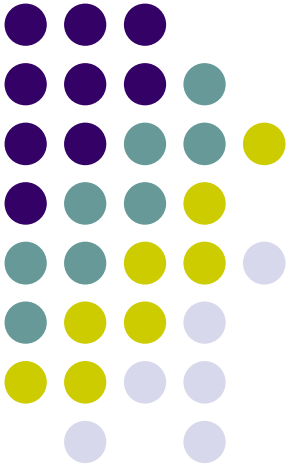
PKI

Views from the Dispassionate "I"

Carlisle Adams

School of Information Technology and Engineering

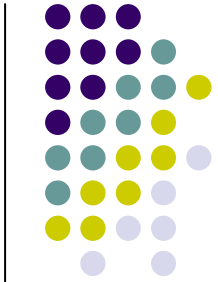
University of Ottawa



Views from the Dispassionate “I”

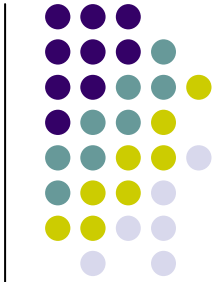
- Three different senses for this title:
  - Play on words (“dispassionate eye”)
  - *“I” = “Carlisle”* (I may now be viewed as more unbiased)
  - *“I” = “Infrastructure”* (without bias toward any particular algorithms, protocols, data structures, toolkits, or products, what do we really want from a public key infrastructure?)

Roadmap



- Defining PKI
- Where have we been?
  - Exploring the reasons for limited success
- Where are we now?
  - Searching for the real requirements for PKI
- Where can we go next?
  - Some technological directions to watch
- Conclusions

Defining PKI (long version)



- A pervasive infrastructure incorporating

  key generation, RA, CA, authority key distribution, repository, backup, recovery, key history, update, usage restrictions, path development, flexible trust models, time stamping, expiry, revocation, archival, stand-alone entity software, and standardized access points

in order to provide a variety of security services, which may include some of

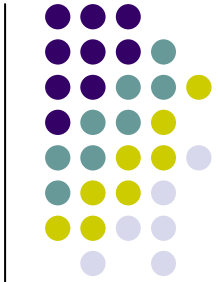
authentication, integrity, confidentiality, secure communication, digital time stamping, electronic notarization, support for non-repudiation, and comprehensive privilege management

Defining PKI (shorter version)

PKI:

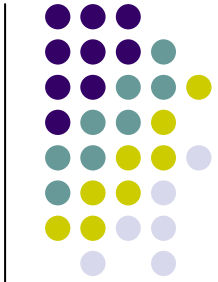
a collection of tools, processes, and policies for associating (often referred to as "binding") public key pairs and identifiers

A Functional View



- Binding of key pair and identifier
  - Validity of bindings
    - Authority (making & breaking)
    - Issuance process (syntax & dissemination)
    - Termination process (alerting)
  - Use of bindings
    - Anchor management process (augment & diminish)
    - Private key management process ("fit for purpose")
    - Binding validation process (trusting someone else's key)

Where Have We Been?

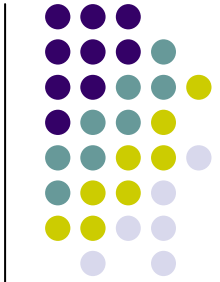


- Public key cryptography proposed in 1976,
  RSA public key algorithm published in 1978,
  Public key certificate proposed in 1978.

  (So, all the essential tools for PKI have been around for almost 30 years…)

- However, PKI has been less than wildly successful
  - Commercially (e.g., in the earnings of PKI companies)
  - Socially (even among the “geeks”)

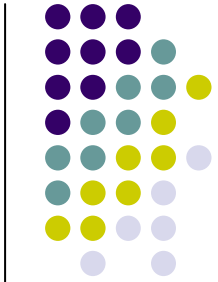
What Determines Success?



