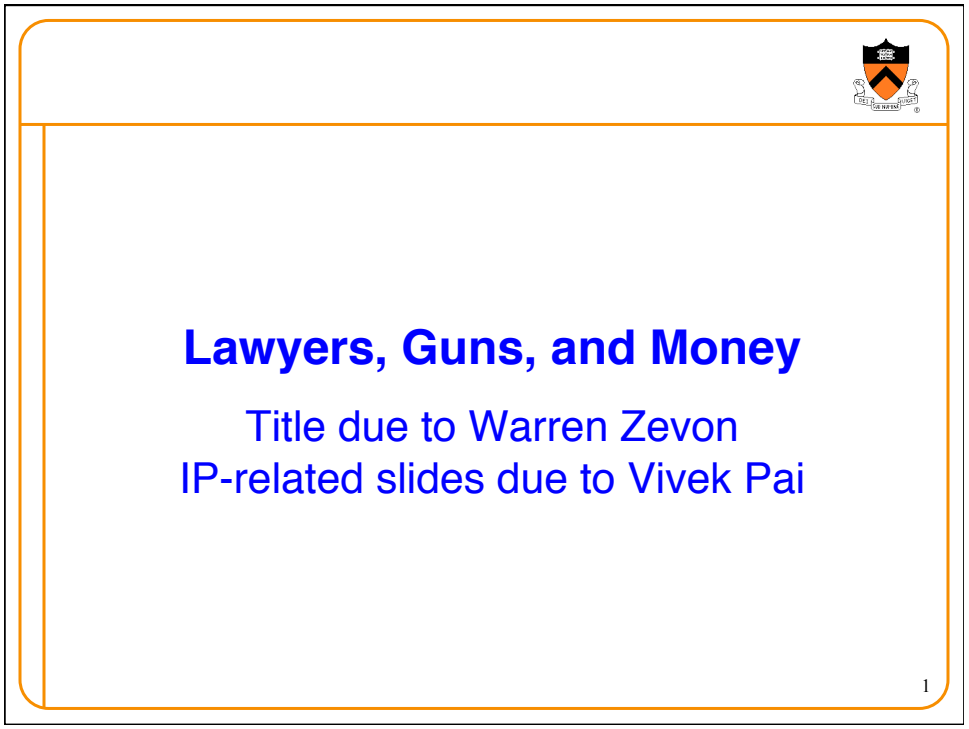
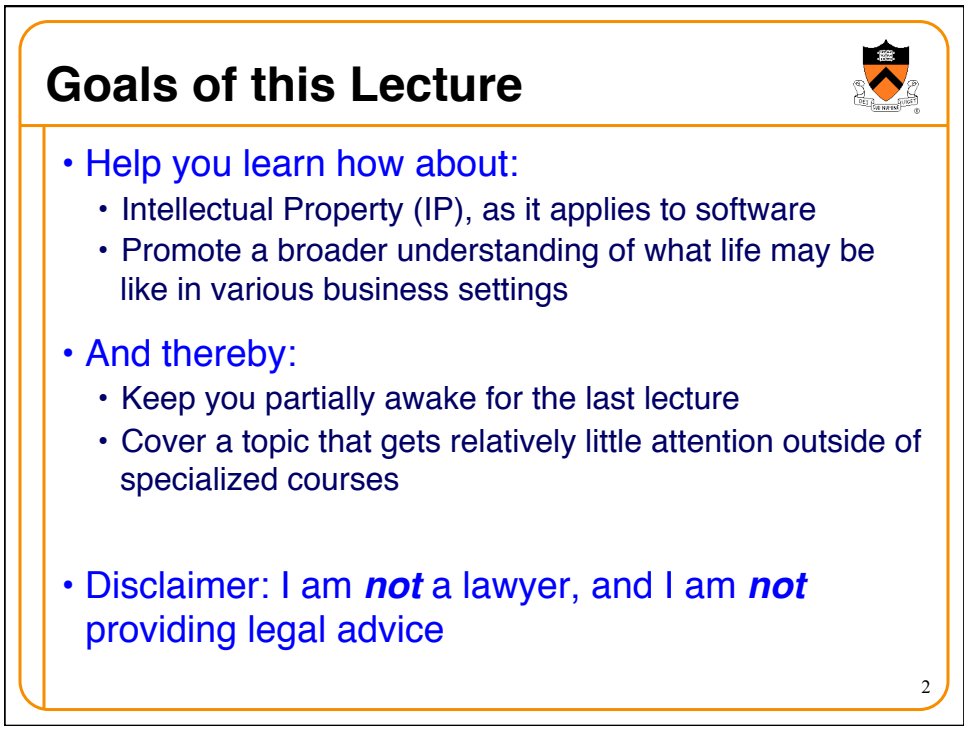

