery:

- Lack of eye contact
- Lecturing to the board/laptop, not the audience
- Blurry fonts
- Too much text
- …

Slide credit: Adam Lee

Example: A Better Talk

Towards Quantitative Analysis of Proofs of Authorization: Applications, Framework, and Techniques

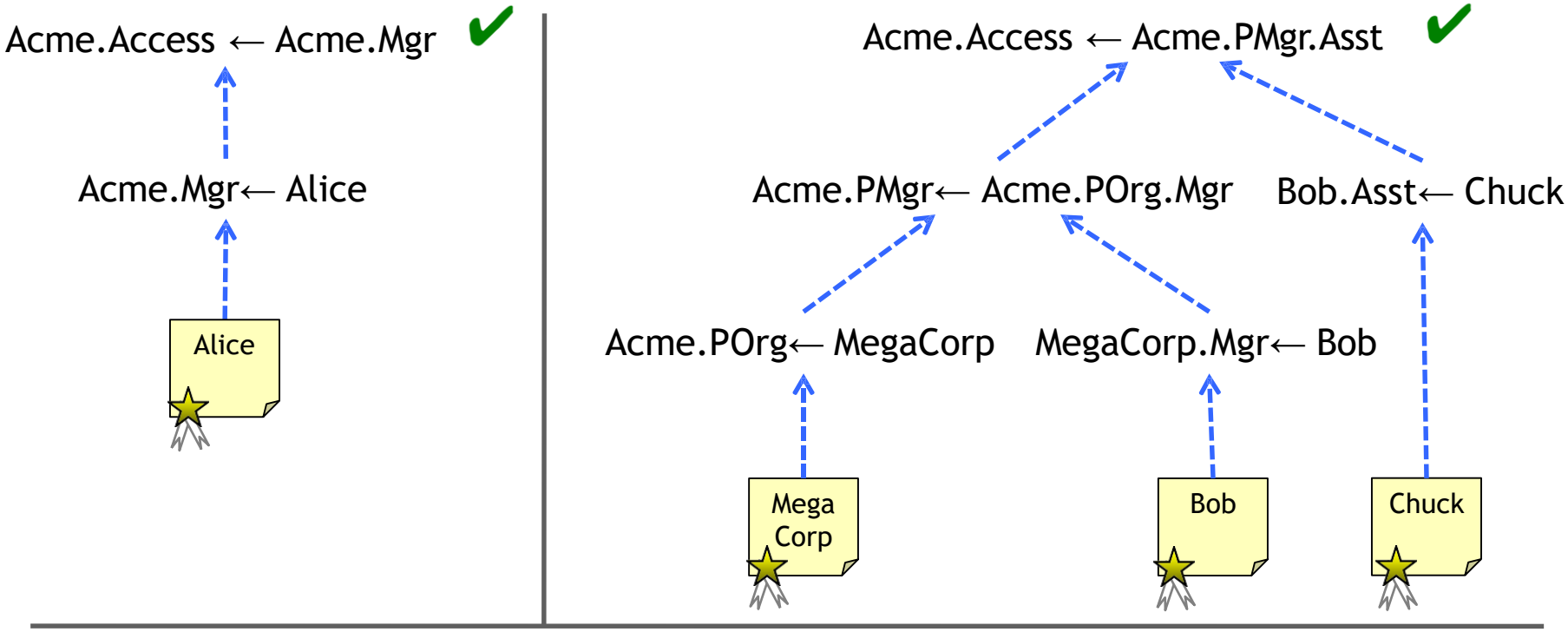
Adam J. Lee
adamlee@cs.pitt.edu
Department of Computer Science
University of Pittsburgh

Ting Yu
yu@csc.ncsu.edu
Department of Computer Science
North Carolina State University

Slide credit: Adam Lee

Like most access control systems, distributed proof construction systems are typically used to support binary decisions

Example: Access to a company database



Note that…

- Both proofs are valid
- The first proof is far simpler than the second
- Why focus only on the *destination* (validity)? What about the *journey* (context)?