- Success (even among the geeks) comes when you give people something they want at a price (processing power, memory, time, money, etc.) they are willing to live with

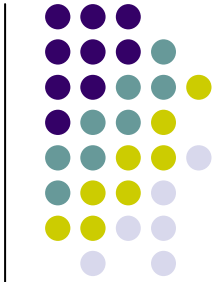
- So, what were people told that PKI would give them, and how was the price determined?

Statements Made About PKI…



- *"scalable key management"*
- *"flexible architecture that can grow as the needs of the organization change over time"*
- *"single infrastructure capable of supporting multiple applications, thus reducing deployment and administrative costs"*
- *"addresses intra- and inter-organizational security needs"*
- *"provides a vehicle for interoperation with several different external trust infrastructures"*
- *"is an e-business enabler"*
- *"provides security services, including: authentication, integrity, confidentiality, secure communication, digital time stamping, electronic notarization, support for non-repudiation, and comprehensive privilege management"*

Reality…



- These are all *potential* outcomes, possibilities
  - All PKI *actually* gave was a binding between a key pair and an identifier

- Unfortunately, the price (typically, monetary cost) was generally set in proportion to the potential outcome, not the actual outcome
  - Analogy: complete set of carpenter's tools is probably worth a few thousand dollars (pounds)

But with these tools you could build this…

(SITE, U. of Ottawa)

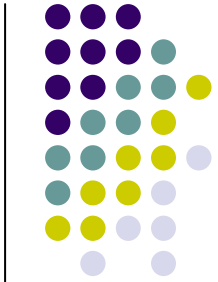
Or this…

(Kasteel Arenberg, K. U. Leuven, Belgium)

Or even this…

(Founder's Quad, Royal Holloway, U. of London)

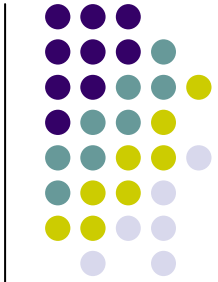
Appropriate Price?



- Should the carpenter's tools be set at 2 Million dollars (pounds)? No. Still worth a few K.

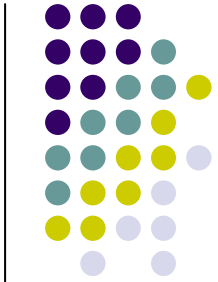
- So, the price needs to be right. But the other side also needs to be right: PKI must deliver something people want
  - Everyone wants the building, not the carpenter's tools
  - Everyone wants security services, not key-identifier bindings
    - Many requirements - and many fears - not addressed by core purchase

Lessons from a $50.00 bill…



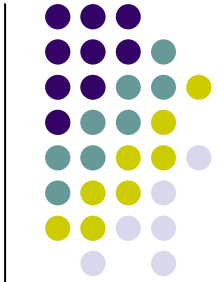
- "Invested in infrastructure, but not using it!"
- Concerns?
  - Theft by cashiers or by public *[Priv. key compromise]*
  - Their partners won't take them *[Interoperability]*
  - On-going cost to keep up with new bills *[Revocation]*
- Ray of hope?
  - Problems not unique to PKI and not insurmountable
    - Risk management, comfort factor, others' success…

Limited Success



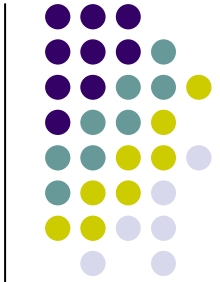
Ultimately, PKI promised too much and delivered too little (and the price usually corresponded to the promises!)

Roadmap



- Defining PKI
- Where have we been?
  - Exploring the reasons for limited success
- Where are we now?
  - Searching for the real requirements for PKI
- Where can we go next?
  - Some technological directions to watch
- Conclusions

Where Are We Now?