Lawyers, Guns, and Money
Title due to Warren Zevon
IP-related slides due to Vivek Pai
1


Goals of this Lecture
• Help you learn how about:
• Intellectual Property (IP), as it applies to software
• Promote a broader understanding of what life may be like in various business settings
• And thereby:
• Keep you partially awake for the last lecture
• Cover a topic that gets relatively little attention outside of specialized courses
• Disclaimer: I am *not* a lawyer, and I am *not* providing legal advice
2

Legal Concepts

- Copyright
- Patents
- Trade Secrets
- Derivative works
- Licenses

3

Copyright

- From http://www.copyright.gov/help/faq/faq-general.html
  - Copyright is a form of protection grounded in the U.S. Constitution and granted by law for original works of authorship fixed in a tangible medium of expression. Copyright covers both published and unpublished works.
- Basically,
  - Anything you write/draw/etc is “automatically” protected from being copied, and you can opt to register your copyright to help enforce it
- Applies regardless of originality, unless you’re copying something else, translating, or mechanically reproducing it
- Protects you from someone literally copying the exact code, file, etc., but you still have to pursue it

4

Copyright Examples

- You write a program in C and compile it
  - The source file in C is protected by copyright
  - The binary file, mechanically produced from the source, is also protected by copyright
  - You can sell the copyrighted binary by itself
- You draw an image
  - The image is protected by copyright
  - Resized, resampled, recolored versions of the image are also protected by that same copyright

5

Copyright Ownership

- Who owns the copyright?
  - If working for yourself, you do
  - If working for someone else, they likely do
  - Even if done on your own time, you may have signed away your rights as part of an employment contract
- What if you are a student?
  - Undergrads typically regarded as individuals
  - Grad students typically regarded as employees
  - Undergrads working on internships, grants, etc., also considered employees
- International issues
  - Copyrights largely governed by Berne Convention
  - Covers 165 countries with largely uniform protection
  - Length of coverage varies, but typically 50+ years

6

Patent

- From http://www.uspto.gov/patents/index.jsp
  - A patent is an intellectual property right granted by the Government of the United States of America to an inventor "to exclude others from making, using, offering for sale, or selling the invention throughout the United States or importing the invention into the United States" for a limited time in exchange for public disclosure of the invention when the patent is granted.
- Basically,
  - You disclose how something new/original works in detail, in exchange for protection against others duplicating it for some time (e.g., 7-14 years)
- Applies only for things that are new and not "obvious"
- Protects you from people copying your invention, at the cost of telling everyone how it works

7

Patent Examples

- You design a new algorithm and implement it
  - E.g., like the RSA crypto algorithm for public keys
- You design a new protocol and implement it
  - E.g., like a remote-screen protocol for Windows
- You design a new shape for a product – design patent
- Patent is ***not*** automatic
  - In US, can apply up to 12 months from first public disclosure
  - In rest of world, no patenting after public disclosure
  - In large # of countries, can file provisional patent that "holds" the patent filing for a year
  - After provisional, must file patent in each country to be protected
  - Patent costs can be very large