Slide credit: Adam Lee

Proofs of authorization reveal a great deal of information about the conditions under which some access was granted

Authorization robustness

- How many proofs can some user generate?
- Are these proofs concise, or do they use odd delegations?
- How dependent on system state are these proofs?
- Applications: Anomaly detection, policy audit

vs.

User-to-user comparison

- Policies are requirements
- How well do various individuals satisfy them?
- Applications: Top-*k* analysis, group formation

Examination of incomplete proofs

- Policies aren't always perfect…
- How close is an unauthorized user to accessing a resource?
- Applications: Risk assessment, policy revision

Slide credit: Adam Lee

What are we not doing?

Point-based access control & trust management

- E.g., Yao et al. 2006
- Privacy-preserving compliance checking of point-based policies

Reputation-based trust management

- E.g., Kamvar et al. 2003, Xiong and Liu 2003, Josang et al. 2007
- Aggregation-based trust, different than credential-based proofs

Risk-based access control

- E.g., MITRE 2003, Aziz et al. 2006, Cheng et al. 2007
- More on this later…

Reasoning under uncertain information

- E.g., Dempster 1976, Shafer 1976, Cox 2004
- Focus is on uncertain information and/or inference rules

Slide credit: Adam Lee

Talk Outline

- Model for quantitative proof analysis
- Proof scoring functions
  - Desiderata
  - An example scoring construction
  - Functional composition
- Scoring incomplete proofs of authorization
- (Lots of) future directions

Slide credit: Adam Lee

RT_0 is the simplest language in the *RT* family

Principals are represented by public keys

Policies are constructed using four basic types of assertion

1. Simple membership: Alice.Friend ← Bob
   - Bob is a member of Alice's "Friend" role

2. Simple containment: Acme.Contractor ← WidgetTech.Employee
   - WidgetTech employees are "Contractors" at Acme

3. Intersection containment: Tech.Disct ← StateU.Student ∩IEEE.member
   - Students at Univ who are IEEE members are eligible for a discount

4. Linking containment: Acme.PMgr← Acme.POrg.Mgr
   - Members of the "Mgr" role defined by any member of "Acme.POrg" are members of Acme's "PMgr" role

Slide credit: Adam Lee

Modeling Authorization Scoring Functions

An RT_0 trust management system consists of:

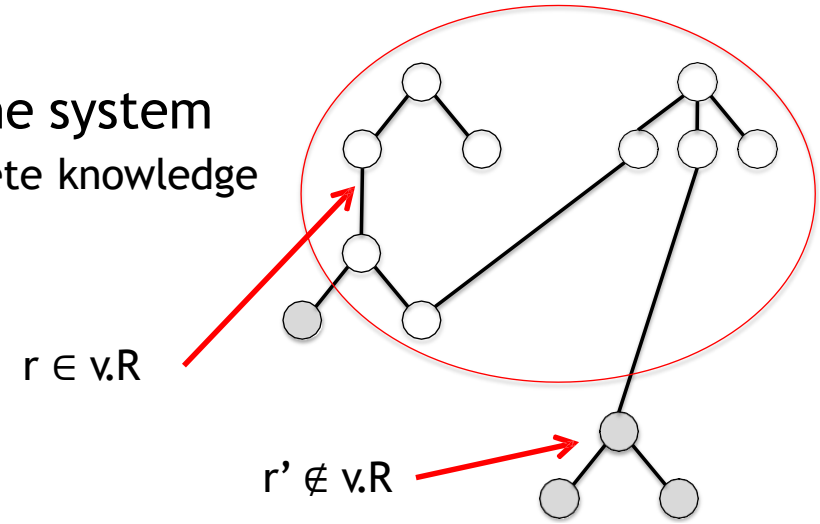
- A set P of principals
- A set R of roles/resources
- A set C of credentials

Very large…

- An inference scheme F : $P \times R \times 2^C \rightarrow$ {True, False}

Each principal has their own view of the system

- A set R $\subseteq R$ for which they have complete knowledge
- A set C $\subseteq C$ of credentials
  - ac(r) ≡{ c ∈ C | head(c) = r }
  - ∀ r ∈ R : ac(r) ⊆ C
- A store of auxiliary information A
  - Ignored in this talk, see paper for details



Proofs are scored relative to some principal's view

- score : $P \times R \times V \rightarrow T$

Slide credit: Adam Lee

What properties should a proof scoring function have?