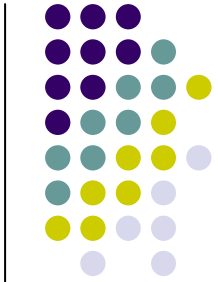


- Success has to do with giving people what they want at a price they can live with

  (I'm not going to make any suggestions about price in this talk…)

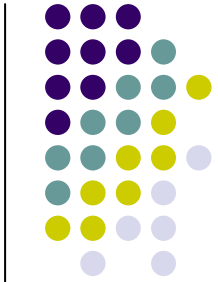
- What kinds of security services do people actually want?

Security Services (1/2)



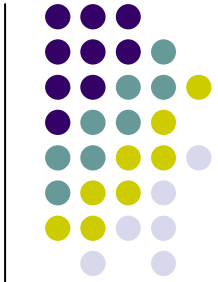
- ***Confidentiality*** (enc. w/o prior key exchange)
  - Not compelling: when do you want to send confidential info to a complete stranger? OTOH, if it's to an acquaintance, symmetric techniques will do, or real-time Diffie-Hellman may be sufficiently low-risk.
- ***Integrity*** (data not changed in transit)
  - Not compelling: clearly important, but meaningless on its own (integrity only makes sense if you know where data originated).
- ***Authentication*** (proof of who you are)
  - Not compelling: in real life, never authenticate just to prove identity; it's always ID as a link to privilege. Authentication is a means, not an end.

Security Services (2/2)



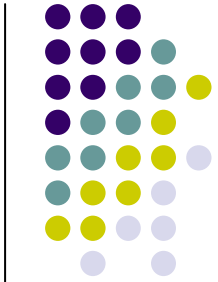
- ***Signature*** (assertion about content)
  - COMPELLING: we need to sign data all the time (business contracts, consent forms, transaction approvals, legal documents, etc.). ***Only PK technology can provide this functionality.***
- ***Access control*** (protecting resources)
  - COMPELLING: objects and services are increasingly networked and we need to properly control who has access to what (and when, where, why, and how this access should happen). ***PK technology increases the security of AC.***

How Can PKI Help?



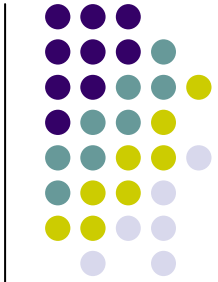
- For **signatures**
  - Typically, want property that a value is created, bound uniquely to an entire document, computed only by a specific individual, but verifiable by anyone in the world (PKI needed for last piece).
  - However, if a signature is an assertion about content, it is imperative to somehow bind that assertion with the signature. (Suggestions to achieve this [e.g., multiple certificates, each with a "purpose" indicator] have not been generally accepted. Need a good way to convey "intent".)

How Can PKI Help? (cont'd)



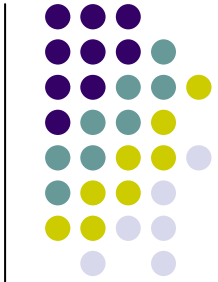
- **For access control**, want a strong way to associate privileges with entities in an environment
  - How can party *A* (Authority) tell party *V* (Verifier) that party *C* (claimant) holds a particular privilege? ["indexical reference problem" - Blakley]
    - *A* gives *C* some token/ticket, and *C* presents this to *V*
      - "bearer token": risky (token subject to theft or loss)
    - *A* gives *V* a description of *C*, and *C* shows up with description
      - "descriptive reference": not applicable in network context
    - *A* gives *V* an id. for *C*, and *C* shows up claiming that id.
      - "nominative reference": tie between auth'n and auth'z

How Can PKI Help? (cont'd)



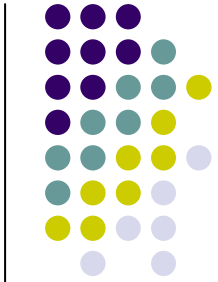
- With nominative reference, there are two considerations in the authentication step:
  - How is the link made between the authentication result and the privilege ownership decision?
  - Is *C* known to *V* prior to this access request?
- Some examples may help to clarify…