8

Patents vs Copyrights

- Scope
  - Patents protect general idea, regardless of implementation
  - Copyrights protect specific implementation & direct derivatives
- Novelty
  - Patents have to be for new, non-obvious things
  - Copyrights can be on almost anything
- Duration
  - Patents are short-lived due to wider scope
  - Copyrights are long-lived
- Prosecution
  - Copyright infringement is usually fairly clear
  - Software patents are often a matter of interpretation
  - Molecular (i.e., drug) patents are typically much clearer

9

Trade Secret

- Another mechanism for protecting something of value
- Basically
  - Do not disclose it
  - Restrict access only to “need to know” employees
  - Clearly identify it as a trade secret
  - Make it clear in employment contract
- Examples
  - The Colonel’s secret recipe of herbs and spices
  - The formula for Coca-Cola
- Recourses for dissemination
  - Sue employee who violates trade secret
  - Use legal mechanisms regarding theft to suppress disclosure

10

Derivative Works

- Main idea: works that derive from copyrighted works inherit those copyrights
  - Pro: provides protection for extending work, compiling it into another form, aggregating it with other works, etc.
  - Con: typically cannot copyright derivative work separately to extend copyright duration, unless significant new material
- Why it matters to developers
  - Who owns derivative works in a contract?
  - Note: adding a .o file into a library (.so, .a) is typically a derivative work

11

License

- The actual text that gives you permission to do something with a copyrighted/patented work
  - The “click-through” license on many software packages
  - The “end user license agreement” on boxed software
  - The text accompanying/following the copyright notice on many source files
- The license is a mini legal contract
  - It can contain whatever the licensor wants it to say
  - The licensee usually has some limited set of options
    - Accept license
    - Reject license, then no right to use/run software

12

Sample Licenses

- Single-user commercial license
  - You can install this software binary on one machine and use it only on that machine
- Commercial site license
  - You can install this software on 100-500 machines owned by some organization
- Permissive open source license
  - You can use this software and source code in any way that you want as long as this copyright notice stays attached to it
- Non-permissive free software license
  - You can do anything you want with this software and source as long as you give the source code along with any binary you distribute

13

Money

- How do you make money?
  - Consult
  - Work for a large company
  - Start a business
  - Work for a startup
  - Other
- Common issues
  - Salary
  - Profits, revenues
  - Stock & stock options
  - Ownership

14

Startup Economics

- Founders – may have salary, outright stock (not options)
  - Founders typically split original stock in company
  - Some portion of stock set aside for employee pool
  - Founders (and other employees) get ownership diluted as more stock is issued to investors
  - Founders own what are called “common shares”
- Employees – salaried, with stock options
  - Stock options may be “relatively” large % of company
  - First employee often 1-3% of company
- Investors
  - Typically angel funding for <$1M, venture capitalists for larger
  - Hold a “higher class” of stock that gets paid first, called preferred
  - May also hold debt, which is foremost a loan (and paid first)

15

How a Small Startup Works

- Two friends form a company, 50% each
  - Work for six months, build viable prototype
  - Release prototype, see market traction, need to grow
  - Approach investors
- Investor evaluates company, puts in money
  - May say that the current company is “worth” $2M
  - Invests $3M to grow the company
  - Post-money company now worth $5M ($2M + $3M)
  - Investor owns 60% of company
  - Investor may dictate that future employees receive 20% of shares
- Original owners now have 100% - 60% - 20% = 20% of shares

16

If Company Gets Acquired

- Acquired for $3M
  - Investor gets back all $3M, weeps
- Acquired for more than $3M
  - Investor may have “preference multiple”
  - If 2x or 3x, first 6M or 9M go to investor
  - If investor has participation, then after payback, investor’s shares become common shares
- Scenario: assume 1x preference + participation, $8M exit
  - Investor first gets $3M back
  - Now, $5M left for all shares
    - Investor gets 60% of this, another $3M
    - Owners get 20% of this, for $1M
    - Employees purchase their stock at strike price, get difference between that and remaining $1M