Necessary properties ensure that proof scores "make sense"

- Deterministic
- Simple ordering:
  - $\forall v \in V$: $F(p_1, r, v.C) \wedge \neg F(p_2, r, v.C) \rightarrow score(p_1, r, v) \geq score(p_2, r, v)$
- Authorization relevant:
  - if F(p, r, C) = True, then C is a proof for p to access r
  - if F(p, r, C') = False for all $C' \subset C$, C is a minimal proof
  - Only credentials belonging to some minimal proof influence score

Desirable properties are beneficial, but not strictly necessary

- Bounded: $\exists\ b_1, b_2 : \forall\ p, r, v : b_1 \leq score(p, r, v) \leq b_2$
- Monotonic: $v \subseteq v' \rightarrow score(p, r, v) \leq score(p, r, v')$

What might some interesting classes of authorization scoring functions look like?

Slide credit: Adam Lee

Scoring proofs generated with incomplete views

Assumption: Principals start with empty views and discover minimal proofs of authorization at runtime

- Credential chain discovery in *RT*
- Distributed proof construction in, e.g., Grey or Cassandra
- Etc.

Let sets(C, r) represent the minimal proofs for r contained in C

One simple scoring construction is the following:

$$\text{score}(p, r, v) = \sum_{i=1}^{|\text{sets}(v.C, r)|} \frac{1}{2}^{i}$$

This function:

- Defines robustness as the number of proofs that a principal can generate
- Exponentially decays the contribution of proofs as they are discovered

Slide credit: Adam Lee

This simple notion of robustness is not very exciting, but can easily be tuned

Consider a function $\omega : 2^{\mathcal{C}} \times 2^{2^{\mathcal{C}}} \rightarrow [0, 1]$ that weights a minimal proof (possibly) by comparing it with other minimal proofs

Examples:

- $\omega_{len}(C_s, \cdot) = \gamma^{\max_{p \in \text{paths}(C_s)}(\text{length}(p))}$
- $\omega_{card}(C_s, \cdot) = \gamma^{|C_s|}$
- $\omega_{ind}(C_s, C) = 1 - \dfrac{\max_{C_i \in C \setminus C_s}(|C_s \cap C_i|)}{|C_s|}$
- Linear combinations of the above