Example #1



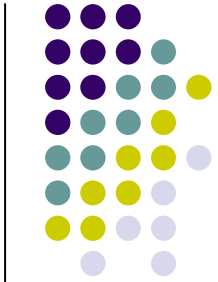
- Attribute cert says “Bob can authorize purchases up to $10,000.”
- Bob presents this cert to *V*, along with the username/password pair {“Bob”, “fluffy”}.
- *V* checks the authentication and checks the name in the attribute cert, and then has to assume that this is the same “Bob”.
  - Guaranteed globally-unique names/identifiers would solve this, but that can be hard (impossible?) to achieve
  - How can this work if *V* doesn’t know Bob’s password?

Example #2



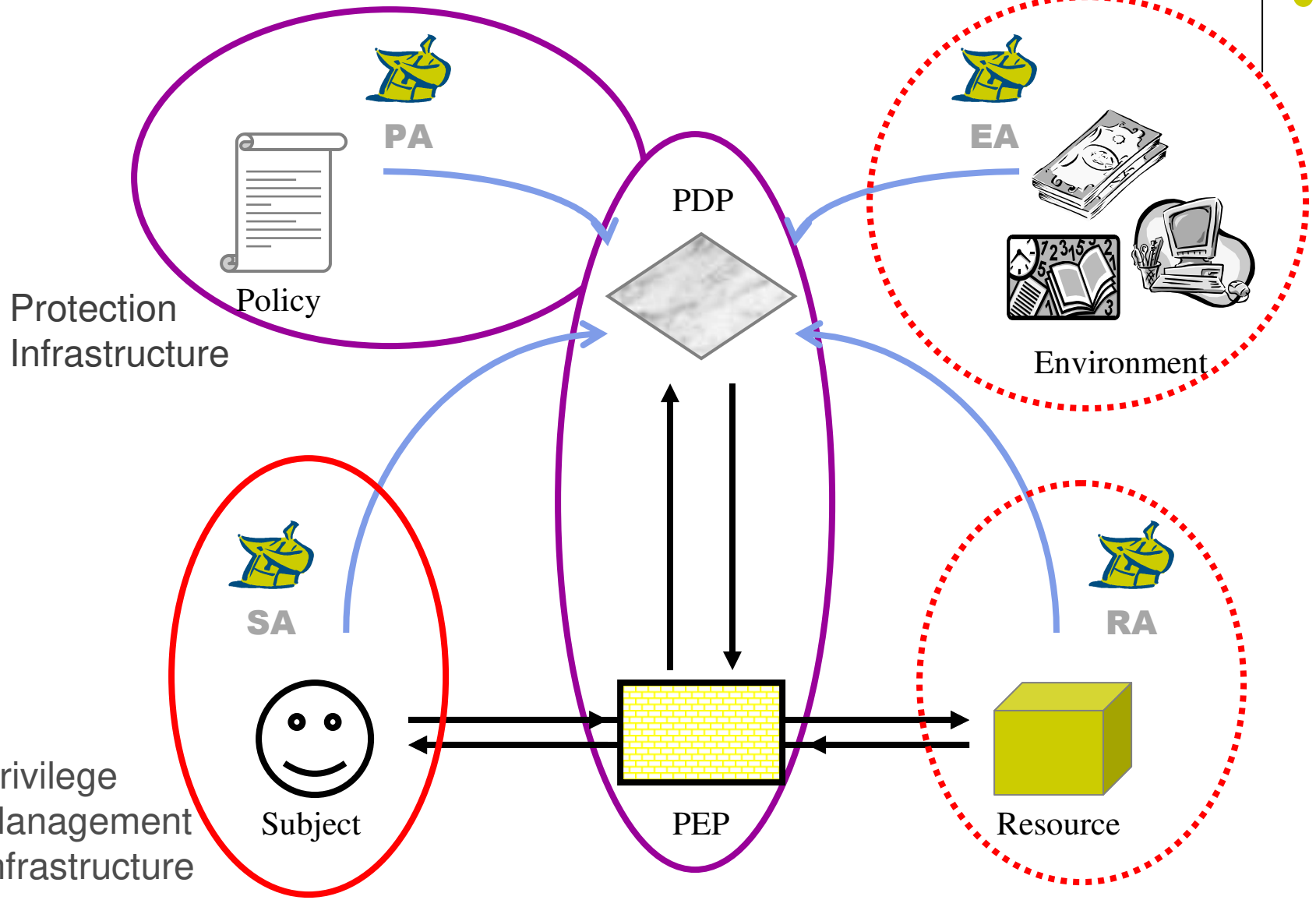
- Don't use name in attribute cert, but use an authentication method and some associated authentication data
- "Whoever can supply a password that hashes to '_____' can authorize purchases up to $10,000."
  - Link between authentication result and privilege ownership is now explicit and cryptographically protected [no assumption on part of *V*]
  - This method can work even if *V* doesn't know requester (important in networked context). But, once a successful auth'n has occurred, *V* can impersonate the requester.