17

Why Take Outside Money

- Much easier to grow
  - Profits small, elusive when starting
  - Organic growth can be very slow
  - First-mover often signs up more customers
  - Market leader more likely to be acquired
- Less upside, less downside
  - In event of failure, investors lose, founders walk
  - In event of success, investors gain, founders gain less
  - Smaller % gain may beat larger % of small profit
- Investors have vested interest in success
  - Investors want to see 3x or more returns
  - Will work to make that happen if possible
  - If you owe $1M, it’s your problem. If you owe $100M, it’s their problem

18

Product Development

- Executive Team – CEO, CTO, VP for *X*…
  - Set strategic direction for product line
- Product Line Management (PLM)
  - Prioritize (triage) customer requirements
  - Work with Engineering to define a product roadmap
- Engineering
  - Work with PLM to define product roadmap
  - Implement requested features for next release
  - Work with QA and Ops to maintain releases
- Quality Assurance (QA)
  - Qualify releases (run regression tests)
- Operations / Customer Support
  - Operate running services / Respond to customer complaints

19

Product Cycle

- Agile Development (aka Scrum)
  - Sprint – Work in 2 or 3 week cycles
    - Deliver a functioning product (with more features) each sprint
  - Scrum Team – Work in a group of 6 – 10 engineers
    - Decide sprint tasks and deliver as a team
  - Product Owner – Team leader (liaison to PLM)
  - Scrum Master – Logistical support
- Release Cycle
  - Every *n* months (for = 3, 4, 6, or 12)
    - Major and minor releases (e.g., 2.5, 3.0)
    - Beta and General Availability (GA) releases (e.g., 2.7-B1)
  - Continue to support GA releases for *x* months (for *x* = 12, 18, or 24)
    - Periodically release maintenance updates (e.g., 2.7-R5)
  - Provide upgrade/downgrade scripts (e.g., from 2.7-R5 to 2.8-R1)
    - ISSU – In Service Software Upgrade

20

Software Business Models

- Licensed Software
  - Deliver software to be installed and used by the customer
  - Charge according to…
    - Installations (per-user, per-machine, per-site)
    - Capacity (e.g., bytes-per-second)
  - Term vs Perpetual (optional on-going maintenance contract)
- Software-as-a-Service (SaaS)
  - Operate a service on behalf of customers
  - Charge by usage (bytes stored/delivered, transactions completed)
  - Or, give service away for free and figure out how to monetize later
- Example – Akamai Technologies
  - Operate a Global Content Delivery Network (CDN) – SaaS
    - Customers are Content Providers (e.g., Netflix, ESPN,…)
  - Also now license CDN Technology to Telcos and MSOs

21

Scaling up a Software Service

- Buy servers and deploy them at hosting (co-location) sites
  - Rent "rack space" – pay for power, real estate, AC, connectivity
- Rent Virtual Machines (VMs) from a Cloud Provider
  - Amazon's EC2 (compute) and S3 (storage) – AWS
  - Microsoft's Window's Azure
  - Google App Engine
- Value Proposition
  - Acquire resources as demand dictates (elastic capacity)
  - Amortize administrative costs across many tenants
- Simple Taxonomy
  - IaaS – Infrastructure-as-a-Service (install your own OS & apps)
  - PaaS – Platform-as-a-Service (complete programming environment)
  - SaaS – Software-as-a-Service (a specific service, e.g., storage)

22

Leveraging Open Source

- Linux Distributions
  - Ubuntu, Debian, RedHat
- Apache Software Foundation
  - Apache Web Server (and much more)
  - Other open source web servers include NGINX and lighttpd
- OpenStack
  - Components to build a private cloud
- Free Software Foundation
  - GNU Project (gcc, gdb, bash, emacs,…)

These organizations also define prototype software licenses

- GNU Public License (GPL) – Free Software
- Apache License – Open Software

23

Free Software (GPL)

- Free derives from freedom, not necessarily free-of-charge
- License allows selling software & associated services
- Source code must be provided
  - Both for the GPL-licensed code & anything linked with it
  - No other restrictions can be added
  - E.g., cannot tell customers that they cannot release source
- How to monetize
  - Can sell the software (with source)
  - Can sell consulting/customization services
  - Can sell update services

24

How This Affects You

- You desire to avoid re-inventing wheel
- You need a component that does X
- You search for X, find it, and use it
- What happens next?