Prefer limited delegation

Prefer simple structure

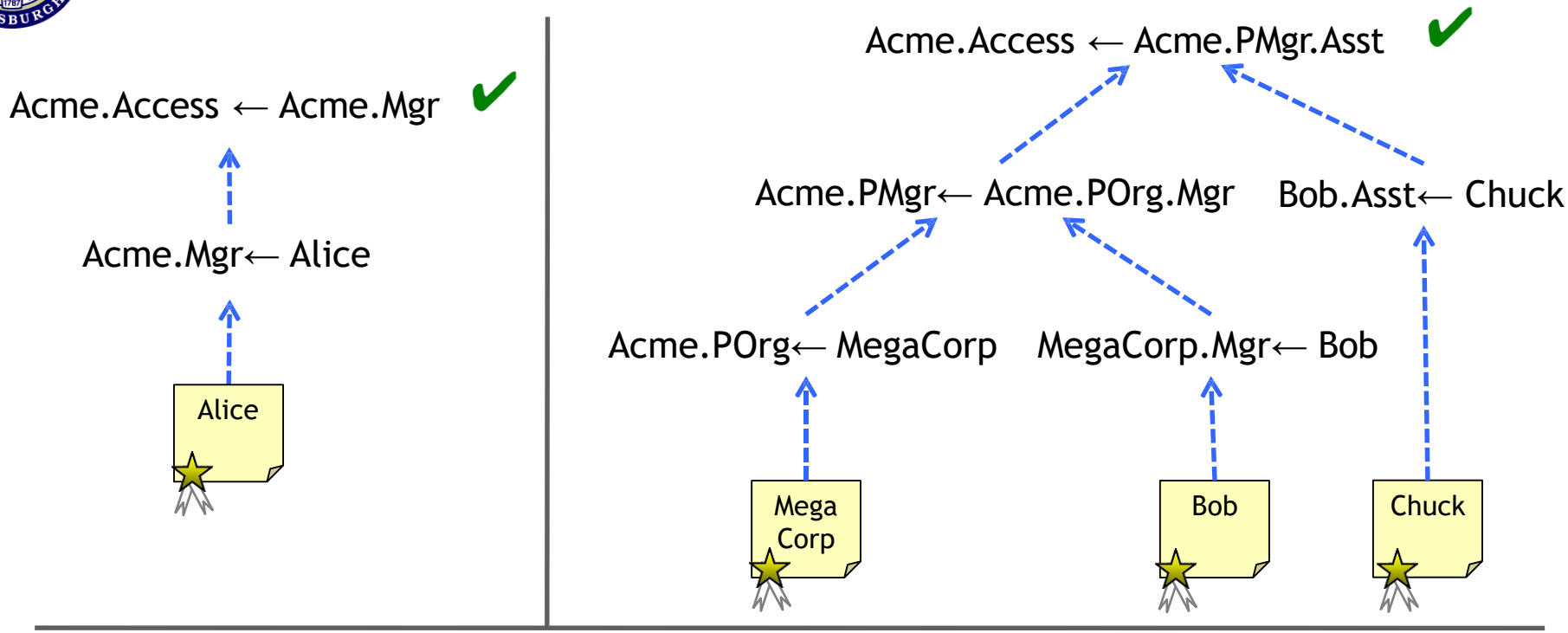
Prefer multiple, largely independent proofs

Our scoring construction can then be rewritten as:

$$\text{score}(p, r, v) = \sum_{(C_i, w_i) \in \text{osets}_\omega(v.C, r)} w_i \cdot \frac{1}{2}^i$$

Slide credit: Adam Lee

Example, Redux



Using ω_{len}:

- score(Alice, Acme.Access, v_1) = 0.365
- score(Chuck, Acme.Access, v_2) = 0.328

Using ω_{card}:

- score(Alice, Acme.Access, v_1) = 0.365
- score(Chuck, Acme.Access, v_2) = 0.215

Note that ω_{ind} is irrelevant in this case…

Slide credit: Adam Lee

This proof scoring function satisfies our desiderata

Theorem: Provided that the function ω used to parameterize osets is deterministic, the authorization scoring function

$$\text{score}(p, r, v) = \sum_{(C_i, w_i) \in \text{osets}_\omega(v.C, r)} w_i \cdot \frac{1}{2}^i$$

satisfies the *deterministic*, *simple ordering*, *authorization relevant*, *bounded*, and *monotonic* properties.

The above scoring function

- is certainly not the only such authorization scoring function
- may not be the best scoring function for all situations
- may only be sensible to use on certain parts of a proof

However, it is an interesting building block…

Slide credit: Adam Lee

In many situations, defining *the* proof scoring function to use could be a difficult task