Example #3

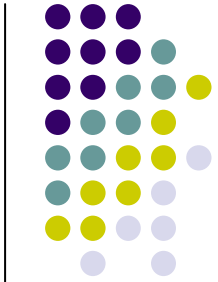


- "Whoever can prove possession of the private key that corresponds to public key '_____' can authorize purchases up to $10,000."
  - This has the advantages of the second example PLUS ensures that *V* cannot subsequently impersonate the requester (since *V* never finds out the private key)
- This is how PKI can support PMI in access control (attribute cert does not contain the name of the privilege holder, but points to the holder's PK cert)
- PKI can also be used to support the protection part of access control (more discussion later…)

Access Control Architecture



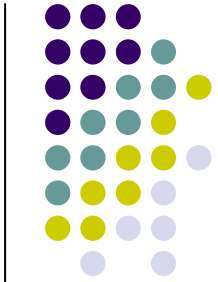
So, where are we now?



- Slowly coming to the realization that a collection of name-identifier bindings which can generically support a large number of security services is of very limited utility (not very many people want to buy the carpenter's tools and build their own castle, no matter how much they want the castle!).

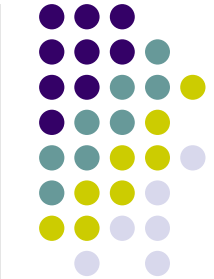
- A better approach may be to really tailor and enhance the PKI to properly support one or two specific security services (suggestion: signatures and access control). This would go a long way toward giving people what they really want. [Setting an appropriate price would then ensure success…]

Roadmap



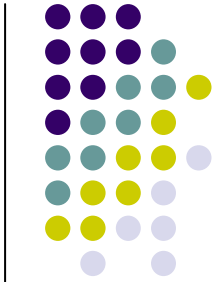
- Defining PKI
- Where have we been?
  - Exploring the reasons for limited success
- Where are we now?
  - Searching for the real requirements for PKI
- Where can we go next?
  - Some technological directions to watch
- Conclusions

Where Can We Go Next?



- A couple of technological directions are worth watching to see how they develop
  - *Identity-Based Cryptography* (IBC, IB-PKI)
  - Stefan Brands' *Digital Credentials*

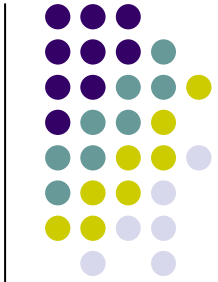
Identity-Based Cryptography



- Idea originally proposed by Shamir in 1984
  - Can an arbitrary string (e.g., a name, an e-mail address, or a mobile phone number) be used as a public key? [Motivation: maybe we don't need certificates…]
- Efficient, secure encryption (IBE) scheme proposed by Boneh and Franklin in 2001
  - Works with points on an elliptic curve (points form a group, so multiplication of a point P by an integer a results in another point aP)
  - Uses a special function f (a bilinear map based on the Weil pairing) that takes 2 points and outputs a number, and has the property:

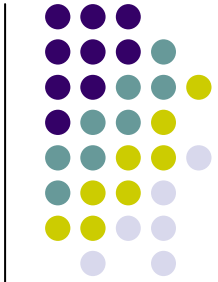
$$f(aP,bQ) = f(P,bQ)^a = f(P,Q)^{ab} = f(bP,aQ)$$

IBE Scheme



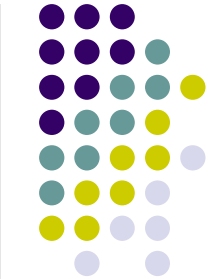
- Setup
  - Key generator generates secret *s*, random point *P*
  - System parameters are *P*, *sP*
- Encryption
  - Alice hashes string for Bob (e.g., Bob@xyz.com) to point *Q*
  - Chooses random value *r*; computes *rQ* and *rP*
  - Encrypts message with symmetric key $k = f(rQ, sP)$
  - Sends encrypted message and *rP* to Bob
- Private Key Generation
  - Bob authenticates to PKG, asks for private key
  - Key generator hashes string for Bob to *Q*, gives back *sQ*
- Decrypt
  - Bob decrypts with key $k = f(sQ, rP) = f(rQ, sP)$

IBC, IB-PKI



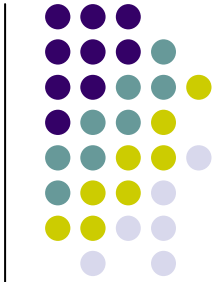
- Along with Identity-Based Encryption, Identity-Based Signature (IBS) schemes have also been proposed
- Given these (i.e., given IBC), it is possible to build an Identity-Based PKI (IB-PKI), but this may not be the panacea that was initially expected (no certificates = no directories & no complicated path validation algorithms!)

IB-PKI Complications (1/2)



- Naming
  - One of the things that makes directories difficult is naming, but the naming problem is not solved in IB-PKI: Alice still needs to associate the *identity* string with the physical, real-world entity, Bob
- Trust
  - Path validation is complex, but at least it gives a mechanical, algorithmic procedure for coming to trust something (a public key). With IB-PKI, Alice has no such automated means for trusting the PKG that will give Bob his private key

IB-PKI Complications (2/2)



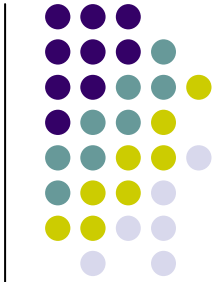
- Revocation
  - Because the public key is the identity string for Bob, it is conceptually more difficult to revoke it (and alert others that they should use some other string instead)
- Binding
  - With traditional PKI we have

    {binding between identity and public key (certificate), and proof of possession of private key}
  - With IB-PKI we have

    {binding between identity and private key, and proof of possession of public key}

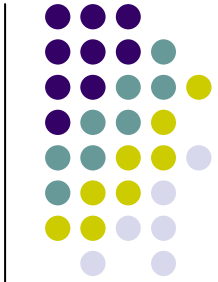
    (Bob needs to prove to PKG that he owns this identity; this seems inherently harder than proving possession of a private key, which is just a chall/rep protocol & math)

Why Consider IB-PKI?