25

Using Licensed Components

- High-order rule: must follow license
- In permissively-licensed code
  - Typically, don't remove copyrights
  - Don't claim that portion of the code as your own
  - Can release source or not, as you choose
- In GPL-licensed code
  - Must release source for any _linked_ code
  - What is linking?
    - Typically, linking of .o files or libraries
    - GNU libc licensed under "lesser GPL" – linking OK
  - Who gets its?
    - Anyone who get binary from you

26

Loophole Case Study #1

- GPL code used to build network service
  - Client connection to GPL-based code via HTTP
  - Client sends request, gets response
  - Client never receives binary – only run on company-owned servers
- Technically, fully compliant with GPL (version 2)
- Violated the spirit, but not the letter, of GPL
- Response: Affero GPL, which extended source release requirements to cover network-based services

27

Loophole Case Study #2

- Device manufacturer uses Linux (GPLv2)
  - Provides all source code
  - All code can be compiled to produce binaries
  - Vendor-custom hardware checks for signed binaries
  - Signed binary has checksum function that produces a value stored with binary
  - Refuses to boot unsigned binaries
- Result: end user cannot boot modified system
  - Fully compliant with letter, not spirit of GPL
- Response: GPLv3 requires release of all signing tools, keys

28

Loophole Case Study #3

- Red Hat Linux wanted to dominate Linux market
  - Would periodically release updates of Red Hat Linux
  - Charged for packaged software & update service
- However, each release was immediately followed by other vendors selling the same CDs at low cost
  - Legal since all of Red Hat's system was GPL
- Red Hat employed trademark protection
  - CDs contained Red Hat trademarks, including name & logo
  - Argued that consumers would be confused by third party sources
- Response: none, really
  - CentOS formed to release RedHat releases w/o RedHat trademarks
  - Sometimes large delay between RedHat release & CentOS release

29

Loophole Case Study #4

- Popular database software released as GPL
- Vendor did not accept outside code contributions
  - Vendor owned copyright to all code
- Software also available as non-GPL commercial license
- GPL version spurred adoption, market share, commercial version paid the bills
- Result: vendor acquired for $1B
- Response: users worried about stagnation of GPL version
  - Nobody can revoke GPL version
  - New features do not have to appear simultaneously (if ever) in GPL version

30

Course Wrap-up

- Final project, final exam
  - Yes, end of semester is stressful
  - Breath deeply, close your eyes, imagine a tranquil place….
- Dean's Date is fixed
  - University deadline
  - Talk to us & res college dean early if many problems
- Final exam is ***closed-book, closed-notes***
  - Format comparable to midterm (except closed-everything)
  - 3 hours provided
  - Unlikely to have time pressure

31

Big Ideas

- Modularity
  - Well-defined interfaces between components
  - Allows changing the implementation of one component without changing another
  - The key to managing complexity in large systems
- Resource sharing
  - Time sharing of the CPU by multiple processes
  - Sharing of the physical memory by multiple processes
- Indirection
  - Representing address space with virtual memory
  - Manipulating data via pointers (or addresses)

32

Big Ideas Continued

- Hierarchy
  - Memory: registers, cache, main memory, disk, tape, …
  - Balancing the trade-off between fast/small and slow/big
- Bits can mean anything
  - Code, addresses, characters, pixels, money, grades, …
  - Arithmetic can be done through logic operations
  - The meaning of the bits depends entirely on how they are accessed, used, and manipulated

33