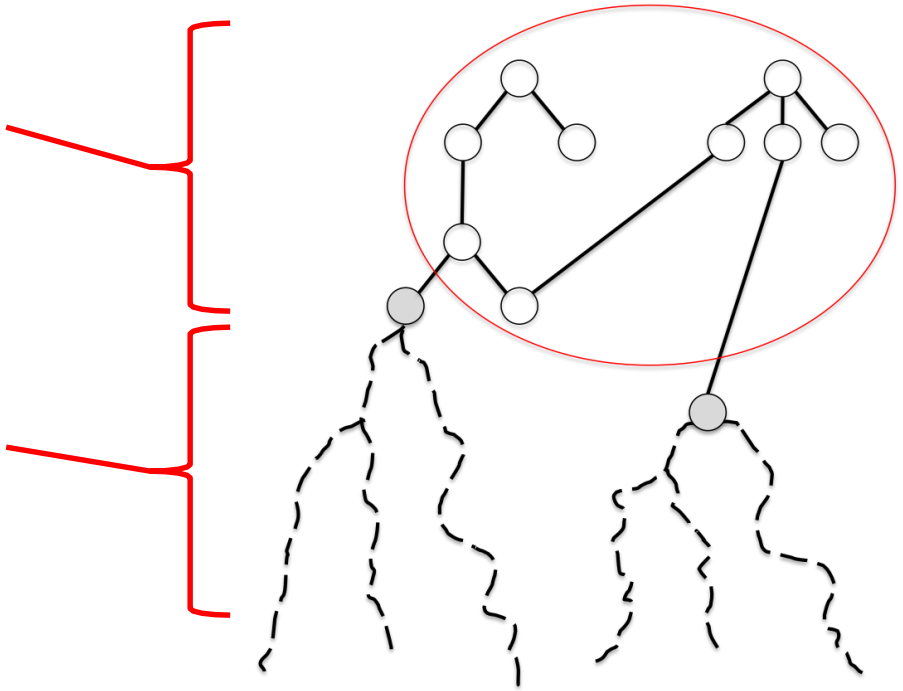
Example: Security administrators within an organization

Perfect information within domain

- Exact knowledge of resource/role semantics
- Very precise weighting and analysis
- Hand-tuned scoring is possible

On-demand information outside of domain

- Known semantics for horizon resources
- Full semantic knowledge of proof is unlikely
- Structure is discovered at runtime



Under what circumstances can good "building block" functions be composed to construct proof scoring functions while still preserving the properties of each building block?

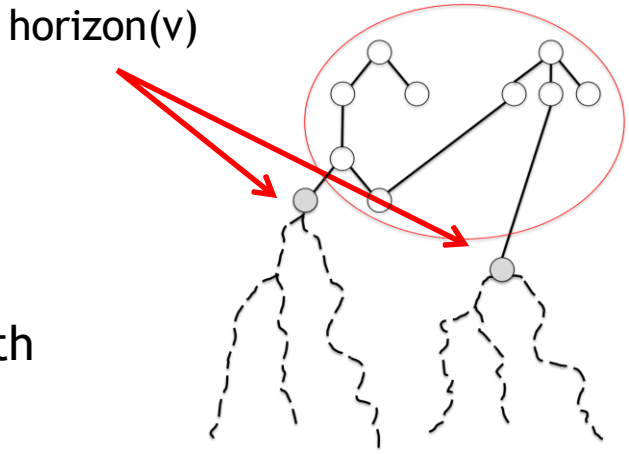
Slide credit: Adam Lee

Fortunately, reasonable proof scoring functions maintain their properties under sequential composition

Definition: Assume that we have

- Principals p and p'
- Resources r and r'
- Views v and v'
- Functions score and score'

We say that score is sequentially composed with score' if r' ∈ horizon(v) and score'(p', r', v') is calculated when calculating score(p, r, v).



Theorem*: Let $\text{score}_1 : P \times S \times V \rightarrow T$ and $\text{score}_2 : P \times S \times V \rightarrow T$ be two authorization scoring functions that satisfy the *deterministic*, *simple ordering*, *authorization relevant*, *bounded*, and *monotonic* properties. The sequential composition of these functions also satisfies the *deterministic*, *simple ordering*, *authorization relevant*, *bounded*, and *monotonic* properties.