- Major advantage:
  - Arbitrary string can be a public key for an entity (doesn't have to be an identity)
    - In particular, proposals have been made to add information to the identity string (e.g., validity period, to help with revocation problem)
    - More generally, string can represent a *policy* to be associated / enforced with the private key operation

IB("Information" Based)-PKI

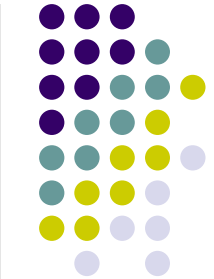


- "Information string" is the policy that is desired
  - Applicability to **Signatures with Intent**
    - "Intent" is explicitly encoded as the public verification key for a digital signature (obvious for verifiers). On request, PKG generates a private key that cryptographically corresponds to this particular intent in order for Bob to sign a document (obvious to signer, and less error-prone than current proposals).
  - Applicability to **Access Control** (Protection Infrastructure)
    - Access Control policy is explicitly encoded as the public key used to encrypt a document (object is encrypted for a set of conditions). Potential recipient needs to prove compliance with this policy (i.e., all conditions must be satisfied) in order to receive private decryption key.

Where Can We Go Next?

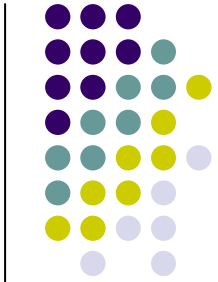
- A couple of technological directions are worth watching to see how they develop
  - *Identity-Based Cryptography* (IBC, IB-PKI)
  - Stefan Brands' *Digital Credentials*

Digital Credentials



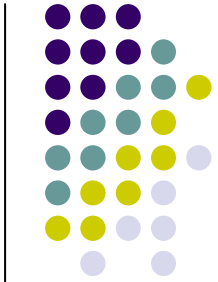
- Idea originally proposed by Brands in 1996
  - Can certificates be created that allow authentication directly to specific, desired attributes? [Motivation: maybe we don't need to reveal identity in all our public key transactions…]
  - Two protocols defined
    - "Issuing protocol": CA signature on Bob's public key
    - "Showing protocol": Bob proves possession of private key, and selectively reveals desired portions of it

Digital Credentials (cont'd)



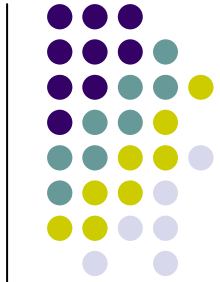
- Bob's private key is (x_1, …, x_n, a), where the x_i are attributes and a is a secret value known only to Bob
- Bob's public key is $h = g_1^{x1} \ldots g_n^{xn} h_0^{a}$
- In the issuing protocol, the CA signs h and gives Bob the signature, which is verifiable by anyone using the CA's public key
- In the showing protocol, Bob gives h, the CA's signature on h, and any of the x_i to the verifier. Bob then proves that the revealed x_i are in his private key and that he possesses the entire private key (i.e., he possesses all the x_i and a).

Why Consider Dig. Credentials?



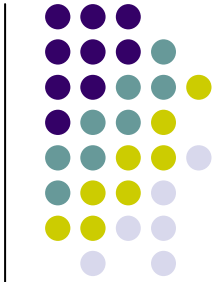
- Major advantage:
  - Allows entities to directly authenticate ownership of specific attributes that are relevant for a transaction (authentication does not happen through “medium” of identity)
- Applicability to **Access Control** (PMI)
  - “Cleaner”, more intuitive notion of privilege management
  - Improves privacy
    - if identity is not revealed, different transactions may not be linked
    - gives entities greater control over what information is stored about them by verifiers (Bob reveals only what is explicitly required for a particular access, no additional information)

Roadmap



- Defining PKI
- Where have we been?
  - Exploring the reasons for limited success
- Where are we now?
  - Searching for the real requirements for PKI
- Where can we go next?
  - Some technological directions to watch
- Conclusions

Conclusions



- The dispassionate "I": a case of **misplaced passions**
  - As a community, we've been passionate about
    - Algorithms, protocols, data structures, toolkits, products

    (where we should have been dispassionate)

    and we've been dispassionate about
    - Security services – trying to generically support all services

    (where we should have been passionate!)

- This talk has suggested that tailoring PKI to specific security services (**signatures** and **access control**) may be useful, and has discussed some recent technologies to watch in this area

- ***Note: everything depends on strong hash functions…***