Slide credit: Adam Lee

So far, we have focused on scoring *complete* proofs of authorization

If a policy is out of date or incomplete, users who should be able to do something might not be able to

Risk-based access control is one approach to limiting inflexibility

- Place a (typically monetary) cap on the amount of risk/damage permissible
- Tokenize this risk/damage and distributed it to users
- Compute "risk prices" for every resource in the system
- If users can pay the access price, they are permitted access

While this would be significantly more flexible than policy-based approaches, pricing access to individual resources is non-trivial

Alternate approach: Rather than pricing resources per user for *every* user, price deviations from expected policies

Slide credit: Adam Lee

To price deviations from an expected policy, we first need to be able to quantify the degree of these deviations

A natural generalization of our framework provides one approach for doing exactly this

Step 1: Find the canonical proofs of authorization for the resource

- All minimal sets of credentials C such that F(p, r, C) = True
  - Note: These credentials may not all be materialized in the system
- Call the result csets(p, r)
- Note: The *RT* credential chain discovery process does this for us

Step 2: Find partial matches between v.C and csets(p, r)

- psets(p, r, v) = $\{(C_p, C_c) \mid C_c \in csets(p, r) \wedge C_p = v.C \cap C_c \wedge C_p \neq C_c\}$

Step 3: Evaluate the quality of each partial match

- leaves(C) = { c ∈ C | c of the form r ← p }
- $\psi(C_p, C_c) = |leaves(C_p \cap C_c)| \ / \ |leaves(C_c)|$
- opsets(p, r, v) = { (w, C_p, C_c) | (C_p, C_c) ∈ psets(p, r, v) ∧ w = $\psi(C_p, C_c)$ }

Slide credit: Adam Lee

Step 4: Tying it all together

$$\phi(x) = \begin{cases} 1 & \text{if } x \geq 1 \\ 0 & \text{otherwise} \end{cases}$$

$$\text{score}(p, r, v) = \phi(|\text{sets}(v.C)|)$$

Ensures non-member scores always below 1

Score complete proofs…

$$+\alpha \sum_{(w_i, C_i) \in \text{osets}_\omega(v.C, r)} w_i \cdot \frac{1}{2}^i$$

…and partial proofs

$$+\beta \sum_{(w, C_p, C_c)_i \in \text{opsets}(p, r, v)} w \cdot \frac{1}{2}^i$$

Note: This function satisfies the *deterministic*, *simple ordering*, *authorization relevant*, *bounded*, and *monotonic* properties

Due to our composition theorem, this function can act as a template function that can be sequentially composed with other reasonable authorization scoring functions

Slide credit: Adam Lee

This work is just a first step…

Question 1: These types of scoring functions *seem* sensible, but do they make sense in the context of real policies?

Medicine

Academic Departments

Defense

Question 2: RT_0 is a very simple language. What would scoring constructions for more feature-rich languages look like?

- Credentials with internal structure (e.g., RT_1)
- Flexible rule structure (e.g., SecPAL, Grey)
- Reasoning over aggregates like reputation (e.g., CTM, WBSNs)
- …

Slide credit: Adam Lee

Efficiency and functional extensions…

Question 3: How can we efficiently construct cost-minimizing approximate proofs of authorization?

- Can we prune the state-space as we search?
- Applications to risk-based access control

Question 4: How can we efficiently execute top-*k* queries over (distributed) authorization datasets?

Group formation

Evaluating Policy Utilization

Slide credit: Adam Lee

Conclusions

Interesting applications of reasoning about proofs of authorization

- User-to-user ranking of proofs
- User-to-ideal assessment of proof quality/robustness/etc.
- Understanding the changing needs of an organization
- Risk-aware authorization reasoning
- …

Our goals for this initial work

- Develop a formal model for proof scoring
- Identify necessary and desirable criteria for scoring functions
- Demonstrate that these criteria are attainable in practice
- Understand the situations in which scoring functions can be composed

There is still much to be done…

Slide credit: Adam Lee

Thank you!

Towards Quantitative Analysis of Proofs of Authorization: Applications, Framework, and Techniques

Adam J. Lee
adamlee@cs.pitt.edu
Department of Computer Science
University of Pittsburgh

Ting Yu
yu@csc.ncsu.edu
Department of Computer Science
North Carolina State University

Questions?

Slide credit: Adam Lee

Why was this better?